MT1000A Network Master Pro
MT1100A Network Master Flex
Remote Scripting Operation Manual

26th Edition

ANRITSU CORPORATION

Document No.: M-W3736AE-26.0

- This is an addendum to “Network Master Pro Operation Manual” and “Network Master Flex Operation Manual”.
- For safety and warning information, please read “Network Master Pro Operation Manual” or “Network Master Flex Operation Manual” before attempting to use the equipment.
- Keep this manual with the equipment.

Notes On Export Management

This product and its manuals may require an Export License/Approval by the Government of the product’s country of origin for re-export from your country. Before re-exporting the product or manuals, please contact us to confirm whether they are export-controlled items or not. When you dispose of export-controlled items, the products/manuals need to be broken/shredded so as not to be unlawfully used for military purpose.

October 6, 2014 (First Edition)
December 4, 2019 (26th Edition)

Copyright ©2014-2019, ANRITSU CORPORATION.
All rights reserved. No part of this manual may be reproduced without the prior written permission of the publisher.
The contents of this manual may be changed without prior notice.

About This Manual

This operation manual describes the SCPI (Standard Commands for Programmable Instruments) commands for Network Master Pro/Flex.

Note: SCPI commands described in this manual are supported in Network Master Pro/Flex version 9.01.

Some commands or queries in this manual may require that specific hardware or software options are installed. These options must be purchased separately.

This operation manual uses the notations described in the following standards:

- IEEE: Std 488.2-1992
- SCPI: VERSION 1999.0 (SCPI Consortium)

Contents

1 Overview **51**
1.1 Ethernet Based Remote Control . . . 51
1.1.1 Connecting Cable . . . 51
1.1.2 Connect to an Instrument . . . 52
1.1.3 Ethernet Remote Control Settings . . . 52
1.1.4 Communication Buffers . . . 52
1.2 Program Messages . . . 53
1.2.1 Program Message Unit . . . 53
1.2.2 Program Headers . . . 54
1.2.3 Program Data . . . 55
1.2.4 Program Message Terminator . . . 56
1.2.5 Compound Program Messages . . . 56
1.2.6 Sequential Execution . . . 56
1.3 Response Messages . . . 57
1.3.1 Response Data . . . 57
1.3.2 Response Messages Terminator . . . 59
1.3.3 Prompt . . . 59
1.4 Status . . . 60
1.4.1 Configuration of Status Registers . . . 60
1.4.2 IEEE488.2-defined Registers . . . 62
1.4.3 SCPI-defined Registers . . . 65
1.4.4 Network Master Device-Dependent Status Registers . . . 67
1.4.5 Reading, Writing and Clearing Status Registers . . . 68
1.5 Controller Example . . . 70
1.5.1 PuTTY . . . 70
1.6 Definitions . . . 73
1.6.1 NaN (Not a Number) . . . 73
1.6.2 → Right Arrow . . . 73
1.6.3 Data Bit (DB) . . . 73
1.6.4 Port Number (Logical Port) . . . 73

2 SCPI Conformance Information **75**
2.1 SCPI Version . . . 75
2.2 IEEE 488.2 Mandatory Commands . . . 76
2.2.1 *CLS . . . 76
2.2.2 *ESE . . . 76
2.2.3 *ESR? . . . 77
2.2.4 *IDN? . . . 77
2.2.5 *OPC . . . 77
2.2.6 *RST . . . 78
2.2.7 *SRE . . . 78
2.2.8 *STB? . . . 79
2.2.9 *TST? . . . 79
2.2.10 *WAI . . . 79
2.3 SCPI System Subsystem Commands . . . 80
2.3.1 SYSTem:VERSion? . . . 80
2.3.2 SYSTem:ERRor[:NEXT]? . . . 80
2.3.3 SYSTem:ERRor:ADDitional[:MESSage] . . . 80
2.3.4 SYSTem:DATE . . . 81

2.3.5 SYSTem:TIME . . . 81
2.3.6 SYSTem:REBoot . . . 81
2.3.7 SYSTem:GPS:NSATellites? . . . 82
2.3.8 SYSTem:GPS:TIME? . . . 82
2.3.9 SYSTem:GPS:LOCation? . . . 82
2.3.10 SYSTem:GPS:POWEr . . . 82
2.3.11 SYSTem:GPS:POWEr? . . . 82
2.3.12 SYSTem:GPS:SYNCmode . . . 82
2.3.13 SYSTem:GPS:SYNCmode? . . . 83
2.3.14 SYSTem:GPS:ANTNpower . . . 83
2.3.15 SYSTem:GPS:ANTNpower? . . . 83
2.3.16 SYSTem:GPS:PWRElapsed? . . . 83
2.3.17 SYSTem:GPS:LOCElapsed? . . . 83
2.3.18 SYSTem:GPS:SYNElapsed? . . . 84
2.3.19 SYSTem:COMMunicate:TERMinator . . . 84
2.3.20 SYSTem:PROMpt . . . 84
2.3.21 SYSTem:LOCal:CONTrol . . . 84
2.3.22 SYSTem:TIMing:EXTernal? . . . 85
2.3.23 SYSTem:STIMuli:INSert . . . 85
2.3.24 SYSTem:STIMuli:CLR . . . 85
2.3.25 SYSTem:WAIT[:IDLE] . . . 86
2.3.26 SYSTem:WAIT:DURation . . . 86
2.3.27 SYSTem:APPlication . . . 86
2.3.28 SYSTem:APPlication? . . . 86
2.4 SCPI Instrument Subsystem Commands . . . 87
2.4.1 Connection to Application Server . . . 87
2.4.2 Connection to multiple applications . . . 87
2.4.3 Connection from multiple users . . . 87
2.4.4 Force Termination of Application Server . . . 87
2.4.5 INSTrument:STARt[:DEFault] . . . 88
2.4.6 INSTrument:STARt:LAST . . . 89
2.4.7 INSTrument:STARt:GUI . . . 89
2.4.8 INSTrument:TERMinate . . . 89
2.4.9 INSTrument:TERMinate:FORCe . . . 90
2.4.10 INSTrument:COUNt? . . . 91
2.4.11 INSTrument:CATalog? . . . 91
2.4.12 INSTrument:STATe? . . . 91
2.4.13 INSTrument:CONNect . . . 91
2.4.14 INSTrument:CONNect:ALL . . . 92
2.4.15 INSTrument:CONNect[:CATalog]? . . . 92
2.4.16 INSTrument:DISConnect . . . 92
2.4.17 INSTrument[:SELect] . . . 92
2.4.18 INSTrument:ERRor[:NEXT]? . . . 93
2.4.19 INSTrument:PORT? . . . 93
2.4.20 INSTrument:PORT:FREE? . . . 93
2.4.21 INSTrument:PORT:CATalog? . . . 94
2.4.22 INSTrument:MODule:CATalog? . . . 94
2.4.23 INSTrument:CTRL:NAME? . . . 94
2.4.24 INSTrument:CTRL:SN? . . . 94
2.4.25 INSTrument:CTRL:TRT? . . . 94
2.4.26 INSTrument:CTRL:OPTion:CATalog? . . . 94
2.4.27 INSTrument:MODule<Md>:NAME? . . . 95
2.4.28 INSTrument:MODule<Md>:SN? . . . 95
2.4.29 INSTrument:MODule<Md>:TRT? . . . 95
2.4.30 INSTrument:MODule<Md>:OPTion:CATalog? . . . 95
2.5 SCPI Status Subsystem Commands . . . 96
2.5.1 STATus:OPERation[:EVENt]? . . . 96
2.5.2 STATus:OPERation:CONDition? . . . 96
2.5.3 STATus:OPERation:ENABle . . . 96

2.5.4 STATus:OPERation:PTRansition . . . 97
2.5.5 STATus:OPERation:NTRansition . . . 97
2.5.6 STATus:QUEStionable[:EVENt]? . . . 97
2.5.7 STATus:QUEStionable:CONDition? . . . 98
2.5.8 STATus:QUEStionable:ENABle . . . 98
2.5.9 STATus:QUEStionable:PTRansition . . . 98
2.5.10 STATus:QUEStionable:NTRansition . . . 99
2.5.11 STATus:PORT[:EVENt]? . . . 99
2.5.12 STATus:PORT:CONDition? . . . 99
2.5.13 STATus:PORT:ENABle . . . 99
2.5.14 STATus:PORT:PTRansition . . . 100
2.5.15 STATus:PORT:NTRansition . . . 100
2.5.16 STATus:PRESet . . . 101
2.5.17 STATus:INTerface:PORT<Pt>[:EVENt]? . . . 101
2.5.18 STATus:INTerface:PORT<Pt>:CONDition? . . . 102
2.6 Mass Memory Subsystem Commands . . . 103
2.6.1 MMEMory:LOAD . . . 103
2.6.2 MMEMory:STORe:STATe . . . 103
2.6.3 MMEMory:STORe:DATA . . . 103
2.6.4 MMEMory:DELete . . . 104
2.6.5 MMEMory:DATA? . . . 104
2.6.6 MMEMory:COPY . . . 104
2.6.7 MMEMory:MOVE . . . 104
2.6.8 MMEMory:INFO? . . . 105
2.6.9 MMEMory:CATalog? . . . 105
2.6.10 MMEMory:DCATalog? . . . 105
2.6.11 MMEMory:MDIRectory . . . 105
2.6.12 MMEMory:RDIRectory . . . 106
2.6.13 MMEMory:SAVE . . . 106

3 No frame **107**
3.1 Interface Setup . . . 107
3.1.1 NFRame:PORT<Pt>:INTerface:TYPE . . . 107
3.1.2 NFRame:PORT<Pt>:INTerface:BITRate . . . 108
3.1.3 NFRame:PORT<Pt>:INTerface:LANE . . . 108
3.1.4 NFRame:PORT<Pt>:TIMing . . . 109
3.1.5 NFRame:PORT<Pt>[:RX]:INTerface:PATTern:FOLLow . . . 109
3.1.6 NFRame:PORT<Pt>:TX:INTerface:PATTern . . . 109
3.1.7 NFRame:PORT<Pt>:RX:INTerface:PATTern . . . 110
3.1.8 NFRame:PORT<Pt>:TX:INTerface:PINVersion . . . 110
3.1.9 NFRame:PORT<Pt>:RX:INTerface:PINVersion . . . 111
3.1.10 NFRame:PORT<Pt>:TX:INTerface:LANE<Ln>:PATTern . . . 111
3.1.11 NFRame:PORT<Pt>:RX:INTerface:LANE<Ln>:PATTern . . . 112
3.1.12 NFRame:PORT<Pt>:TX:INTerface:LANE<Ln>:PINVersion . . . 112
3.1.13 NFRame:PORT<Pt>:RX:INTerface:LANE<Ln>:PINVersion . . . 112
3.1.14 NFRame:PORT<Pt>:TX:INTerface:PATTern:LANE:FOLLow . . . 113
3.1.15 NFRame:PORT<Pt>:RX:INTerface:PATTern:LANE:FOLLow . . . 113
3.2 Stimuli . . . 114
3.2.1 NFRame:PORT<Pt>:STIMuli:TX:FOFFset . . . 114
3.2.2 NFRame:PORT<Pt>:STIMuli:TX:LANE . . . 114
3.3 Result . . . 114
3.3.1 NFRame:PORT<Pt>[:RX]:IFETch? . . . 114
3.4 Status . . . 115
3.4.1 NFRame:STATus:PORT<Pt>[:RX]:AESummary[:EVENt]? . . . 115
3.4.2 NFRame:STATus:PORT<Pt>[:RX]:AESummary:CONDition? . . . 115
3.4.3 NFRame:STATus:PORT<Pt>[:RX]:ALARm[:EVENt]? . . . 116
3.4.4 NFRame:STATus:PORT<Pt>[:RX]:ALARm:CONDition? . . . 116
3.4.5 NFRame:STATus:PORT<Pt>[:RX]:ERRor[:EVENt]? . . . 116
3.4.6 NFRame:STATus:PORT<Pt>[:RX]:ERRor:CONDition? . . . 116

3.4.7 NFRame:STATus:PORT<Pt>[:RX]:PSLevel? . . . 117
3.4.8 NFRame:STATus:PORT<Pt>:TX:PSLevel? . . . 117
3.4.9 NFRame:STATus:PORT<Pt>[:RX]:PFRequency? . . . 117
3.4.10 NFRame:STATus:PORT<Pt>[:RX]:PDEViation? . . . 117
3.4.11 NFRame:STATus:PORT<Pt>[:RX]:PCDRlock? . . . 117

4 CPRI/OBSAI 119
4.1 Port . . . 119
4.1.1 CPRI:PORT<Pt>:MODE . . . 119
4.1.2 CPRI:PORT<Pt>:TIMing:SOURce . . . 119
4.1.3 CPRI:PORT<Pt>:LRATe . . . 120
4.1.4 CPRI:PORT<Pt>:LRATe:MODE . . . 120
4.1.5 CPRI:PORT<Pt>:CONTents . . . 121
4.1.6 CPRI:PORT<Pt>:FEC . . . 121
4.1.7 CPRI:PORT<Pt>:PATTern . . . 121
4.1.8 CPRI:PORT<Pt>:PINVersion . . . 122
4.1.9 CPRI:PORT<Pt>:UP32 . . . 122
4.1.10 CPRI:PORT<Pt>:PVERsion . . . 122
4.1.11 CPRI:PORT<Pt>:PROLe . . . 123
4.1.12 CPRI:PORT<Pt>:SSEQence . . . 123
4.1.13 CPRI:PORT<Pt>:CMHDlc . . . 123
4.1.14 CPRI:PORT<Pt>:EENable . . . 124
4.1.15 CPRI:PORT<Pt>:CMEPointer . . . 124
4.1.16 CPRI:PORT<Pt>:OBSai:FSSeed . . . 124
4.1.17 CPRI:PORT<Pt>:OBSai:TX:FIDLe . . . 125
4.1.18 CPRI:PORT<Pt>:OBSai:TX:SSINdex . . . 125
4.1.19 CPRI:PORT<Pt>:OBSai:TX:SSValue . . . 125
4.1.20 CPRI:PORT<Pt>:OBSai:TX:ADDRess . . . 126
4.1.21 CPRI:PORT<Pt>:OBSai:TX:TYPE . . . 126
4.1.22 CPRI:PORT<Pt>:OBSai:RX:FILTer . . . 126
4.1.23 CPRI:PORT<Pt>:OBSai:RX:SSINdex . . . 127
4.1.24 CPRI:PORT<Pt>:OBSai:RX:SSValue . . . 127
4.1.25 CPRI:PORT<Pt>:OBSai:RX:ADDRess . . . 127
4.1.26 CPRI:PORT<Pt>:OBSai:RX:TYPE . . . 127
4.1.27 CPRI:PORT<Pt>:SETup:FOLLow . . . 128
4.2 Stimuli . . . 129
4.2.1 CPRI:PORT<Pt>:STIMuli:ALARm . . . 129
4.2.2 CPRI:PORT<Pt>:STIMuli:AINSertion . . . 129
4.2.3 CPRI:PORT<Pt>:STIMuli:ERRor . . . 130
4.2.4 CPRI:PORT<Pt>:STIMuli:EINSertion . . . 130
4.2.5 CPRI:PORT<Pt>:STIMuli:EBLength . . . 130
4.2.6 CPRI:PORT<Pt>:STIMuli:FOFFset . . . 131
4.3 Result . . . 132
4.3.1 CPRI:PORT<Pt>:IFETch? . . . 132
4.4 Status . . . 133
4.4.1 CPRI:PORT<Pt>:STATus:AESummary[:EVENt]? . . . 133
4.4.2 CPRI:PORT<Pt>:STATus:AESummary:CONDiton? . . . 133
4.4.3 CPRI:PORT<Pt>:STATus:ALARm<section>[:EVENt]? . . . 133
4.4.4 CPRI:PORT<Pt>:STATus:ALARm<section>:CONDition? . . . 134
4.4.5 CPRI:PORT<Pt>:STATus:ERRor[:EVENt]? . . . 134
4.4.6 CPRI:PORT<Pt>:STATus:ERRor:CONDition? . . . 134
4.4.7 CPRI:PORT<Pt>:STATus:FEC[:EVENt]? . . . 134
4.4.8 CPRI:PORT<Pt>:STATus:FEC:CONDition? . . . 134
4.4.9 CPRI:PORT<Pt>:STATus:TX:PSLevel? . . . 135
4.4.10 CPRI:PORT<Pt>:STATus:RX:PSLevel? . . . 135
4.4.11 CPRI:PORT<Pt>:STATus:TX:PDEViation? . . . 135
4.4.12 CPRI:PORT<Pt>:STATus:RX:PDEViation? . . . 135
4.4.13 CPRI:PORT<Pt>:STATus:TX:PBRate? . . . 135
4.4.14 CPRI:PORT<Pt>:STATus:RX:PBRate? . . . 136

4.4.15 CPRI:PORT<Pt>:STATus:TX:PPBRate? . . . 136
4.4.16 CPRI:PORT<Pt>:STATus:RX:PPBRate? . . . 136
4.4.17 CPRI:PORT<Pt>:STATus:LINK:STATe? . . . 136
4.4.18 CPRI:PORT<Pt>:STATus:LINK:PROLe? . . . 136
4.4.19 CPRI:PORT<Pt>:STATus:RX:LINK:PVERsion? . . . 136
4.4.20 CPRI:PORT<Pt>:STATus:RX:LINK:HRATe? . . . 137
4.4.21 CPRI:PORT<Pt>:STATus:RX:LINK:POINterp? . . . 137
4.4.22 CPRI:PORT<Pt>:STATus:TX:LINK:PVERsion? . . . 137
4.4.23 CPRI:PORT<Pt>:STATus:TX:LINK:HRATe? . . . 137
4.4.24 CPRI:PORT<Pt>:STATus:TX:LINK:POINterp? . . . 137
4.4.25 CPRI:PORT<Pt>:STATus:OBSai:TX:LINK:STATe? . . . 137
4.4.26 CPRI:PORT<Pt>:STATus:OBSai:RX:LINK:STATe? . . . 138
4.5 RTD . . . 139
4.5.1 CPRI:PORT<Pt>:RTD:ENABle . . . 139
4.5.2 CPRI:PORT<Pt>:RTD:MLIMit . . . 139
4.5.3 CPRI:PORT<Pt>:RTD:NUMBer? . . . 139
4.5.4 CPRI:PORT<Pt>:RTD:ATIMe? . . . 139
4.5.5 CPRI:PORT<Pt>:RTD:MTIMe? . . . 140
4.5.6 CPRI:PORT<Pt>:RTD:LTIMe? . . . 140
4.6 APS . . . 141
4.6.1 CPRI:PORT<Pt>:APS:ENABle . . . 141
4.6.2 CPRI:PORT<Pt>:APS:EVENt . . . 141
4.6.3 CPRI:PORT<Pt>:APS:PERiod . . . 142
4.6.4 CPRI:PORT<Pt>:APS:MLIMit . . . 142
4.6.5 CPRI:PORT<Pt>:APS:NUMBer? . . . 142
4.6.6 CPRI:PORT<Pt>:APS:ATIMe? . . . 142
4.6.7 CPRI:PORT<Pt>:APS:MTIMe? . . . 143
4.6.8 CPRI:PORT<Pt>:APS:LTIMe? . . . 143
4.6.9 CPRI:PORT<Pt>:APS:CTIMe? . . . 143

5 2 Mbps 145
5.1 Receiver . . . 145
5.1.1 TMBPs:RX<Pt>[:ENABled] . . . 145
5.1.2 TMBPs:RX<Pt>:FOLLow . . . 145
5.1.3 TMBPs:RX<Pt>:CONNector . . . 146
5.1.4 TMBPs:RX<Pt>:MODE . . . 146
5.1.5 TMBPs:RX<Pt>:SENSitivity . . . 146
5.1.6 TMBPs:RX<Pt>:CODE . . . 147
5.1.7 TMBPs:RX<Pt>:PCMFrame . . . 147
5.1.8 TMBPs:RX<Pt>:CRC4 . . . 147
5.1.9 TMBPs:RX<Pt>:EBITs . . . 148
5.1.10 TMBPs:RX<Pt>:PATTern . . . 148
5.1.11 TMBPs:RX<Pt>:PINVersion . . . 149
5.1.12 TMBPs:RX<Pt>:PTSLots . . . 149
5.1.13 TMBPs:RX<Pt>:UP16 . . . 150
5.1.14 TMBPs:RX<Pt>:UP32 . . . 150
5.1.15 TMBPs:RX<Pt>:UP2K . . . 150
5.1.16 TMBPs:RX<Pt>:AUDio . . . 151
5.1.17 TMBPs:RX<Pt>:ATSLot . . . 151
5.1.18 TMBPs:RX<Pt>:CAS . . . 151
5.2 Transmitter . . . 152
5.2.1 TMBPs:TX<Pt>[:ENABled] . . . 152
5.2.2 TMBPs:TX<Pt>:FOLLow . . . 152
5.2.3 TMBPs:TX<Pt>:CONNector . . . 152
5.2.4 TMBPs:TX<Pt>:DINSert[:ENABle] . . . 153
5.2.5 TMBPs:TX<Pt>:TIMing . . . 153
5.2.6 TMBPs:TX<Pt>:FOFFset . . . 153
5.2.7 TMBPs:TX<Pt>:CODE . . . 154
5.2.8 TMBPs:TX<Pt>:PCMFrame . . . 154

5.2.9 TMBPs:TX<Pt>:CRC4 . . . 154
5.2.10 TMBPs:TX<Pt>:SABits . . . 155
5.2.11 TMBPs:TX<Pt>:SABMode . . . 155
5.2.12 TMBPs:TX<Pt>:SABMode? . . . 156
5.2.13 TMBPs:TX<Pt>:PATTern . . . 156
5.2.14 TMBPs:TX<Pt>:PINVersion . . . 156
5.2.15 TMBPs:TX<Pt>:PTSLots . . . 157
5.2.16 TMBPs:TX<Pt>:UP16 . . . 157
5.2.17 TMBPs:TX<Pt>:UP32 . . . 157
5.2.18 TMBPs:TX<Pt>:UP2K . . . 158
5.2.19 TMBPs:TX<Pt>:UTSLots . . . 158
5.2.20 TMBPs:TX<Pt>:SCContent . . . 159
5.2.21 TMBPs:TX<Pt>:SCTSlot . . . 159
5.2.22 TMBPs:TX<Pt>:TFR . . . 159
5.2.23 TMBPs:TX<Pt>:TLEVel . . . 160
5.2.24 TMBPs:TX<Pt>:CAS . . . 160
5.2.25 TMBPs:TX<Pt>:CASChannel . . . 160
5.2.26 TMBPs:TX<Pt>:CASBits . . . 161
5.2.27 TMBPs:TX<Pt>:COCBits . . . 161
5.3 Stimuli . . . 162
5.3.1 TMBPs:STIMuli:TX<Pt>:ALARm . . . 162
5.3.2 TMBPs:STIMuli:TX<Pt>:ERRor . . . 162
5.3.3 TMBPs:STIMuli:TX<Pt>:EINSert . . . 163
5.3.4 TMBPs:STIMuli:TX<Pt>:EBLength . . . 163
5.4 Result . . . 163
5.4.1 TMBPs:RX<Pt>:IFETch? . . . 163
5.5 Status . . . 165
5.5.1 TMBPs:STATus:RX<Pt>:AESummary[:EVENt]? . . . 165
5.5.2 TMBPs:STATus:RX<Pt>:AESummary:CONDition? . . . 165
5.5.3 TMBPs:STATus:RX<Pt>:ALARm[:EVENt]? . . . 165
5.5.4 TMBPs:STATus:RX<Pt>:ALARm:CONDition? . . . 166
5.5.5 TMBPs:STATus:RX<Pt>:ERRor[:EVENt]? . . . 166
5.5.6 TMBPs:STATus:RX<Pt>:ERRor:CONDition? . . . 166
5.5.7 TMBPs:STATus:RX<Pt>:PSLevel? . . . 167
5.5.8 TMBPs:STATus:RX<Pt>:PDEViation? . . . 167
5.5.9 TMBPs:STATus:RX<Pt>:PBRate? . . . 167
5.5.10 TMBPs:STATus:RX<Pt>:PPBRate? . . . 167
5.5.11 TMBPs:STATus:RX<Pt>:FNFWord? . . . 167
5.5.12 TMBPs:STATus:RX<Pt>:FDUMp? . . . 168
5.5.13 TMBPs:STATus:RX<Pt>:ACONtent? . . . 168
5.5.14 TMBPs:STATus:RX<Pt>:AEBContent? . . . 168
5.5.15 TMBPs:STATus:RX<Pt>:APPeak? . . . 168
5.5.16 TMBPs:STATus:RX<Pt>:ANPeak? . . . 168
5.5.17 TMBPs:STATus:RX<Pt>:ALEVel? . . . 168
5.5.18 TMBPs:STATus:RX<Pt>:AFRequency? . . . 169
5.5.19 TMBPs:STATus:RX<Pt>:ACOFfset? . . . 169
5.5.20 TMBPs:STATus:RX<Pt>:CMSignal? . . . 169
5.5.21 TMBPs:STATus:RX<Pt>:CBITs? . . . 169
5.6 APS . . . 170
5.6.1 TMBPs:APS:STARt . . . 170
5.6.2 TMBPs:APS:STOP . . . 170
5.6.3 TMBPs:APS:RX<Pt>:NUMBer? . . . 170
5.6.4 TMBPs:APS:RX<Pt>:ATIMe? . . . 170
5.6.5 TMBPs:APS:RX<Pt>:MTIMe? . . . 170
5.6.6 TMBPs:APS:RX<Pt>:LTIMe? . . . 171
5.6.7 TMBPs:APS:RX<Pt>:EVENt . . . 171
5.6.8 TMBPs:APS:RX<Pt>:MLIMit . . . 171
5.7 RTD . . . 172
5.7.1 TMBPs:RTD:RX<Pt>:NUMBer? . . . 172

5.7.2 TMBPs:RTD:RX<Pt>:ATIMe? . . . 172
5.7.3 TMBPs:RTD:RX<Pt>:MTIMe? . . . 172
5.7.4 TMBPs:RTD:RX<Pt>:LTIMe? . . . 172

6 T1 173
6.1 Receiver . . . 173
6.1.1 T1:RX<Pt>[:ENABled] . . . 173
6.1.2 T1:RX<Pt>:FOLLow . . . 173
6.1.3 T1:RX<Pt>:MODE . . . 174
6.1.4 T1:RX<Pt>:CODE . . . 174
6.1.5 T1:RX<Pt>:PCMFrame . . . 174
6.1.6 T1:RX<Pt>:FTYPe . . . 175
6.1.7 T1:RX<Pt>:PATTern . . . 175
6.1.8 T1:RX<Pt>:PINVersion . . . 176
6.1.9 T1:RX<Pt>:PTSLots . . . 176
6.1.10 T1:RX<Pt>:UP16 . . . 176
6.1.11 T1:RX<Pt>:UP32 . . . 177
6.1.12 T1:TX<Pt>:SCContent . . . 177
6.1.13 T1:RX<Pt>:AUDio . . . 177
6.1.14 T1:RX<Pt>:ATSLot . . . 178
6.1.15 T1:RX<Pt>:UP2K . . . 178
6.1.16 T1:RX<Pt>:CAS . . . 178
6.2 Transmitter . . . 179
6.2.1 T1:TX<Pt>[:ENABled] . . . 179
6.2.2 T1:TX<Pt>:FOLLow . . . 179
6.2.3 T1:TX<Pt>:DINSert[:ENABle] . . . 180
6.2.4 T1:TX<Pt>:TIMing . . . 180
6.2.5 T1:TX<Pt>:FOFFset . . . 180
6.2.6 T1:TX<Pt>:CODE . . . 181
6.2.7 T1:TX<Pt>:LBOut . . . 181
6.2.8 T1:TX<Pt>:PCMFrame . . . 181
6.2.9 T1:TX<Pt>:FTYPe . . . 182
6.2.10 T1:TX<Pt>:PATTern . . . 182
6.2.11 T1:TX<Pt>:PINVersion . . . 183
6.2.12 T1:TX<Pt>:PTSLots . . . 183
6.2.13 T1:TX<Pt>:UP16 . . . 184
6.2.14 T1:TX<Pt>:UP32 . . . 184
6.2.15 T1:TX<Pt>:UP2K . . . 184
6.2.16 T1:TX<Pt>:UTSLots . . . 185
6.2.17 T1:TX<Pt>:SCTSlot . . . 185
6.2.18 T1:TX<Pt>:TFR . . . 185
6.2.19 T1:TX<Pt>:TLEVel . . . 186
6.2.20 T1:TX<Pt>:CAS . . . 186
6.2.21 T1:TX<Pt>:CASChannel . . . 186
6.2.22 T1:TX<Pt>:CASBits . . . 187
6.2.23 T1:TX<Pt>:COCBits . . . 187
6.2.24 T1:TX<Pt>:SF:CASBits . . . 187
6.2.25 T1:TX<Pt>:SF:COCBits . . . 188
6.3 Stimuli . . . 188
6.3.1 T1:STIMuli:TX<Pt>:ALARm . . . 188
6.3.2 T1:STIMuli:TX<Pt>:ERRor . . . 189
6.3.3 T1:STIMuli:TX<Pt>:EINSert . . . 189
6.3.4 T1:STIMuli:TX<Pt>:EBLength . . . 190
6.3.5 T1:STIMuli:TX<Pt>:FDLink . . . 190
6.3.6 T1:STIMuli:TX<Pt>:FDLink:UFDL . . . 191
6.3.7 T1:STIMuli:TX<Pt>:FDLink:IBCode . . . 191
6.4 Result . . . 192
6.4.1 T1:RX<Pt>:IFETch? . . . 192
6.5 Status . . . 193

6.5.1 T1:STATus:RX<Pt>:AESummary[:EVENt]? . . . 193
6.5.2 T1:STATus:RX<Pt>:AESummary:CONDition? . . . 193
6.5.3 T1:STATus:RX<Pt>:ALARm[:EVENt]? . . . 193
6.5.4 T1:STATus:RX<Pt>:ALARm:CONDition? . . . 194
6.5.5 T1:STATus:RX<Pt>:ERRor[:EVENt]? . . . 194
6.5.6 T1:STATus:RX<Pt>:ERRor:CONDition? . . . 194
6.5.7 T1:STATus:RX<Pt>:PSLevel? . . . 195
6.5.8 T1:STATus:RX<Pt>:PDEViation? . . . 195
6.5.9 T1:STATus:RX<Pt>:PBRate? . . . 195
6.5.10 T1:STATus:RX<Pt>:FDUMp? . . . 195
6.5.11 T1:STATus:RX<Pt>:ACONtent? . . . 195
6.5.12 T1:STATus:RX<Pt>:AIBContent? . . . 195
6.5.13 T1:STATus:RX<Pt>:APPeak? . . . 196
6.5.14 T1:STATus:RX<Pt>:ANPeak? . . . 196
6.5.15 T1:STATus:RX<Pt>:ALEVel? . . . 196
6.5.16 T1:STATus:RX<Pt>:AFRequency? . . . 196
6.5.17 T1:STATus:RX<Pt>:ACOFfset? . . . 196
6.5.18 T1:STATus:RX<Pt>:PPBRate? . . . 196
6.5.19 T1:STATus:RX<Pt>:FBIT? . . . 197
6.5.20 T1:STATus:RX<Pt>:SBIT? . . . 197
6.5.21 T1:STATus:RX<Pt>:MBIT? . . . 197
6.5.22 T1:STATus:RX<Pt>:FDL:MODE . . . 197
6.5.23 T1:STATus:RX<Pt>:FDL:MBIT:CODE? . . . 197
6.5.24 T1:STATus:RX<Pt>:FDL:INBand:CODE? . . . 198
6.5.25 T1:STATus:RX<Pt>:FDL:UDEFined . . . 198
6.5.26 T1:STATus:RX<Pt>:FDL:UCODe . . . 198
6.5.27 T1:STATus:RX<Pt>:FDL:TRIGger . . . 198
6.5.28 T1:STATus:RX<Pt>:CBITs? . . . 199
6.6 APS . . . 200
6.6.1 T1:APS:STARt . . . 200
6.6.2 T1:APS:STOP . . . 200
6.6.3 T1:APS:RX<Pt>:NUMBer? . . . 200
6.6.4 T1:APS:RX<Pt>:ATIMe? . . . 200
6.6.5 T1:APS:RX<Pt>:MTIMe? . . . 200
6.6.6 T1:APS:RX<Pt>:LTIMe? . . . 201
6.6.7 T1:APS:RX<Pt>:EVENt . . . 201
6.6.8 T1:APS:RX<Pt>:MLIMit . . . 201
6.7 RTD . . . 202
6.7.1 T1:RTD:RX<Pt>:NUMBer? . . . 202
6.7.2 T1:RTD:RX<Pt>:ATIMe? . . . 202
6.7.3 T1:RTD:RX<Pt>:MTIMe? . . . 202
6.7.4 T1:RTD:RX<Pt>:LTIMe? . . . 202

7 E3 . . . 203
7.1 Receiver . . . 203
7.1.1 E3:RX<Pt>[:ENABled] . . . 203
7.1.2 E3:RX<Pt>:MODE . . . 203
7.1.3 E3:RX<Pt>:FOLLow . . . 204
7.1.4 E3:RX<Pt>:PCMFrame . . . 204
7.1.5 E3:RX<Pt>:PATTern . . . 205
7.1.6 E3:RX<Pt>:PINVersion . . . 205
7.1.7 E3:RX<Pt>:UP16 . . . 206
7.1.8 E3:RX<Pt>:UP32 . . . 206
7.1.9 E3:RX<Pt>:UP2K . . . 206
7.2 Transmitter . . . 207
7.2.1 E3:TX<Pt>[:ENABled] . . . 207
7.2.2 E3:TX<Pt>:FOLLow . . . 207
7.2.3 E3:TX<Pt>:TIMing . . . 207
7.2.4 E3:TX<Pt>:FOFFset . . . 208

7.2.5 E3:TX<Pt>:PCMFrame . . . 208
7.2.6 E3:TX<Pt>:PATTern . . . 209
7.2.7 E3:TX<Pt>:PINVersion . . . 209
7.2.8 E3:TX<Pt>:UP16 . . . 210
7.2.9 E3:TX<Pt>:UP32 . . . 210
7.2.10 E3:TX<Pt>:UP2K . . . 210
7.3 Stimuli . . . 211
7.3.1 E3:STIMuli:TX<Pt>:ALARm . . . 211
7.3.2 E3:STIMuli:TX<Pt>:ERRor . . . 211
7.3.3 E3:STIMuli:TX<Pt>:EINSert . . . 212
7.3.4 E3:STIMuli:TX<Pt>:EBLength . . . 212
7.4 Results . . . 213
7.4.1 E3:RX<Pt>:IFETch? . . . 213
7.5 Status . . . 214
7.5.1 E3:STATus:RX<Pt>:PSLevel? . . . 214
7.5.2 E3:STATus:RX<Pt>:PDEViation? . . . 214
7.5.3 E3:STATus:RX<Pt>:PBRate? . . . 214
7.5.4 E3:STATus:RX<Pt>:PPBRate? . . . 214
7.5.5 E3:STATus:RX<Pt>:AESummary[:EVENt]? . . . 214
7.5.6 E3:STATus:RX<Pt>:AESummary:CONDition? . . . 215
7.5.7 E3:STATus:RX<Pt>:ALARm[:EVENt]? . . . 215
7.5.8 E3:STATus:RX<Pt>:ALARm:CONDition? . . . 215
7.5.9 E3:STATus:RX<Pt>:ERRor[:EVENt]? . . . 216
7.5.10 E3:STATus:RX<Pt>:ERRor:CONDition? . . . 216
7.6 RTD . . . 217
7.6.1 E3:RTD:RX<Pt>:NUMBer? . . . 217
7.6.2 E3:RTD:RX<Pt>:ATIMe? . . . 217
7.6.3 E3:RTD:RX<Pt>:MTIMe? . . . 217
7.6.4 E3:RTD:RX<Pt>:LTIMe? . . . 217

8 T3 . . . 219
8.1 Receiver . . . 219
8.1.1 T3:RX<Pt>[:ENABled] . . . 219
8.1.2 T3:RX<Pt>:MODE . . . 219
8.1.3 T3:RX<Pt>:FOLLow . . . 220
8.1.4 T3:RX<Pt>:PCMFrame . . . 220
8.1.5 T3:RX<Pt>:FTYPe . . . 220
8.1.6 T3:RX<Pt>:PATTern . . . 221
8.1.7 T3:RX<Pt>:PINVersion . . . 221
8.1.8 T3:RX<Pt>:UP16 . . . 222
8.1.9 T3:RX<Pt>:UP32 . . . 222
8.1.10 T3:RX<Pt>:UP2K . . . 222
8.2 Transmitter . . . 223
8.2.1 T3:TX<Pt>[:ENABled] . . . 223
8.2.2 T3:TX<Pt>:FOLLow . . . 223
8.2.3 T3:TX<Pt>:TIMing . . . 223
8.2.4 T3:TX<Pt>:FOFFset . . . 224
8.2.5 T3:TX<Pt>:PCMFrame . . . 224
8.2.6 T3:TX<Pt>:FTYPe . . . 224
8.2.7 T3:TX<Pt>:LBOut . . . 225
8.2.8 T3:TX<Pt>:PATTern . . . 225
8.2.9 T3:TX<Pt>:PINVersion . . . 226
8.2.10 T3:TX<Pt>:UP16 . . . 226
8.2.11 T3:TX<Pt>:UP32 . . . 226
8.2.12 T3:TX<Pt>:UP2K . . . 227
8.3 Stimuli . . . 228
8.3.1 T3:STIMuli:TX<Pt>:ALARm . . . 228
8.3.2 T3:STIMuli:TX<Pt>:ERRor . . . 228
8.3.3 T3:STIMuli:TX<Pt>:EINSert . . . 229

8.3.4 T3:STIMuli:TX<Pt>:EBLength . . . 229
8.4 Results . . . 230
8.4.1 T3:RX<Pt>:IFETch? . . . 230
8.5 Status . . . 231
8.5.1 T3:STATus:RX<Pt>:PSLevel? . . . 231
8.5.2 T3:STATus:RX<Pt>:PDEViation? . . . 231
8.5.3 T3:STATus:RX<Pt>:PBRate? . . . 231
8.5.4 T3:STATus:RX<Pt>:PPBRate? . . . 231
8.5.5 T3:STATus:RX<Pt>:AESummary[:EVENt]? . . . 231
8.5.6 T3:STATus:RX<Pt>:AESummary:CONDition? . . . 232
8.5.7 T3:STATus:RX<Pt>:ALARm[:EVENt]? . . . 232
8.5.8 T3:STATus:RX<Pt>:ALARm:CONDition? . . . 232
8.5.9 T3:STATus:RX<Pt>:ERRor[:EVENt]? . . . 233
8.5.10 T3:STATus:RX<Pt>:ERRor:CONDition? . . . 233
8.6 RTD . . . 234
8.6.1 T3:RTD:RX<Pt>:NUMBer? . . . 234
8.6.2 T3:RTD:RX<Pt>:ATIMe? . . . 234
8.6.3 T3:RTD:RX<Pt>:MTIMe? . . . 234
8.6.4 T3:RTD:RX<Pt>:LTIMe? . . . 234

9 E4 . . . 235
9.1 Receiver . . . 235
9.1.1 E4:RX<Pt>[:ENABled] . . . 235
9.1.2 E4:RX<Pt>:GAIN . . . 235
9.1.3 E4:RX<Pt>:FOLLow . . . 236
9.1.4 E4:RX<Pt>:PCMFrame . . . 236
9.1.5 E4:RX<Pt>:PATTern . . . 237
9.1.6 E4:RX<Pt>:PINVersion . . . 237
9.1.7 E4:RX<Pt>:UP16 . . . 238
9.1.8 E4:RX<Pt>:UP32 . . . 238
9.1.9 E4:RX<Pt>:UP2K . . . 238
9.2 Transmitter . . . 239
9.2.1 E4:TX<Pt>[:ENABled] . . . 239
9.2.2 E4:TX<Pt>:FOLLow . . . 239
9.2.3 E4:TX<Pt>:TIMing . . . 239
9.2.4 E4:TX<Pt>:FOFFset . . . 240
9.2.5 E4:TX<Pt>:PCMFrame . . . 240
9.2.6 E4:TX<Pt>:PATTern . . . 241
9.2.7 E4:TX<Pt>:PINVersion . . . 241
9.2.8 E4:TX<Pt>:UP16 . . . 242
9.2.9 E4:TX<Pt>:UP32 . . . 242
9.2.10 E4:TX<Pt>:UP2K . . . 242
9.3 Stimuli . . . 243
9.3.1 E4:STIMuli:TX<Pt>:ALARm . . . 243
9.3.2 E4:STIMuli:TX<Pt>:ERRor . . . 243
9.3.3 E4:STIMuli:TX<Pt>:EINSert . . . 244
9.3.4 E4:STIMuli:TX<Pt>:EBLength . . . 244
9.4 Status . . . 245
9.4.1 E4:STATus:RX<Pt>:PSLevel? . . . 245
9.4.2 E4:STATus:RX<Pt>:PDEViation? . . . 245
9.4.3 E4:STATus:RX<Pt>:PBRate? . . . 245
9.4.4 E4:STATus:RX<Pt>:PPBRate? . . . 245
9.4.5 E4:STATus:RX<Pt>:AESummary[:EVENt]? . . . 245
9.4.6 E4:STATus:RX<Pt>:AESummary:CONDition? . . . 246
9.4.7 E4:STATus:RX<Pt>:ALARm[:EVENt]? . . . 246
9.4.8 E4:STATus:RX<Pt>:ALARm:CONDition? . . . 246
9.4.9 E4:STATus:RX<Pt>:ERRor[:EVENt]? . . . 247
9.4.10 E4:STATus:RX<Pt>:ERRor:CONDition? . . . 247
9.5 Results . . . 248

9.5.1 E4:RX<Pt>:IFETch? . . . 248
9.6 RTD . . . 249
9.6.1 E4:RTD:RX<Pt>:NUMBer? . . . 249
9.6.2 E4:RTD:RX<Pt>:ATIMe? . . . 249
9.6.3 E4:RTD:RX<Pt>:MTIMe? . . . 249
9.6.4 E4:RTD:RX<Pt>:LTIMe? . . . 249

10 SDH **251**
10.1 Receiver . . . 251
10.1.1 SDH:RX<Pt>:INTerface . . . 251
10.1.2 SDH:RX<Pt>:STMLevel . . . 251
10.1.3 SDH:RX<Pt>:AUGRoup . . . 252
10.1.4 SDH:RX<Pt>:CONCatenation . . . 252
10.1.5 SDH:RX<Pt>:CONTainer . . . 253
10.1.6 SDH:RX<Pt>:SSTRucture . . . 253
10.1.7 SDH:RX<Pt>:PATTern . . . 254
10.1.8 SDH:RX<Pt>:UP16 . . . 254
10.1.9 SDH:RX<Pt>:UP32 . . . 255
10.1.10 SDH:RX<Pt>:UP2K . . . 255
10.1.11 SDH:RX<Pt>:PINVersion . . . 255
10.1.12 SDH:RX<Pt>:TCM . . . 256
10.1.13 SDH:RX<Pt>:TUG3 . . . 256
10.1.14 SDH:RX<Pt>:TUG2 . . . 256
10.1.15 SDH:RX<Pt>:TU12 . . . 257
10.1.16 SDH:RX<Pt>:TU11 . . . 257
10.1.17 SDH:RX<Pt>:CHANnel . . . 258
10.1.18 SDH:RX<Pt>:MAPPing . . . 258
10.1.19 SDH:RX<Pt>:GAIN . . . 258
10.1.20 SDH:RX<Pt>:FOLLow . . . 259
10.1.21 SDH:RX<Pt>:MEASurement:HPLM . . . 259
10.1.22 SDH:RX<Pt>:MEASurement:LPLM . . . 259
10.2 Transmitter . . . 260
10.2.1 SDH:TX<Pt>:INTerface . . . 260
10.2.2 SDH:TX<Pt>[:ENABle] . . . 260
10.2.3 SDH:TX<Pt>:OPTical . . . 260
10.2.4 SDH:TX<Pt>:ELECtrical . . . 261
10.2.5 SDH:TX<Pt>:STMLevel . . . 261
10.2.6 SDH:TX<Pt>:AUGRoup . . . 262
10.2.7 SDH:TX<Pt>:CONCatenation . . . 262
10.2.8 SDH:TX<Pt>:CONTainer . . . 263
10.2.9 SDH:TX<Pt>:SSTRucture . . . 263
10.2.10 SDH:TX<Pt>:PATTern . . . 264
10.2.11 SDH:TX<Pt>:UP16 . . . 264
10.2.12 SDH:TX<Pt>:UP32 . . . 265
10.2.13 SDH:TX<Pt>:UP2K . . . 265
10.2.14 SDH:TX<Pt>:PINVersion . . . 265
10.2.15 SDH:TX<Pt>:TIMing . . . 266
10.2.16 SDH:TX<Pt>:TCM . . . 266
10.2.17 SDH:TX<Pt>:TUG3 . . . 266
10.2.18 SDH:TX<Pt>:TUG2 . . . 267
10.2.19 SDH:TX<Pt>:TU12 . . . 267
10.2.20 SDH:TX<Pt>:TU11 . . . 268
10.2.21 SDH:TX<Pt>:CHANnel . . . 268
10.2.22 SDH:TX<Pt>:MAPPing . . . 268
10.2.23 SDH:TX<Pt>:SOH:DEFault . . . 269
10.2.24 SDH:TX<Pt>:SOH:TRACe . . . 269
10.2.25 SDH:TX<Pt>:SOH:TRACe:CRC . . . 269
10.2.26 SDH:TX<Pt>:SOH . . . 270
10.2.27 SDH:TX<Pt>:VC4:POH:DEFault . . . 271

10.2.28 SDH:TX<Pt>:VC4:POH:TRACe . . . 271
10.2.29 SDH:TX<Pt>:VC4:POH:TRACe:CRC . . . 271
10.2.30 SDH:TX<Pt>:VC4:POH:TTCM . . . 272
10.2.31 SDH:TX<Pt>:VC4:POH . . . 272
10.2.32 SDH:TX<Pt>:VC3:POH:DEFault . . . 273
10.2.33 SDH:TX<Pt>:VC3:POH:TRACe . . . 273
10.2.34 SDH:TX<Pt>:VC3:POH:TRACe:CRC . . . 273
10.2.35 SDH:TX<Pt>:VC3:POH:TTCM . . . 274
10.2.36 SDH:TX<Pt>:VC3:POH . . . 274
10.2.37 SDH:TX<Pt>:VC12:POH:DEFault . . . 274
10.2.38 SDH:TX<Pt>:VC12:POH:TRACe . . . 275
10.2.39 SDH:TX<Pt>:VC12:POH:TRACe:CRC . . . 275
10.2.40 SDH:TX<Pt>:VC12:POH:TTCM . . . 275
10.2.41 SDH:TX<Pt>:VC12:POH . . . 276
10.2.42 SDH:TX<Pt>:VC11:POH:DEFault . . . 276
10.2.43 SDH:TX<Pt>:VC11:POH:TRACe . . . 276
10.2.44 SDH:TX<Pt>:VC11:POH:TRACe:CRC . . . 277
10.2.45 SDH:TX<Pt>:VC11:POH:TTCM . . . 277
10.2.46 SDH:TX<Pt>:VC11:POH . . . 278
10.2.47 SDH:TX<Pt>:AU3:VC3:POH:DEFault . . . 278
10.2.48 SDH:TX<Pt>:AU3:VC3:POH:TRACe . . . 278
10.2.49 SDH:TX<Pt>:AU3:VC3:POH:TRACe:CRC . . . 279
10.2.50 SDH:TX<Pt>:AU3:VC3:POH:TTCM . . . 279
10.2.51 SDH:TX<Pt>:AU3:VC3:POH . . . 279
10.2.52 SDH:TX<Pt>:FOLLow . . . 280
10.2.53 SDH:TX<Pt>:OTHRough . . . 280
10.2.54 SDH:TX<Pt>:OTHRough:BYTE . . . 281
10.2.55 SDH:TX<Pt>:STL:MMAPing:LANE . . . 281
10.3 Stimuli . . . 282
10.3.1 SDH:STIMuli:TX<Pt>:FOFFset . . . 282
10.3.2 SDH:STIMuli:TX<Pt>:ALARm . . . 283
10.3.3 SDH:STIMuli:TX<Pt>:AINSert . . . 283
10.3.4 SDH:STIMuli:TX<Pt>:ANLength . . . 284
10.3.5 SDH:STIMuli:TX<Pt>:AALength . . . 284
10.3.6 SDH:STIMuli:TX<Pt>:ERRor . . . 285
10.3.7 SDH:STIMuli:TX<Pt>:EINSert . . . 286
10.3.8 SDH:STIMuli:TX<Pt>:EBLength . . . 286
10.3.9 SDH:STIMuli:TX<Pt>:ENLength . . . 287
10.3.10 SDH:STIMuli:TX<Pt>:EELength . . . 287
10.3.11 SDH:STIMuli:TX<Pt>:PTSequence . . . 287
10.3.12 SDH:STIMuli:TX<Pt>:PMOVe . . . 288
10.3.13 SDH:STIMuli:TX<Pt>:PJUMp . . . 289
10.3.14 SDH:STIMuli:TX<Pt>:STL:AERRor:LANE . . . 289
10.3.15 SDH:STIMuli:TX<Pt>:STL:ALARm . . . 290
10.3.16 SDH:STIMuli:TX<Pt>:STL:AINSert . . . 290
10.3.17 SDH:STIMuli:TX<Pt>:STL:ABLength . . . 290
10.3.18 SDH:STIMuli:TX<Pt>:STL:AALength . . . 291
10.3.19 SDH:STIMuli:TX<Pt>:STL:ANLength . . . 291
10.3.20 SDH:STIMuli:TX<Pt>:STL:ERRor . . . 291
10.3.21 SDH:STIMuli:TX<Pt>:STL:EINSert . . . 292
10.3.22 SDH:STIMuli:TX<Pt>:STL:EELength . . . 292
10.3.23 SDH:STIMuli:TX<Pt>:STL:ENLength . . . 292
10.3.24 SDH:STIMuli:TX<Pt>:STL:SKEW:BIT . . . 293
10.3.25 SDH:STIMuli:TX<Pt>:STL:SKEW:NS? . . . 293
10.3.26 SDH:STIMuli:TX<Pt>:STL:SKEW:LANE . . . 293
10.4 Result . . . 294
10.4.1 SDH:RX<Pt>:IFETch? . . . 294
10.4.2 SDH:PMOVement:RX<Pt>:FETCh? . . . 298
10.5 Status . . . 298

10.5.1 SDH:STATus:RX<Pt>:AESummary[:EVENt]? . . . 298
10.5.2 SDH:STATus:RX<Pt>:AESummary:CONDition? . . . 298
10.5.3 SDH:STATus:RX<Pt>:ALARm<section>[:EVENt]? . . . 299
10.5.4 SDH:STATus:RX<Pt>:ALARm<section>:CONDition? . . . 300
10.5.5 SDH:STATus:RX<Pt>:ERRor<section>[:EVENt]? . . . 300
10.5.6 SDH:STATus:RX<Pt>:ERRor<section>:CONDition? . . . 301
10.5.7 SDH:STATus:RX<Pt>:PSLevel? . . . 301
10.5.8 SDH:STATus:TX<Pt>:PSLevel? . . . 301
10.5.9 SDH:STATus:RX<Pt>:PDEViation? . . . 301
10.5.10 SDH:STATus:RX<Pt>:PBRate? . . . 301
10.5.11 SDH:STATus:RX<Pt>:DIFFerence? . . . 302
10.5.12 SDH:STATus:RX<Pt>:RACCumulated . . . 302
10.5.13 SDH:STATus:RX<Pt>:PPBRate? . . . 302
10.5.14 SDH:STATus:RX<Pt>:CAPTure:SOH:TRACe? . . . 302
10.5.15 SDH:STATus:RX<Pt>:CAPTure<Frame>:SOH? . . . 303
10.5.16 SDH:STATus:RX<Pt>:CAPTure:VC4:POH:TRACe? . . . 303
10.5.17 SDH:STATus:RX<Pt>:CAPTure<Frame>:VC4:POH? . . . 304
10.5.18 SDH:STATus:RX<Pt>:CAPTure:VC3:POH:TRACe? . . . 304
10.5.19 SDH:STATus:RX<Pt>:CAPTure<Frame>:VC3:POH? . . . 304
10.5.20 SDH:STATus:RX<Pt>:CAPTure:VC12:POH:TRACe? . . . 305
10.5.21 SDH:STATus:RX<Pt>:CAPTure<Frame>:VC12:POH? . . . 305
10.5.22 SDH:STATus:RX<Pt>:CAPTure:VC11:POH:TRACe? . . . 305
10.5.23 SDH:STATus:RX<Pt>:CAPTure<Frame>:VC11:POH? . . . 305
10.5.24 SDH:STATus:RX<Pt>:CAPTure:AU3:VC3:POH:TRACe? . . . 306
10.5.25 SDH:STATus:RX<Pt>:CAPTure<Frame>:AU3:VC3:POH? . . . 306
10.6 APS . . . 307
10.6.1 SDH:APS:STARt . . . 307
10.6.2 SDH:APS:STOP . . . 307
10.6.3 SDH:APS:RX<Pt>:PINTerpret? . . . 307
10.6.4 SDH:APS:RX<Pt>:NUMBer? . . . 307
10.6.5 SDH:APS:RX<Pt>:ATIMe? . . . 307
10.6.6 SDH:APS:RX<Pt>:MTIMe? . . . 308
10.6.7 SDH:APS:RX<Pt>:LTIMe? . . . 308
10.6.8 SDH:APS:RX<Pt>:CTIMe? . . . 308
10.6.9 SDH:APS:RX<Pt>:EVENt . . . 309
10.6.10 SDH:APS:RX<Pt>:MLIMit . . . 309
10.6.11 SDH:APS:RX<Pt>:PERiod . . . 310
10.6.12 SDH:APS:RX<Pt>:CONFig . . . 310
10.6.13 SDH:APS:RX<Pt>:RTYPe . . . 311
10.6.14 SDH:APS:RX<Pt>:K1 . . . 312
10.6.15 SDH:APS:RX<Pt>:K2 . . . 312
10.6.16 SDH:APS:RX<Pt>:APPLy . . . 312
10.7 Tributary Scan . . . 312
10.7.1 SDH:TSCan:STARt . . . 312
10.7.2 SDH:TSCan:STOP . . . 313
10.7.3 SDH:TSCan:RX<Pt>:NHOContainer? . . . 313
10.7.4 SDH:TSCan:RX<Pt>:GHOContainer? . . . 313
10.7.5 SDH:TSCan:RX<Pt>:SHOContainer . . . 313
10.7.6 SDH:TSCan:RX<Pt>:SHOContainer? . . . 313
10.7.7 SDH:TSCan:RX<Pt>:DHOContainer? . . . 314
10.7.8 SDH:TSCan:RX<Pt>:NLOContainer? . . . 314
10.7.9 SDH:TSCan:RX<Pt>:GLOContainer? . . . 314
10.7.10 SDH:TSCan:RX<Pt>:DLOContainer? . . . 314
10.7.11 SDH:TSCan:RX<Pt>:DMUX? . . . 315
10.8 RTD . . . 316
10.8.1 SDH:RTD:RX<Pt>:MLIMit . . . 316
10.8.2 SDH:RTD:RX<Pt>:NUMBer? . . . 316
10.8.3 SDH:RTD:RX<Pt>:ATIMe? . . . 316
10.8.4 SDH:RTD:RX<Pt>:MTIMe? . . . 316

10.8.5 SDH:RTD:RX<Pt>:LTIMe? . . . 317
10.8.6 SDH:RTD:RX<Pt>:CTIMe? . . . 317

11 SONET **319**
11.1 Receiver . . . 319
11.1.1 SONet:RX<Pt>:INTerface . . . 319
11.1.2 SONet:RX<Pt>:OCLevel . . . 319
11.1.3 SONet:RX<Pt>:STSLevel . . . 320
11.1.4 SONet:RX<Pt>:STS3c . . . 320
11.1.5 SONet:RX<Pt>:CONCatenation . . . 321
11.1.6 SONet:RX<Pt>:CONTainer . . . 321
11.1.7 SONet:RX<Pt>:SSTRucture . . . 322
11.1.8 SONet:RX<Pt>:PATTern . . . 322
11.1.9 SONet:RX<Pt>:UP32 . . . 323
11.1.10 SONet:RX<Pt>:UP2K . . . 323
11.1.11 SONet:RX<Pt>:PINVersion . . . 323
11.1.12 SONet:RX<Pt>:TCM . . . 324
11.1.13 SONet:RX<Pt>:TUG3 . . . 324
11.1.14 SONet:RX<Pt>:VTG . . . 324
11.1.15 SONet:RX<Pt>:VT2 . . . 325
11.1.16 SONet:RX<Pt>:VT15 . . . 325
11.1.17 SONet:RX<Pt>:CHANnel . . . 326
11.1.18 SONet:RX<Pt>:MAPPing . . . 326
11.1.19 SONet:RX<Pt>:GAIN . . . 326
11.1.20 SONet:RX<Pt>:FOLLow . . . 327
11.1.21 SONet:RX<Pt>:MEASurement:PLMP . . . 327
11.1.22 SONet:RX<Pt>:MEASurement:PLMV . . . 327
11.2 Transmitter . . . 328
11.2.1 SONet:TX<Pt>:INTerface . . . 328
11.2.2 SONet:TX<Pt>[:ENABle] . . . 328
11.2.3 SONet:TX<Pt>:OPTical . . . 328
11.2.4 SONet:TX<Pt>:ELECtrical . . . 329
11.2.5 SONet:TX<Pt>:OCLevel . . . 329
11.2.6 SONet:TX<Pt>:STSLevel . . . 330
11.2.7 SONet:TX<Pt>:STS3c . . . 330
11.2.8 SONet:TX<Pt>:CONCatenation . . . 331
11.2.9 SONet:TX<Pt>:CONTainer . . . 331
11.2.10 SONet:TX<Pt>:SSTRucture . . . 332
11.2.11 SONet:TX<Pt>:PATTern . . . 332
11.2.12 SONet:TX<Pt>:UP32 . . . 333
11.2.13 SONet:TX<Pt>:UP2K . . . 333
11.2.14 SONet:TX<Pt>:PINVersion . . . 333
11.2.15 SONet:TX<Pt>:TIMing . . . 334
11.2.16 SONet:TX<Pt>:TCM . . . 334
11.2.17 SONet:TX<Pt>:TUG3 . . . 334
11.2.18 SONet:TX<Pt>:VTG . . . 335
11.2.19 SONet:TX<Pt>:VT2 . . . 335
11.2.20 SONet:TX<Pt>:VT15 . . . 336
11.2.21 SONet:TX<Pt>:CHANnel . . . 336
11.2.22 SONet:TX<Pt>:MAPPing . . . 336
11.2.23 SONet:TX<Pt>:TOH:DEFault . . . 337
11.2.24 SONet:TX<Pt>:TOH:TRACe . . . 337
11.2.25 SONet:TX<Pt>:TOH:TRACe:CRC . . . 337
11.2.26 SONet:TX<Pt>:TOH . . . 338
11.2.27 SONet:TX<Pt>:STS3:POH:DEFault . . . 339
11.2.28 SONet:TX<Pt>:STS3:POH:TRACe . . . 339
11.2.29 SONet:TX<Pt>:STS3:POH:TRACe:CRC . . . 339
11.2.30 SONet:TX<Pt>:STS3:POH:TTCM . . . 340
11.2.31 SONet:TX<Pt>:STS3:POH . . . 340

11.2.32 SONet:TX<Pt>:STS1:POH:DEFault . . . 341
11.2.33 SONet:TX<Pt>:STS1:POH:TRACe . . . 341
11.2.34 SONet:TX<Pt>:STS1:POH:TRACe:CRC . . . 341
11.2.35 SONet:TX<Pt>:STS1:POH:TTCM . . . 342
11.2.36 SONet:TX<Pt>:STS1:POH . . . 342
11.2.37 SONet:TX<Pt>:VT2:POH:DEFault . . . 342
11.2.38 SONet:TX<Pt>:VT2:POH:TRACe . . . 343
11.2.39 SONet:TX<Pt>:VT2:POH:TRACe:CRC . . . 343
11.2.40 SONet:TX<Pt>:VT2:POH:TTCM . . . 343
11.2.41 SONet:TX<Pt>:VT2:POH . . . 344
11.2.42 SONet:TX<Pt>:VT15:POH:DEFault . . . 344
11.2.43 SONet:TX<Pt>:VT15:POH:TRACe . . . 344
11.2.44 SONet:TX<Pt>:VT15:POH:TRACe:CRC . . . 345
11.2.45 SONet:TX<Pt>:VT15:POH:TTCM . . . 345
11.2.46 SONet:TX<Pt>:VT15:POH . . . 346
11.2.47 SONet:TX<Pt>:STS1:STS1:POH:DEFault . . . 346
11.2.48 SONet:TX<Pt>:STS1:STS1:POH:TRACe . . . 346
11.2.49 SONet:TX<Pt>:STS1:STS1:POH:TRACe:CRC . . . 347
11.2.50 SONet:TX<Pt>:STS1:STS1:POH:TTCM . . . 347
11.2.51 SONet:TX<Pt>:STS1:STS1:POH . . . 347
11.2.52 SONet:TX<Pt>:FOLLow . . . 348
11.2.53 SONet:TX<Pt>:OTHRough . . . 348
11.2.54 SONet:TX<Pt>:OTHRough:BYTE . . . 349
11.2.55 SONet:TX<Pt>:STL:MMAPing:LANE . . . 349
11.3 Stimuli . . . 350
11.3.1 SONet:STIMuli:TX<Pt>:FOFFset . . . 350
11.3.2 SONet:STIMuli:TX<Pt>:ALARm . . . 351
11.3.3 SONet:STIMuli:TX<Pt>:AINSert . . . 351
11.3.4 SONet:STIMuli:TX<Pt>:ANLength . . . 352
11.3.5 SONet:STIMuli:TX<Pt>:AALength . . . 352
11.3.6 SONet:STIMuli:TX<Pt>:ERRor . . . 353
11.3.7 SONet:STIMuli:TX<Pt>:EINSert . . . 354
11.3.8 SONet:STIMuli:TX<Pt>:EBLength . . . 354
11.3.9 SONet:STIMuli:TX<Pt>:ENLength . . . 355
11.3.10 SONet:STIMuli:TX<Pt>:EELength . . . 355
11.3.11 SONet:STIMuli:TX<Pt>:PTSequence . . . 355
11.3.12 SONet:STIMuli:TX<Pt>:PMOVe . . . 356
11.3.13 SONet:STIMuli:TX<Pt>:PJUMp . . . 357
11.3.14 SONet:STIMuli:TX<Pt>:STL:AERRor:LANE . . . 357
11.3.15 SONet:STIMuli:TX<Pt>:STL:ALARm . . . 358
11.3.16 SONet:STIMuli:TX<Pt>:STL:AINSert . . . 358
11.3.17 SONet:STIMuli:TX<Pt>:STL:ABLength . . . 358
11.3.18 SONet:STIMuli:TX<Pt>:STL:AALength . . . 359
11.3.19 SONet:STIMuli:TX<Pt>:STL:ANLength . . . 359
11.3.20 SONet:STIMuli:TX<Pt>:STL:ERRor . . . 359
11.3.21 SONet:STIMuli:TX<Pt>:STL:EINSert . . . 360
11.3.22 SONet:STIMuli:TX<Pt>:STL:EELength . . . 360
11.3.23 SONet:STIMuli:TX<Pt>:STL:ENLength . . . 360
11.3.24 SONet:STIMuli:TX<Pt>:STL:SKEW:BIT . . . 361
11.3.25 SONet:STIMuli:TX<Pt>:STL:SKEW:NS? . . . 361
11.3.26 SONet:STIMuli:TX<Pt>:STL:SKEW:LANE . . . 361
11.3.27 SONet:STIMuli:TX<Pt>:STL:SKEW:LANE? . . . 361
11.4 Result . . . 362
11.4.1 SONet:RX<Pt>:IFETch? . . . 362
11.4.2 SONet:PMOVement:RX<Pt>:FETCh? . . . 366
11.5 Status . . . 366
11.5.1 SONet:STATus:RX<Pt>:AESummary[:EVENt]? . . . 366
11.5.2 SONet:STATus:RX<Pt>:AESummary:CONDition? . . . 366
11.5.3 SONet:STATus:RX<Pt>:ALARm<section>[:EVENt]? . . . 367

11.5.4 SONet:STATus:RX<Pt>:ALARm<section>:CONDition? . . . 368
11.5.5 SONet:STATus:RX<Pt>:ERRor<section>[:EVENt]? . . . 368
11.5.6 SONet:STATus:RX<Pt>:ERRor<section>:CONDition? . . . 369
11.5.7 SONet:STATus:RX<Pt>:PSLevel? . . . 369
11.5.8 SONet:STATus:TX<Pt>:PSLevel? . . . 369
11.5.9 SONet:STATus:RX<Pt>:PDEViation? . . . 369
11.5.10 SONet:STATus:RX<Pt>:PBRate? . . . 369
11.5.11 SONet:STATus:RX<Pt>:DIFFerence? . . . 370
11.5.12 SONet:STATus:RX<Pt>:RACCumulated . . . 370
11.5.13 SONet:STATus:RX<Pt>:PPBRate? . . . 370
11.5.14 SONet:STATus:RX<Pt>:CAPTure:TOH:TRACe? . . . 370
11.5.15 SONet:STATus:RX<Pt>:CAPTure<Frame>:TOH? . . . 371
11.5.16 SONet:STATus:RX<Pt>:CAPTure:STS3:POH:TRACe? . . . 371
11.5.17 SONet:STATus:RX<Pt>:CAPTure<Frame>:STS3:POH? . . . 372
11.5.18 SONet:STATus:RX<Pt>:CAPTure:STS1:POH:TRACe? . . . 372
11.5.19 SONet:STATus:RX<Pt>:CAPTure<Frame>:STS1:POH? . . . 372
11.5.20 SONet:STATus:RX<Pt>:CAPTure:VT2:POH:TRACe? . . . 373
11.5.21 SONet:STATus:RX<Pt>:CAPTure<Frame>:VT2:POH? . . . 373
11.5.22 SONet:STATus:RX<Pt>:CAPTure:VT15:POH:TRACe? . . . 373
11.5.23 SONet:STATus:RX<Pt>:CAPTure<Frame>:VT15:POH? . . . 373
11.5.24 SONet:STATus:RX<Pt>:CAPTure:STS1:STS1:POH:TRACe? . . . 374
11.5.25 SONet:STATus:RX<Pt>:CAPTure<Frame>:STS1:STS1:POH? . . . 374
11.6 APS . . . 375
11.6.1 SONet:APS:STARt . . . 375
11.6.2 SONet:APS:STOP . . . 375
11.6.3 SONet:APS:RX<Pt>:PINTerpret? . . . 375
11.6.4 SONet:APS:RX<Pt>:NUMBer? . . . 375
11.6.5 SONet:APS:RX<Pt>:ATIMe? . . . 375
11.6.6 SONet:APS:RX<Pt>:MTIMe? . . . 376
11.6.7 SONet:APS:RX<Pt>:LTIMe? . . . 376
11.6.8 SONet:APS:RX<Pt>:CTIMe? . . . 376
11.6.9 SONet:APS:RX<Pt>:EVENt . . . 377
11.6.10 SONet:APS:RX<Pt>:MLIMit . . . 377
11.6.11 SONet:APS:RX<Pt>:PERiod . . . 378
11.6.12 SONet:APS:RX<Pt>:CONFig . . . 378
11.6.13 SONet:APS:RX<Pt>:RTYPe . . . 379
11.6.14 SONet:APS:RX<Pt>:K1 . . . 380
11.6.15 SONet:APS:RX<Pt>:K2 . . . 380
11.6.16 SONet:APS:RX<Pt>:APPLy . . . 380
11.7 Tributary Scan . . . 380
11.7.1 SONet:TSCan:STARt . . . 380
11.7.2 SONet:TSCan:STOP . . . 381
11.7.3 SONet:TSCan:RX<Pt>:NHOContainer? . . . 381
11.7.4 SONet:TSCan:RX<Pt>:GHOContainer? . . . 381
11.7.5 SONet:TSCan:RX<Pt>:SHOContainer . . . 381
11.7.6 SONet:TSCan:RX<Pt>:SHOContainer? . . . 381
11.7.7 SONet:TSCan:RX<Pt>:DHOContainer? . . . 382
11.7.8 SONet:TSCan:RX<Pt>:NLOContainer? . . . 382
11.7.9 SONet:TSCan:RX<Pt>:GLOContainer? . . . 382
11.7.10 SONet:TSCan:RX<Pt>:DLOContainer? . . . 382
11.7.11 SONet:TSCan:RX<Pt>:DMUX? . . . 383
11.8 RTD . . . 384
11.8.1 SONet:RTD:RX<Pt>:MLIMit . . . 384
11.8.2 SONet:RTD:RX<Pt>:NUMBer? . . . 384
11.8.3 SONet:RTD:RX<Pt>:ATIMe? . . . 384
11.8.4 SONet:RTD:RX<Pt>:MTIMe? . . . 384
11.8.5 SONet:RTD:RX<Pt>:LTIMe? . . . 385
11.8.6 SONet:RTD:RX<Pt>:CTIMe? . . . 385

12 Ethernet **387**
12.1 Port Setup . . . 387
12.1.1 ETHernet:PORT<Pt>:ITYPe . . . 387
12.1.2 ETHernet:PORT<Pt>:MODE . . . 388
12.1.3 ETHernet:PORT<Pt>:ANEGotiate . . . 388
12.1.4 ETHernet:PORT<Pt>:FMODe . . . 389
12.1.5 ETHernet:PORT<Pt>:CMODe . . . 389
12.1.6 ETHernet:PORT<Pt>:MDICrossover . . . 390
12.1.7 ETHernet:PORT<Pt>:TIMing:SOURce . . . 390
12.1.8 ETHernet:PORT<Pt>:PFRames . . . 390
12.1.9 ETHernet:PORT<Pt>:FEC . . . 391
12.2 WAN . . . 392
12.2.1 ETHernet:PORT<Pt>:WAN[:ENABle] . . . 392
12.2.2 ETHernet:PORT<Pt>:WAN:TERMinology . . . 392
12.2.3 ETHernet:PORT<Pt>:WAN:SOH:DEFault . . . 392
12.2.4 ETHernet:PORT<Pt>:WAN:SOH:TRACe . . . 393
12.2.5 ETHernet:PORT<Pt>:WAN:SOH:TRACe:CRC . . . 393
12.2.6 ETHernet:PORT<Pt>:WAN:SOH . . . 393
12.2.7 ETHernet:PORT<Pt>:WAN:POH:DEFault . . . 395
12.2.8 ETHernet:PORT<Pt>:WAN:POH:TRACe . . . 395
12.2.9 ETHernet:PORT<Pt>:WAN:POH:TRACe:CRC . . . 395
12.2.10 ETHernet:PORT<Pt>:WAN:POH . . . 396
12.3 Reflector . . . 397
12.3.1 ETHernet:PORT<Pt>:REFLector:SWAP[:ENABle] . . . 397
12.3.2 ETHernet:PORT<Pt>:REFLector:SSMac . . . 397
12.3.3 ETHernet:PORT<Pt>:REFLector:SMAC . . . 397
12.3.4 ETHernet:PORT<Pt>:REFLector:IPSWap . . . 398
12.3.5 ETHernet:PORT<Pt>:REFLector:PSWap . . . 398
12.3.6 ETHernet:PORT<Pt>:REFLector:ATCP . . . 398
12.4 PCS . . . 399
12.4.1 ETHernet:PORT<Pt>:PCS:MMAPing:LANE . . . 399
12.5 Traffic . . . 399
12.5.1 ETHernet:TRAFfic:GENerator:STARt . . . 399
12.5.2 ETHernet:PORT<Pt>:TRAFfic:GENerator:STARt . . . 399
12.5.3 ETHernet:TRAFfic:GENerator:STOP . . . 400
12.5.4 ETHernet:PORT<Pt>:TRAFfic:GENerator:STOP . . . 400
12.5.5 ETHernet:PORT<Pt>:TRAFfic:GENerator:STATus? . . . 400
12.5.6 ETHernet:PORT<Pt>:TRAFfic:DMODe . . . 400
12.5.7 ETHernet:PORT<Pt>:TRAFfic:DURation . . . 401
12.5.8 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:TXMode . . . 401
12.5.9 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:PROFile . . . 401
12.5.10 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:PROFile:MRAMp [:MODe] . . . 402
12.5.11 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad[:CONStant] . . . 402
12.5.12 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP[:MODE] . . . 403
12.5.13 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:STARt . . . 403
12.5.14 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:END . . . 404
12.5.15 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:STEP . . . 404
12.5.16 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:DURation . . . 405
12.5.17 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:PROFile:MRAMp :DURation . . . 405
12.5.18 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:PROFile . . . 406
12.5.19 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe[:STARt] . . . 406
12.5.20 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:END . . . 406
12.5.21 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:STEP . . . 407
12.5.22 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:DURation . . . 407
12.5.23 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:PROFile . . . 408
12.5.24 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:ENCoding:VIDeo . . . 408
12.5.25 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:ENCoding:VOICe . . . 409

12.5.26 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:NCHannels . . . 409
12.5.27 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:MODe . . . 409
12.5.28 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:NBURst:MODe . . . 410
12.5.29 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:NBURst:BNUMber . . . 410
12.5.30 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:PDUTy [:DUTY] . . . 411
12.5.31 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:PDUTy :PERiod . . . 411
12.5.32 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:DUTY? . . . 411
12.5.33 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:PERiod? . . . 411
12.5.34 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:BLENgth . . . 412
12.5.35 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:BGAP . . . 412
12.5.36 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:RMODe . . . 413
12.5.37 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:BLENgth :STARt . . . 413
12.5.38 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:BLENgth :END . . . 414
12.5.39 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:BGAP . . . 414
12.6 Frame Content . . . 415
12.6.1 ETHernet:PORT<Pt>:STReam<St>:FRAMed . . . 415
12.6.2 ETHernet:PORT<Pt>:STReam<St>:MAC:SOURce . . . 415
12.6.3 ETHernet:PORT<Pt>:STReam<St>:MAC:DESTination . . . 415
12.6.4 ETHernet:PORT<Pt>:STReam<St>:MAC:ARP . . . 416
12.6.5 ETHernet:PORT<Pt>:STReam<St>:MAC:ARP:LKUP . . . 416
12.6.6 ETHernet:PORT<Pt>:STReam<St>:MAC:ARP:RESult? . . . 416
12.6.7 ETHernet:PORT<Pt>:STReam<St>:MAC:NDP . . . 417
12.6.8 ETHernet:PORT<Pt>:STReam<St>:MAC:NDP:LKUP . . . 417
12.6.9 ETHernet:PORT<Pt>:STReam<St>:MAC:NDP:RESult? . . . 417
12.6.10 ETHernet:PORT<Pt>:STReam<St>:MAC:BROadcast . . . 417
12.6.11 ETHernet:PORT<Pt>:STReam<St>:MAC:DEFault . . . 418
12.6.12 ETHernet:PORT<Pt>:STReam<St>:MAC:ETYPe . . . 418
12.6.13 ETHernet:PORT<Pt>:STReam<St>:MAC:L3EType . . . 419
12.6.14 ETHernet:PORT<Pt>:STReam<St>:MPLS[:ENABle] . . . 419
12.6.15 ETHernet:PORT<Pt>:STReam<St>:MPLS:LCount . . . 419
12.6.16 ETHernet:PORT<Pt>:STReam<St>:MPLS:LEVel<Lv>:LABel . . . 420
12.6.17 ETHernet:PORT<Pt>:STReam<St>:MPLS:LEVel<Lv>:EBITs . . . 420
12.6.18 ETHernet:PORT<Pt>:STReam<St>:MPLS:LEVel<Lv>:TTL . . . 421
12.6.19 ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet[:ENABle] . . . 421
12.6.20 ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:AINCrement . . . 421
12.6.21 ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:SMAC . . . 422
12.6.22 ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:DMAC . . . 422
12.6.23 ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:ETYPe . . . 423
12.6.24 ETHernet:PORT<Pt>:STReam<St>:MIM[:ENABle] . . . 423
12.6.25 ETHernet:PORT<Pt>:STReam<St>:MIM:BTAG:DEI . . . 423
12.6.26 ETHernet:PORT<Pt>:STReam<St>:MIM:BTAG:PRiority . . . 424
12.6.27 ETHernet:PORT<Pt>:STReam<St>:MIM:BTAG:VID . . . 424
12.6.28 ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:UCA . . . 425
12.6.29 ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:DEI . . . 425
12.6.30 ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:PRiority . . . 425
12.6.31 ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:SID . . . 426
12.6.32 ETHernet:PORT<Pt>:STReam<St>:MIM:ETHernet:SMAC . . . 426
12.6.33 ETHernet:PORT<Pt>:STReam<St>:MIM:ETHernet:DMAC . . . 426
12.6.34 ETHernet:PORT<Pt>:STReam<St>:MIM:ETHernet:ETYPe . . . 427
12.6.35 ETHernet:PORT<Pt>:STReam<St>:VLAN[:ENABle] . . . 427
12.6.36 ETHernet:PORT<Pt>:STReam<St>:VLAN:LCount . . . 428
12.6.37 ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:ID . . . 428
12.6.38 ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:CFI . . . 428
12.6.39 ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:DEI . . . 429
12.6.40 ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:PRiority . . . 429

12.6.41 ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:ETYPe . . . 430
12.6.42 ETHernet:PORT<Pt>:STReam<St>:LLC[:ENABle] . . . 430
12.6.43 ETHernet:PORT<Pt>:STReam<St>:SNAP[:ENABle] . . . 431
12.6.44 ETHernet:PORT<Pt>:STReam<St>:L2Custom[:ENABle] . . . 431
12.6.45 ETHernet:PORT<Pt>:STReam<St>:IPV4[:ENABle] . . . 432
12.6.46 ETHernet:PORT<Pt>:STReam<St>:IPV4:SOURce . . . 432
12.6.47 ETHernet:PORT<Pt>:STReam<St>:IPV4:DESTination . . . 432
12.6.48 ETHernet:PORT<Pt>:STReam<St>:IPV4:GATeway[:ENABle] . . . 433
12.6.49 ETHernet:PORT<Pt>:STReam<St>:IPV4:GATeway:ADDRess . . . 433
12.6.50 ETHernet:PORT<Pt>:STReam<St>:IPV4:NETMask . . . 434
12.6.51 ETHernet:PORT<Pt>:STReam<St>:IPV4:HOST . . . 434
12.6.52 ETHernet:PORT<Pt>:STReam<St>:IPV4:TTL . . . 434
12.6.53 ETHernet:PORT<Pt>:STReam<St>:IPV4:TOS . . . 435
12.6.54 ETHernet:PORT<Pt>:STReam<St>:IPV4:IDENtifier . . . 435
12.6.55 ETHernet:PORT<Pt>:STReam<St>:IPV4:AINCrement . . . 436
12.6.56 ETHernet:PORT<Pt>:STReam<St>:IPV4:MF . . . 436
12.6.57 ETHernet:PORT<Pt>:STReam<St>:IPV4:DF . . . 436
12.6.58 ETHernet:PORT<Pt>:STReam<St>:IPV4:RES . . . 437
12.6.59 ETHernet:PORT<Pt>:STReam<St>:IPV4:PROTocol . . . 437
12.6.60 ETHernet:PORT<Pt>:STReam<St>:IPV4:TLENgth? . . . 438
12.6.61 ETHernet:PORT<Pt>:STReam<St>:IPV4:CHECksum? . . . 438
12.6.62 ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP[:ENABle] . . . 438
12.6.63 ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:VERSion . . . 438
12.6.64 ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:GROup[:ADDRess] . . . 439
12.6.65 ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:SOURce[:ENABle] . . . 439
12.6.66 ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:SOURce:ADDRess . . . 440
12.6.67 ETHernet:PORT<Pt>:STReam<St>:DNS[:ENABle] . . . 440
12.6.68 ETHernet:PORT<Pt>:STReam<St>:DNS:PRIMary . . . 440
12.6.69 ETHernet:PORT<Pt>:STReam<St>:DNS:SECondary . . . 441
12.6.70 ETHernet:PORT<Pt>:STReam<St>:IPV6[:ENABle] . . . 441
12.6.71 ETHernet:PORT<Pt>:STReam<St>:IPV6:TCLass . . . 442
12.6.72 ETHernet:PORT<Pt>:STReam<St>:IPV6:FLABel . . . 442
12.6.73 ETHernet:PORT<Pt>:STReam<St>:IPV6:NHeader . . . 442
12.6.74 ETHernet:PORT<Pt>:STReam<St>:IPV6:HLIMit . . . 443
12.6.75 ETHernet:PORT<Pt>:STReam<St>:IPV6:SOURce . . . 443
12.6.76 ETHernet:PORT<Pt>:STReam<St>:IPV6:DESTination . . . 444
12.6.77 ETHernet:PORT<Pt>:STReam<St>:IPV6:GATeway[:ENABle] . . . 444
12.6.78 ETHernet:PORT<Pt>:STReam<St>:IPV6:GATeway:ADDRess . . . 444
12.6.79 ETHernet:PORT<Pt>:STReam<St>:IPV6:PREFix . . . 445
12.6.80 ETHernet:PORT<Pt>:STReam<St>:IPV6:ADDRconfig . . . 445
12.6.81 ETHernet:PORT<Pt>:STReam<St>:IPV6:ADDRconfig:RESult? . . . 446
12.6.82 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:IID . . . 446
12.6.83 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:IID:AUTO . . . 446
12.6.84 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:LINKlocal? . . . 447
12.6.85 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:RAFLag? . . . 447
12.6.86 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:SRCMac? . . . 447
12.6.87 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:PREFix? . . . 447
12.6.88 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:PRFLag? . . . 447
12.6.89 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:LTIMe? . . . 448
12.6.90 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:RENew . . . 448
12.6.91 ETHernet:PORT<Pt>:STReam<St>:IPV6:PLENgth? . . . 448
12.6.92 ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD[:ENABle] . . . 448
12.6.93 ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:VERSion . . . 449
12.6.94 ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:GROup[:ADDRess] . . . 449
12.6.95 ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:SOURce[:ENABle] . . . 449
12.6.96 ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:SOURce:ADDRess . . . 450
12.6.97 ETHernet:PORT<Pt>:STReam<St>:L3Custom[:ENABle] . . . 450
12.6.98 ETHernet:PORT<Pt>:STReam<St>:DHCP[:ENABle] . . . 451
12.6.99 ETHernet:PORT<Pt>:STReam<St>:DHCP:RENew . . . 451

12.6.100 ETHernet:PORT<Pt>:STReam<St>:DHCP:DNS . . . 451
12.6.101 ETHernet:PORT<Pt>:STReam<St>:DHCP:GATeway . . . 452
12.6.102 ETHernet:PORT<Pt>:STReam<St>:DHCP:LETime? . . . 452
12.6.103 ETHernet:PORT<Pt>:STReam<St>:UDP[:ENABle] . . . 452
12.6.104 ETHernet:PORT<Pt>:STReam<St>:UDP:SPORt . . . 453
12.6.105 ETHernet:PORT<Pt>:STReam<St>:UDP:DPORt . . . 453
12.6.106 ETHernet:PORT<Pt>:STReam<St>:UDP:LENGth? . . . 453
12.6.107 ETHernet:PORT<Pt>:STReam<St>:UDP:CNULl . . . 454
12.6.108 ETHernet:PORT<Pt>:STReam<St>:UDP:CHECksum? . . . 454
12.6.109 ETHernet:PORT<Pt>:STReam<St>:TCP[:ENABle] . . . 454
12.6.110 ETHernet:PORT<Pt>:STReam<St>:TCP:ACONnect . . . 455
12.6.111 ETHernet:PORT<Pt>:STReam<St>:TCP:LMODe . . . 455
12.6.112 ETHernet:PORT<Pt>:STReam<St>:TCP:SPORt . . . 455
12.6.113 ETHernet:PORT<Pt>:STReam<St>:TCP:DPORt . . . 456
12.6.114 ETHernet:PORT<Pt>:STReam<St>:TCP:SEQuence . . . 456
12.6.115 ETHernet:PORT<Pt>:STReam<St>:TCP:AINCrement . . . 457
12.6.116 ETHernet:PORT<Pt>:STReam<St>:TCP:ACKNowledge . . . 457
12.6.117 ETHernet:PORT<Pt>:STReam<St>:TCP:REServed . . . 457
12.6.118 ETHernet:PORT<Pt>:STReam<St>:TCP:FCWR . . . 458
12.6.119 ETHernet:PORT<Pt>:STReam<St>:TCP:FECE . . . 458
12.6.120 ETHernet:PORT<Pt>:STReam<St>:TCP:FURG . . . 459
12.6.121 ETHernet:PORT<Pt>:STReam<St>:TCP:FACK . . . 459
12.6.122 ETHernet:PORT<Pt>:STReam<St>:TCP:FPSH . . . 459
12.6.123 ETHernet:PORT<Pt>:STReam<St>:TCP:FRST . . . 460
12.6.124 ETHernet:PORT<Pt>:STReam<St>:TCP:FSYN . . . 460
12.6.125 ETHernet:PORT<Pt>:STReam<St>:TCP:FFIN . . . 461
12.6.126 ETHernet:PORT<Pt>:STReam<St>:TCP:WINDow . . . 461
12.6.127 ETHernet:PORT<Pt>:STReam<St>:TCP:UPOinter . . . 461
12.6.128 ETHernet:PORT<Pt>:STReam<St>:TCP:CHECksum? . . . 462
12.6.129 ETHernet:PORT<Pt>:STReam1:BER:UTGenerator . . . 462
12.6.130 ETHernet:PORT<Pt>:STReam1:BER:SEQuence . . . 462
12.6.131 ETHernet:PORT<Pt>:STReam1:BER:SDMeasure[:ENABled] . . . 463
12.6.132 ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:DTYPe . . . 463
12.6.133 ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:MLIMit . . . 463
12.6.134 ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:MDISruption . . . 464
12.6.135 ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:EFPeriod . . . 464
12.6.136 ETHernet:PORT<Pt>:STReam1:BER:STHResholds[:ENABle] . . . 464
12.6.137 ETHernet:PORT<Pt>:STReam1:BER:STHResholds:VALue . . . 465
12.6.138 ETHernet:PORT<Pt>:STReam<St>:PAYLoad . . . 465
12.6.139 ETHernet:PORT<Pt>:STReam<St>:UP16 . . . 466
12.6.140 ETHernet:PORT<Pt>:STReam<St>:UP32 . . . 466
12.6.141 ETHernet:PORT<Pt>:STReam:PCMA . . . 466
12.6.142 ETHernet:PORT<Pt>:STReam:CROSsprbs . . . 467
12.6.143 ETHernet:PORT<Pt>:STReam<St>:LATency . . . 467
12.6.144 ETHernet:PORT<Pt>:STReam<St>:JITTer . . . 467
12.6.145 ETHernet:PORT<Pt>:STReam<St>:FLOSs . . . 468
12.6.146 ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds[:ENABle] . . . 468
12.6.147 ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds:MODE . . . 468
12.6.148 ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds:COUNt . . . 469
12.6.149 ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds:RATio . . . 469
12.6.150 ETHernet:PORT<Pt>:STReam<St>:LATency:THResholds[:ENABle] . . . 470
12.6.151 ETHernet:PORT<Pt>:STReam<St>:LATency:THResholds:VALue . . . 470
12.6.152 ETHernet:PORT<Pt>:STReam<St>:JITTer:THResholds[:ENABle] . . . 470
12.6.153 ETHernet:PORT<Pt>:STReam<St>:JITTer:THResholds:VALue . . . 471
12.6.154 ETHernet:PORT<Pt>:STReam:TSTamp:CCTimestamp . . . 471
12.6.155 ETHernet:PORT<Pt>:STReam:TSTamp:TSRC . . . 472
12.6.156 ETHernet:PORT<Pt>:STReam:TSTamp:GCABle . . . 472
12.6.157 ETHernet:PORT<Pt>:STReam:TSTamp:SYNC? . . . 472
12.6.158 ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:FIELd . . . 473

12.6.159 ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:TYPE . . . 474
12.6.160 ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:OFFSet . . . 474
12.6.161 ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:LENGth . . . 475
12.6.162 ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:STARt . . . 475
12.6.163 ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:END . . . 475
12.6.164 ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:STEP . . . 476
12.6.165 ETHernet:PORT<Pt>:STReam<St>:CUSTom:LENGth . . . 476
12.6.166 ETHernet:PORT<Pt>:STReam<St>:CUSTom:DATa . . . 477
12.7 Settings . . . 478
12.7.1 ETHernet:PORT<Pt>:SETTings:AARP . . . 478
12.7.2 ETHernet:PORT<Pt>:SETTings:ANDP . . . 478
12.7.3 ETHernet:PORT<Pt>:SETTings:APINg . . . 478
12.7.4 ETHernet:PORT<Pt>:SETTings:RLFSignaling . . . 479
12.7.5 ETHernet:PORT<Pt>:SETTings:ACMA . . . 479
12.7.6 ETHernet:PORT<Pt>:SETTings:PLENenght . . . 479
12.7.7 ETHernet:PORT<Pt>:SETTings:IPViolations . . . 480
12.7.8 ETHernet:PORT<Pt>:SETTings:ILThreshold . . . 480
12.7.9 ETHernet:PORT<Pt>:SETTings:FIViolations . . . 480
12.7.10 ETHernet:PORT<Pt>:SETTings:JFSize . . . 481
12.7.11 ETHernet:PORT<Pt>:SETTings:ERFault . . . 481
12.7.12 ETHernet:PORT<Pt>:SETTings:FRESults . . . 481
12.7.13 ETHernet:PORT<Pt>:SETTings:ACISetup . . . 482
12.7.14 ETHernet:PORT<Pt>:SETTings:ASTGenerator . . . 482
12.7.15 ETHernet:PORT<Pt>:SETTings:BER:OBAMeasuring . . . 482
12.7.16 ETHernet:PORT<Pt>:SETTings:BER:IAFFilter . . . 483
12.7.17 ETHernet:PORT<Pt>:SETTings:BER:CLFrames . . . 483
12.7.18 ETHernet:PORT<Pt>:SETTings:BER:HIDeframeloss . . . 483
12.7.19 ETHernet:PORT<Pt>:SETTings:CINBand[:ENABle] . . . 484
12.7.20 ETHernet:PORT<Pt>:SETTings:CINBand:PASSword[:ENABle] . . . 484
12.7.21 ETHernet:PORT<Pt>:SETTings:CINBand:PASSword:VALue . . . 484
12.7.22 ETHernet:PORT<Pt>:SETTings:CINBand:MAC:SOURce . . . 485
12.7.23 ETHernet:PORT<Pt>:SETTings:CINBand:MAC:DEFault . . . 485
12.7.24 ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:SOURce . . . 485
12.7.25 ETHernet:PORT<Pt>:SETTings:CINBand:DHCP[:ENABle] . . . 486
12.7.26 ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:NETMask . . . 486
12.7.27 ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:GATeway[:ENABle] . . . 486
12.7.28 ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:GATeway:ADDRess . . . 487
12.7.29 ETHernet:PORT<Pt>:SETTings:CINBand:VLAN:LCount . . . 487
12.7.30 ETHernet:PORT<Pt>:SETTings:CINBand:VLAN:LEVel<Lv>:ID . . . 487
12.8 Filter . . . 488
12.8.1 ETHernet:PORT<Pt>:FILTer[:ENABle] . . . 488
12.8.2 ETHernet:PORT<Pt>:FILTer:RESet . . . 488
12.8.3 ETHernet:PORT<Pt>:FILTer:ENCapsulation . . . 488
12.8.4 ETHernet:PORT<Pt>:FILTer:PROTocols . . . 489
12.8.5 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>[:ENABle] . . . 490
12.8.6 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SMAC . . . 490
12.8.7 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:DMAC . . . 490
12.8.8 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:EMSMac . . . 491
12.8.9 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:EMDMac . . . 491
12.8.10 ETHernet:PORT<Pt>:FILTer:POFFset . . . 492
12.8.11 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:PATTern . . . 492
12.8.12 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SPORt . . . 492
12.8.13 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:DPORt . . . 493
12.8.14 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SIV4 . . . 493
12.8.15 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:DIV4 . . . 494
12.8.16 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SIV6 . . . 494
12.8.17 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:HIV6 . . . 495
12.8.18 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:FIV6 . . . 495
12.8.19 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:LABel . . . 496

12.8.20 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:EBITs . . . 496
12.8.21 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:BOLStack . . . 497
12.8.22 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:TTL . . . 497
12.8.23 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:ID . . . 498
12.8.24 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:CFI . . . 498
12.8.25 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:DEI . . . 499
12.8.26 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:PRiority . . . 499
12.8.27 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMSMac . . . 500
12.8.28 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMDMac . . . 500
12.8.29 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMBTAG:ID . . . 500
12.8.30 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMBTAG:PRiority . . . 501
12.8.31 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMBTAG:DEI . . . 501
12.8.32 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:ID . . . 502
12.8.33 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:PRiority . . . 502
12.8.34 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:DEI . . . 503
12.8.35 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:UCA . . . 503
12.9 Thresholds . . . 504
12.9.1 ETHernet:PORT<Pt>:THResholds[:ENABle] . . . 504
12.9.2 ETHernet:PORT<Pt>:THResholds:SELect . . . 504
12.9.3 ETHernet:PORT<Pt>:THResholds:VALue . . . 505
12.10 SyncE . . . 507
12.10.1 ETHernet:SYNCe:PTHRough:FCAPture . . . 507
12.10.2 ETHernet:PORT<Pt>:SYNCe:PTHRough:QLEVel[:ENABle] . . . 507
12.10.3 ETHernet:PORT<Pt>:SYNCe:PTHRough:QLEVel:USER . . . 507
12.10.4 ETHernet:PORT<Pt>:SYNCe[:ENABle] . . . 508
12.10.5 ETHernet:PORT<Pt>:SYNCe:MODE . . . 508
12.10.6 ETHernet:PORT<Pt>:SYNCe:FCAPture . . . 508
12.10.7 ETHernet:PORT<Pt>:SYNCe:QLEVel . . . 509
12.10.8 ETHernet:PORT<Pt>:SYNCe:MAC . . . 509
12.10.9 ETHernet:PORT<Pt>:SYNCe:MAC:AUTO . . . 509
12.10.10 ETHernet:PORT<Pt>:SYNCe:EFLag . . . 510
12.11 Precision Time Protocol - IEEE 1588v2 . . . 511
12.11.1 ETHernet:PORT<Pt>:PTP[:ENABle] . . . 511
12.11.2 ETHernet:PORT<Pt>:PTP:SONLy . . . 511
12.11.3 ETHernet:PORT<Pt>:PTP:CAST . . . 511
12.11.4 ETHernet:PORT<Pt>:PTP:UNEGotiate . . . 512
12.11.5 ETHernet:PORT<Pt>:PTP:PROFile . . . 512
12.11.6 ETHernet:PORT<Pt>:PTP:DOMain . . . 512
12.11.7 ETHernet:PORT<Pt>:PTP:TPRotocol . . . 513
12.11.8 ETHernet:PORT<Pt>:PTP:MAC[:SOURce] . . . 513
12.11.9 ETHernet:PORT<Pt>:PTP:MAC:AUTO . . . 514
12.11.10 ETHernet:PORT<Pt>:PTP:MAC:DESTination . . . 514
12.11.11 ETHernet:PORT<Pt>:PTP:MAC:SMULticast . . . 514
12.11.12 ETHernet:PORT<Pt>:PTP:MAC:ARP . . . 515
12.11.13 ETHernet:PORT<Pt>:PTP:MAC:ARP:RESult? . . . 515
12.11.14 ETHernet:PORT<Pt>:PTP:MAC:NDP . . . 515
12.11.15 ETHernet:PORT<Pt>:PTP:MAC:NDP:RESult? . . . 515
12.11.16 ETHernet:PORT<Pt>:PTP:PTYPe . . . 516
12.11.17 ETHernet:PORT<Pt>:PTP:EXT . . . 516
12.11.18 ETHernet:PORT<Pt>:PTP:EXT:TYPE . . . 516
12.11.19 ETHernet:PORT<Pt>:PTP:EXT:LENGth . . . 517
12.11.20 ETHernet:PORT<Pt>:PTP:EXT:DATA . . . 517
12.11.21 ETHernet:PORT<Pt>:PTP:IP:IPV4:SOURce . . . 518
12.11.22 ETHernet:PORT<Pt>:PTP:IP:IPV4:DESTination . . . 518
12.11.23 ETHernet:PORT<Pt>:PTP:IPV4:IGMP[:ENABle] . . . 518
12.11.24 ETHernet:PORT<Pt>:PTP:IPV4:IGMP:VERSion . . . 519
12.11.25 ETHernet:PORT<Pt>:PTP:IP:IPV6:SOURce . . . 519
12.11.26 ETHernet:PORT<Pt>:PTP:IP:IPV6:DESTination . . . 519
12.11.27 ETHernet:PORT<Pt>:PTP:IPV6:ADDRconfig . . . 520

12.11.28 ETHernet:PORT<Pt>:PTP:IPV6:ADDRconfig:RESult? . . . 520
12.11.29 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:IID . . . 520
12.11.30 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:IID:AUTO . . . 521
12.11.31 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:LINKlocal? . . . 521
12.11.32 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:RAFLag? . . . 521
12.11.33 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:SRCMac? . . . 521
12.11.34 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:PREFix? . . . 521
12.11.35 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:PRFLag? . . . 522
12.11.36 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:LTIMe? . . . 522
12.11.37 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:RENew . . . 522
12.11.38 ETHernet:PORT<Pt>:PTP:IPV6:MLD[:ENABle] . . . 522
12.11.39 ETHernet:PORT<Pt>:PTP:IPV6:MLD:VERSion . . . 523
12.11.40 ETHernet:PORT<Pt>:PTP:IP:DSCP:EVENt . . . 523
12.11.41 ETHernet:PORT<Pt>:PTP:IP:DSCP:NORMal . . . 523
12.11.42 ETHernet:PORT<Pt>:PTP:TUNNeling . . . 524
12.11.43 ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LCount . . . 524
12.11.44 ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:ID . . . 524
12.11.45 ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:CFI . . . 525
12.11.46 ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:DEI . . . 525
12.11.47 ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:PRiority . . . 526
12.11.48 ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:ETYPe . . . 526
12.11.49 ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LCount . . . 527
12.11.50 ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LEVel<Lv>:LABel . . . 527
12.11.51 ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LEVel<Lv>:EBITs . . . 527
12.11.52 ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LEVel<Lv>:TTL . . . 528
12.11.53 ETHernet:PORT<Pt>:PTP:FCAPture . . . 528
12.11.54 ETHernet:PORT<Pt>:PTP:FCAPture:DIRecotry . . . 528
12.11.55 ETHernet:PORT<Pt>:PTP:ELOG . . . 529
12.11.56 ETHernet:PORT<Pt>:PTP:SMODe . . . 529
12.11.57 ETHernet:PORT<Pt>:PTP:DMEChanism . . . 530
12.11.58 ETHernet:PORT<Pt>:PTP:CLOCk:SOURce . . . 530
12.11.59 ETHernet:PORT<Pt>:PTP:CLOCk:IDENtity . . . 530
12.11.60 ETHernet:PORT<Pt>:PTP:CLOCk:IDENtity:AUTO . . . 531
12.11.61 ETHernet:PORT<Pt>:PTP:CLOCk:P1 . . . 531
12.11.62 ETHernet:PORT<Pt>:PTP:CLOCk:P2 . . . 531
12.11.63 ETHernet:PORT<Pt>:PTP:CLOCk:CLASs . . . 532
12.11.64 ETHernet:PORT<Pt>:PTP:CLOCk:ACCuracy . . . 532
12.11.65 ETHernet:PORT<Pt>:PTP:CLOCk:ACCuracy:USER . . . 533
12.11.66 ETHernet:PORT<Pt>:PTP:CLOCk:TSOurce . . . 533
12.11.67 ETHernet:PORT<Pt>:PTP:CLOCk:TSOurce:USER . . . 534
12.11.68 ETHernet:PORT<Pt>:PTP:TIMing:AINTerval . . . 534
12.11.69 ETHernet:PORT<Pt>:PTP:TIMing:ATIMeout . . . 534
12.11.70 ETHernet:PORT<Pt>:PTP:TIMing:UOFFset . . . 535
12.11.71 ETHernet:PORT<Pt>:PTP:TIMing:SINTerval . . . 535
12.11.72 ETHernet:PORT<Pt>:PTP:TIMing:DRINterval . . . 536
12.11.73 ETHernet:PORT<Pt>:PTP:TIMing:UDURation . . . 536
12.11.74 ETHernet:PORT<Pt>:PTP:TIMing:GCABle . . . 537
12.11.75 ETHernet:PORT<Pt>:PTP:FFIeld? . . . 537
12.11.76 ETHernet:PORT<Pt>:PTP:FFIeld:AMFLag . . . 537
12.11.77 ETHernet:PORT<Pt>:PTP:FFIeld:L61 . . . 537
12.11.78 ETHernet:PORT<Pt>:PTP:FFIeld:L59 . . . 538
12.11.79 ETHernet:PORT<Pt>:PTP:FFIeld:CUOValid . . . 538
12.11.80 ETHernet:PORT<Pt>:PTP:FFIeld:PTIMescale . . . 538
12.11.81 ETHernet:PORT<Pt>:PTP:FFIeld:TTRaceable . . . 539
12.11.82 ETHernet:PORT<Pt>:PTP:FFIeld:FTRaceable . . . 539
12.11.83 ETHernet:PORT<Pt>:PTP:FFIeld:SUNCertain . . . 539
12.11.84 ETHernet:PORT<Pt>:PTP:POPTions:IRTLv . . . 540
12.11.85 ETHernet:PORT<Pt>:PTP:POPTions:IRTLv:SPEed . . . 540
12.11.86 ETHernet:PORT<Pt>:PTP:LOG? . . . 541

12.12 Follow . . . 542
12.12.1 ETHernet:FOLLow:TRAFfic . . . 542
12.12.2 ETHernet:FOLLow:STReam<St> . . . 542
12.12.3 ETHernet:FOLLow:GATeway<St> . . . 542
12.12.4 ETHernet:FOLLow:SETTings . . . 543
12.12.5 ETHernet:FOLLow:FILTer . . . 543
12.12.6 ETHernet:FOLLow:THResholds . . . 543
12.12.7 ETHernet:FOLLow:SYNCe . . . 544
12.12.8 ETHernet:FOLLow:PTP . . . 544
12.12.9 ETHernet:PORT<Pt>:FOLLow:TRAFfic . . . 544
12.12.10 ETHernet:PORT<Pt>:FOLLow:STReam<St> . . . 545
12.12.11 ETHernet:PORT<Pt>:FOLLow:GATeway<St> . . . 545
12.12.12 ETHernet:PORT<Pt>:FOLLow:SETTings . . . 546
12.12.13 ETHernet:PORT<Pt>:FOLLow:FILTer . . . 546
12.12.14 ETHernet:PORT<Pt>:FOLLow:THResholds . . . 546
12.12.15 ETHernet:PORT<Pt>:FOLLow:SYNCe . . . 547
12.12.16 ETHernet:PORT<Pt>:FOLLow:PTP . . . 547
12.12.17 ETHernet:PORT<Pt>:FOLLow:WAN . . . 547
12.13 Stimuli . . . 548
12.13.1 ETHernet:PORT<Pt>:STIMuli:ALARm . . . 548
12.13.2 ETHernet:PORT<Pt>:STIMuli:ERRor . . . 548
12.13.3 ETHernet:PORT<Pt>:STIMuli:EINSertion . . . 549
12.13.4 ETHernet:PORT<Pt>:STIMuli:EBLength . . . 549
12.13.5 ETHernet:PORT<Pt>:STIMuli:FEC:ERRor:LANE . . . 550
12.13.6 ETHernet:PORT<Pt>:STIMuli:WAN:ALARm . . . 550
12.13.7 ETHernet:PORT<Pt>:STIMuli:WAN:ERRor . . . 551
12.13.8 ETHernet:PORT<Pt>:STIMuli:WAN:EINSertion . . . 551
12.13.9 ETHernet:PORT<Pt>:STIMuli:WAN:EBLength . . . 551
12.13.10 ETHernet:PORT<Pt>:STIMuli:DEViation . . . 552
12.13.11 ETHernet:PORT<Pt>:STIMuli:PCS:ALARm . . . 552
12.13.12 ETHernet:PORT<Pt>:STIMuli:PCS:ERRor . . . 553
12.13.13 ETHernet:PORT<Pt>:STIMuli:PCS:ERRor:LANE . . . 553
12.13.14 ETHernet:PORT<Pt>:STIMuli:PCS:EINSertion . . . 554
12.13.15 ETHernet:PORT<Pt>:STIMuli:PCS:EBLength . . . 554
12.13.16 ETHernet:PORT<Pt>:STIMuli:PCS:ERATe . . . 554
12.13.17 ETHernet:PORT<Pt>:STIMuli:PCS:EELength . . . 555
12.13.18 ETHernet:PORT<Pt>:STIMuli:PCS:ENLength . . . 555
12.13.19 ETHernet:PORT<Pt>:STIMuli:PCS:SKEW:BIT . . . 555
12.13.20 ETHernet:PORT<Pt>:STIMuli:PCS:SKEW:TYPE . . . 556
12.13.21 ETHernet:PORT<Pt>:STIMuli:PCS:SKEW:LANE . . . 556
12.13.22 ETHernet:PORT<Pt>:STIMuli:PCS:SKEW:NS? . . . 556
12.14 Result . . . 557
12.14.1 ETHernet:PORT<Pt>:IFETch? . . . 557
12.14.2 ETHernet:PORT<Pt>:TFETch? . . . 563
12.14.3 ETHernet:PORT<Pt>:TFRDiffrence:PSELection . . . 564
12.15 Status . . . 565
12.15.1 ETHernet:STATus:PORT<Pt>:AESummary[:EVENt]? . . . 565
12.15.2 ETHernet:STATus:PORT<Pt>:AESummary:CONDition? . . . 565
12.15.3 ETHernet:STATus:PORT<Pt>:ALARm<section>[:EVENt]? . . . 565
12.15.4 ETHernet:STATus:PORT<Pt>:ALARm<section>:CONDition? . . . 566
12.15.5 ETHernet:STATus:PORT<Pt>:ERRor<section>[:EVENt]? . . . 567
12.15.6 ETHernet:STATus:PORT<Pt>:ERRor<section>:CONDition? . . . 567
12.15.7 ETHernet:STATus:PORT<Pt>:LINK? . . . 568
12.15.8 ETHernet:STATus:PORT<Pt>:INTerface? . . . 568
12.15.9 ETHernet:STATus:PORT<Pt>:FRAMes? . . . 568
12.15.10 ETHernet:STATus:PORT<Pt>:MPLS? . . . 568
12.15.11 ETHernet:STATus:PORT<Pt>:VLAN? . . . 568
12.15.12 ETHernet:STATus:PORT<Pt>:SPeed? . . . 569
12.15.13 ETHernet:STATus:PORT<Pt>:DUPLex? . . . 569

12.15.14 ETHernet:STATus:PORT<Pt>:MDI? . . . 569
12.15.15 ETHernet:STATus:PORT<Pt>:L10G? . . . 569
12.15.16 ETHernet:STATus:PORT<Pt>:LFS? . . . 569
12.15.17 ETHernet:STATus:PORT<Pt>:ANComplete? . . . 570
12.15.18 ETHernet:STATus:PORT<Pt>:PCAPable? . . . 570
12.15.19 ETHernet:STATus:PORT<Pt>:APRequest? . . . 570
12.15.20 ETHernet:STATus:PORT<Pt>:RFAult? . . . 570
12.15.21 ETHernet:STATus:PORT<Pt>:LCLock? . . . 570
12.15.22 ETHernet:STATus:PORT<Pt>:SADuplex? . . . 571
12.15.23 ETHernet:STATus:PORT<Pt>:UTILization? . . . 571
12.15.24 ETHernet:STATus:PORT<Pt>:THRoughput? . . . 571
12.15.25 ETHernet:STATus:PORT<Pt>:EFRames? . . . 571
12.15.26 ETHernet:STATus:PORT<Pt>:PCS:SHLock? . . . 572
12.15.27 ETHernet:STATus:PORT<Pt>:PCS:AMLock? . . . 572
12.15.28 ETHernet:STATus:PORT<Pt>:TIMing? . . . 572
12.15.29 ETHernet:STATus:PORT<Pt>:TIMing:SOURce? . . . 572
12.15.30 ETHernet:STATus:PORT<Pt>:SYNCe? . . . 572
12.15.31 ETHernet:STATus:PORT<Pt>:PTP:LCLock? . . . 573
12.15.32 ETHernet:STATus:PORT<Pt>:PTP:WCLock? . . . 573
12.15.33 ETHernet:STATus:PORT<Pt>:PTP:PCLock? . . . 573
12.15.34 ETHernet:STATus:PORT<Pt>:PTP:GCLock? . . . 573
12.15.35 ETHernet:STATus:PORT<Pt>:PTP:FMASters? . . . 574
12.15.36 ETHernet:STATus:PORT<Pt>:WAN:CAPTure:SOH:TRACe? . . . 574
12.15.37 ETHernet:STATus:PORT<Pt>:WAN:CAPTure<Frame>:SOH? . . . 574
12.15.38 ETHernet:STATus:PORT<Pt>:WAN:CAPTure:POH:TRACe? . . . 575
12.15.39 ETHernet:STATus:PORT<Pt>:WAN:CAPTure<Frame>:POH? . . . 575
12.15.40 ETHernet:STATus:PORT<Pt>:TSRC:R10M? . . . 576
12.15.41 ETHernet:STATus:PORT<Pt>:TSRC:RPPS? . . . 576
12.15.42 ETHernet:STATus:PORT<Pt>:TSRC:GPS? . . . 576
12.16 Ping . . . 577
12.16.1 ETHernet:PING:STARt . . . 577
12.16.2 ETHernet:PING:STOP . . . 577
12.16.3 ETHernet:PING:SETup:PORT<Pt>:DMODe . . . 577
12.16.4 ETHernet:PING:SETup:PORT<Pt>:SDURation . . . 577
12.16.5 ETHernet:PING:SETup:PORT<Pt>:RDURation . . . 578
12.16.6 ETHernet:PING:SETup:PORT<Pt>:INTerval . . . 578
12.16.7 ETHernet:PING:SETup:PORT<Pt>:FSIZe . . . 578
12.16.8 ETHernet:PING:SETup:PORT<Pt>:TOUT . . . 579
12.16.9 ETHernet:PING:RESults:SUMMary? . . . 579
12.16.10 ETHernet:PING:RESults:RTT? . . . 579
12.16.11 ETHernet:PING:RESults:NREQuests? . . . 580
12.16.12 ETHernet:PING:RESults:SREQuest? . . . 580
12.17 Traceroute . . . 581
12.17.1 ETHernet:TRACeroute:STARt . . . 581
12.17.2 ETHernet:TRACeroute:STOP . . . 581
12.17.3 ETHernet:TRACeroute:SETup:NATTempts . . . 581
12.17.4 ETHernet:TRACeroute:SETup:MNHops . . . 581
12.17.5 ETHernet:TRACeroute:SETup:TOUT . . . 582
12.17.6 ETHernet:TRACeroute:SETup:NTPHosts . . . 582
12.17.7 ETHernet:TRACeroute:RESults:NHOPs? . . . 582
12.17.8 ETHernet:TRACeroute:RESults:HOP? . . . 582
12.18 RFC2544 . . . 584
12.18.1 ETHernet:RFC:STARt . . . 584
12.18.2 ETHernet:RFC:STOP . . . 584
12.18.3 ETHernet:RFC:SETup:GENeral:MODE . . . 584
12.18.4 ETHernet:RFC:SETup:GENeral:TSELection . . . 584
12.18.5 ETHernet:RFC:SETup:GENeral:E2E:OWAY . . . 585
12.18.6 ETHernet:RFC:SETup:GENeral:E2E:ADDRess . . . 585
12.18.7 ETHernet:RFC:SETup:GENeral:E2E:TDIRection . . . 585

12.18.8 ETHernet:RFC:SETup:GENeral:E2E:SSTore . . . 586
12.18.9 ETHernet:RFC:SETup:GENeral:TLFrames . . . 586
12.18.10 ETHernet:RFC:SETup:GENeral:IAFFilter . . . 586
12.18.11 ETHernet:RFC:SETup:GENeral:JITTer . . . 587
12.18.12 ETHernet:RFC:SETup:GENeral:ACCumulate . . . 587
12.18.13 ETHernet:RFC:SETup:GENeral:TCLayer . . . 587
12.18.14 ETHernet:RFC:SETup:GENeral:TTYPe . . . 588
12.18.15 ETHernet:RFC:SETup:GENeral:VERCompati . . . 588
12.19 RFC2544 - Throughput . . . 589
12.19.1 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:MODE . . . 589
12.19.2 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:USER . . . 589
12.19.3 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:JUMBo . . . 590
12.19.4 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:BEGin . . . 590
12.19.5 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:END . . . 590
12.19.6 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:STEP . . . 591
12.19.7 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:CONStant . . . 591
12.19.8 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:FLEXible:COUNt . . . 591
12.19.9 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:FLEXible<Count>:SIZE . . . 592
12.19.10 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:STOP . . . 592
12.19.11 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:MINimum . . . 593
12.19.12 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:MAXimum . . . 593
12.19.13 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:STEP . . . 594
12.19.14 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch[:ENABle] . . . 594
12.19.15 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch:MODE . . . 594
12.19.16 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch :RESolution . . . 595
12.19.17 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:DURation:STEP . . . 595
12.19.18 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:DURation:REPeats . . . 596
12.20 RFC2544 - Frame Loss . . . 597
12.20.1 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:MODE . . . 597
12.20.2 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:USER . . . 597
12.20.3 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:JUMBo . . . 598
12.20.4 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:BEGin . . . 598
12.20.5 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:END . . . 598
12.20.6 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:STEP . . . 599
12.20.7 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:CONStant . . . 599
12.20.8 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:FLEXible:COUNt . . . 599
12.20.9 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:FLEXible<Count>:SIZE . . . 600
12.20.10 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:STOP . . . 600
12.20.11 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:MINimum . . . 601
12.20.12 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:MAXimum . . . 601
12.20.13 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:STEP . . . 602
12.20.14 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:ASEarch[:ENABle] . . . 602
12.20.15 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:ASEarch:MODE . . . 602
12.20.16 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:ASEarch:RESolution . . . 603
12.20.17 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:DURation:STEP . . . 603
12.20.18 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:DURation:REPeats . . . 604
12.21 RFC2544 - Throughput and Frame Loss . . . 605
12.21.1 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:MODE . . . 605
12.21.2 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:USER . . . 605
12.21.3 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:JUMBo . . . 606
12.21.4 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:BEGin . . . 606
12.21.5 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:END . . . 606
12.21.6 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:STEP . . . 607
12.21.7 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:CONStant . . . 607
12.21.8 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:FLEXible:COUNt . . . 607
12.21.9 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:FLEXible<Count>:SIZE . . . 608
12.21.10 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:STOP . . . 608
12.21.11 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:MINimum . . . 609

12.21.12 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:MAXimum . . . 609
12.21.13 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:STEP . . . 610
12.21.14 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:ASEarch [:ENABle] . . . 610
12.21.15 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:ASEarch:MODE . . . 611
12.21.16 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:ASEarch :RESolution . . . 611
12.21.17 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:DURation:STEP . . . 611
12.21.18 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:DURation:REPeats . . . 612
12.22 RFC2544 - Latency . . . 613
12.22.1 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:MODE . . . 613
12.22.2 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:USER . . . 613
12.22.3 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:JUMBo . . . 614
12.22.4 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:BEGin . . . 614
12.22.5 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:END . . . 614
12.22.6 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:STEP . . . 615
12.22.7 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:CONStant . . . 615
12.22.8 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:FLEXible:COUNt . . . 615
12.22.9 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:FLEXible<Count>:SIZE . . . 616
12.22.10 ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:MINimum . . . 616
12.22.11 ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:MAXimum . . . 617
12.22.12 ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:STEP . . . 617
12.22.13 ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:ONLY . . . 618
12.22.14 ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:OTESt . . . 618
12.22.15 ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:TLEVel . . . 618
12.22.16 ETHernet:RFC:SETup:PORT<Pt>:LATency:DURation:STEP . . . 619
12.22.17 ETHernet:RFC:SETup:PORT<Pt>:LATency:DURation:REPeats . . . 619
12.23 RFC2544 - Burst . . . 620
12.23.1 ETHernet:RFC:SETup:PORT<Pt>:BURSt:MODE . . . 620
12.23.2 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:MODE . . . 620
12.23.3 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:USER . . . 620
12.23.4 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:JUMBo . . . 621
12.23.5 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:BEGin . . . 621
12.23.6 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:END . . . 622
12.23.7 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:STEP . . . 622
12.23.8 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:CONStant . . . 622
12.23.9 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:FLEXible:COUNt . . . 623
12.23.10 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:FLEXible<Count>:SIZE . . . 623
12.23.11 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:MODE . . . 624
12.23.12 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:BEGin . . . 624
12.23.13 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:END . . . 624
12.23.14 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:STEP . . . 625
12.23.15 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:CONStant . . . 625
12.23.16 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:STOP . . . 625
12.23.17 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch[:ENABle] . . . 626
12.23.18 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:MODE . . . 626
12.23.19 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:RESolution . . . 627
12.23.20 ETHernet:RFC:SETup:PORT<Pt>:BURSt:SPBurst:SEConds . . . 627
12.23.21 ETHernet:RFC:SETup:PORT<Pt>:BURSt:SPBurst:ASEarch:MODE . . . 627
12.23.22 ETHernet:RFC:SETup:PORT<Pt>:BURSt:SPBurst:ASEarch:RESolution . . . 628
12.23.23 ETHernet:RFC:SETup:PORT<Pt>:BURSt:DURation:STEP . . . 628
12.23.24 ETHernet:RFC:SETup:PORT<Pt>:BURSt:DURation:REPeats . . . 629
12.24 RFC2544 - Result . . . 630
12.24.1 ETHernet:RFC:RESult:TEST . . . 630
12.24.2 ETHernet:RFC:RESult:PORT<Pt>:NSTep? . . . 630
12.24.3 ETHernet:RFC:RESult:PORT<Pt>:SELect . . . 630
12.24.4 ETHernet:RFC:RESult:PORT<Pt>:FETCh? . . . 631
12.25 Y.1564 Service Activation Test . . . 632
12.25.1 ETHernet:SATest:STARt . . . 632

12.25.2 ETHernet:SATest:STOP . . . 632
12.25.3 ETHernet:SATest:SETup:TMODe . . . 632
12.25.4 ETHernet:SATest:SETup:OWTest:LTRemote[:ENABle] . . . 632
12.25.5 ETHernet:SATest:SETup:OWTest:RTLocal[:ENABle] . . . 633
12.25.6 ETHernet:SATest:SETup:OWTest:BSYMmetry . . . 633
12.25.7 ETHernet:SATest:SETup:OWTest:SYNChronization . . . 633
12.25.8 ETHernet:SATest:SETup:OWTest:TLSAddresses[:ENABle] . . . 634
12.25.9 ETHernet:SATest:SETup:CCFC . . . 634
12.25.10 ETHernet:SATest:SETup:CBRate . . . 634
12.25.11 ETHernet:SATest:SETup:SPTDuration . . . 635
12.25.12 ETHernet:SATest:SETup:SPTDuration:CDURation . . . 635
12.25.13 ETHernet:SATest:SETup:COFail . . . 636
12.25.14 ETHernet:SATest:SETup:GENeral:IAFFilter . . . 636
12.25.15 ETHernet:SATest:SETup:SERVice<no>[:ENABle] . . . 636
12.25.16 ETHernet:SATest:SETup:SERVice<no>:NAME . . . 637
12.25.17 ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate] . . . 637
12.25.18 ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate] . . . 637
12.25.19 ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:PROFile . . . 638
12.25.20 ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:PROFile . . . 638
12.25.21 ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:ENCoding :VIDeo . . . 638
12.25.22 ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:ENCoding :VIDeo . . . 639
12.25.23 ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:ENCoding :VOICe . . . 639
12.25.24 ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:ENCoding :VOICe . . . 640
12.25.25 ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:NCHannels . . . 640
12.25.26 ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:NCHannels . . . 640
12.25.27 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:EIRate . . . 641
12.25.28 ETHernet:SATest:SETup:SERVice<no>:RTLocal:EIRate . . . 641
12.25.29 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:TPOLicing[:ENABle] . . . 641
12.25.30 ETHernet:SATest:SETup:SERVice<no>:RTLocal:TPOLicing[:ENABle] . . . 642
12.25.31 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:TPOLicing:MARGin . . . 642
12.25.32 ETHernet:SATest:SETup:SERVice<no>:RTLocal:TPOLicing:MARGin . . . 642
12.25.33 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CBSize . . . 643
12.25.34 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CBSize . . . 643
12.25.35 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:EBSize . . . 643
12.25.36 ETHernet:SATest:SETup:SERVice<no>:RTLocal:EBSize . . . 644
12.25.37 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare[:ENABle] . . . 644
12.25.38 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare[:ENABle] . . . 644
12.25.39 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:CMEThod . . . 645
12.25.40 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:CMEThod . . . 645
12.25.41 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:DSCP:GREen . . . 645
12.25.42 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:DSCP:GREen . . . 646
12.25.43 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:DSCP :YELLow . . . 646
12.25.44 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:DSCP:YELLow . . . 646
12.25.45 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:PCP:GREen . . . 647
12.25.46 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:PCP:GREen . . . 647
12.25.47 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:PCP :YELLow . . . 647
12.25.48 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:PCP:YELLow . . . 648
12.25.49 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria:FTDelay . . . 648
12.25.50 ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:FTDelay . . . 648
12.25.51 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria :FDVariation . . . 649
12.25.52 ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:FDVariation . . . 649
12.25.53 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria:FLRatio . . . 649

12.25.54 ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:FLRatio 650
12.25.55 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria :AVAilability 650
12.25.56 ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:AVAilability 650
12.25.57 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:FSMode 651
12.25.58 ETHernet:SATest:SETup:SERVice<no>:RTLocal:FSMode 651
12.25.59 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CFSize 651
12.25.60 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CFSize 652
12.25.61 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:EMIX 652
12.25.62 ETHernet:SATest:SETup:SERVice<no>:RTLocal:EMIX 653
12.25.63 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:MTU 654
12.25.64 ETHernet:SATest:SETup:SERVice<no>:RTLocal:MTU 654
12.25.65 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:UDFSize 654
12.25.66 ETHernet:SATest:SETup:SERVice<no>:RTLocal:UDFSize 655
12.25.67 ETHernet:SATest:SETup:SERVice<no>:SCTSteps:SDURation 655
12.25.68 ETHernet:SATest:SETup:SERVice<no>:SCTSteps:NSTep 655
12.25.69 ETHernet:SATest:SETup:SERVice<no>:SCTSteps:SLOPe 656
12.25.70 ETHernet:SATest:RESult:TSYNchronization? 657
12.25.71 ETHernet:SATest:RESult:SCTest[:LTRemote]? 657
12.25.72 ETHernet:SATest:RESult:SCTest:RTLocal? 657
12.25.73 ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:CIRate? 657
12.25.74 ETHernet:SATest:RESult:SCTest:SERVice<no>:RTLocal:CIRate? 658
12.25.75 ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:EIRate? 658
12.25.76 ETHernet:SATest:RESult:SCTest:SERVice<no>:RTLocal:EIRate? 658
12.25.77 ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:TPOLicing? 659
12.25.78 ETHernet:SATest:RESult:SCTest:SERVice<no>:RTLocal:TPOLicing? 659
12.25.79 ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:CBSize? 659
12.25.80 ETHernet:SATest:RESult:SCTest:SERVice<no>:RTLocal:CBSize? 659
12.25.81 ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:EBSize? 660
12.25.82 ETHernet:SATest:RESult:SCTest:SERVice<no>:RTLocal:EBSize? 660
12.25.83 ETHernet:SATest:RESult:SPTest[:LTRemote]? 661
12.25.84 ETHernet:SATest:RESult:SPTest:RTLocal? 661
12.25.85 ETHernet:SATest:RESult:SPTest:SERVice<no>[:LTRemote]? 661
12.25.86 ETHernet:SATest:RESult:SPTest:SERVice<no>:RTLocal? 662
12.26 Cable Test 663
12.26.1 ETHernet:CABLe:STARt 663
12.26.2 ETHernet:CABLe:STOP 663
12.26.3 ETHernet:CABLe:CCStandard 663
12.26.4 ETHernet:CABLe:RESults:PAIR<Pr>? 664
12.27 IP Channel Statistics 665
12.27.1 ETHernet:CSTat:STARt 665
12.27.2 ETHernet:CSTat:STOP 665
12.27.3 ETHernet:CSTat:SETup:FOLLow 665
12.27.4 ETHernet:CSTat:SETup:PORT<Pt>:FOLLow 665
12.27.5 ETHernet:CSTat:SETup:PORT<Pt>:CDEFinitions 666
12.27.6 ETHernet:CSTat:SETup:PORT<Pt>:MERGe 666
12.27.7 ETHernet:CSTat:SETup:PORT<Pt>:FRESults 667
12.27.8 ETHernet:CSTat:RESults:PORT<Pt>:NCHannels? 667
12.27.9 ETHernet:CSTat:RESults:PORT<Pt>:DFRames? 667
12.27.10 ETHernet:CSTat:RESults:PORT<Pt>:FETCh? 667
12.27.11 ETHernet:CSTat:RESults:RESet 669
12.28 Monitor Generator 670
12.28.1 ETHernet:MONGen:SETup:GENeral:IAFFilter 670
12.29 Sync Test 671
12.29.1 ETHernet:SYNTest:SETup:MPERiod 671
12.29.2 ETHernet:SYNTest:SETup:MPERiod:USERdefined 671
12.29.3 ETHernet:SYNTest:SETup:MPERiod:USERdefined:TIMe 671
12.29.4 ETHernet:SYNTest:SETup:PPS:MODE[:ENABle] 672
12.29.5 ETHernet:SYNTest:SETup:PPS:CABLecorrection 672

12.29.6 ETHernet:SYNTest:SETup:PACKet:CABLecorrection . . . 672
12.29.7 ETHernet:THResholds:PPS:PHASe[:ENABle] . . . 673
12.29.8 ETHernet:THResholds:PPS:PHASe:MINValue . . . 673
12.29.9 ETHernet:THResholds:PPS:PHASe:MAXValue . . . 673
12.29.10 ETHernet:THResholds:PPS:FILTered[:ENABle] . . . 674
12.29.11 ETHernet:THResholds:PPS:FILTered:MINValue . . . 674
12.29.12 ETHernet:THResholds:PPS:FILTered:MAXValue . . . 674
12.29.13 ETHernet:THResholds:PPS:DEViation[:ENABle] . . . 675
12.29.14 ETHernet:THResholds:PPS:DEViation:MINValue . . . 675
12.29.15 ETHernet:THResholds:PPS:DEViation:MAXValue . . . 675
12.29.16 ETHernet:THResholds:OWD:SYNC[:ENABle] . . . 676
12.29.17 ETHernet:THResholds:OWD:SYNC:MINValue . . . 676
12.29.18 ETHernet:THResholds:OWD:SYNC:MAXValue . . . 676
12.29.19 ETHernet:THResholds:OWD:FOLLow[:ENABle] . . . 677
12.29.20 ETHernet:THResholds:OWD:FOLLow:MINValue . . . 677
12.29.21 ETHernet:THResholds:OWD:FOLLow:MAXValue . . . 677
12.29.22 ETHernet:THResholds:OWD:DELay[:ENABle] . . . 678
12.29.23 ETHernet:THResholds:OWD:DELay:MINValue . . . 678
12.29.24 ETHernet:THResholds:OWD:DELay:MAXValue . . . 678
12.29.25 ETHernet:PORT<Pt>:SYNTest:ADVanced:ADAPte[:ENABle] . . . 679
12.29.26 ETHernet:SYNTest:SETup:ADVanced:FOLLow . . . 679
12.29.27 ETHernet:PORT<Pt>:SYNCtest:RESults:BOUNds? . . . 679
12.29.28 ETHernet:SYNCtest:1PPS:RESults? . . . 679
12.29.29 ETHernet:PORT<Pt>:SYNCtest:OWD:RESults? . . . 680
12.29.30 ETHernet:PORT<Pt>:SYNCtest:TERR:RESults? . . . 680
12.30 Frame Capture . . . 680
12.30.1 ETHernet:PORT<Pt>:CAPTure:STARt . . . 680
12.30.2 ETHernet:PORT<Pt>:CAPTure:STOP . . . 680
12.30.3 ETHernet:PORT<Pt>:CAPTure:STATus? . . . 681
12.30.4 ETHernet:PORT<Pt>:CAPTure:BUFSize . . . 681
12.30.5 ETHernet:PORT<Pt>:CAPTure:SLICing . . . 681
12.30.6 ETHernet:PORT<Pt>:CAPTure:ALOCation . . . 682
12.30.7 ETHernet:PORT<Pt>:CAPTure:TXFRame . . . 682
12.30.8 ETHernet:PORT<Pt>:CAPTure:TRIGger:TYPE . . . 682
12.30.9 ETHernet:PORT<Pt>:CAPTure:TRIGger:POSition . . . 683
12.30.10 ETHernet:PORT<Pt>:CAPTure:TRIGger:ERRor . . . 683
12.30.11 ETHernet:PORT<Pt>:CAPTure:TRIGger:OFFSet . . . 684
12.30.12 ETHernet:PORT<Pt>:CAPTure:TRIGger:PATTern . . . 684
12.30.13 ETHernet:PORT<Pt>:CAPTure:STORe . . . 684
12.30.14 ETHernet:PORT<Pt>:CAPTure:BUFFer? . . . 685
12.31 Discovery . . . 686
12.31.1 ETHernet:DISCovery:SETup:TYPE . . . 686
12.31.2 ETHernet:DISCovery:SETup:TARGet:COUNt . . . 686
12.31.3 ETHernet:DISCovery:SETup:TARGet:IPV4[:BEGin] . . . 686
12.31.4 ETHernet:DISCovery:RESults:NREQuest? . . . 687
12.31.5 ETHernet:DISCovery:RESults:SREQuest? . . . 687
12.31.6 ETHernet:DISCovery:RESults:SREQuest:DETail? . . . 687
12.31.7 ETHernet:DISCovery:RESults:UPDate . . . 688
12.31.8 ETHernet:DISCovery:RESults:UPDate:RESult? . . . 688
12.31.9 ETHernet:DISCovery:CAPPlication[:EXECute] . . . 688
12.31.10 ETHernet:DISCovery:CAPPlication:PASSword . . . 688
12.31.11 ETHernet:DISCovery:CAPPlication:SAPPlication . . . 689
12.31.12 ETHernet:DISCovery:CAPPlication:TARGet:MAC:SOURce . . . 689
12.31.13 ETHernet:DISCovery:CAPPlication:TARGet:IPV4:SOURce . . . 689
12.31.14 ETHernet:DISCovery:CAPPlication:TARGet:IPV4:DESTination . . . 690
12.31.15 ETHernet:DISCovery:CAPPlication:TARGet:IPV4:NETMask . . . 690
12.31.16 ETHernet:DISCovery:CAPPlication:TARGet:IPV4:GATeway[:ENABle] . . . 690
12.31.17 ETHernet:DISCovery:CAPPlication:TARGet:IPV4:GATeway:ADDRess . . . 691
12.31.18 ETHernet:DISCovery:CAPPlication:TARGet:VLAN:LCount . . . 691

12.31.19 ETHernet:DISCovery:CAPPlication:TARGet:VLAN:LEVel<Lv>:ID 691
12.31.20 ETHernet:DISCovery:CAPPlication:IMULti:MAC:SOURce 692
12.31.21 ETHernet:DISCovery:CAPPlication:IMULti:IPV4:SOURce 692
12.31.22 ETHernet:DISCovery:CAPPlication:IMULti:VLAN:LEVel<Lv>:ID 692
12.32 OAM 694
12.32.1 ETHernet:PORT<Pt>:OAM:PROTocol 694
12.32.2 ETHernet:PORT<Pt>:OAM:SMAC 694
12.32.3 ETHernet:PORT<Pt>:OAM:MAC:ETYPe 695
12.32.4 ETHernet:PORT<Pt>:OAM:AH:DISCovery[:ENABle] 695
12.32.5 ETHernet:PORT<Pt>:OAM:AH:DISCovery:LMODe 695
12.32.6 ETHernet:PORT<Pt>:OAM:AH:DISCovery:VOUI 696
12.32.7 ETHernet:PORT<Pt>:OAM:AH:DISCovery:VSINfo 696
12.32.8 ETHernet:PORT<Pt>:OAM:AH:DISCovery:UNIDirectional 696
12.32.9 ETHernet:PORT<Pt>:OAM:AH:DISCovery:VRETrival 697
12.32.10 ETHernet:PORT<Pt>:OAM:AH:LDEFects:LFAults 697
12.32.11 ETHernet:PORT<Pt>:OAM:AH:LDEFects:DGASp 697
12.32.12 ETHernet:PORT<Pt>:OAM:AH:LDEFects:CEvent 698
12.32.13 ETHernet:PORT<Pt>:OAM:AG[:ENABle] 698
12.32.14 ETHernet:PORT<Pt>:OAM:AG:MEPid 698
12.32.15 ETHernet:PORT<Pt>:OAM:AG:MDLevel 699
12.32.16 ETHernet:PORT<Pt>:OAM:AG:DOMain 699
12.32.17 ETHernet:PORT<Pt>:OAM:AG:MAIDentifier 699
12.32.18 ETHernet:PORT<Pt>:OAM:AG:CCM:INTerval 700
12.32.19 ETHernet:PORT<Pt>:OAM:AG:LBM:OTLV 700
12.32.20 ETHernet:PORT<Pt>:OAM:AG:LBM:TLVLength 701
12.32.21 ETHernet:PORT<Pt>:OAM:AG:LBM:DTLV:VALue 701
12.32.22 ETHernet:PORT<Pt>:OAM:AG:LBM:TTLV:VALue 701
12.32.23 ETHernet:PORT<Pt>:OAM:AG:LTM:TTL 702
12.32.24 ETHernet:PORT<Pt>:OAM:AG:LTM:TID 702
12.32.25 ETHernet:PORT<Pt>:OAM:AG:MEPList:ADD 702
12.32.26 ETHernet:PORT<Pt>:OAM:AG:MEPList:DELete[:MEP] 703
12.32.27 ETHernet:PORT<Pt>:OAM:AG:MEPList:CATalog? 703
12.32.28 ETHernet:PORT<Pt>:OAM:Y[:ENABle] 703
12.32.29 ETHernet:PORT<Pt>:OAM:Y:MEPid 703
12.32.30 ETHernet:PORT<Pt>:OAM:Y:MEGid 704
12.32.31 ETHernet:PORT<Pt>:OAM:Y:MEGLevel 704
12.32.32 ETHernet:PORT<Pt>:OAM:Y:CCM:INTerval 705
12.32.33 ETHernet:PORT<Pt>:OAM:Y:LBM:OTLV 705
12.32.34 ETHernet:PORT<Pt>:OAM:Y:LBM:TLVLength 705
12.32.35 ETHernet:PORT<Pt>:OAM:Y:LBM:DTLV:VALue 706
12.32.36 ETHernet:PORT<Pt>:OAM:Y:LBM:TTLV:VALue 706
12.32.37 ETHernet:PORT<Pt>:OAM:Y:LTM:TTL 706
12.32.38 ETHernet:PORT<Pt>:OAM:Y:LTM:TID 707
12.32.39 ETHernet:PORT<Pt>:OAM:Y:TST:FTSend 707
12.32.40 ETHernet:PORT<Pt>:OAM:Y:TST:RATE 707
12.32.41 ETHernet:PORT<Pt>:OAM:Y:TST:OTLV 708
12.32.42 ETHernet:PORT<Pt>:OAM:Y:TST:TLVLength 708
12.32.43 ETHernet:PORT<Pt>:OAM:Y:TST:TTLV:VALue 708
12.32.44 ETHernet:PORT<Pt>:OAM:Y:MCC:OUI 709
12.32.45 ETHernet:PORT<Pt>:OAM:Y:MCC:DATA 709
12.32.46 ETHernet:PORT<Pt>:OAM:Y:LCK:CMEG 709
12.32.47 ETHernet:PORT<Pt>:OAM:Y:LCK:RATE 710
12.32.48 ETHernet:PORT<Pt>:OAM:Y:AIS:CMEG 710
12.32.49 ETHernet:PORT<Pt>:OAM:Y:AIS:RATE 710
12.32.50 ETHernet:PORT<Pt>:OAM:Y:DMONe:TYPE 711
12.32.51 ETHernet:PORT<Pt>:OAM:Y:DMONe:RATE 711
12.32.52 ETHernet:PORT<Pt>:OAM:Y:DMONe:FTSend 712
12.32.53 ETHernet:PORT<Pt>:OAM:Y:DMONe:OTLV 712
12.32.54 ETHernet:PORT<Pt>:OAM:Y:DMONe:TLVLength 712

12.32.55 ETHernet:PORT<Pt>:OAM:Y:DMONe:DTLV:VALue . . . 713
12.32.56 ETHernet:PORT<Pt>:OAM:Y:DMONe:TTLV:VALue . . . 713
12.32.57 ETHernet:PORT<Pt>:OAM:Y:DMM:TYPE . . . 713
12.32.58 ETHernet:PORT<Pt>:OAM:Y:DMM:RATE . . . 714
12.32.59 ETHernet:PORT<Pt>:OAM:Y:DMM:FTSend . . . 714
12.32.60 ETHernet:PORT<Pt>:OAM:Y:DMM:FVTHreshold . . . 714
12.32.61 ETHernet:PORT<Pt>:OAM:Y:DMM:OTLV . . . 715
12.32.62 ETHernet:PORT<Pt>:OAM:Y:DMM:TLVLength . . . 715
12.32.63 ETHernet:PORT<Pt>:OAM:Y:DMM:DTLV:VALue . . . 715
12.32.64 ETHernet:PORT<Pt>:OAM:Y:DMM:TTLV:VALue . . . 716
12.32.65 ETHernet:PORT<Pt>:OAM:Y:LMM:RATE . . . 716
12.32.66 ETHernet:PORT<Pt>:OAM:Y:LMM:FTSend . . . 716
12.32.67 ETHernet:PORT<Pt>:OAM:Y:LMM:FLTHreshold . . . 717
12.32.68 ETHernet:PORT<Pt>:OAM:Y:SLM:RATE . . . 717
12.32.69 ETHernet:PORT<Pt>:OAM:Y:SLM:FTSend . . . 717
12.32.70 ETHernet:PORT<Pt>:OAM:Y:SLM:FLTHreshold . . . 718
12.32.71 ETHernet:PORT<Pt>:OAM:Y:EXM:DATA . . . 718
12.32.72 ETHernet:PORT<Pt>:OAM:Y:VSM:DATA . . . 718
12.32.73 ETHernet:PORT<Pt>:OAM:Y:MEPList:ADD . . . 719
12.32.74 ETHernet:PORT<Pt>:OAM:Y:MEPList:DELete[:MEP] . . . 719
12.32.75 ETHernet:PORT<Pt>:OAM:Y:MEPList:CATalog? . . . 719
12.32.76 ETHernet:PORT<Pt>:OAM:DISCovery:TYPE . . . 719
12.32.77 ETHernet:PORT<Pt>:OAM:DISCovery:INTerval . . . 720
12.32.78 ETHernet:PORT<Pt>:OAM:DISCovery:DMAX . . . 720
12.32.79 ETHernet:PORT<Pt>:OAM:DISCovery:CATalog? . . . 721
12.32.80 ETHernet:PORT<Pt>:OAM:MPLS[:ENABle] . . . 721
12.32.81 ETHernet:PORT<Pt>:OAM:MPLS:LCOunt . . . 721
12.32.82 ETHernet:PORT<Pt>:OAM:MPLS:LEVel<Lv>:LABel . . . 722
12.32.83 ETHernet:PORT<Pt>:OAM:MPLS:LEVel<Lv>:TCLass . . . 722
12.32.84 ETHernet:PORT<Pt>:OAM:MPLS:LEVel<Lv>:TTL . . . 722
12.32.85 ETHernet:PORT<Pt>:OAM:MPLS:TPRofile[:ENABle] . . . 723
12.32.86 ETHernet:PORT<Pt>:OAM:MIM[:ENABle] . . . 723
12.32.87 ETHernet:PORT<Pt>:OAM:MIM:BTAG:DEI . . . 723
12.32.88 ETHernet:PORT<Pt>:OAM:MIM:BTAG:PRIority . . . 724
12.32.89 ETHernet:PORT<Pt>:OAM:MIM:BTAG:VID . . . 724
12.32.90 ETHernet:PORT<Pt>:OAM:MIM:ITAG:UCA . . . 724
12.32.91 ETHernet:PORT<Pt>:OAM:MIM:ITAG:DEI . . . 725
12.32.92 ETHernet:PORT<Pt>:OAM:MIM:ITAG:PRIority . . . 725
12.32.93 ETHernet:PORT<Pt>:OAM:MIM:ITAG:SID . . . 725
12.32.94 ETHernet:PORT<Pt>:OAM:MIM:SMAC . . . 726
12.32.95 ETHernet:PORT<Pt>:OAM:MIM:ETHernet:DMAC . . . 726
12.32.96 ETHernet:PORT<Pt>:OAM:VLAN[:ENABle] . . . 726
12.32.97 ETHernet:PORT<Pt>:OAM:VLAN:LCOunt . . . 727
12.32.98 ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:ID . . . 727
12.32.99 ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:CFI . . . 728
12.32.100 ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:DEI . . . 728
12.32.101 ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:PRIority . . . 729
12.32.102 ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:ETYPe . . . 729
12.32.103 ETHernet:PORT<Pt>:OAM:OUTPut:ULBM . . . 730
12.32.104 ETHernet:PORT<Pt>:OAM:OUTPut:MLBM . . . 730
12.32.105 ETHernet:PORT<Pt>:OAM:OUTPut:LTM . . . 730
12.32.106 ETHernet:PORT<Pt>:OAM:OUTPut:MCC . . . 730
12.32.107 ETHernet:PORT<Pt>:OAM:OUTPut:EXM . . . 730
12.32.108 ETHernet:PORT<Pt>:OAM:OUTPut:VSM . . . 731
12.32.109 ETHernet:PORT<Pt>:OAM:OUTPut:TST . . . 731
12.32.110 ETHernet:PORT<Pt>:OAM:OUTPut:LMM . . . 731
12.32.111 ETHernet:PORT<Pt>:OAM:OUTPut:SLM . . . 731
12.32.112 ETHernet:PORT<Pt>:OAM:OUTPut:DMONe . . . 731
12.32.113 ETHernet:PORT<Pt>:OAM:OUTPut:DMM . . . 732

12.32.114 ETHernet:PORT<Pt>:OAM:OUTPut:CCM . . . 732
12.32.115 ETHernet:PORT<Pt>:OAM:OUTPut:AIS . . . 732
12.32.116 ETHernet:PORT<Pt>:OAM:OUTPut:LCK . . . 732
12.32.117 ETHernet:PORT<Pt>:OAM:OUTPut:LOOP . . . 733
12.32.118 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:MODE? . . . 733
12.32.119 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:PARSer? . . . 733
12.32.120 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:MUXer? . . . 733
12.32.121 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:OUI? . . . 734
12.32.122 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:VSI? . . . 734
12.32.123 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:UNIDirectional? . . . 734
12.32.124 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:LEVents? . . . 734
12.32.125 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:LBACk? . . . 734
12.32.126 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:VRETrieval? . . . 734
12.32.127 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:REVision? . . . 735
12.32.128 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:MAC? . . . 735
12.32.129 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:MPDU? . . . 735
12.32.130 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:MODE? . . . 735
12.32.131 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:PARSer? . . . 735
12.32.132 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:MUXer? . . . 735
12.32.133 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:UNIDirectional? . . . 736
12.32.134 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:LEVents? . . . 736
12.32.135 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:LBACk? . . . 736
12.32.136 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:VRETrieval? . . . 736
12.32.137 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:REVision? . . . 736
12.32.138 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:MPDU? . . . 736
12.32.139 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:STATe? . . . 737
12.32.140 ETHernet:PORT<Pt>:OAM:STATus:AH:VARiable:REQuest . . . 737
12.32.141 ETHernet:PORT<Pt>:OAM:STATus:AH:VARiable:RESPonse? . . . 738
12.32.142 ETHernet:PORT<Pt>:OAM:LOG? . . . 738
12.33 TCP Throughput Test - RFC-6349 . . . 739
12.33.1 ETHernet:TTTest:SETup:TCPPort . . . 739
12.33.2 ETHernet:TTTest:SETup:FATSequence . . . 739
12.33.3 ETHernet:TTTest:SETup:CIPerf . . . 739
12.33.4 ETHernet:TTTest:SETup:SLSName . . . 740
12.33.5 ETHernet:TTTest:SETup:LLSName . . . 740
12.33.6 ETHernet:TTTest:SETup:SRSName . . . 740
12.33.7 ETHernet:TTTest:SETup:LRSName . . . 741
12.33.8 ETHernet:TTTest:SETup:TDLR . . . 741
12.33.9 ETHernet:TTTest:SETup:TDRL . . . 741
12.33.10 ETHernet:TTTest:SETup:TDSimultaneos . . . 742
12.33.11 ETHernet:TTTest:SETup[:LTRemote]:CIRate . . . 742
12.33.12 ETHernet:TTTest:SETup[:LTRemote]:MTUSize . . . 742
12.33.13 ETHernet:TTTest:SETup:RTLocal:CIRate . . . 743
12.33.14 ETHernet:TTTest:SETup:RTLocal:MTUSize . . . 743
12.33.15 ETHernet:TTTest:SETup:PMTU:DISCover[:ENABle] . . . 743
12.33.16 ETHernet:TTTest:SETup:PMTU:DISCover:MINimum . . . 744
12.33.17 ETHernet:TTTest:SETup:PMTU:DISCover:MAXimum . . . 744
12.33.18 ETHernet:TTTest:SETup:BRTT . . . 744
12.33.19 ETHernet:TTTest:SETup:BRTT:AUTO . . . 745
12.33.20 ETHernet:TTTest:SETup:BRTT:AUTO:DURation . . . 745
12.33.21 ETHernet:TTTest:SETup:WSTTest[:ENABle] . . . 745
12.33.22 ETHernet:TTTest:SETup:WSTTest:MODE . . . 746
12.33.23 ETHernet:TTTest:SETup:WSCan[:LTRemote]:SDURation . . . 746
12.33.24 ETHernet:TTTest:SETup:WSCan[:LTRemote]:AMODe:MWSize . . . 746
12.33.25 ETHernet:TTTest:SETup:WSCan:RTLocal:SDURation . . . 747
12.33.26 ETHernet:TTTest:SETup:WSCan:RTLocal:AMODe:MWSize . . . 747
12.33.27 ETHernet:TTTest:SETup:WSCan:AMODe[:LTRemote]:STEP<no>[:ENABle] . . . 747
12.33.28 ETHernet:TTTest:SETup:WSCan:AMODe:RTLocal:STEP<no>[:ENABle] . . . 748

12.33.29 ETHernet:TTTest:SETup:WSCan:EMODe[:LTRemote]:STEP<no>[:ENABle] . . . 748
12.33.30 ETHernet:TTTest:SETup:WSCan:EMODe[:LTRemote]:STEP<no>:WSIZe . . . 748
12.33.31 ETHernet:TTTest:SETup:WSCan:EMODe[:LTRemote]:STEP<no>CONNections . . . 749
12.33.32 ETHernet:TTTest:SETup:WSCan:EMODe:RTLocal:STEP<no>[:ENABle] . . . 749
12.33.33 ETHernet:TTTest:SETup:WSCan:EMODe:RTLocal:STEP<no>:WSIZe . . . 749
12.33.34 ETHernet:TTTest:SETup:WSCan:EMODe:RTLocal:STEP<no>:CONNections . . . 750
12.33.35 ETHernet:TTTest:SETup:TTESt[:LTRemote]:THReshold[:ENABle] . . . 750
12.33.36 ETHernet:TTTest:SETup:TTESt[:LTRemote]:THReshold:PERCentage . . . 750
12.33.37 ETHernet:TTTest:SETup:TTESt[:LTRemote]:SDURation . . . 751
12.33.38 ETHernet:TTTest:SETup:TTESt:RTLocal:THReshold[:ENABle] . . . 751
12.33.39 ETHernet:TTTest:SETup:TTESt:RTLocal:THReshold:PERCentage . . . 751
12.33.40 ETHernet:TTTest:SETup:TTESt:RTLocal:SDURation . . . 752
12.33.41 ETHernet:TTTest:SETup:TTESt:EMODe[:LTRemote][:ENABle] . . . 752
12.33.42 ETHernet:TTTest:SETup:TTESt:EMODe[:LTRemote]:WSIZe . . . 752
12.33.43 ETHernet:TTTest:SETup:TTESt:EMODe[:LTRemote]:CONNections . . . 753
12.33.44 ETHernet:TTTest:SETup:TTESt:EMODe:RTLocal[:ENABle] . . . 753
12.33.45 ETHernet:TTTest:SETup:TTESt:EMODe:RTLocal:WSIZe . . . 753
12.33.46 ETHernet:TTTest:SETup:TTESt:EMODe:RTLocal:CONNections . . . 754
12.33.47 ETHernet:TTTest:SETup:MSERvice[:ENABle] . . . 754
12.33.48 ETHernet:TTTest:SETup:MSERvice[:LTRemote]:TDURation . . . 754
12.33.49 ETHernet:TTTest:SETup:MSERvice[:LTRemote]:WSIZe . . . 755
12.33.50 ETHernet:TTTest:SETup:MSERvice[:LTRemote]:WSIZe:AUTO . . . 755
12.33.51 ETHernet:TTTest:SETup:MSERvice:RTLocal:TDURation . . . 755
12.33.52 ETHernet:TTTest:SETup:MSERvice:RTLocal:WSIZe . . . 756
12.33.53 ETHernet:TTTest:SETup:MSERvice:RTLocal:WSIZe:AUTO . . . 756
12.33.54 ETHernet:TTTest:SETup:MSERvice:SERVice<no>[:ENABle] . . . 756
12.33.55 ETHernet:TTTest:SETup:MSERvice:SERVice<no>:DPORt . . . 757
12.33.56 ETHernet:TTTest:SETup:MSERvice:SERVice<no>:DOT . . . 757
12.33.57 ETHernet:TTTest:SETup:MSERvice:SERVice<no>:TCLass . . . 757
12.33.58 ETHernet:TTTest:SETup:MSERvice:SERVice<no>:FLABel . . . 758
12.33.59 ETHernet:TTTest:SETup:MSERvice:SERVice<no>:NAME . . . 758
12.33.60 ETHernet:TTTest:RESult:PARameters? . . . 759
12.33.61 ETHernet:TTTest:RESult:WSCan? . . . 759
12.33.62 ETHernet:TTTest:RESult:THRoughput? . . . 759
12.33.63 ETHernet:TTTest:RESult:THRoughput:EXPort . . . 760
12.33.64 ETHernet:TTTest:RESult:MSERvice? . . . 761
12.33.65 ETHernet:TTTest:RESult:MSERvice:TOTal? . . . 761
12.33.66 ETHernet:TTTest:RESult:MSERvice:EXPort . . . 762

13 eCPRI/RoE BERT **763**
13.1 Port Setup . . . 763
13.1.1 ETHernet:PORT<Pt>:DPIReference . . . 763
13.2 PCS . . . 763
13.3 Traffic . . . 763
13.4 Settings . . . 763
13.5 Thresholds . . . 764
13.6 SyncE . . . 764
13.7 Precision Time Protocol - IEEE 1588v2 . . . 764
13.8 Follow . . . 764
13.9 Stimuli . . . 764
13.10 Result . . . 764
13.11 Status . . . 764
13.12 Frame Content . . . 765
13.12.1 ETHernet:PORT<Pt>:STReam<St>:RFRame . . . 765
13.12.2 ETHernet:PORT<Pt>:STReam<St>:ECPRi:MESSage . . . 765
13.12.3 ETHernet:PORT<Pt>:STReam<St>:ECPRi:MVAL . . . 766

13.12.4 ETHernet:PORT<Pt>:STReam<St>:ECPRi:LENGth . . . 766
13.12.5 ETHernet:PORT<Pt>:STReam<St>:ECPRi:IQData:PCID . . . 766
13.12.6 ETHernet:PORT<Pt>:STReam<St>:ECPRi:IQData:SEQId . . . 767
13.12.7 ETHernet:PORT<Pt>:STReam<St>:ECPRi:BITSeq:PCID . . . 767
13.12.8 ETHernet:PORT<Pt>:STReam<St>:ECPRi:BITSeq:SEQId . . . 767
13.12.9 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RTCData:PCID . . . 768
13.12.10 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RTCData:SEQId . . . 768
13.12.11 ETHernet:PORT<Pt>:STReam<St>:ECPRi:GDTRans:PCID . . . 768
13.12.12 ETHernet:PORT<Pt>:STReam<St>:ECPRi:GDTRans:SEQId . . . 769
13.12.13 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:ACID . . . 769
13.12.14 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:RW . . . 769
13.12.15 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:RRESp . . . 770
13.12.16 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:ELID . . . 770
13.12.17 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:ADDRess . . . 771
13.12.18 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:LENGth . . . 771
13.12.19 ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:MESId . . . 771
13.12.20 ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:ACTType . . . 772
13.12.21 ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:TSTamp . . . 772
13.12.22 ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:CVALue . . . 772
13.12.23 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RRESet:REID . . . 773
13.12.24 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RRESet:COPtion . . . 773
13.12.25 ETHernet:PORT<Pt>:STReam<St>:ECPRi:CDATa . . . 774
13.12.26 ETHernet:PORT<Pt>:STReam<St>:ECPRi:CLENgth . . . 774
13.12.27 ETHernet:PORT<Pt>:STReam<St>:ECPRi:CENable . . . 774
13.12.28 ETHernet:PORT<Pt>:STReam<St>:ROE:SUBType . . . 775
13.12.29 ETHernet:PORT<Pt>:STReam<St>:ROE:SVALue . . . 775
13.12.30 ETHernet:PORT<Pt>:STReam<St>:ROE:FLID . . . 775
13.12.31 ETHernet:PORT<Pt>:STReam<St>:ROE:LENGth . . . 776
13.12.32 ETHernet:PORT<Pt>:STReam<St>:ROE:OINFo . . . 776
13.12.33 ETHernet:PORT<Pt>:STReam<St>:ROE:CONTrol:OPCode . . . 776
13.12.34 ETHernet:PORT<Pt>:STReam<St>:ROE:CDATa . . . 777
13.12.35 ETHernet:PORT<Pt>:STReam<St>:ROE:CLENgth . . . 777
13.12.36 ETHernet:PORT<Pt>:STReam<St>:ROE:CENable . . . 777

14 Fibre Channel **779**
14.1 Port . . . 779
14.1.1 FCHannel:PORT<Pt>:MODE . . . 779
14.1.2 FCHannel:TX<Pt>:TIMing . . . 779
14.2 Interface . . . 780
14.2.1 FCHannel:PORT<Pt>:TOPology . . . 780
14.2.2 FCHannel:PORT<Pt>:PTP:LOGin . . . 780
14.2.3 FCHannel:PORT<Pt>:FABRic:LOGin . . . 780
14.2.4 FCHannel:PORT<Pt>:EPORt:ELP . . . 781
14.2.5 FCHannel:PORT<Pt>:FCONtrol[:ENABle] . . . 781
14.2.6 FCHannel:PORT<Pt>:FCONtrol:LCRedit . . . 781
14.2.7 FCHannel:PORT<Pt>:FCONtrol:RCRedit . . . 781
14.2.8 FCHannel:PORT<Pt>:SOURce:WWN . . . 782
14.2.9 FCHannel:PORT<Pt>:SOURce:WWN:DEFault . . . 782
14.2.10 FCHannel:PORT<Pt>:SOURce:ID . . . 782
14.2.11 FCHannel:PORT<Pt>:DESTination:WWN . . . 783
14.2.12 FCHannel:PORT<Pt>:DESTination:ID . . . 783
14.2.13 FCHannel:PORT<Pt>:DESTination:LOGin . . . 783
14.3 Frame . . . 784
14.3.1 FCHannel:PORT<Pt>:FRAMe:FRAMing . . . 784
14.3.2 FCHannel:PORT<Pt>:FRAMe:CONTent . . . 784
14.3.3 FCHannel:PORT<Pt>:FRAMe:UP32 . . . 785
14.3.4 FCHannel:PORT<Pt>:FRAMe:HEADer:SID . . . 785
14.3.5 FCHannel:PORT<Pt>:FRAMe:HEADer:OID . . . 785
14.3.6 FCHannel:PORT<Pt>:FRAMe:HEADer:RID . . . 786

14.4 Generator . . . 786
14.4.1 FCHannel:PORT<Pt>:TRAFfic:DMODe . . . 786
14.4.2 FCHannel:PORT<Pt>:TRAFfic:DSEConds . . . 786
14.4.3 FCHannel:PORT<Pt>:TRAFfic:DFRames . . . 787
14.4.4 FCHannel:PORT<Pt>:TRAFfic:ASTGenerator . . . 787
14.4.5 FCHannel:TRAFfic:GENerator:STARt . . . 787
14.4.6 FCHannel:TRAFfic:GENerator:STOP . . . 788
14.4.7 FCHannel:PORT<Pt>:TRAFfic:GENerator:STARt . . . 788
14.4.8 FCHannel:PORT<Pt>:TRAFfic:GENerator:STOP . . . 788
14.4.9 FCHannel:PORT<Pt>:TRAFfic:GENerator:STATus? . . . 788
14.5 Stream . . . 788
14.5.1 FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:PROFile . . . 788
14.5.2 FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad[:CONStant] . . . 789
14.5.3 FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP[:MODE] . . . 789
14.5.4 FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:STARt . . . 789
14.5.5 FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:END . . . 790
14.5.6 FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:STEP . . . 790
14.5.7 FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:DURation . . . 790
14.5.8 FCHannel:PORT<Pt>:TRAFfic:STReam:FSIZe . . . 791
14.5.9 FCHannel:PORT<Pt>:TRAFfic:STReam:LATency[:ENABle] . . . 791
14.5.10 FCHannel:PORT<Pt>:TRAFfic:STReam:LATency:THResholds[:ENABle] . . . 791
14.5.11 FCHannel:PORT<Pt>:TRAFfic:STReam:LATency:THResholds:VALue . . . 792
14.5.12 FCHannel:PORT<Pt>:TRAFfic:STReam:JITTer[:ENABle] . . . 792
14.5.13 FCHannel:PORT<Pt>:TRAFfic:STReam:JITTer:THResholds[:ENABle] . . . 792
14.5.14 FCHannel:PORT<Pt>:TRAFfic:STReam:JITTer:THResholds:VALue . . . 793
14.5.15 FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure[:ENABle] . . . 793
14.5.16 FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure:MDISruption . . . 793
14.5.17 FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure:THResholds [:ENABle] . . . 794
14.5.18 FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure:THResholds :VALue . . . 794
14.6 Settings . . . 795
14.6.1 FCHannel:PORT<Pt>:SETTings:BER:OBAMeasuring . . . 795
14.6.2 FCHannel:PORT<Pt>:SETTings:BER:IAFFilter . . . 795
14.6.3 FCHannel:PORT<Pt>:SETTings:BER:CLFrames . . . 795
14.6.4 FCHannel:PORT<Pt>:SETTings:BER:HIDeframeloss . . . 796
14.7 Thresholds . . . 797
14.7.1 FCHannel:PORT<Pt>:THResholds[:ENABle] . . . 797
14.7.2 FCHannel:PORT<Pt>:THResholds:SELect . . . 797
14.7.3 FCHannel:PORT<Pt>:THResholds:VALue . . . 798
14.8 Follow . . . 798
14.8.1 FCHannel:PORT<Pt>:FOLLow:GENerator . . . 798
14.8.2 FCHannel:PORT<Pt>:FOLLow:THResholds . . . 799
14.8.3 FCHannel:PORT<Pt>:FOLLow:FRAMe . . . 799
14.8.4 FCHannel:PORT<Pt>:FOLLow:STReam . . . 799
14.9 Stimuli . . . 800
14.9.1 FCHannel:PORT<Pt>:STIMuli:ALARm . . . 800
14.9.2 FCHannel:PORT<Pt>:STIMuli:ERRor . . . 800
14.9.3 FCHannel:PORT<Pt>:STIMuli:EINSertion . . . 801
14.9.4 FCHannel:PORT<Pt>:STIMuli:EBLength . . . 801
14.9.5 FCHannel:PORT<Pt>:STIMuli:FOFFset . . . 802
14.10 Results . . . 803
14.10.1 FCHannel:PORT<Pt>:IFETch? . . . 803
14.10.2 FCHannel:PORT<Pt>:TFETch? . . . 804
14.11 Status . . . 805
14.11.1 FCHannel:STATus:PORT<Pt>:AESummary[:EVENt]? . . . 805
14.11.2 FCHannel:STATus:PORT<Pt>:AESummary:CONDition? . . . 805
14.11.3 FCHannel:STATus:PORT<Pt>:ALARm[:EVENt]? . . . 805
14.11.4 FCHannel:STATus:PORT<Pt>:ALARm:CONDition? . . . 806

14.11.5 FCHannel:STATus:PORT<Pt>:ERRor[:EVENt]? . . . 806
14.11.6 FCHannel:STATus:PORT<Pt>:ERRor:CONDition? . . . 806
14.11.7 FCHannel:STATus:PORT<Pt>:SID? . . . 806
14.11.8 FCHannel:STATus:PORT<Pt>:DID? . . . 807
14.11.9 FCHannel:STATus:PORT<Pt>:PDEViation? . . . 807
14.12 Performance Test . . . 808
14.12.1 FCHannel:PERF:STARt . . . 808
14.12.2 FCHannel:PERF:STOP . . . 808
14.12.3 FCHannel:PERF:SETup:GENeral:MODE . . . 808
14.12.4 FCHannel:PERF:SETup:GENeral:TSELection . . . 808
14.12.5 FCHannel:PERF:SETup:GENeral:TCLayer . . . 809
14.12.6 FCHannel:PERF:SETup:GENeral:TTYPe . . . 809
14.12.7 FCHannel:PERF:SETup:FSIZe:MODE . . . 810
14.12.8 FCHannel:PERF:SETup:FSIZe:USER . . . 810
14.12.9 FCHannel:PERF:SETup:FSIZe:JUMBo . . . 810
14.12.10 FCHannel:PERF:SETup:FSIZe:BEGin . . . 811
14.12.11 FCHannel:PERF:SETup:FSIZe:END . . . 811
14.12.12 FCHannel:PERF:SETup:FSIZe:STEP . . . 811
14.12.13 FCHannel:PERF:SETup:FSIZe:CONStant . . . 812
14.12.14 FCHannel:PERF:SETup:PORT<Pt>:THRoughput:LLoad:MAXimum . . . 812
14.12.15 FCHannel:PERF:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch :MODE . . . 813
14.12.16 FCHannel:PERF:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch :RESolution . . . 813
14.12.17 FCHannel:PERF:SETup:PORT<Pt>:THRoughput:DURation:STEP . . . 813
14.13 Performance Test - Traffic Profile . . . 815
14.13.1 FCHannel:PERF:SETup:PORT<Pt>:PROFile:LLoad:MINimum . . . 815
14.13.2 FCHannel:PERF:SETup:PORT<Pt>:PROFile:LLoad:MAXimum . . . 815
14.13.3 FCHannel:PERF:SETup:PORT<Pt>:PROFile:LLoad:STEP . . . 816
14.13.4 FCHannel:PERF:SETup:PORT<Pt>:PROFile:DURation:STEP . . . 816
14.14 Performance Test - Latency . . . 817
14.14.1 FCHannel:PERF:SETup:PORT<Pt>:LATency:DURation:STEP . . . 817
14.15 Performance Test - Burst . . . 818
14.15.1 FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:MODE . . . 818
14.15.2 FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:BEGin . . . 818
14.15.3 FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:END . . . 818
14.15.4 FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:STEP . . . 819
14.15.5 FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:MODE . . . 819
14.15.6 FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:RESolution . . . 820
14.16 Performance Test - Credit . . . 821
14.16.1 FCHannel:PERF:SETup:PORT<Pt>:CREDit:MODE . . . 821
14.16.2 FCHannel:PERF:SETup:PORT<Pt>:CREDit:BEGin . . . 821
14.16.3 FCHannel:PERF:SETup:PORT<Pt>:CREDit:END . . . 821
14.16.4 FCHannel:PERF:SETup:PORT<Pt>:CREDit:STEP . . . 822
14.16.5 FCHannel:PERF:SETup:PORT<Pt>:CREDit:ASEarch:MODE . . . 822
14.16.6 FCHannel:PERF:SETup:PORT<Pt>:CREDit:ASEarch:RESolution . . . 822
14.16.7 FCHannel:PERF:SETup:PORT<Pt>:CREDit:DURation:STEP . . . 823
14.17 Performance Test - Result . . . 824
14.17.1 FCHannel:PERF:RESult:TEST . . . 824
14.17.2 FCHannel:PERF:RESult:PORT<Pt>:NSTep? . . . 824
14.17.3 FCHannel:PERF:RESult:PORT<Pt>:SELect . . . 824
14.17.4 FCHannel:PERF:RESult:PORT<Pt>:FETCh? . . . 825

15 OTN **827**
15.1 Transmitter . . . 827
15.1.1 OTN:TX<Pt>:INTerface . . . 827
15.1.2 OTN:TX<Pt>[:ENABled] . . . 827
15.1.3 OTN:TX<Pt>:TIMing . . . 828
15.1.4 OTN:TX<Pt>:MAPPing:CSIGnal . . . 828

15.1.5 OTN:TX<Pt>:PATTern . . . 830
15.1.6 OTN:TX<Pt>:PINVersion . . . 830
15.1.7 OTN:TX<Pt>:UP32 . . . 830
15.1.8 OTN:TX<Pt>:UP2K . . . 831
15.1.9 OTN:TX<Pt>:MAPPing:OUTPutsignal . . . 831
15.1.10 OTN:TX<Pt>:FEC . . . 832
15.1.11 OTN:TX<Pt>:MAPPing:MULTiplexing<stage> . . . 832
15.1.12 OTN:TX<Pt>:MAPPing:LANE . . . 833
15.1.13 OTN:TX<Pt>:COPY . . . 833
15.1.14 OTN:TX<Pt>:CSF . . . 834
15.1.15 OTN:TX<Pt>:FOLLow . . . 835
15.1.16 OTN:OH:TX<Pt>:OTN:OH . . . 835
15.1.17 OTN:OH:TX<Pt>:OTN:OH:DEFault . . . 836
15.1.18 OTN:TX<Pt>:TP . . . 836
15.1.19 OTN:TX<Pt>:TS . . . 838
15.1.20 OTN:TX<Pt>:ODUFlex:TS . . . 839
15.1.21 OTN:TX<Pt>:ALLocated . . . 840
15.1.22 OTN:TX<Pt>:GFP:PTI . . . 841
15.1.23 OTN:TX<Pt>:GFP:PFI . . . 841
15.1.24 OTN:TX<Pt>:GFP:EXI . . . 841
15.1.25 OTN:TX<Pt>:GFP:UPI . . . 842
15.1.26 OTN:TX<Pt>:GFP:EXTHeader:CID . . . 842
15.1.27 OTN:TX<Pt>:GFP:EXTHeader:SPARe . . . 842
15.1.28 OTN:TX<Pt>:GFP:PTIMes . . . 843
15.2 Receiver . . . 844
15.2.1 OTN:RX<Pt>:INTerface . . . 844
15.2.2 OTN:RX<Pt>:MAPPing:CSIGnal . . . 844
15.2.3 OTN:RX<Pt>:PATTern . . . 846
15.2.4 OTN:RX<Pt>:PINVersion . . . 846
15.2.5 OTN:RX<Pt>:UP32 . . . 846
15.2.6 OTN:RX<Pt>:UP2K . . . 847
15.2.7 OTN:RX<Pt>:MAPPing:INPutsignal . . . 847
15.2.8 OTN:RX<Pt>:MAPPing:MULTiplexing<stage> . . . 848
15.2.9 OTN:RX<Pt>:FEC . . . 848
15.2.10 OTN:RX<Pt>:FOLLow . . . 849
15.2.11 OTN:RX<Pt>:TP . . . 849
15.2.12 OTN:RX<Pt>:TS . . . 850
15.2.13 OTN:RX<Pt>:ODUFlex:TS . . . 851
15.2.14 OTN:RX<Pt>:TPTS:DETect . . . 852
15.2.15 OTN:RX<Pt>:MEAS:TTI:DET . . . 853
15.3 Measurement Setup . . . 854
15.3.1 OTN:TX<Pt>:MEAS:TTI:ENCoding . . . 854
15.3.2 OTN:RX<Pt>:MEAS:TCM:ENABle . . . 854
15.3.3 OTN:RX<Pt>:MEASurement:PLM . . . 855
15.3.4 MEASurement:SETup:OTN:RX<Pt>:MSIMdetection<stage>:SETup . . . 855
15.3.5 MEASurement:SETup:OTN:RX<Pt>:MSIMdetection<stage>:MSIM? . . . 856
15.3.6 MEASurement:SETup:PERFormance:OTN:RX<Pt>:PARameter . . . 856
15.3.7 MEASurement:SETup:PERFormance:OTN:RX<Pt>:ALLocation . . . 857
15.3.8 MEASurement:SETup:PERFormance:OTN:TPERiod . . . 857
15.3.9 MEASurement:SETup:PERFormance:OTN:RX<Pt>:OBJectives . . . 858
15.4 Result . . . 859
15.4.1 OTN:RX<Pt>:IFETch? . . . 859
15.5 Status . . . 862
15.5.1 OTN:STATus:RX<Pt>:AESummary[:EVENt]? . . . 862
15.5.2 OTN:STATus:RX<Pt>:AESummary:CONDition? . . . 862
15.5.3 OTN:STATus:RX<Pt>:ALARm<section>[:EVENt]? . . . 863
15.5.4 OTN:STATus:RX<Pt>:ALARm<section>:CONDition? . . . 863
15.5.5 OTN:STATus:RX<Pt>:ERRor<section>[:EVENt]? . . . 865
15.5.6 OTN:STATus:RX<Pt>:ERRor<section>:CONDition? . . . 865

15.5.7 OTN:STATus:RX<Pt>:MSTage:AESummary[:EVENt]? . . . 867
15.5.8 OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition? . . . 867
15.5.9 OTN:STATus:RX<Pt>:OTU:ASUMmary[:EVENt]? . . . 868
15.5.10 OTN:STATus:RX<Pt>:OTU:ASUMmary:CONDition? . . . 868
15.5.11 OTN:STATus:RX<Pt>:OTU:ESUMmary[:EVENt]? . . . 868
15.5.12 OTN:STATus:RX<Pt>:OTU:ESUMmary:CONDition? . . . 869
15.5.13 OTN:STATus:RX<Pt>:OTU:ALARm<section>[:EVENt]? . . . 869
15.5.14 OTN:STATus:RX<Pt>:OTU:ALARm<section>:CONDition? . . . 869
15.5.15 OTN:STATus:RX<Pt>:OTU:ERRor<section>[:EVENt]? . . . 870
15.5.16 OTN:STATus:RX<Pt>:OTU:ERRor<section>:CONDition? . . . 871
15.5.17 OTN:STATus:RX<Pt>:SONE:ASUMmary[:EVENt]? . . . 872
15.5.18 OTN:STATus:RX<Pt>:SONE:ASUMmary:CONDition? . . . 872
15.5.19 OTN:STATus:RX<Pt>:SONE:ESUMmary[:EVENt]? . . . 872
15.5.20 OTN:STATus:RX<Pt>:SONE:ESUMmary:CONDition? . . . 872
15.5.21 OTN:STATus:RX<Pt>:SONE:ALARm<section>[:EVENt]? . . . 873
15.5.22 OTN:STATus:RX<Pt>:SONE:ALARm<section>:CONDition? . . . 873
15.5.23 OTN:STATus:RX<Pt>:SONE:ERRor<section>[:EVENt]? . . . 874
15.5.24 OTN:STATus:RX<Pt>:SONE:ERRor<section>:CONDition? . . . 874
15.5.25 OTN:STATus:RX<Pt>:STWO:ASUMmary[:EVENt]? . . . 875
15.5.26 OTN:STATus:RX<Pt>:STWO:ASUMmary:CONDition? . . . 875
15.5.27 OTN:STATus:RX<Pt>:STWO:ESUMmary[:EVENt]? . . . 875
15.5.28 OTN:STATus:RX<Pt>:STWO:ESUMmary:CONDition? . . . 875
15.5.29 OTN:STATus:RX<Pt>:STWO:ALARm<section>[:EVENt]? . . . 876
15.5.30 OTN:STATus:RX<Pt>:STWO:ALARm<section>:CONDition? . . . 876
15.5.31 OTN:STATus:RX<Pt>:STWO:ERRor<section>[:EVENt]? . . . 876
15.5.32 OTN:STATus:RX<Pt>:STWO:ERRor<section>:CONDition? . . . 876
15.5.33 OTN:STATus:RX<Pt>:STHRee:ASUMmary[:EVENt]? . . . 877
15.5.34 OTN:STATus:RX<Pt>:STHRee:ASUMmary:CONDition? . . . 877
15.5.35 OTN:STATus:RX<Pt>:STHRee:ESUMmary[:EVENt]? . . . 877
15.5.36 OTN:STATus:RX<Pt>:STHRee:ESUMmary:CONDition? . . . 877
15.5.37 OTN:STATus:RX<Pt>:STHRee:ALARm<section>[:EVENt]? . . . 878
15.5.38 OTN:STATus:RX<Pt>:STHRee:ALARm<section>:CONDition? . . . 878
15.5.39 OTN:STATus:RX<Pt>:STHRee:ERRor<section>[:EVENt]? . . . 878
15.5.40 OTN:STATus:RX<Pt>:STHRee:ERRor<section>:CONDition? . . . 878
15.5.41 OTN:STATus:RX<Pt>:CLIent:ASUMmary[:EVENt]? . . . 879
15.5.42 OTN:STATus:RX<Pt>:CLIent:ASUMmary:CONDition? . . . 879
15.5.43 OTN:STATus:RX<Pt>:CLIent:ESUMmary[:EVENt]? . . . 879
15.5.44 OTN:STATus:RX<Pt>:CLIent:ESUMmary:CONDition? . . . 879
15.5.45 OTN:STATus:RX<Pt>:CLIent:ALARm<section>[:EVENt]? . . . 880
15.5.46 OTN:STATus:RX<Pt>:CLIent:ALARm<section>:CONDition? . . . 880
15.5.47 OTN:STATus:RX<Pt>:CLIent:ERRor<section>[:EVENt]? . . . 881
15.5.48 OTN:STATus:RX<Pt>:CLIent:ERRor<section>:CONDition? . . . 881
15.5.49 OTN:STATus:RX<Pt>:PSLevel? . . . 881
15.5.50 OTN:STATus:TX<Pt>:PSLevel? . . . 882
15.5.51 OTN:STATus:RX<Pt>:FREQuency? . . . 882
15.5.52 OTN:STATus:TX<Pt>:FREQuency? . . . 882
15.5.53 OTN:STATus:RX<Pt>:CAPTure:OH? . . . 883
15.6 Stimuli . . . 884
15.6.1 OTN:STIMuli:TX<Pt>:TYPE . . . 884
15.6.2 OTN:STIMuli:TX<Pt>:AEINsert:LEVel . . . 884
15.6.3 OTN:STIMuli:TX<Pt>:AEINsert:TYPE . . . 885
15.6.4 OTN:STIMuli:TX<Pt>:AEINsert:INSert . . . 886
15.6.5 OTN:STIMuli:TX<Pt>:AEINsert:BURSt . . . 886
15.6.6 OTN:STIMuli:TX<Pt>:AEINsert:AERRor . . . 886
15.6.7 OTN:STIMuli:TX<Pt>:AEINsert:NORMal . . . 887
15.6.8 OTN:STIMuli:TX<Pt>:AEINsert:RATE . . . 887
15.6.9 OTN:STIMuli:TX<Pt>:JUSTification:LEVel . . . 888
15.6.10 OTN:STIMuli:TX<Pt>:JUSTification:BURSt . . . 888
15.6.11 OTN:STIMuli:TX<Pt>:JUSTification:MOVement . . . 888

15.6.12 OTN:STIMuli:TX<Pt>:JMOVe . . . 889
15.6.13 OTN:STIMuli:TX<Pt>:FREQuency:OFFSet . . . 889
15.6.14 OTN:STIMuli:TX<Pt>:PAYLoad:OFFSet<stage> . . . 889
15.6.15 OTN:STIMuli:TX<Pt>:GMP:AEINsert:LEVel . . . 890
15.6.16 OTN:STIMuli:TX<Pt>:GMP:AEINsert:TYPE . . . 890
15.6.17 OTN:STIMuli:TX<Pt>:GMP:AEINsert:INSert . . . 891
15.6.18 OTN:STIMuli:TX<Pt>:GMP:AEINsert:EBITs . . . 891
15.6.19 OTN:STIMuli:TX<Pt>:GFP:AEINsert:TYPE . . . 892
15.6.20 OTN:STIMuli:TX<Pt>:GFP:AEINsert:INSert . . . 892
15.6.21 OTN:STIMuli:TX<Pt>:GFP:AEINsert:EBITs . . . 893
15.6.22 OTN:STIMuli:TX<Pt>:GFP:AEINsert:BURSt . . . 893
15.6.23 OTN:STIMuli:TX<Pt>:GFP:AEINsert:CMF:INTerval . . . 893
15.6.24 OTN:STIMuli:TX<Pt>:FEC:AEINsert:TYPE . . . 894
15.6.25 OTN:STIMuli:TX<Pt>:FEC:AEINsert:INSert . . . 894
15.6.26 OTN:STIMuli:TX<Pt>:FEC:AEINsert:RMANtissa . . . 894
15.6.27 OTN:STIMuli:TX<Pt>:FEC:AEINsert:REXPnent . . . 895
15.6.28 OTN:STIMuli:TX<Pt>:FEC:AEINsert:EFAS . . . 895
15.6.29 OTN:STIMuli:TX<Pt>:FEC:AEINsert:SUBRow . . . 896
15.6.30 OTN:STIMuli:TX<Pt>:LLD:AEINsert:TYPE . . . 896
15.6.31 OTN:STIMuli:TX<Pt>:LLD:AEINsert:ALARm . . . 896
15.6.32 OTN:STIMuli:TX<Pt>:LLD:AEINsert:ERRor . . . 897
15.6.33 OTN:STIMuli:TX<Pt>:LLD:AEINsert:INSert . . . 897
15.6.34 OTN:STIMuli:TX<Pt>:LLD:AEINsert:BURSt . . . 897
15.6.35 OTN:STIMuli:TX<Pt>:LLD:AEINsert:AERRor . . . 898
15.6.36 OTN:STIMuli:TX<Pt>:LLD:AEINsert:NORMal . . . 898
15.6.37 OTN:STIMuli:TX<Pt>:LLD:AEINsert:LANE . . . 898
15.6.38 OTN:STIMuli:TX<Pt>:LLD:SKEW:LANE . . . 899
15.6.39 OTN:STIMuli:TX<Pt>:LLD:SKEW:BIT . . . 899
15.6.40 OTN:STIMuli:TX<Pt>:LLD:SKEW:TYPE . . . 900
15.7 Tributary Scan . . . 901
15.7.1 OTN:TSCan:STARt . . . 901
15.7.2 OTN:TSCan:STOP . . . 901
15.7.3 OTN:TSCan:RX<Pt>:ODU2:NUMBer? . . . 901
15.7.4 OTN:TSCan:RX<Pt>:ODU2:GET? . . . 901
15.7.5 OTN:TSCan:RX<Pt>:ODU2:DETailed? . . . 901
15.7.6 OTN:TSCan:RX<Pt>:ODU1:NUMBer? . . . 902
15.7.7 OTN:TSCan:RX<Pt>:ODU1:GET? . . . 902
15.7.8 OTN:TSCan:RX<Pt>:ODU1:SCANning . . . 902
15.7.9 OTN:TSCan:RX<Pt>:ODU1:DETailed? . . . 902
15.7.10 OTN:TSCan:RX<Pt>:ODU0:NUMBer? . . . 903
15.7.11 OTN:TSCan:RX<Pt>:ODU0:GET? . . . 903
15.7.12 OTN:TSCan:RX<Pt>:ODU0:DETailed? . . . 903
15.8 APS . . . 904
15.8.1 OTN:APS:RX<Pt>:BEVent . . . 904
15.8.2 OTN:APS:RX<Pt>:EEVent . . . 905
15.8.3 OTN:APS:RX<Pt>:PERiod . . . 905
15.8.4 OTN:APS:RX<Pt>:MLIMit . . . 906
15.8.5 OTN:APS:STARt . . . 906
15.8.6 OTN:APS:STOP . . . 906
15.8.7 OTN:APS:RX<Pt>:NUMBer? . . . 906
15.8.8 OTN:APS:RX<Pt>:MTIMe? . . . 907
15.8.9 OTN:APS:RX<Pt>:LTIMe? . . . 907
15.8.10 OTN:APS:RX<Pt>:ATIMe? . . . 907
15.8.11 OTN:APS:RX<Pt>:CTIMe? . . . 907
15.9 RTD . . . 908
15.9.1 OTN:RTD:RX<Pt>:MLIMit . . . 908
15.9.2 OTN:RTD:RX<Pt>:NUMBer? . . . 908
15.9.3 OTN:RTD:RX<Pt>:ATIMe? . . . 908
15.9.4 OTN:RTD:RX<Pt>:MTIMe? . . . 908

15.9.5 OTN:RTD:RX<Pt>:LTIMe? . . . 909
15.9.6 OTN:RTD:RX<Pt>:CTIMe? . . . 909

16 Pluggable Module **911**
16.1 Status . . . 911
16.1.1 PMODule:STATus:PORT<Pt>:PRESent? . . . 911
16.1.2 PMODule:STATus:PORT<Pt>:VENDor? . . . 911
16.1.3 PMODule:STATus:PORT<Pt>:WAVelength? . . . 911
16.1.4 PMODule:STATus:PORT<Pt>:BRATe? . . . 911
16.1.5 PMODule:STATus:PORT<Pt>:COMPliance? . . . 912
16.1.6 PMODule:STATus:PORT<Pt>:PNUMber? . . . 912
16.1.7 PMODule:STATus:PORT<Pt>:REVision? . . . 912
16.1.8 PMODule:STATus:PORT<Pt>:SNUMber? . . . 912
16.1.9 PMODule:STATus:PORT<Pt>:DATE? . . . 912
16.1.10 PMODule:STATus:PORT<Pt>:LOT? . . . 912
16.1.11 PMODule:STATus:PORT<Pt>:TPOWer? . . . 913
16.1.12 PMODule:STATus:PORT<Pt>:RPOWer? . . . 913
16.1.13 PMODule:STATus:PORT<Pt>:STATus? . . . 913
16.1.14 PMODule:STATus:PORT<Pt>:GALarm? . . . 913
16.1.15 PMODule:STATus:PORT<Pt>:PALarm? . . . 913
16.1.16 PMODule:STATus:PORT<Pt>:AESummary[:EVENt]? . . . 914
16.1.17 PMODule:STATus:PORT<Pt>:AESummary:CONDition? . . . 914
16.1.18 PMODule:STATus:PORT<Pt>:ALARm[:EVENt]? . . . 914
16.1.19 PMODule:STATus:PORT<Pt>:ALARm:CONDition? . . . 914
16.2 MDIO . . . 915
16.2.1 PMODule:PORT<Pt>:MDIO:WRITe . . . 915
16.2.2 PMODule:PORT<Pt>:MDIO:READ? . . . 915
16.3 I2C . . . 915
16.3.1 PMODule:PORT<Pt>:I2C:WRITe . . . 915
16.3.2 PMODule:PORT<Pt>:I2C:READ? . . . 916
16.4 CFP2-QSFP28 Adaptor . . . 916
16.4.1 PMODule:PORT<Pt>:MDIO:ADPT:WRITe . . . 916
16.4.2 PMODule:PORT<Pt>:MDIO:ADPT:READ? . . . 916
16.5 Tx reference clock output . . . 916
16.5.1 PMODule:TOUTput:PORT<Pt>[:ENABle] . . . 916
16.5.2 PMODule:TOUTput:PORT<Pt>:FREQuency . . . 917
16.5.3 PMODule:TOUTput:PORT<Pt>:SYNC:FREQuency . . . 917
16.6 tunable SFP . . . 918
16.6.1 PMODule:PORT<Pt>:TSFP:CHANnel . . . 918
16.6.2 PMODule:PORT<Pt>:TSFP:CHANnel? . . . 918
16.6.3 PMODule:PORT<Pt>:TSFP:WLENgth . . . 918
16.6.4 PMODule:PORT<Pt>:TSFP:WLENgth? . . . 918
16.7 Transceiver settings - CFP2-QSFP28 Adaptor . . . 919
16.7.1 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:ATTN:TRACking . . . 919
16.7.2 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:ATTN . . . 919
16.7.3 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:PRE:TRACking . . . 919
16.7.4 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:PRE . . . 920
16.7.5 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:POST:TRACking . . . 920
16.7.6 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:POST . . . 920
16.7.7 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:EAUTo . . . 921
16.7.8 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:EQUalizer:TRACking . . . 921
16.7.9 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:EQUalizer . . . 921
16.7.10 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:RESTore . . . 922
16.7.11 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:AUTotune . . . 922
16.7.12 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:DEFault . . . 922
16.8 Transceiver settings - CFP2 . . . 923
16.8.1 PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:ATTN:TRACking . . . 923
16.8.2 PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:ATTN . . . 923
16.8.3 PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:PRE:TRACking . . . 923

16.8.4 PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:PRE . . . 924
16.8.5 PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:POST:TRACking . . . 924
16.8.6 PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:POST . . . 924
16.8.7 PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:EAUTo . . . 925
16.8.8 PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:EQUalizer:TRACking . . . 925
16.8.9 PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:EQUalizer . . . 925
16.8.10 PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:RESTore . . . 926
16.8.11 PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:AUTotune . . . 926
16.8.12 PMODule:PORT<Pt>:TRANsceiver:CFP2:DEFault . . . 926
16.9 Transceiver settings - CXP . . . 927
16.9.1 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:VOD:TRACking . . . 927
16.9.2 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:VOD . . . 927
16.9.3 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:FIRSt:TRACking . . . 927
16.9.4 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:FIRSt . . . 928
16.9.5 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:PRE:TRACking . . . 928
16.9.6 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:PRE . . . 928
16.9.7 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:SECond:TRACking . . . 929
16.9.8 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:SECond . . . 929
16.9.9 PMODule:PORT<Pt>:TRANsceiver:CXP:RX:CONTrol:TRACking . . . 929
16.9.10 PMODule:PORT<Pt>:TRANsceiver:CXP:RX:CONTrol . . . 930
16.9.11 PMODule:PORT<Pt>:TRANsceiver:CXP:RX:DCGain:TRACking . . . 930
16.9.12 PMODule:PORT<Pt>:TRANsceiver:CXP:RX:DCGain . . . 931
16.9.13 PMODule:PORT<Pt>:TRANsceiver:CXP:DEFault . . . 931
16.10 Transceiver settings - CFP . . . 932
16.10.1 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:VOD:TRACking . . . 932
16.10.2 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:VOD . . . 932
16.10.3 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:FIRSt:TRACking . . . 932
16.10.4 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:FIRSt . . . 933
16.10.5 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:PRE:TRACking . . . 933
16.10.6 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:PRE . . . 933
16.10.7 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:SECond:TRACking . . . 934
16.10.8 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:SECond . . . 934
16.10.9 PMODule:PORT<Pt>:TRANsceiver:CFP:RX:CONTrol:TRACking . . . 935
16.10.10 PMODule:PORT<Pt>:TRANsceiver:CFP:RX:CONTrol . . . 935
16.10.11 PMODule:PORT<Pt>:TRANsceiver:CFP:RX:DCGain:TRACking . . . 935
16.10.12 PMODule:PORT<Pt>:TRANsceiver:CFP:RX:DCGain . . . 936
16.10.13 PMODule:PORT<Pt>:TRANsceiver:CFP:DEFault . . . 936
16.11 Transceiver settings - QSFP . . . 937
16.11.1 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:VOD:TRACking . . . 937
16.11.2 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:VOD . . . 937
16.11.3 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:FIRSt:TRACking . . . 937
16.11.4 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:FIRSt . . . 938
16.11.5 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:PRE:TRACking . . . 938
16.11.6 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:PRE . . . 938
16.11.7 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:SECond:TRACking . . . 939
16.11.8 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:SECond . . . 939
16.11.9 PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:CONTrol:TRACking . . . 939
16.11.10 PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:CONTrol . . . 940
16.11.11 PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DCGain:TRACking . . . 940
16.11.12 PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DCGain . . . 940
16.11.13 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:POST:TRACking . . . 941
16.11.14 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:POST . . . 941
16.11.15 PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DFE:TRACking . . . 941
16.11.16 PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DFE . . . 942
16.11.17 PMODule:PORT<Pt>:TRANsceiver:QSFP:DEFault . . . 942
16.12 Transceiver settings - SFP28 . . . 943
16.12.1 PMODule:PORT<Pt>:TRANsceiver:SFP28:TX:VOD . . . 943
16.12.2 PMODule:PORT<Pt>:TRANsceiver:SFP28:TX:PRE . . . 943
16.12.3 PMODule:PORT<Pt>:TRANsceiver:SFP28:TX:POST . . . 943

16.12.4 PMODule:PORT<Pt>:TRANsceiver:SFP28:RX:DFE . . . 944
16.12.5 PMODule:PORT<Pt>:TRANsceiver:SFP28:DEFault . . . 944
16.13 Transceiver settings - CFP4 . . . 945
16.13.1 PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:VOD:TRACking . . . 945
16.13.2 PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:VOD . . . 945
16.13.3 PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:PRE:TRACking . . . 945
16.13.4 PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:PRE . . . 946
16.13.5 PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:POST:TRACking . . . 946
16.13.6 PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:POST . . . 947
16.13.7 PMODule:PORT<Pt>:TRANsceiver:CFP4:RX:DFE:TRACking . . . 947
16.13.8 PMODule:PORT<Pt>:TRANsceiver:CFP4:RX:DFE . . . 947
16.13.9 PMODule:PORT<Pt>:TRANsceiver:CFP4:DEFault . . . 948
16.14 Transceiver settings - QSFP28 . . . 949
16.14.1 PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:VOD:TRACking . . . 949
16.14.2 PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:VOD . . . 949
16.14.3 PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:PRE:TRACking . . . 949
16.14.4 PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:PRE . . . 950
16.14.5 PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:POST:TRACking . . . 950
16.14.6 PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:POST . . . 951
16.14.7 PMODule:PORT<Pt>:TRANsceiver:QSFP28:RX:DFE:TRACking . . . 951
16.14.8 PMODule:PORT<Pt>:TRANsceiver:QSFP28:RX:DFE . . . 951
16.14.9 PMODule:PORT<Pt>:TRANsceiver:QSFP28:DEFault . . . 952

17 Common Application Settings 953
17.1 RTD . . . 953
17.1.1 RTD:MODE . . . 953
17.1.2 RTD:PERiod . . . 953
17.1.3 RTD:IFMData . . . 954
17.1.4 RTD:PDH:RX<Pt>:MLIMit . . . 954
17.2 Hierarchy . . . 955
17.2.1 HIERarchy:PORT<Pt>:OTN:TX . . . 955
17.2.2 HIERarchy:PORT<Pt>:OTN:RX . . . 955
17.2.3 HIERarchy:PORT<Pt>:SDH:TX . . . 955
17.2.4 HIERarchy:PORT<Pt>:SDH:RX . . . 956
17.2.5 HIERarchy:PORT<Pt>:PDH:TX . . . 956
17.2.6 HIERarchy:PORT<Pt>:PDH:RX . . . 956

18 Measurement 957
18.1 Application, Start and Stop . . . 958
18.1.1 MEASurement:APPLication? . . . 958
18.1.2 MEASurement:STARt . . . 959
18.1.3 MEASurement:STOP . . . 959
18.2 Setup . . . 959
18.2.1 MEASurement:SETup:PORT<Pt>:TERMinology . . . 959
18.2.2 MEASurement:SETup:SELect . . . 960
18.2.3 MEASurement:SETup:ILENgth . . . 960
18.2.4 MEASurement:SETup:STARt . . . 961
18.2.5 MEASurement:SETup:STAT . . . 961
18.2.6 MEASurement:SETup:STOP . . . 962
18.2.7 MEASurement:SETup:SPAT . . . 962
18.2.8 MEASurement:SETup:SDURation . . . 963
18.2.9 MEASurement:SETup:MALLocation . . . 963
18.2.10 MEASurement:SETup:PERFormance:TMBPs:PARameter . . . 964
18.2.11 MEASurement:SETup:PERFormance:T1:PARameter . . . 964
18.2.12 MEASurement:SETup:PERFormance:E3:PARameter . . . 964
18.2.13 MEASurement:SETup:PERFormance:T3:PARameter . . . 965
18.2.14 MEASurement:SETup:PERFormance:E4:PARameter . . . 965
18.2.15 MEASurement:SETup:PERFormance:SDH:PARameter . . . 965
18.2.16 MEASurement:SETup:PERFormance:SDH:MUX . . . 966
18.2.17 MEASurement:SETup:PERFormance:SDH:VC4 . . . 966

18.2.18 MEASurement:SETup:PERFormance:SDH:VC3 . . . 966
18.2.19 MEASurement:SETup:PERFormance:SDH:VC12 . . . 967
18.2.20 MEASurement:SETup:PERFormance:SDH:TPERiod . . . 967
18.2.21 MEASurement:SETup:PERFormance:SONet:PARameter . . . 968
18.2.22 MEASurement:SETup:PERFormance:SONet:MUX . . . 968
18.2.23 MEASurement:SETup:PERFormance:SONet:STS3 . . . 968
18.2.24 MEASurement:SETup:PERFormance:SONet:STS1 . . . 969
18.2.25 MEASurement:SETup:PERFormance:SONet:VT2 . . . 969
18.2.26 MEASurement:SETup:PERFormance:SONet:TPERiod . . . 969
18.2.27 MEASurement:SETup:PERFormance:WAN:PARameter . . . 970
18.2.28 MEASurement:SETup:EVALuation:RX<Pt>[:ENABle] . . . 970
18.2.29 MEASurement:SETup:EVALuation:RX<Pt>:INTerface . . . 970
18.2.30 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:ITEM . . . 971
18.2.31 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:ITEM:ALARm . . . 971
18.2.32 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:ITEM:ERRor . . . 972
18.2.33 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:TYPE . . . 972
18.2.34 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:CNT:PASS . . . 973
18.2.35 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:CNT:FAIL . . . 973
18.2.36 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:RATio:PASS . . . 973
18.2.37 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:RATio:FAIL . . . 974
18.2.38 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:ITEM . . . 974
18.2.39 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:TYPE . . . 975
18.2.40 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:CNT:PASS . . . 975
18.2.41 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:CNT:FAIL . . . 975
18.2.42 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:RATio:PASS . . . 976
18.2.43 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:RATio:FAIL . . . 976
18.2.44 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:PCT:PASS . . . 976
18.2.45 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:PCT:FAIL . . . 977
18.2.46 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:HREFerence . . . 977
18.2.47 MEASurement:SETup:EVALuation:E3:RX<Pt>:ITEM . . . 978
18.2.48 MEASurement:SETup:EVALuation:E3:RX<Pt>:TYPE . . . 978
18.2.49 MEASurement:SETup:EVALuation:E3:RX<Pt>:CNT:PASS . . . 979
18.2.50 MEASurement:SETup:EVALuation:E3:RX<Pt>:CNT:FAIL . . . 979
18.2.51 MEASurement:SETup:EVALuation:E3:RX<Pt>:RATio:PASS . . . 979
18.2.52 MEASurement:SETup:EVALuation:E3:RX<Pt>:RATio:FAIL . . . 980
18.2.53 MEASurement:SETup:EVALuation:E4:RX<Pt>:ITEM . . . 980
18.2.54 MEASurement:SETup:EVALuation:E4:RX<Pt>:TYPE . . . 981
18.2.55 MEASurement:SETup:EVALuation:E4:RX<Pt>:CNT:PASS . . . 981
18.2.56 MEASurement:SETup:EVALuation:E4:RX<Pt>:CNT:FAIL . . . 981
18.2.57 MEASurement:SETup:EVALuation:E4:RX<Pt>:RATio:PASS . . . 982
18.2.58 MEASurement:SETup:EVALuation:E4:RX<Pt>:RATio:FAIL . . . 982
18.2.59 MEASurement:SETup:EVALuation:SDH:RX<Pt>:ITEM . . . 982
18.2.60 MEASurement:SETup:EVALuation:SDH:RX<Pt>:ITEM:ALARm . . . 983
18.2.61 MEASurement:SETup:EVALuation:SDH:RX<Pt>:ITEM:ERRor . . . 984
18.2.62 MEASurement:SETup:EVALuation:SDH:RX<Pt>:TYPE . . . 984
18.2.63 MEASurement:SETup:EVALuation:SDH:RX<Pt>:CNT:PASS . . . 985
18.2.64 MEASurement:SETup:EVALuation:SDH:RX<Pt>:CNT:FAIL . . . 985
18.2.65 MEASurement:SETup:EVALuation:SDH:RX<Pt>:RATio:PASS . . . 985
18.2.66 MEASurement:SETup:EVALuation:SDH:RX<Pt>:RATio:FAIL . . . 986
18.2.67 MEASurement:SETup:EVALuation:SONet:RX<Pt>:ITEM . . . 986
18.2.68 MEASurement:SETup:EVALuation:SONet:RX<Pt>:ITEM:ALARm . . . 987
18.2.69 MEASurement:SETup:EVALuation:SONet:RX<Pt>:ITEM:ERRor . . . 988
18.2.70 MEASurement:SETup:EVALuation:SONet:RX<Pt>:TYPE . . . 988
18.2.71 MEASurement:SETup:EVALuation:SONet:RX<Pt>:CNT:PASS . . . 989
18.2.72 MEASurement:SETup:EVALuation:SONet:RX<Pt>:CNT:FAIL . . . 989
18.2.73 MEASurement:SETup:EVALuation:SONet:RX<Pt>:RATio:PASS . . . 989
18.2.74 MEASurement:SETup:EVALuation:SONet:RX<Pt>:RATio:FAIL . . . 990
18.2.75 MEASurement:SETup:EVALuation:OTN:RX<Pt>:ITEM . . . 990
18.2.76 MEASurement:SETup:EVALuation:OTN:RX<Pt>:ITEM:ALARm . . . 990

18.2.77 MEASurement:SETup:EVALuation:OTN:RX<Pt>:ITEM:ERRor . . . 992
18.2.78 MEASurement:SETup:EVALuation:OTN:RX<Pt>:TYPE . . . 993
18.2.79 MEASurement:SETup:EVALuation:OTN:RX<Pt>:CNT:PASS . . . 993
18.2.80 MEASurement:SETup:EVALuation:OTN:RX<Pt>:CNT:FAIL . . . 993
18.2.81 MEASurement:SETup:EVALuation:OTN:RX<Pt>:RATio:PASS . . . 994
18.2.82 MEASurement:SETup:EVALuation:OTN:RX<Pt>:RATio:FAIL . . . 994
18.2.83 MEASurement:SETup:EVALuation:BER:OBAMeasuring . . . 994
18.2.84 MEASurement:SETup:EVALuation:BER:IAFFilter . . . 995
18.2.85 MEASurement:SETup:EVALuation:BER:CLFrames . . . 995
18.2.86 MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds [:ENABle] . . . 995
18.2.87 MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds:MODE . . . 996
18.2.88 MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds:COUNt . . . 996
18.2.89 MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds:RATio . . . 996
18.2.90 MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds :PERCent . . . 997
18.2.91 MEASurement:SETup:POINtermovement:SDH:MODE . . . 997
18.2.92 MEASurement:SETup:POINtermovement:SONet:MODE . . . 997
18.3 Information . . . 998
18.3.1 MEASurement:INFO:TBEGin? . . . 998
18.3.2 MEASurement:INFO:TEND? . . . 998
18.3.3 MEASurement:INFO:MDURation? . . . 998
18.3.4 MEASurement:INFO:STATus<Pt>? . . . 998
18.3.5 MEASurement:INFO:IMIN? . . . 999
18.3.6 MEASurement:INFO:IMAX? . . . 999
18.3.7 MEASurement:INFO:IFIRst? . . . 999
18.3.8 MEASurement:INFO:ILASt? . . . 999
18.4 Event Log . . . 999
18.4.1 MEASurement:ELOG:MINimum? . . . 999
18.4.2 MEASurement:ELOG:MAXimum? . . . 1000
18.4.3 MEASurement:ELOG:FETCh? . . . 1000
18.4.4 MEASurement:ELOG:FETCh:NEXT? . . . 1000
18.4.5 MEASurement:ELOG:EXPort . . . 1000
18.5 Result . . . 1001
18.5.1 MEASurement:RESult:SUMMary? . . . 1001
18.5.2 MEASurement:RESult:SUMMary:STATistics:PORT<Pt>? . . . 1001

A Example Scripts **1003**
A.1 Hints . . . 1003
A.2 2 Mbps BERT . . . 1004
A.3 2 Mbps Status & Stimuli . . . 1005
A.4 E3 BERT . . . 1006
A.5 E4 BERT . . . 1007
A.6 SDH BERT . . . 1008
A.7 Ethernet BERT . . . 1009
A.8 Ethernet Discovery . . . 1010
A.9 Fibre Channel BERT . . . 1011
A.10 OTN BERT . . . 1012
A.11 CPRI BERT . . . 1013

List of Figures

1.1 System setup using Ethernet . . . 51

1.2 Connector panel . . . 51
1.3 Configure TCP **Port** for Remote Control . . . 52
1.4 Program message structure . . . 53
1.5 Program message unit . . . 53
1.6 Response message structure . . . 57
1.7 Configuration of Status Registers in the Network Master ⊕ means logical OR. . . . 60
1.8 The register model for the Status registers. . . . 61
1.9 Configuration of Status Byte Register and Service Request Enable Register . . . 62
1.10 Configuration of Standard Event Status Register and Standard Event Status Enable Register . . . 63
1.11 Configuration of Questionable Status Register . . . 65
1.12 Configuration of Operation Status Register . . . 66
1.13 Configuration of Port Status Register . . . 67
1.14 The structure for the Port Interface Status Register . . . 68
1.15 The generic structure for the AESummary register of interfaces . . . 68
1.16 Enable **Implicit CR in every LF** in PuTTY . . . 71
1.17 **IP Address** on the instrument . . . 71
1.18 Specify the destination and **Open** the connection in PuTTY . . . 72
1.19 Connection established with PuTTY . . . 72
1.20 Data bit . . . 73
1.21 Correspondence between physical and logical port numbers (Three ports) . . . 73
1.22 Correspondence between physical and logical port numbers (One port) . . . 74
1.23 Correspondence between physical and logical port numbers (Two ports) . . . 74

2.1 `INST:STAR` automatically *connects* and selects started application server. . . . 90
2.2 Original client session has to *disconnect*, before another client session can *connect*. . . . 90
2.3 When connected to multiple application servers, client session *selects* to which application server the application specific commands are dispatched. . . . 90

Chapter 1

Overview

The Network Master command based remote control functions support the built-in Ethernet service interface. Software specifications are in conformity with the IEEE488.2 standard based on SCPI version 1999 (Standard Commands for Programmable Instruments). Network Master becomes an automated measurement instrument when it is connected to an external controller.

Figure 1.1: System setup using Ethernet

1.1 Ethernet Based Remote Control

1.1.1 Connecting Cable

To use remote control via the Ethernet service interface, connect an Ethernet cable to the Ethernet connector next to the power socket.

Figure 1.2: Connector panel

1.1.2 Connect to an Instrument

Remote commands are exchanged with the Network Master over a raw TCP/IP connection. Connect TCP port 56001 for program/response communication, see section 1.5 Controller Example.

1.1.3 Ethernet Remote Control Settings

Port Number

To change a TCP port number (for a valid range, see Table 1.1) type the number in the **TCP Port** field (see Figure 1.3).

Figure 1.3: Configure TCP **Port** for Remote Control

Table 1.1: Allowable TCP port range

Setup item	Description	Allowable range
Port Number	TCP Port Number	1024 to 65535 (default: 56001)

1.1.4 Communication Buffers

The input- and output streams are buffered. Besides the TCP receive buffer (87380 bytes) and the TCP transmit buffer (16384 bytes), the two streams share a common command/response buffer of 32 entries. Each buffer entry can hold a compound program message of maximum 4 KB or a response message of maximum 64 KB.

Program data transferred as <ARBITRARY BLOCK PROGRAM DATA> does not go through the internal buffer, but is streamed directly from the TCP receive buffer to the internal file system. Similar for response data of type <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>; it is streamed directly from the internal file system to the TCP transmit buffer.

1.2 Program Messages

Program messages are the remote commands sent to Network Master as shown in Figure 1.4.

Figure 1.4: Program message structure

A program message consists of one or more program message units separated with a semicolon (;). Space(s) before or after a semicolon is ignored (space has no meaning). For more information on program message units, see section 1.2.1 Program Message Unit.

When a program message is sent to Network Master, a terminator is appended after it. Network Master receives the program message by detecting the terminator. For a description of the terminator, see section 1.2.4 Program Message Terminator.

The Network Master is able to handle program messages with a maximum length of 4096 characters including the message terminator.

1.2.1 Program Message Unit

A program message unit consists of a program header and a program data, see Figure 1.5.

Figure 1.5: Program message unit

There must be one or more spaces between a program header and a program data. Network Master recognizes the program header and program data using the space(s). One or more spaces before a program header are ignored.

1.2.2 Program Headers

The program header specifies the function of the command message unit sent from the controller to Network Master. There are two types of program headers:

- Program headers for command message units.
- Program headers for query message units. Similar to headers for command message unit, but are always followed by a question mark "?".

The Network Master supports some of the common commands defined in the IEEE488.2 standard. These common commands are special in the way that they are always preceded by an asterisk "*"; e.g. `*IDN?`. All other commands are referred to as "device specific commands". Device specific commands consists of two or more <program mnemonic>'s (hereinafter called "mnemonic") separated with a colon "(:)".

[:]<program mnemonic>[:<program mnemonic>]... e.g. `SYSTem:TIME`

A mnemonic is a character string, which consists of capital and small letters. The capital part of the mnemonic is also referred to as the short form of the mnemonic.

- Long form program header: `INSTrument:STARt`
- Short form program header: `INST:STAR`

The Network Master recognizes a mnemonic even if only the short form is sent. For example, mnemonic `INSTrument` is recognized as a normal mnemonic when `INST` is sent.

In this way, capital and small letters are used for recognizing long and short forms of a mnemonic, The Network Master does not distinguish between capital and small letters when reading the program header. However, the Network Master only accepts the short form or the complete long form of a mnemonic. Hence `INSTru` is **not** a valid mnemonic. The following program headers are all acceptable and assumed to be the same:

- `SYSTEM:POWER:SOURCE?`
- `system:power:source?`
- `SySteM:PoWeR:SoUr?`
- `syst:POW:sour?`

1.2.3 Program Data

Program data is sent following the program header as parameters specified in the command message unit. This operation manual uses the notations given below in Table 1.2 for indicating the program data format. Most of them are defined in the IEEE488.2 standard.

Table 1.2: Acceptable program data

Program data type	Description
<BOOLEAN PROGRAM DATA>	Defined in IEEE488.2 Indicates On/Off, Enable/Disable, or Yes/No. To specify On/Enable state, set {ON|1}. To specify Off/Disable state, set {OFF|0}.
<NUMERIC PROGRAM DATA>	Comprises <DECIMAL NUMERIC PROGRAM DATA> and <NON-DECIMAL NUMERIC PROGRAM DATA> as defined in IEEE488.2 The Network Master accepts both decimal and non-decimal entries for the <NUMERIC PROGRAM DATA>.
<DECIMAL NUMERIC PROGRAM DATA>	Defined in IEEE488.2 Comprises <NR1>,<NR2> and <NR3> decimal values, where <NR1> indicates an integer value. <NR2> indicates a numeric value in fixed point format. <NR3> indicates a numeric value in floating point format. Examples: <NR1>: 123 <NR2>: -123.456 <NR3>: 1.23E-3
<NONDECIMAL NUMERIC PROGRAM DATA>	Defined in IEEE488.2 Comprises <HEXADECIMAL>, <OCTAL> or <BINARY> program data. See below for further details.

<HEXADECIMAL>	Conforms to the hexadecimal format defined in IEEE488.2 as follows:
	`#{H|h}{A|a|B|b|C|c|D|d|E|e|F|f|<digit>}...`
	<digit> is an ASCII character with a value in the range of 0x30 to 0x39 (48 to 57 in decimal), that is, a numeric 0 to 9.
	Examples:
	`#h1234ABCD`
	`#Hfe1a9`

<OCTAL>	Conforms to the octal format defined in IEEE488.2 as follows:
	`#{Q|q}{0|1|2|3|4|5|6|7}...`
	Examples:
	`#q12345670`
	`#Q77`

<BINARY>	Conforms to the binary format defined in IEEE488.2 as follows:
	`#{B|b}{0|1}...`
	Examples:
	`#b10101010`
	`#B110`

<STRING PROGRAM DATA>	Defined in IEEE488.2
	A character string in a pair of single quotation marks (') or double quotation marks (").
	Examples:
	`"Network Master"`
	`'Testing the network'`

continued on next page...

. . . continued from previous page

Program data type	Description
<CHARACTER PROGRAM DATA>	Defined in IEEE488.2 Indicates two or more mnemonics for selections. Like program header mnemonics, <CHARACTER PROGRAM DATA> mnemonics can have a short and a long form. The syntax used in the Network Master additionally allows a digit as the first character of a mnemonic and also allows a dash (-) inside a mnemonic.

1.2.4 Program Message Terminator

A program message terminator indicates the end of the program message. Upon reception of a terminator, the Network Master assumes that the program message is complete and starts processing the message. A terminator must always be added to the end of a program message. For Network Master the program message terminator is:

[<`WHITE SPACE`>]{`NL`} for Ethernet based remote control

<WHITE SPACE> is one or more ASCII characters with a value in the range of 0x00 to 0x09 or 0x0B to 0x20 (0 to 9 or 11 to 32 in decimal). These ranges include the ASCII control characters and space, except NL (newline). Since <WHITE SPACE> includes CR (0x0D) (13 in decimal), {CR}{NL} is also interpreted as a terminator by Network Master in Ethernet based remote control - to keep compatibility with conventional models.

1.2.5 Compound Program Messages

Compound headers are supported by the Network Master. Examples of the use of the compound headers are shown below.
The three program message units:

```
SYSTem:TIME?
SYSTem:DATE?
SYSTem:POWer:SOURce?
```

can be combined in one program message as follows:

```
SYSTem:TIME?; :SYSTem:DATE?; :SYSTem:POWer:SOURce?
```

or just:

```
SYSTem:TIME?; DATE?; POWer:SOURce?
```

(`SYST:` mnemonic can be omitted in the second and third program data units)
For further information on compound headers, see Appendix A of the IEEE488.2 standard.

1.2.6 Sequential Execution

The Network Master processes one program message unit at a time and in the same order in which they are arranged within the program message. The Network Master will not start processing a new program message until the processing of the current program message is finished.

1.3 Response Messages

Response messages are messages sent from a Network Master to a controller as reply to queries, see Figure 1.6.

Figure 1.6: Response message structure

A response message consists of one or more response data separated with a semicolon (;). The response message is terminated with the response message terminator.

1.3.1 Response Data

Response data is a data returned by Network Master as reply to a query received from the controller. Table 1.3 shows examples of the response data format used in this manual.

Table 1.3: Network Master response data

Response data type	Description
<BOOLEAN RESPONSE DATA>	Defined in SCPI-99 Indicates On/Off, Enable/Disable, or Yes/No. When "1" is returned, it indicates an On/Enable state. When "0" is returned, it indicates an Off/Disable state.

<NR1 NUMERIC RESPONSE DATA>	Defined in IEEE488.2
	Indicates an decimal integer value.
	Examples:
	`123`
	`-500`

<NR2 NUMERIC RESPONSE DATA>	Defined in IEEE488.2
	Indicates a numeric value in fixed point format.
	Examples:
	`123.45`
	`-500.0`

continued on next page...

. . . continued from previous page

Response data type	Description

<NR3 NUMERIC RESPONSE DATA>	Defined in IEEE488.2
	Indicates a numeric value in floating point format.
	Examples:
	`1.23E3`
	`-5.67E-4`

<HEXADECIMAL NUMERIC RESPONSE DATA>	Conforms to the hexadecimal format defined in IEEE488.2 as follows:
	`#H{A|B|C|D|E|F|<digit>}...`
	<digit> is an ASCII character with a value in the range of 0x30 to 0x39 (48 to 57 in decimal), a numeric 0 to 9.
	Example:
	`#H0011EEFF`

<BINARY NUMERIC RESPONSE DATA>	Conforms to the binary format defined in IEEE488.2 as follows:
	`#B{0|1}...`
	Example:
	`#B10101010`

<STRING RESPONSE DATA>	Defined in IEEE488.2 A character string enclosed in a pair of double quotation marks (”). Example: `"Network Master - Testing the network."`
<CHARACTER RESPONSE DATA>	Defined in IEEE488.2 Indicates two or more mnemonics for selections. Like program header mnemonics, <CHARACTER RESPONSE DATA> mnemonics can have a short and a long form. The Network Master always returns the short form. The syntax used in the Network Master additionally allows a digit as the first character of a mnemonic and also allows a dash (-) inside a mnemonic.
<EXPRESSION RESPONSE DATA>	Defined in IEEE488.2 A Network Master-defined set of <RESPONSE DATA> elements separated by a comma (,) and enclosed by a set of parenthesis. Example: `(2,0.5),(3,0.25),(4,1.75)` For further details refer to the detailed description of the Network Master specific commands.
<DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>	Defined in IEEE488.2 This response data type is used when the instrument streams binary data (typically PDF files) to the controller. It is defined as #<nonzero digit><digits><8 bit data bytes> , where: <nonzero digit> is a single ASCII character in the range of ’1’-’9’. It represents the length of <digits> in number of bytes. <digits> is a number of ASCII characters in the range of ’0’-’9’, which together are a decimal representation of the number of succeeding data bytes.
	Example: `#49137<9137 bytes of binary data>`

1.3.2 Response Messages Terminator

A response message terminator indicates the end of the response message. Network Master appends the terminator to the end of a response message to indicate the end of the message. For Network Master the response message terminator is {`NL`} .

1.3.3 Prompt

For Ethernet based remote control a prompt can optionally be returned by the Network Master when all commands in a program message has completed. The prompt is inserted after the response message if any. It can be useful to enable the prompt when manually typing commands on the command line of the remote control interface. The prompt inserted is:

```
SCPI:>
```

See section 2.3.21 on page 84 and section 1.5 Controller Example.

1.4 Status

Network Master has registers which show instrument status such as the error occurrence and the command procession. This section describes these registers.

1.4.1 Configuration of Status Registers

Figure 1.7: Configuration of Status Registers in the Network Master ⊕ means logical OR.

Each register has eight bits data or 16 bits and the bit enable register is provided against each register The register model for SCPI defined status registers and Network Master device dependant registers is shown in Figure 1.8.

Figure 1.8: The register model for the Status registers.

Condition Registers

The Condition registers reflect the real-time status of the instrument or summary-message bits of other status registers. Thus, this register does not memorize the status.

Transition Filters

Sets the Condition Register data in the Event Register. The following three types of transition filters are available depending on which change of the Condition Register is to be evaluated.

- Positive direction change: The event becomes true only when the corresponding condition changes from false to true.
- Negative direction change: The event becomes true only when the corresponding condition changes from true to false.
- Bi-directional change: The event becomes true when a change arise either in the positive or negative direction.

The transition filters for the Network Master's Device-Dependent Status registers are locked to the "Positive transition" criteria. This means that events will be generated when the condition bits goes from 0 to 1. It is not possible to change the transition criteria.

Event Registers

The Event registers store the transition filter output. These registers are cleared when read.

Event Enable Registers

The Event Enable registers cannot be read or written and on power-on they are all set to zero. To enable the Event Enable registers and hence generation of summary-messages, the STATus:PRESet command must be send. The STATus:PRESet command changes all the bits in the Event Enable registers to 1.

1.4.2 IEEE488.2-defined Registers

Status Byte

Figure 1.9: Configuration of Status Byte Register and Service Request Enable Register

Note:
When using the GPIB interface (J1667A GPIB-USB Converter), the service request is enabled. For more

Table 1.4: Bits in the Status Byte register

Bit	Name	Description
7	OPERation	Summary-message bit for the Operation Status register. Use the `STATus:OPERation:ENABle` command described in section 2.5.2 to enable generation of this summary-message.
6	Master Summary (MSS)	The Master Summary Status message. Use the `*SRE` command described in section 2.2.7 to enable generation of this summary-message.
5	Standard Event (ESB)	Summary-message bit for the Standard Event Status register. Use the `*ESE` command described in section 2.2.2 to enable generation of this summary-message.
4	Message Available (MAV)	Summary-message bit for the Output Queue.
3	QUEStionable	Summary-message bit for the Questionable Status register. Use the `STATus:QUEStionable:ENABle` command described in section 2.5.8 to enable generation of this summary-message.
2	Error Queue	Summary-message bit for the Error queue for all connected application servers. Use the `INSTrument:ERRor[:NEXT]?` command described in section 2.4.18 to retrieve the messages.
1	Event Queue	Summary-message bit for the Event Queue for the currently selected application server. Use the `SYSTem:ERRor[:NEXT]?` command described in section 2.3.2 to retrieve the messages.
0	Port	Summary-message bit for the Port Event Summary register. Use the `STATus:PRESet` command described in section 2.5.16 and the `STATus:PORT:ENABle` command described in section 2.5.13 to enable generation of this summary-message.

information about the Status Byte register, see section 2.2.8 on page 79.

Standard Event Status

Figure 1.10: Configuration of Standard Event Status Register and Standard Event Status Enable Register

Table 1.5: Bits in the Standard Event Status register (unused bits are not listed)

Bit	Name	Description
7	Power On	The condition bit changes to 1 when the external power supply is connected.
5	Command Error	The condition bit changes to 1 when a unknown or errored command is received.
4	Execution Error	The condition bit changes to 1 when a command fail to execute properly.
3	Device Dependent Error	The condition bit changes to 1 when a required SW or HW options is missing or the Error/Event queue is full.
0	Operation Complete	The condition bit changes to 1 when *OPC command is received.

All condition bits are immediately changed back to 0 after they are set. This means that the only way to check the bits is to read the Event register. For more information on what triggers the Device Dependent, Execution and Command Errors see the Error/Event Queue section on page page 64.

For more information on the Standard Event Status register see section 2.2.2 on page 76.

Error/Event Queue

When an unexpected error or event occurs, an entry is added to the Error/Event queue. This queue can hold 4 errors or events. If the queue overflows, the most recent events are discarded. A summary-message in bit 2 of the Status Byte is 1 when the queue is not empty. Table 1.6 gives an overview of the different errors and events inserted in the queue.

For more information about the Event queue, see section 2.4.18 on page 93.

Table 1.6: Errors and events that can occur in the Error/Event queue

Event Number	**Error Description**
0	No Error (when queue is empty)
Command errors (Command Error bit is simultaneously set)	
-100	Command error
-102	Syntax error
-104	Data type error
-115	Unexpected number of parameters
-130	Suffix error
-131	Invalid suffix
-138	Suffix not allowed
Execution errors (Execution Error bit is simultaneously set)	
-200	Execution error
-220	Parameter error
-221	Settings conflict
-222	Data out of range
-224	Illegal parameter value
-250	Mass storage error
Device Dependent errors (Device Dependent Error bit is simultaneously set)	
1	Options Missing
-350	Queue overflow

For more information about the Error queue, see section 2.3.2 on page 80.

1.4.3 SCPI-defined Registers

Questionable Status

Figure 1.11: Configuration of Questionable Status Register

Table 1.7: Bit in the Questionable Status register (unused bits are not listed)

Bit	Name	Description
14	Command warning	The command warnig bit changes to 1 when the command which could not be processed was sent.

For more information about the Questionable Status register, see section 2.5.6 on page 97.

Operation Status

Figure 1.12: Configuration of Operation Status Register

Table 1.8: Bit in the Operation Status register (unused bits are not listed)

Bit	Name	Description
4	Measuring	The measuring condition bit changes to 1 when the an Application Server is running a measurement or a test. It returns to 0 when the measurement or test is stopped.

For more information about the Operation Status register, see section 2.5.1 on page 96.

1.4.4 Network Master Device-Dependent Status Registers

Figure 1.13 shows the structure of Post Status registers as the Network Master Device-Dependent status registers.

There is a Port Status register for each port on the Network Master. The Port Status registers summarize the

Figure 1.13: Configuration of Port Status Register

AESummary (Alarm and Error Summary) registers from the active interfaces. The Port Status registers are again summarized in bit 0 of the Status Byte, see Figure 1.15.

For more information about the Port Interface Status register, see section 2.5.17 on page 101.

Summary Register

The summarized status of the following registers is stored in the Summary Register (Refer to Table 1.14). The Network Master Device-Dependent Status registers are used to report alarms and errors for all interfaces. Each interface has one or more registers to represent the current alarm and error status. Each of these Alarm and Error registers are summarized in a General Interface Summery register (AESummery), see Figure 1.14. The exact layout of each register is found under the Status section for each interface.

- Alarm event registers
- Error event registers
- Event registers of lower layer summary registers

Like other status registers, the Summary register consists of the Condition register and the Event register, according to the register model defined in section 11.4.2 of IEEE488.2. The Summary register is locked to positive transition criteria because it consists of lower layer Event registers.

Figure 1.14: The structure for the Port Interface Status Register

Figure 1.15: The generic structure for the AESummary register of interfaces

1.4.5 Reading, Writing and Clearing Status Registers

The following two tables list the possibilities for reading and writing the various status registers and queues. They also show when and how registers are cleared or enabled.

Table 1.9: Reading and writing of Status registers

Registers		**Reading**	**Writing**
IEEE488.2 standard status registers	Status Byte	*STB?	Not possible
	Service Request Enable	*SRE?	*SRE
	Standard Event Status	*ESR? After reading, the register content is cleared.	Not possible
	Standard Event Status Enable	*ESE?	*ESE
SCPI defined status registers	Error/Event Queue	SYST:ERR? After reading, the error/event is removed from the queue.	Not possible
	Operation Event	STAT:OPER? After reading, the register content is cleared.	Not possible
	Operation Enable	STAT:OPER:ENAB?	STAT:OPER:ENAB
	Questionable Event	STAT:QUES? After reading, the register content is cleared.	Not possible
	Questionable Enable	STAT:QUES:ENAB?	STAT:QUES:ENAB
Network Master device-dependent status registers	Condition	<Interface>:STAT:<Port>:<Register>:COND?	Not possible
	Transition Filter	Not possible	Not possible
	Event	<Interface>:STAT:<Port>:<Register>? After reading, the register content is cleared.	Not possible
	Enable	Not possible	Not possible

Notes

The Condition resister of the Summary register is locked to positive transition criteria. Therefore, if clearing the register (*CLS) while an alarm or error is occurring, the register bits stay in ”0” (cleared) in spite of the alarm or error occurrence.

Table 1.10: Status registers behaviour for different commands/events

Registers		***RST**	***CLS**	**PowerOn**	**STAT:PRES**
IEEE488.2 standard status registers	Status Byte	No Changes	Cleared	Cleared	No Changes
	Service Request Enable	No Changes	No Changes	Cleared	No Changes
	Standard Event Status	No Changes	Cleared	Cleared	No Changes
	Standard Event Status Enable	No Changes	No Changes	Cleared	No Changes
SCPI defined status registers	Error/Event Queue	No Changes	Cleared	Cleared	No Changes
	Operation Event	No Changes	Cleared	Cleared	No Changes
	Operation Enable	No Changes	No Changes	Cleared	No Changes
	Questionable Event	No Changes	Cleared	Cleared	No Changes
	Questionable Enable	No Changes	No Changes	Cleared	No Changes
Network Master Unique Status registers	Condition	No Changes	No Changes	Cleared	No Changes
	Transition Filter	No Changes	No Changes	No Changes	No Changes
	Event	No Changes	Cleared	Cleared	No Changes
	Enable	No Changes	No Changes	Cleared	Enabled (all 1's)

1.5 Controller Example

One example of how to connect a controller to the Network Master instrument is described in this section.

1.5.1 PuTTY

PuTTY is a free Telnet/SSH client which supports raw TCP connections. With PuTTY it is possible to get terminal emulation access to the instrument. It is recommended to enable the prompt when using PuTTY. PuTTY does not support file streaming like the MEAS:EXP command.
PuTTY can be downloaded from http://www.chiark.greenend.org.uk/~sgtatham/putty/

Setup

1. Install PuTTY.
2. Start PuTTY.
3. In the PuTTY Configuration enable **Implicit CR in every LF** at **Category:→Terminal**.

Figure 1.16: Enable **Implicit CR in every LF** in PuTTY

4. In the instrument GUI, find the instrument's **IP Address** information, see Figure 1.17. Then type it in PuTTY, see Figure 1.18.

Figure 1.17: **IP Address** on the instrument

5. In PuTTY, type 56001 in the **Port** field, select the **Raw** radio button in the **Connection type** field, and click the **Open** button. see Figure 1.18.

Figure 1.18: Specify the destination and **Open** the connection in PuTTY

6. A window appears, see Figure 1.19.

Figure 1.19: Connection established with PuTTY

1.6 Definitions

1.6.1 NaN (Not a Number)

NaN is defined in SCPI-99. NaN is represented as 9.91E37 (<NR3 NUMERIC RESPONSE DATA>) as defined in IEEE 754. NaN is also used to represent missing data.

1.6.2 → Right Arrow

The right arrow → used in this document has two meanings:

- On the left side of the arrow is a query and returned value on the right hand side.
  Example: `TMBP:RX1:PATT?` → `PRBS11`

1.6.3 Data Bit (DB)

Data bit is represented as DBx where x represents the bit index in a register. DB1 is always LSB.

Figure 1.20: Data bit

1.6.4 Port Number (Logical Port)

Specify a logical port number assigned for each started application as a port number <Pt> in the SCPI command. The logical port numbers will be assigned in the order of Module1-Port1, Module1-Port2, Module2-Port1, and Module2-Port2. The logical port number starts with "1" regardless of the physical port numbers.

Example of logical port assignment when using the application start command

- INST:STAR TP-BERT-OTN,1-PORT2,1-PORT1,2-PORT1 (Figure 1.21)
  Physical Port : Logical Port
  1-PORT1 : PORT1
  1-PORT2 : PORT2
  2-PORT1 : PORT3

  When setting CFP for 2-PORT1 by using command, specify "3" for <Pt> in OTN:TX<PT>:INTerface.
  Example: OTN:TX3:INT CFP

Figure 1.21: Correspondence between physical and logical port numbers (Three ports)

- INST:STAR TP-BERT-OTN,2-PORT2 (Figure 1.22)
  Physical Port : Logical Port
  2-PORT2 : PORT1

Figure 1.22: Correspondence between physical and logical port numbers (One port)

- INST:STAR TP-BERT-ETH,1-PORT2,2-PORT1 (Figure 1.23)
  Physical Port : Logical Port
  1-PORT2 : PORT1
  2-PORT1 : PORT2

Figure 1.23: Correspondence between physical and logical port numbers (Two ports)

Notes

- The physical ports (1-PORT1, 2-PORT1 etc.) specified at application startup basically do not match the logical port numbers assigned in each application.
- The logical ports are not assigned in the same order of the physical ports described in the SCPI command for starting application.(Figure 1.17) For example, the following two commands include the same physical port numbers in different orders. However, for the both cases, 1-PORT is assigned to logical PORT1 and 1-PORT2 is assigned to logical PORT2.
  INST:STAR TP-BERT-OTN,1-PORT1,1-PORT2
  INST:STAR TP-BERT-OTN,1-PORT2,1-PORT1
- When multiple applications are started, the logical ports are numbered from 1 for each application.

Chapter 2

SCPI Conformance Information

2.1 SCPI Version

The Network Master Remote Control application conforms to SCPI 1999.0

2.2 IEEE 488.2 Mandatory Commands

2.2.1 *CLS

Syntax	*CLS
Description	This command clears all the event registers summarized in the Status Byte register. The error queue is emptied. Neither the Standard Event Status Enable register, nor the Service Request Enable register are affected by this command.
Parameter	None.
Response	None.
Example	`*CLS`
Note	All active (SCPI) sessions has their own set of standard registers.

2.2.2 *ESE

Syntax	*ESE <mask>
Description	This command sets bits in the Standard Event Status Enable register. A 1 in a bit in the enable register enables the corresponding bit in the Standard Event Status register. This register is cleared at power-on. The *RST and *CLS commands do not affect this register.
Parameter	<mask> = <NUMERIC PROGRAM DATA> The bits and their values for the enable mask: DB1 (1) = Operation Complete DB2 = NOT USED DB3 = NOT USED DB4 (8) = Device Dependent Error DB5 (16) = Execution Error DB6 (32) = Command Error DB7 = NOT USED DB8 (128) = Power On *MINimum=0, MAXimum=255*
Response	None.
Example	`*ESE 16`
Note	All active sessions has their own Standard Event Status Enable register.

Syntax	*ESE?
Description	This query returns the contents of the Standard Event Status Enable register.
Parameter	None.
Response	<mask> = <NR1 NUMERIC RESPONSE DATA> See the *ESE command for bit values for the enable mask.
Example	`*ESE? → 16`
Note	

2.2.3 *ESR?

Syntax	*ESR?
Description	This query returns the contents of the Standard Event Status register. This register is cleared after being read.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA> The bits and their values for the register: DB1 (1) = Operation Complete DB2 = NOT USED DB3 = NOT USED DB4 (8) = Device Dependent Error DB5 (16) = Execution Error DB6 (32) = Command Error DB7 = NOT USED DB8 (128) = Power On
Example	`*ESR?` → `49`
Note	All active sessions has their own Standard Event Status register.

2.2.4 *IDN?

Syntax	*IDN?
Description	This query returns the instrument identification over the interface.
Parameter	None.
Response	<manufacturer>,<model>,<serial>,<version> = <ARBITRARY ASCII RESPONSE DATA>
Example	`*IDN?` → `Anritsu,MT1000A,6123456789,1.00`
Note	

2.2.5 *OPC

Syntax	*OPC
Description	This command causes the instrument to generate the operation complete message in the Standard Event Status register when all pending selected instrument operations have been finished.
Parameter	None.
Response	None.
Example	`*OPC`
Note	All active sessions has their own Standard Event Status register.

Syntax	*OPC?
Description	This query places the ASCII character '1' into the instrument's output queue when all pending operations have been finished.
Parameter	None.
Response	<operation complete> = <NR1 NUMERIC RESPONSE DATA>
Example	`*OPC?` → `1`
Note	Only application servers connected to by the current session are synchronized.

2.2.6 *RST

Syntax	*RST
Description	This command sets the instrument to reset setting (standard setting) stored in internal storage. The instrument is placed in the idle state awaiting a command. All running application/servers are closed when the *RST command is issued. The following are not changed: - Service Request Enable register (SRE) - Standard Event Status register (ESR) - Standard Event Status Enable register (ESE) - Any instrument specific Status Event or Status Event Enable registers
Parameter	None.
Response	None.
Example	`*RST`
Note	*RST is specially configured to be compatible with SCPI remote control.

2.2.7 *SRE

Syntax	*SRE <enable mask>
Description	This command sets bits in the Service Request Enable register. A 1 in a bit in the enable register enables the corresponding bit in the Status Byte, also sets the Master Summary Status bit (DB7) in the Status Byte. The register is cleared at power-on. The *RST and *CLS commands do not affect the register.
Parameter	<enable mask> = <NUMERIC PROGRAM DATA> The bits and their values for the register: DB1 (1) = Port Event Summary DB2 (2) = Event Queue Summary for the currently selected application server. DB3 (4) = Error Queue Summary for all connected application servers. DB4 (8) = Questionable Status Summary DB5 (16) = Message Available (MAV) DB6 (32) = Standard Event Status Summary (ESB) DB7 = NOT USED DB8 (128) = Operation Status Summary *MINimum=0, MAXimum=255*
Response	None.
Example	`*SRE 255`
Note	All active sessions has their own Service Request Enable register.

Syntax	*SRE?
Description	This query returns the contents of the Service Request Enable register.
Parameter	None.
Response	<mask> = <NR1 NUMERIC RESPONSE DATA> See the *SRE command for bit values for the enable mask.
Example	`*SRE? → 255`
Note	

2.2.8 *STB?

Syntax	*STB?
Description	This query returns the contents of the Status Byte register. The Master Summary Status (MSS) bit is true when any bit of the STB register is set and a matching bit in the Service Request Enable Register is set, see `*SRE`. The Status Byte register including the MSS is not altered by this query.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA> The bits and their values for the register: DB1 (1) = Port Event Summary DB2 (2) = Event Queue Summary for the currently selected application server. DB3 (4) = Error Queue Summary for all connected application servers. DB4 (8) = Questionable Status Summary DB5 (16) = Message Available (MAV) DB6 (32) = Standard Event Status Summary (ESB) DB7 (64) = Master Summary Status (MSS) DB8 (128) = Operation Status Summary
Example	`*STB? → 7`
Note	

2.2.9 *TST?

Syntax	*TST?
Description	This query returns whether or not the instrument completed the self-test without any detected errors.
Parameter	None.
Response	<result> = <NR1 NUMERIC RESPONSE DATA> 0: No self-test errors detected 1: Self-test error detected
Example	`*TST? → 0`
Note	Self-test is performed automatically at the time of power up.

2.2.10 *WAI

Syntax	*WAI
Description	This command prevents the instrument from executing any further commands until the current command has been finished. All pending operations are completed during the wait period.
Parameter	None.
Response	None.
Example	`*WAI`
Note	*WAI functions for commands called "overlap command". For now Network Master does not have any overlap commands. So *WAI does not work on the current Network Master.

2.3 SCPI System Subsystem Commands

2.3.1 SYSTem:VERSion?

Syntax	SYSTem:VERSion?
Description	This query returns the SCPI revision to which the system complies.
Parameter	None.
Response	<version> = <NR2 NUMERIC RESPONSE DATA>
Example	`SYST:VERS?` → `1999.0`
Note	

2.3.2 SYSTem:ERRor[:NEXT]?

Syntax	SYSTem:ERRor[:NEXT]?
Description	This query returns the oldest entry of the error queue and removes the returned entry from the queue.
Parameter	None.
Response	<error number> = <NR1 NUMERIC RESPONSE DATA>
	<description> = <STRING RESPONSE DATA>
Example	`SYST:ERR?` → `-222,"Data out of range"`
Note	All active sessions has their own error queue. Application server ID is added to each error message when the additional message is selected TEST or BOTH. Application server ID is -1 for system errors. Application server ID is fixed to 0 if no error message is in error queue. Error command is added to each error message when the additional message is selected COMMand or BOTH.

2.3.3 SYSTem:ERRor:ADDitional[:MESSage]

Syntax	SYSTem:ERRor:ADDitional[:MESSage] <message>
Description	This command select additional message in the error message.
Parameter	<message> = <CHARACTER PROGRAM DATA> NONE TEST COMMand BOTH
Response	None.
Example	`SYST:ERR:ADD BOTH` `SYST:ERR?` → `-115,"Unexpected number of parameters:-1:INST:TERM"`
Note	This setting is applied only for the current session and defaulted to NONE when session closed. See also SYSTem:ERRor[:NEXT]?

Syntax	SYSTem:ERRor:ADDitional[:MESSage]?
Description	This query returns additional message in the error message.
Parameter	None.
Response	<message> = <CHARACTER RESPONSE DATA>
Example	`SYST:ERR:ADD?` → `NON`
Note	

2.3.4 SYSTem:DATE

Syntax	SYSTem:DATE <year>,<month>,<day>
Description	This command sets the date of the internal calender.
Parameters	<year> = <NUMERIC PROGRAM DATA> *MINimum = 1997, MAXimum = 2036*
	<month> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 12*
	<day> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 31*
Response	None.
Example	`SYST:DATE 2009,12,31`
Note	

Syntax	SYSTem:DATE?
Description	This query returns the date of the internal calender.
Parameter	None.
Response	<year>,<month>,<day> = <NR1 NUMERIC RESPONSE DATA>
Example	`SYST:DATE?` → `2009,07,04`
Note	

2.3.5 SYSTem:TIME

Syntax	SYSTem:TIME <hour>,<minute>,<second>
Description	This command sets the time of the internal clock.
Parameters	<hour> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 23*
	<minute> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 59*
	<second> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 59*
Response	None.
Example	`SYST:TIME 23,59,59`
Note	

Syntax	SYSTem:TIME?
Description	This query gets the time of the internal clock.
Parameter	None.
Response	<hour>,<minute>,<second> = <NR1 NUMERIC RESPONSE DATA>
Example	`SYST:TIME?` → `15,45,03`
Note	

2.3.6 SYSTem:REBoot

Syntax	SYSTem:REBoot
Description	This command will force a reboot of the instrument. A TCP remote connection to the instrument will be lost.
Parameter	None.
Response	None.
Example	`SYST:REB`
Note	

2.3.7 SYSTem:GPS:NSATellites?

Syntax	SYSTem:GPS:NSATellites?
Description	This query returns the number of satellites found by GPS.
Parameter	None.
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`SYST:GPS:NSAT?` → `5`
Note	Return "0" if GPS is not available.

2.3.8 SYSTem:GPS:TIME?

Syntax	SYSTem:GPS:TIME?
Description	This query returns the GPS time.
Parameter	None.
Response	<time> = <CHARACTER RESPONSE DATA>
Example	`SYST:GPS:TIME?` → `2014-01-01T12:34:56`
Note	Return "0" if GPS is not available.

2.3.9 SYSTem:GPS:LOCation?

Syntax	SYSTem:GPS:LOCation?
Description	This query returns the location.
Parameter	None.
Response	<location> = <NR2 NUMERIC RESPONSE DATA>
Example	`SYST:GPS:LOC?` → `85 26.8444N, 22 20.4508E`
Note	Return "0" if GPS is not available.

2.3.10 SYSTem:GPS:POWEr

Syntax	SYSTem:GPS:POWEr <power>
Description	This command turns GPS module's power ON/OFF (1 - ON, 0 - OFF).
Parameters	<power> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1*
Response	None.
Example	`SYST:GPS:POWE 0`
Note	

2.3.11 SYSTem:GPS:POWEr?

Syntax	SYSTem:GPS:POWEr?
Description	This query returns GPS module's power ON/OFF (1 - ON, 0 - OFF).
Parameter	None.
Response	<power> = <NR1 NUMERIC RESPONSE DATA>
Example	`SYST:GPS:POWE?` → `0`
Note	

2.3.12 SYSTem:GPS:SYNCmode

Syntax	SYSTem:GPS:SYNCmode <mode>
Description	This command turns GPS module's sync mode to GPS/EXT (1 - GPS, 0 - EXT).
Parameters	<mode> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1*
Response	None.
Example	`SYST:GPS:SYNC 0`
Note	

2.3.13 SYSTem:GPS:SYNCmode?

Syntax	SYSTem:GPS:SYNCmode?
Description	This query returns GPS module's sync mode GPS/EXT (1 - GPS, 0 - EXT).
Parameter	None.
Response	<mode> = <NR1 NUMERIC RESPONSE DATA>
Example	`SYST:GPS:SYNC?` → `0`
Note	

2.3.14 SYSTem:GPS:ANTNpower

Syntax	SYSTem:GPS:ANTNpower <power>
Description	This command turns GPS module's antenna power 5V/3.3V (1 - 5V, 0 - 3.3V).
Parameters	<power> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1*
Response	None.
Example	`SYST:GPS:ANTN 0`
Note	

2.3.15 SYSTem:GPS:ANTNpower?

Syntax	SYSTem:GPS:ANTNpower?
Description	This query returns GPS module's antenna power 5V/3.3V (1 - 5V, 0 - 3.3V).
Parameter	None.
Response	<power> = <NR1 NUMERIC RESPONSE DATA>
Example	`SYST:GPS:ANTN?` → `0`
Note	

2.3.16 SYSTem:GPS:PWRElapsed?

Syntax	SYSTem:GPS:PWRElapsed?
Description	This query returns GPS module's power on elapsed seconds.
Parameter	None.
Response	<time> = <NR1 NUMERIC RESPONSE DATA>
Example	`SYST:GPS:PWRE?` → `0`
Note	

2.3.17 SYSTem:GPS:LOCElapsed?

Syntax	SYSTem:GPS:LOCElapsed?
Description	This query returns GPS module's oscillator lock elapsed seconds.
Parameter	None.
Response	<time> = <NR1 NUMERIC RESPONSE DATA>
Example	`SYST:GPS:LOCE?` → `0`
Note	

2.3.18 SYSTem:GPS:SYNElapsed?

Syntax	SYSTem:GPS:SYNElapsed?
Description	This query returns GPS module's sync mode elapsed seconds.
Parameter	None.
Response	<time> = <NR1 NUMERIC RESPONSE DATA>
Example	`SYST:GPS:SYNE?` → `0`
Note	

2.3.19 SYSTem:COMMunicate:TERMinator

Syntax	SYSTem:COMMunicate:TERMinator <terminator>
Description	This command sets the terminator code which is appended to the query response.
Parameter	<terminator> = <CHARACTER PROGRAM DATA> NONE(only GPIB) LF CRLF
Response	None.
Example	`SYST:COMM:TERM LF`
Note	This setting is applied only for the current session and defaulted to CRLF when session closed.

Syntax	SYSTem:COMMunicate:TERMinator?
Description	This query returns the terminator code which is appended to the query response.
Parameter	None.
Response	<terminator> = <CHARACTER RESPONSE DATA>
Example	`SYST:COMM:TERM?` → `LF`
Note	

2.3.20 SYSTem:PROMpt

Syntax	SYSTem:PROMpt <enable>
Description	This command enables/disables appending of prompt to all replies from Remote Control interface.
Parameter	<enable> = <BOOLEAN PROGRAM DATA>
Response	None.
Example	`SYST:PROM 1`
Note	The prompt string is "SCPI:> " This setting is applied only for the current session and forgets when session closed.

Syntax	SYSTem:PROMpt?
Description	This query returns status of the prompt.
Parameter	None.
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`SYST:PROM?` → `SCPI:>1`
Note	

2.3.21 SYSTem:LOCal:CONTrol

Syntax	SYSTem:LOCal:CONTrol <enable>
Description	This command enables/disables local control.
Parameter	<enable> = <BOOLEAN PROGRAM DATA>
Response	None.
Example	`SYST:LOC:CONT 1`
Note	This setting is applied all connected sessions and forgets when turn off SCPI.

Syntax	SYSTem:LOCal:CONTrol?
Description	This query returns enables/disables local control.
Parameter	None.
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`SYST:LOC:CONT?` → `1`
Note	

2.3.22 SYSTem:TIMing:EXTernal?

Syntax	SYSTem:TIMing:EXTernal?
Description	This query returns external clock input signal status.
Response	<clock> = <STRING RESPONSE DATA> "2M" : 2MHz clock or E1 (SETS) signal "1.5M" : T1 (BITS) signal "10M" : 10MHz clock signal "N/A" : No Input
Example	`SYST:TIM:EXT?` → `"2M"`
Note	There must be a connected application server for this command to be recognized as a legal command.

2.3.23 SYSTem:STIMuli:INSert

Syntax	SYSTem:STIMuli:INSert
Description	This command emulates pressing the Error Insert button of the GUI. Errors are inserted in the outgoing data stream according to the configuration of the active Stimuli setup(s).
Parameter	None.
Response	None.
Example	`SYST:STIM:INS`
Note	There must be a connected application server for this command to be recognized as a legal command.

2.3.24 SYSTem:STIMuli:CLR

Syntax	SYSTem:STIMuli:CLR
Description	This command clears the Stimuli setup(s) for all interfaces.
Parameter	None.
Response	None.
Example	`SYST:STIM:CLR`
Note	There must be a connected application server for this command to be recognized as a legal command.

2.3.25 SYSTem:WAIT[:IDLE]

Syntax	SYSTem:WAIT[:IDLE]
Description	This command waits for the instrument to go into IDLE state, i.e. no measurement or test is pending, running, loading or being stored. It also waits for load and save of settings to finish.
Parameter	None.
Response	None.
Example	`SYST:WAIT`
Note	There must be a connected application server for this command to be recognized as a legal command.
	Be careful when using this command as it may lead to undesired blocking of the remote interface. In some situations the instrument requires a remote command or other user intervention in order to return to IDLE state; e.g. when a measurement is running and measurement stop mode is set to MANual (`MEAS:SET:STOP` → `MAN`). In this situation, for the instrument to return to IDLE state press the START/STOP button on the GUI or apply the `MEASurement:STOP` command. The latter is NOT possible if `SYST:WAIT:IDLE` is currently being executed. If an undesired blocking occurs close and re-open the remote connection. And then send *RST command to reset undesired blocking remote connection.

2.3.26 SYSTem:WAIT:DURation

Syntax	SYSTem:WAIT:DURation <seconds>
Description	This command waits for the specified number of seconds.
Parameter	<seconds> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 3600*
Response	None.
Example	`SYST:WAIT:DUR 5`
Note	There must be a connected application server for this command to be recognized as a legal command.

2.3.27 SYSTem:APPlication

Syntax	SYSTem:APPlication <application>
Description	This command selects between the BERT, APS and RTD applications
	<application> = <CHARACTER PROGRAM DATA> BERT: BERT Application APS: APS Application RTD: RTD Application *DEFault = BERT*
Response	None.
Example	`SYST:APP BERT`
Note	This command only applies in the OTN or SDH/SONET/PDH/DSn BERT applications

2.3.28 SYSTem:APPlication?

Syntax	SYSTem:APPlication?
Description	This command returns the application type
Response	<app> = <CHARACTER PROGRAM DATA> BERT: BERT Application APS: APS Application RTD: RTD Application
Example	`SYST:APP?` → `BERT`
Note	This command only applies in the OTN or SDH/SONET/PDH/DSn BERT applications

2.4 SCPI Instrument Subsystem Commands

To use the application-specific SCPI commands, you need to connect the client session to the application server. By sending SCPI commands to the application server connected, you can control the application.

2.4.1 Connection to Application Server

To connect the client session to the application server, do one of the following:

- Use the `INST:CONN:ALL` command.
  This command connects a client session to the application server which is not occupied.
  If a different client session is already connected to the server (application server is occupied), the connection attempt fails.
- Use the `INST:CONN` command.
  This command connects a client session to all application servers which are not occupied.
- Use the `INST:STAR` command.
  This command starts a new application server and connects a client session to the application server.

2.4.2 Connection to multiple applications

A client session can connect to multiple application servers simultaneously. In this case, the SCPI command destination will be the selected application server.
The destination can be set by using the `INST:SEL` command. To confirm the selected application server, use the `INST?` command.
When a new application server has started by the `INST:STAR` command, the SCPI command destination will be changed to the newly started application server.

2.4.3 Connection from multiple users

When using a Network Master by multi-users, multiple client sessions connect to a Network Master. Under this condition, make sure that the application server is not occupied by any other user.

If a client session has already connected to an application server, Network Master cannot connect any other client session to the application server.
To connect to the application server occupied by a client session, you need to release the application server by disconnecting the client session which is connecting currently. To release the application server, do one of the following:

- Use the `INST:DISC` command.
  You can disconnect the server by sending the `INST:DISC` command from the connecting client session.
- Terminate the Client Session.
  After the client session terminated, all application servers to which the client session was connected are released.
  You can terminate the client session by disconnecting connection to Network Master from the User PC.

You can connect a client session to the released application server again by using the `INST:CONN` command or the `INST:CONN:ALL` command.

2.4.4 Force Termination of Application Server

An application server to which a session is connecting doesn't receive SCPI commands from any other session. An exception is the `INST:TERM:FORC` command. This command is always accepted even if sent from the other session and terminates the application server to which a session is connecting.

continued on next page...

. . . continued from previous page

2.4.5 INSTrument:STARt[:DEFault]

Syntax	INSTrument:STARt[:DEFault] <app name>[, <port name>[, <port name>,...]]
Description	This command starts an application server with default settings.
Parameters	<app name> = <CHARACTER PROGRAM DATA> TP-APS-OTN: OTN Automatic Protection Switching application. TP-APS-SDHPDH: SDH/PDH Automatic Protection Switching application. TP-APS-SDHPDH-OTN: SDH/PDH over OTN Automatic Protection Switching application. TP-BERT-CPRI: CPRI Bit Error Rate Test application. TP-BERT-CPRI-OTN: CPRI over OTN Bit Error Rate Test application. TP-BERT-ETH: Ethernet Bit Error Rate Test application. TP-BERT-ETH-OTN: Ethernet over OTN Bit Error Rate Test application. TP-BERT-FC: Fibre Channel Bit Error Rate Test application. TP-BERT-FC-OTN: Fibre Channel over OTN Bit Error Rate Test application. TP-BERT-OTN: OTN Bit Error Rate Test application. TP-BERT-ROE: eCPRI/RoE Bit Error Rate Test application. TP-BERT-SDHPDH: PDH/SDH Bit Error Rate Test application. TP-BERT-SDHPDH-OTN: PDH/SDH over OTN Bit Error Rate Test application. TP-CABLE-ETH: Ethernet cable test application. TP-CHSTAT-ETH: Ethernet channel statistics application. TP-DISC-ETH: Ethernet discovery application. TP-MONGEN-ETH: Ethernet monitor/generate application. TP-MONGEN-ETH-OTN: Ethernet over OTN monitor/generate application. TP-NOFRAME-DEVICE: No frame device test (Unframed Bit Error Rate Test) application. TP-PASS-CPRI: CPRI pass-through application. TP-PASS-ETH: Ethernet pass-through application. TP-PERF-FC: Fibre Channel performance test application. TP-PERF-FC-OTN: Fibre Channel over OTN performance test application. TP-PING-ETH: Ethernet ICMP ping application. TP-REFL-ETH: Ethernet reflector application. TP-REFL-ETH-OTN: Ethernet over OTN reflector application. TP-REFL-FC: Fibre Channel reflector application. TP-REFL-FC-OTN: Fibre Channel over OTN reflector application. TP-RFC-ETH: Ethernet RFC-2544 test application. TP-RFC-ETH-OTN: Ethernet over OTN RFC-2544 test application. TP-RFC6349-ETH: Ethernet RFC-6349 test application. TP-RTD-OTN: OTN Round Trip Delay test application. TP-RTD-SDHPDH: SDH/PDH Round Trip Delay test application. TP-RTD-SDHPDH-OTN: SDH/PDH Round Trip Delay test application. TP-SAT-ETH: Ethernet Service Activation Test application. TP-SAT-ETH-OTN: Ethernet over OTN Service Activation Test application. TP-SYNCTEST-ETH: Ethernet sync test application. TP-TRACE-ETH: Ethernet trace-route application. OTDR-OTDR: OTDR application.
	<port name> = <CHARACTER PROGRAM DATA> 1-PORT1: Port 1 on module 1 1-PORT2: Port 2 on module 1 2-PORT1: Port 1 on module 2 2-PORT2: Port 2 on module 2 1-PORT-SING1: Single port 1 on module 1 (for MU110011A/MU110012A/MU110013A) 1-PORT-MULT1: Port 1 on module 1 (for TP-BERT-ROE with "25G eCPRI/RoE Dual Ability") 1-PORT-MULT2: Port 2 on module 1 (for TP-BERT-ROE with "25G eCPRI/RoE Dual Ability")

continued on next page. . .

. . . continued from previous page

	The physical port(s) given as parameters are assigned to logical port numbers in the application server. The logical port number range is from 1 to the actual number of port assigned to the application server. The list of ports returned by the `INSTrument:STATe? <id>` command reveals the virtual port number sequence. The ports will be shown sorted in (1st) module order and (2nd) port order.
Response	None.
Example	`INST:STAR TP-BERT-OTN,1-PORT1` `INST? → 2`
Note	Operators can get started application server ID by using the `INSTrument:SELect?` command. When you start an application server, the application server will be connected and selected automatically. When using this command, the application server will be started with DEFault setup.

2.4.6 INSTrument:STARt:LAST

Syntax	INSTrument:STARt:LAST <app name>[, <port name>[, <port name>,...]]
Description	This command starts an application server and loads the applicable auto saved settings.
Parameters	The parameters of this command are similar to the parameters of the `INSTrument:STARt[:DEFault]` command above.
Response	None.
Example	`INST:STAR:LAST TP-BERT-OTN,1-PORT1` `INST? → 2`
Note	Operators can get started application server ID by using the `INSTrument:SELect?` command. When you start an application server, the application server will be connected and selected automatically. When using this command, the application server will be started with LAST setup.

2.4.7 INSTrument:STARt:GUI

Syntax	INSTrument:STARt:GUI [<test index>]
Description	This command starts GUI for the application server.
Parameter	<test index> = <NUMERIC PROGRAM DATA> Defaults to the current application server if a value is omitted.
Response	None.
Example	`INST:STAR TP-BERT-OTN,1-PORT1` `INST:STAR:GUI`
Note	Must connect to the application server first.

2.4.8 INSTrument:TERMinate

Syntax	INSTrument:TERMinate [<test index>]
Description	This command terminates an application server.
Parameter	<test index> = <NUMERIC PROGRAM DATA> Defaults to the current application server if a value is omitted.
Response	None.

Example	`INST:STAR TP-BERT-OTN,1-PORT1`
	`INST? → 2`
	`INST:TERM 2`

Note	Must connect to the application server first.

Figure 2.1: `INST:STAR` automatically *connects* and selects started application server.

Figure 2.2: Original client session has to *disconnect*, before another client session can *connect*.

Figure 2.3: When connected to multiple application servers, client session *selects* to which application server the application specific commands are dispatched.

2.4.9 INSTrument:TERMinate:FORCe

Syntax	INSTrument:TERMinate:FORCe [<test index>]
Description	This command force terminates an application server.
Parameter	<test index> = <NUMERIC PROGRAM DATA> Defaults to the current application server if a value is omitted.
Response	None.

Example	`INST:STAR TP-BERT-OTN,1-PORT1`
	`INST?` → `2`
	`INST:TERM:FORC 2`

Note	WARNING. This command can terminate the application to which the other session communicating.

2.4.10 INSTrument:COUNt?

Syntax	INSTrument:COUNt?
Description	This query returns the number of active application servers.
Parameter	None.
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`INST:COUN?` → `2`
Note	

2.4.11 INSTrument:CATalog?

Syntax	INSTrument:CATalog?
Description	This query returns test indices, application name and port name of all active application servers.
Parameter	None.
Response	<test index> = <EXPRESSION RESPONSE DATA> Expression format: (<test index>,<app name>,<port name>)
Example	`INST:CAT?` → `(1,TP-SAT-ETH,1-PORT1),(2,TP-BERT-SDHPDH,1-PORT2)`
Note	Return -1 if no application server is running.

2.4.12 INSTrument:STATe?

Syntax	INSTrument:STATe? <test index>
Description	This query returns status information about a given application server.
Parameter	<test index> = <NUMERIC PROGRAM DATA>
Response	<app name> = <CHARACTER RESPONSE DATA>
	<client connection> = <CHARACTER RESPONSE DATA> NON IP Address
	<select status> = <CHARACTER RESPONSE DATA> NON SELECTED
	<port name> = <CHARACTER RESPONSE DATA> Expression format: Character list
Example	`INST:STAT? 1` → `TP-BERT-OTN,192.168.128.21,SELECTED,1-PORT1,2-PORT2...`
Note	

2.4.13 INSTrument:CONNect

Syntax	INSTrument:CONNect <test index>
Description	This command allows client session to connect to an existing application server.
Parameter	<test index> = <NUMERIC PROGRAM DATA>
Response	None.
Example	`INST:CONN 1`
Note	This command fails if the application server is already connect to by another client session. Use `INSTrument:CATalog?` query to acquire the list of all existing application servers. If command succeeds, the application server will be selected automatically.

2.4.14 INSTrument:CONNect:ALL

Syntax	INSTrument:CONNect:ALL
Description	This command allows client session to connect to all existing application servers.
Parameter	None.
Response	None.
Example	`INST:CONN:ALL`
Note	This command fails if no application server is selected when the command exits, e.g. because all application servers was already connected to by other client sessions, or because there are no application servers at all. If connected to multiple application servers, the application server with the lowest index will be selected, but selected index will not change if an application server was already selected prior to issuing this command.

2.4.15 INSTrument:CONNect[:CATalog]?

Syntax	INSTrument:CONNect[:CATalog]?
Description	This query returns indices of all application servers for the current client session.
Parameter	None.
Response	<test index> = <EXPRESSION RESPONSE DATA> Expression format: Numeric list
Example	`INST:CONN? → 0,1,...`
Note	Return -1 if current client session has no application servers.

2.4.16 INSTrument:DISConnect

Syntax	INSTrument:DISConnect <test index>
Description	This command disconnect the application server from the client session.
Parameter	<test index> = <NUMERIC PROGRAM DATA>
Response	None.
Example	`INST:DISC 1`
Note	When current application is disconnected, the application server lowest ID will be selected automatically. When a client session is disconnected all the currently connected application servers will be disconnected automatically.

2.4.17 INSTrument[:SELect]

Syntax	INSTrument[:SELect] <test index>
Description	This command select the current application server.
Parameter	<test index> = <NUMERIC PROGRAM DATA>
Response	None.

Example	`INST:STAR TP-BERT-OTN,1-PORT1`
	`INST? → 1`
	`INST:STAR TP-BERT-OTN,1-PORT2`
	`INST? → 2`
	`INST 1`

Note	All future commands are forwarded to the current application server until the current application server is changed.

Syntax	INSTrument[:SELect]?
Description	This query returns index of the currently selected application server.
Parameter	None.
Response	<count> = <NR1 NUMERIC RESPONSE DATA>

Example	`INST:STAR TP-BERT-OTN,1-PORT1`
	`INST? → 1`
	`INST:STAR TP-BERT-OTN,1-PORT2`
	`INST? → 2`
	`INST 1`

Note	Return -1 if current client session does not have a currently selected application server.

2.4.18 INSTrument:ERRor[:NEXT]?

Syntax	INSTrument:ERRor[:NEXT]?
Description	This query returns the oldest entry of the event queue for the currently selected application server and removes the returned entry from the queue.
Parameter	None.
Response	<description> = <STRING RESPONSE DATA>
Example	`INST:ERR? → "Signal abnormal"`
Note	Application server has its own event queue. This event queue is not destroyed when the session is closed. Before checking the server status by using this command, controller may check DB2 (Event Queue Summary) of the Status Byte Register to see if it is set. DB2 of the Status Byte Register will aggregate input from potentially many application servers, and clear if event queues of all servers are empty.

2.4.19 INSTrument:PORT?

Syntax	INSTrument:PORT?
Description	This query returns ports assigned for the currently selected application server.
Parameter	None.
Response	<port name> = <CHARACTER RESPONSE DATA> Expression format: Character list
Example	`INST:STAR TP-BERT-OTN,1-PORT1,1-PORT2` `INST:PORT? → 1-PORT1,1-PORT2`
Note	Operates on the application server currently selected by the `INSTrument:SELect command`. Returns `NON` if the client session does not have a currently selected application server. Returns `NON` if no ports are assigned for the currently selected application server.

2.4.20 INSTrument:PORT:FREE?

Syntax	INSTrument:PORT:FREE? <app name>
Description	This query returns all unused ports for the target application name.
Parameter	<app name> = <CHARACTER PROGRAM DATA>
Response	<port name> = <CHARACTER RESPONSE DATA> Expression format: Character list
Example	`INST:PORT:FREE? TP-BERT-OTN → 1-PORT1,1-PORT2`
Note	

2.4.21 INSTrument:PORT:CATalog?

Syntax	INSTrument:PORT:CATalog?
Description	This query returns all ports of device.
Parameter	None.
Response	<port name> = <CHARACTER RESPONSE DATA> Expression format: Character list
Example	`INST:PORT:CAT?` → `1-PORT1,1-PORT2`
Note	When SCPI client uses any of the STATus:PORT: commands, <bit> index is the same as returned by the INSTrument:PORT:CATalog?

2.4.22 INSTrument:MODule:CATalog?

Syntax	INSTrument:MODule:CATalog?
Description	This query returns module names of device.
Parameter	None.
Response	{<module n>, }* = <CHARACTER RESPONSE DATA>
Example	`INST:MOD:CAT?` → `MU100010A,MU100011A`
Note	

2.4.23 INSTrument:CTRL:NAME?

Syntax	INSTrument:CTRL:NAME?
Description	This query returns model name.
Parameter	None.
Response	<model name> = <CHARACTER RESPONSE DATA>
Example	`INST:CTRL:NAME?` → `MT1000A`
Note	

2.4.24 INSTrument:CTRL:SN?

Syntax	INSTrument:CTRL:SN?
Description	This query returns controller serial number.
Parameter	None.
Response	<serial number> = <CHARACTER RESPONSE DATA>
Example	`INST:CTRL:SN?` → `1234567890`
Note	

2.4.25 INSTrument:CTRL:TRT?

Syntax	INSTrument:CTRL:TRT?
Description	This query returns controller total run time(sec).
Parameter	None.
Response	<time> = <NR1 NUMERIC RESPONSE DATA>
Example	`INST:CTRL:TRT?` → `5000000`
Note	

2.4.26 INSTrument:CTRL:OPTion:CATalog?

Syntax	INSTrument:CTRL:OPTion:CATalog?
Description	This query returns controller enabled options.
Parameter	None.
Response	{<option n>, }* = <CHARACTER RESPONSE DATA>
Example	`INST:CTRL:OPT:CAT?` → `MT1000A-303,MT1000A-005`
Note	

2.4.27 INSTrument:MODule<Md>:NAME?

Syntax	INSTrument:MODule<Md>:NAME?
Description	This query returns module model number.
Parameter	None.
Response	<model name> = <CHARACTER RESPONSE DATA>
Example	`INST:MOD1:NAME?` → `MU100010A`
Note	

2.4.28 INSTrument:MODule<Md>:SN?

Syntax	INSTrument:MODule<Md>:SN?
Description	This query returns module serial number.
Parameter	None.
Response	<serial number> = <CHARACTER RESPONSE DATA>
Example	`INST:MOD1:SN?` → `1234567890`
Note	

2.4.29 INSTrument:MODule<Md>:TRT?

Syntax	INSTrument:MODule<Md>:TRT?
Description	This query returns module total run time(sec).
Parameter	None.
Response	<time> = <NR1 NUMERIC RESPONSE DATA>
Example	`INST:MOD1:TRT?` → `5000000`
Note	

2.4.30 INSTrument:MODule<Md>:OPTion:CATalog?

Syntax	INSTrument:MODule<Md>:OPTion:CATalog?
Description	This query returns module enabled options.
Parameter	None.
Response	{<option n>, }* = <CHARACTER RESPONSE DATA>
Example	`INST:MOD1:OPT:CAT?` → `MU100010A-001,MU100010A-002`
Note	

2.5 SCPI Status Subsystem Commands

2.5.1 STATus:OPERation[:EVENt]?

Syntax	STATus:OPERation[:EVENt]?
Description	This query returns and clears the operation event register.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA> The bits and their values for the register: DB1-DB4 = NOT USED DB5 (16) = Measuring DB6-DB16 = NOT USED
Example	`STAT:OPER?` → `16`
Note	All active sessions has their own register and it is cleared when the session starts.

2.5.2 STATus:OPERation:CONDition?

Syntax	STATus:OPERation:CONDition?
Description	This query returns the operation condition register.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA> The bits and their values for the register: DB1-DB4 = NOT USED DB5 (16) = Measuring DB6-DB16 = NOT USED
Example	`STAT:OPER:COND?` → `16`
Note	

2.5.3 STATus:OPERation:ENABle

Syntax	STATus:OPERation:ENABle <mask>
Description	This command sets the enable mask for the operation event register.
Parameter	<mask> = <NUMERIC PROGRAM DATA> The bits and their values for the register: DB1-DB4 = NOT USED DB5 (16) = Measuring DB6-DB16 = NOT USED *MINimum = 0, MAXimum = 65535*
Response	None.
Example	`STAT:OPER:ENAB 65535`
Note	

Syntax	STATus:OPERation:ENABle?
Description	This query returns the enable mask for the operation event register.
Parameter	None.
Response	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`STAT:OPER:ENAB?` → `16`
Note	

2.5.4 STATus:OPERation:PTRansition

Syntax	STATus:OPERation:PTRansition <mask>
Description	This command sets the positive transition filter for the operation event register.
Parameter	<mask> = <NUMERIC PROGRAM DATA> The bits and their values for the register: DB1-DB4 = NOT USED DB5 (16) = Measuring DB6-DB16 = NOT USED *MINimum = 0, DEFault = 65535, MAXimum = 65535*
Response	None.
Example	`STAT:OPER:PTR 16384`
Note	

Syntax	STATus:OPERation:PTRansition?
Description	This query returns the positive transition filter for the operation event register.
Parameter	None.
Response	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`STAT:OPER:PTR?` → `16384`
Note	

2.5.5 STATus:OPERation:NTRansition

Syntax	STATus:OPERation:NTRansition <mask>
Description	This command sets the negative transition filter for the operation event register.
Parameter	<mask> = <NUMERIC PROGRAM DATA> The bits and their values for the register: DB1-DB4 = NOT USED DB5 (16) = Measuring DB6-DB16 = NOT USED *MINimum = 0, MAXimum = 65535*
Response	None.
Example	`STAT:OPER:NTR 16384`
Note	

Syntax	STATus:OPERation:NTRansition?
Description	This query returns the negative transition filter for the operation event register.
Parameter	None.
Response	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`STAT:OPER:NTR?` → `16384`
Note	

2.5.6 STATus:QUEStionable[:EVENt]?

Syntax	STATus:QUEStionable[:EVENt]?
Description	This query returns and clears the questionable event register.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA> The bits and their values for the register: DB1-DB14 = NOT USED DB15 (16384) = Command Warning DB16 = NOT USED
Example	`STAT:QUES?` → `16384`
Note	All active sessions has their own register and it is cleared when the session starts.

2.5.7 STATus:QUEStionable:CONDition?

Syntax	STATus:QUEStionable:CONDition?
Description	This query returns the questionable condition register.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA> The bits and their values for the register: DB1-DB14 = NOT USED DB15 (16384) = Command Warning DB16 = NOT USED
Example	`STAT:QUES:COND?` → `16384`
Note	

2.5.8 STATus:QUEStionable:ENABle

Syntax	STATus:QUEStionable:ENABle <mask>
Description	This command sets the enable mask for the questionable event register.
Parameter	<mask> = <NUMERIC PROGRAM DATA> The bits and their values for the register: DB1-DB14 = NOT USED DB15 (16384) = Command Warning DB16 = NOT USED *MINimum = 0, MAXimum = 65535*
Response	None.
Example	`STAT:QUES:ENAB 16384`
Note	

Syntax	STATus:QUEStionable:ENABle?
Description	This query returns the enable mask for the questionable event register.
Parameter	None.
Response	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`STAT:QUES:ENAB?` → `16384`
Note	

2.5.9 STATus:QUEStionable:PTRansition

Syntax	STATus:QUEStionable:PTRansition <mask>
Description	This command sets the positive transition filter for the questionable event register.
Parameter	<mask> = <NUMERIC PROGRAM DATA> The bits and their values for the register: DB1-DB14 = NOT USED DB15 (16384) = Command Warning DB16 = NOT USED *MINimum = 0, DEFault = 65535, MAXimum = 65535*
Response	None.
Example	`STAT:QUES:PTR 16384`
Note	

Syntax	STATus:QUEStionable:PTRansition?
Description	This query returns the positive transition filter for the questionable event register.
Parameter	None.
Response	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`STAT:QUES:PTR?` → `16384`
Note	

2.5.10 STATus:QUEStionable:NTRansition

Syntax	STATus:QUEStionable:NTRansition <mask>
Description	This command sets the negative transition filter for the questionable event register.
Parameter	<mask> = <NUMERIC PROGRAM DATA> The bits and their values for the register: DB1-DB14 = NOT USED DB15 (16384) = Command Warning DB16 = NOT USED *MINimum = 0, MAXimum = 65535*
Response	None.
Example	`STAT:QUES:NTR 16384`
Note	

Syntax	STATus:QUEStionable:NTRansition?
Description	This query returns the negative transition filter for the questionable event register.
Parameter	None.
Response	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`STAT:QUES:NTR? → 16384`
Note	

2.5.11 STATus:PORT[:EVENt]?

Syntax	STATus:PORT[:EVENt]? <port name>
Description	This query returns and clears the port event register.
Parameter	<port name> = <CHARACTER PROGRAM DATA> Expression format: Character list
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`STAT:PORT? 1-PORT1 → 1`
Note	All active sessions has their own register and it is cleared when the session starts. <port name> is the same as the one returned by `INSTrument:PORT:CATalog?`.

2.5.12 STATus:PORT:CONDition?

Syntax	STATus:PORT:CONDition? <port name>
Description	This query returns the port condition register.
Parameter	<port name> = <CHARACTER PROGRAM DATA> Expression format: Character list
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`STAT:PORT:COND? 1-PORT1 → 1`
Note	<port name> is the same as one of the returned by the INSTrument:PORT:CATalog?

2.5.13 STATus:PORT:ENABle

Syntax	STATus:PORT:ENABle <port name>,<enable>
Description	This command sets the enable mask for the port event register.
Parameters	<port name> = <CHARACTER PROGRAM DATA> Expression format: Character list
	<enable> = <BOOLEAN PROGRAM DATA>
Response	None.
Example	`STAT:PORT:ENAB 1-PORT1,ON`
Note	<port name> is the same as one of the returned by the INSTrument:PORT:CATalog?

Syntax	STATus:PORT:ENABle? <port name>
Description	This query returns the enable mask for the port event register.
Parameter	<port name> = <CHARACTER PROGRAM DATA> Expression format: Character list
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`STAT:PORT:ENAB? 1-PORT1` → `1`
Note	<port name> is the same as one of the returned by the INSTrument:PORT:CATalog?

2.5.14 STATus:PORT:PTRansition

Syntax	STATus:PORT:PTRansition <port name>,<enable>
Description	This command sets the positive transition filter for the port event register.
Parameters	<port name> = <CHARACTER PROGRAM DATA> Expression format: Character list
	<enable> = <BOOLEAN PROGRAM DATA>
Response	None.
Example	`STAT:PORT:PTR 1-PORT1,ON`
Note	<port name> is the same as one of the returned by the INSTrument:PORT:CATalog?

Syntax	STATus:PORT:PTRansition? <port name>
Description	This query returns the positive transition filter for the port event register.
Parameter	<port name> = <CHARACTER PROGRAM DATA> Expression format: Character list
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`STAT:PORT:PTR? 1-PORT1` → `1`
Note	<port name> is the same as one of the returned by the INSTrument:PORT:CATalog?

2.5.15 STATus:PORT:NTRansition

Syntax	STATus:PORT:NTRansition <port name>,<enable>
Description	This command sets the negative transition filter for the port event register.
Parameters	<port name> = <CHARACTER PROGRAM DATA> Expression format: Character list
	<enable> = <BOOLEAN PROGRAM DATA>
Response	None.
Example	`STAT:PORT:NTR 1-PORT1,OFF`
Note	<port name> is the same as one of the returned by the INSTrument:PORT:CATalog?

Syntax	STATus:PORT:NTRansition? <port name>
Description	This query returns the negative transition filter for the port event register.
Parameter	<port name> = <CHARACTER PROGRAM DATA> Expression format: Character list
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`STAT:PORT:NTR? 1-PORT1` → `0`
Note	<port name> is the same as one of the returned by the INSTrument:PORT:CATalog?

2.5.16 STATus:PRESet

Syntax	STATus:PRESet
Description	For the instrument-dependent status data structures, the PRESet commands sets the enable register to all 1's and the transition filter register to recognize only positive transitions. For the SCPI-mandated status structures (operation, questionable and port status) the PRESet command sets the transition filter registers to recognize only positive transitions and set the enable registers to 0's. This command does not affect either the Status Byte or the Standard Event Status register. PRESet does not clear any of the event registers or any item from the error or event queues.
Parameter	None.
Response	None.
Example	`STAT:PRES`
Note	All active sessions has their own set of registers. This command affects registers in all connected application servers.

2.5.17 STATus:INTerface:PORT<Pt>[:EVENt]?

Syntax	STATus:INTerface:PORT<Pt>[:EVENt]?
Description	This query returns and clears the device dependent interface event status register. Events are summarized in Status Byte DB1 for port, DB2 for port 2 and so on.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> The bits and their values for the register: DB1 (1) = 2 Mbps Summary DB2 (2) = SDH/SONET Summary DB3 (4) = E3 Summary DB4 (8) = E4 Summary DB5 (16) = NOT USED. Reserved for V-Series/Codir. Summary DB6 (32) = Ethernet Summary DB7 (64) = T1 Summary DB8 (128) = OTN Summary DB9 (256) = NoFrame Summary DB10 (512) = Physical Summary DB11 (1024) = T3 Summary DB12 (2048) = Fibre Channel Summary DB13 (4096) = CPRI Summary DB14 - DB16 = NOT USED
Example	`STAT:INT:PORT1?` → `16`
Note	There must be a connected application server for this command to be recognized as a legal command.

2.5.18 STATus:INTerface:PORT<Pt>:CONDition?

Syntax	STATus:INTerface:PORT<Pt>:CONDition?
Description	This query returns the instrument dependent interface status condition register. This register summarize all events/alarms from the different interfaces.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> The bits and their values for the register: DB1 (1) = 2 Mbps Summary DB2 (2) = SDH/SONET Summary DB3 (4) = E3 Summary DB4 (8) = E4 Summary DB5 (16) = NOT USED. Reserved for V-Series/Codir. Summary DB6 (32) = Ethernet Summary DB7 (64) = T1 Summary DB8 (128) = OTN Summary DB9 (256) = NoFrame Summary DB10 (512) = Physical Summary DB11 (1024) = T3 Summary DB12 (2048) = Fibre Channel Summary DB13 (4096) = CPRI Summary DB14 - DB16 = NOT USED
Example	`STAT:INT:PORT1:COND?` → `16`
Note	There must be a connected application server for this command to be recognized as a legal command.

2.6 Mass Memory Subsystem Commands

The commands in this section operates on files and directories placed in the following storage areas:

Location	Description
`Internal/`	The internal storage of the Network Master.
`Usb/`	An USB connected storage device. This location is only accessible when a USB storage device is mounted.
`Internal/remote/`	A remote network drive. Refer to the User Manual for information on how to connect to an external storage location. This location is only accessible when external storage is configured and the Network Master is able to connect to it.

Files must be located in one of the locations described in the table above - or a in a sub-directory of one of these.

2.6.1 MMEMory:LOAD

Syntax	MMEMory:LOAD <file>
Description	This command loads a file into the currently selected application server. The file may contain settings only or both settings and results data.
Parameter	<file> = <STRING PROGRAM DATA> The path and name of the file to be loaded.
Response	None
Example	`MMEM:LOAD "Internal/SAT-settings.cfg"`
Note	There must be a connected application server for this command to be recognized as a legal command.
	The application server must be in the idle state and the content of the loaded file must match the application server type.

2.6.2 MMEMory:STORe:STATe

Syntax	MMEMory:STORe:STATe <file>
Description	This command stores the current settings to a file on the instrument.
Parameter	<file> = <STRING PROGRAM DATA> The path and name of the file to store the data.
Response	None
Example	`MMEM:STOR:STAT "Internal/my-bert-settings.cfg"`
Note	There must be a connected application server for this command to be recognized as a legal command.
	The application server must be in the idle state.

2.6.3 MMEMory:STORe:DATA

Syntax	MMEMory:STORe:DATA <file>
Description	This command stores the current settings and result data to a file on the instrument.
Parameter	<file> = <STRING PROGRAM DATA> The path and name of the file to store the data.
Response	None
Example	`MMEM:STOR:DATA "Usb/my-bert-result.res"`
Note	There must be a connected application server for this command to be recognized as a legal command.
	The application server must be in the idle state.

2.6.4 MMEMory:DELete

Syntax	MMEMory:DELete <file>
Description	This command deletes a file.
Parameter	<file> = <STRING PROGRAM DATA> The path to the file to be deleted.
Response	None.
Example	`MMEM:DEL "Internal/report.pdf"`
Note	

2.6.5 MMEMory:DATA?

Syntax	MMEMory:DATA? <file>
Description	This command retrieves a file.
Parameter	<file> = <STRING PROGRAM DATA> The path to the file to be retrieved.
Response	<DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA> = #<nonzero digit><digits><8 bit data bytes>, where: <nonzero digit> is a single ASCII character in the range of '1'-'9'. It represents the length of <digits> in number of bytes. <digits> is a number of ASCII characters in the range of '0'-'9', which together are a decimal representation of the number of succeeding data bytes.
Example	`MMEM:DATA? "Internal/report.pdf"` → #49137<9137 bytes of binary data>
Note	This command cannot be used together with other commands in a compound command.

2.6.6 MMEMory:COPY

Syntax	MMEMory:COPY <source-file>,<destination-file>
Description	This command copies a file.
Parameter	<source-file> = <STRING PROGRAM DATA> The path to the file to be copied.
	<destination-file> = <STRING PROGRAM DATA> The path to the new file.
Response	None.
Example	`MMEM:COPY "Internal/report.pdf","Usb/report.pdf"`
Note	

2.6.7 MMEMory:MOVE

Syntax	MMEMory:MOVE <old-file>,<new-file>
Description	This command moves or renames a file.
Parameter	<old-file> = <STRING PROGRAM DATA> The path to the file to be moved or renamed.
	<new-file> = <STRING PROGRAM DATA> The new path to the file.
Response	None.
Example	`MMEM:MOVE "Internal/report.pdf","Usb/report.pdf"`
Note	

2.6.8 MMEMory:INFO?

Syntax	MMEMory:INFO? <file>
Description	This command retrieves information about a file.
Parameter	<file> = <STRING PROGRAM DATA> The path to the file to retrieve file information about.
Response	<file-date-time> = <STRING RESPONSE DATA> Last file modification date.
	<file-size> = <NR1 NUMERIC RESPONSE DATA> The file size in bytes.
Example	`MMEM:INFO? "Internal/report.pdf"` → `"2015-05-29 16:02:20",9137`
Note	

2.6.9 MMEMory:CATalog?

Syntax	MMEMory:CATalog? <directory>[,<pattern>]
Description	This command lists the files present in a directory.
Parameters	<directory> = <STRING PROGRAM DATA> The path to the directory to be listed.
	<pattern> = <STRING PROGRAM DATA> An optional case sensitive file name pattern. Wildcard characters are * and ?.
Response	({<item>} + {,}∗) = <EXPRESSION RESPONSE DATA> A list of quoted file and directory names.
Example	`MMEM:CAT? "Internal/reports"` → `("report.pdf","setup.cfg")`
Note	

2.6.10 MMEMory:DCATalog?

Syntax	MMEMory:DCATalog? <directory>
Description	This command lists the sub-directories present in a directory.
Parameter	<directory> = <STRING PROGRAM DATA> The path to the directory to be listed.
Response	({<directory>} + {,}∗) = <EXPRESSION RESPONSE DATA> A list for quoted directory names.
Example	`MMEM:DCAT? "Internal/"` → `("diagnostics","favorites","logs","remote","screens","windowsinstaller")`
Note	

2.6.11 MMEMory:MDIRectory

Syntax	MMEMory:MDIRectory <directory>
Description	This command makes a new sub-directory.
Parameter	<directory> = <STRING PROGRAM DATA> The path to the directory to be created.
Response	None.
Example	`MMEM:MDIR "Internal/reports"`
Note	

2.6.12 MMEMory:RDIRectory

Syntax	MMEMory:RDIRectory <directory>[,<force>]
Description	This command removes an existing directory.
Parameter	<directory> = <STRING PROGRAM DATA> The path to the directory to be created.
	<force> = <BOOLEAN PROGRAM DATA> If set to `ON` then ALL CONTENTS i.e. files and sub-directories will be deleted.
Response	None.
Example	`MMEM:RDIR "Internal/reports"`
Note	None.

2.6.13 MMEMory:SAVE

Syntax	MMEMory:SAVE <enable>
Description	This command enables or disables auto saving configuration when the application is terminated
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.

Example	`INST:STAR:LAST TP-BERT-OTN,1-PORT1`
	`MMEM:SAVE ON`
	`INST? → 2`
	`INST:TERM 2`

Note	

Syntax	MMEMory:SAVE?
Description	This query returns whether or not auto saving configuration enabled when the application is terminated .
Parameter	None.
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`MMEM:SAVE? → 1`
Note	

Chapter 3

No frame

3.1 Interface Setup

3.1.1 NFRame:PORT<Pt>:INTerface:TYPE

Syntax	NFRame:PORT<Pt>:INTerface:TYPE <type>
Description	This command sets the interface type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> OFF QSFP: QSFP optical interface (40 Gbps) CXP: CXP optical interface (100 Gbps) CFP: CFP optical interface (40/100 Gbps) CFP2: CFP2 optical interface (100 Gbps) QSFP28ADpt: CFP2-QSFP28 Adaptor interface (100 Gbps) CFP4: CFP4 optical interface (100 Gbps) QSFP28: QSFP28 optical interface (100 Gbps) SFP28: SFP28 optical interface (25 Gbps) *DEFault = OFF*
Response	None.
Example	`NFR:PORT1:INT:TYPE QSFP`
Note	

Syntax	NFRame:PORT<Pt>:INTerface:TYPE?
Description	This query returns the interface type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`NFR:PORT1:INT:TYPE?` → `QSFP`
Note	

3.1.2 NFRame:PORT<Pt>:INTerface:BITRate

Syntax	NFRame:PORT<Pt>:INTerface:BITRate <bitrate>
Description	This command sets the interface bitrate.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> E100G: 100G Ethernet E40G: 40G Ethernet E25G: 25G Ethernet STM256: STM256/OC768 OTU4 OTU3 OTU3E1 OTU3E2 *DEFault = E100G*
Response	None.
Example	`NFR:PORT1:INT:BITR E100G`
Note	

Syntax	NFRame:PORT<Pt>:INTerface:BITRate?
Description	This query returns the interface bitrate.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`NFR:PORT1:INT:BITR?` → `E100G`
Note	

3.1.3 NFRame:PORT<Pt>:INTerface:LANE

Syntax	NFRame:PORT<Pt>:INTerface:LANE <type>
Description	This command sets the lane type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> LANE10: 10 Lanes LANE20: 20 Lanes LANE4 : 4 Lanes LANE1 : 1 Lane *DEFault = LANE10*
Response	None.
Example	`NFR:PORT1:INT:LANE LANE10`
Note	

Syntax	NFRame:PORT<Pt>:INTerface:LANE?
Description	This query returns the lane type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`NFR:PORT1:INT:LANE?` → `LANE10`
Note	

3.1.4 NFRame:PORT<Pt>:TIMing

Syntax	NFRame:PORT<Pt>:TIMing <source>
Description	This command sets the timing source.
Parameters	<Pt> = Port number
	<source> = <CHARACTER PROGRAM DATA> INTernal: Internal clock. EXTernal: External clock. *DEFault = INT*
Response	None.
Example	`NFR:PORT1:TIM INT`
Note	

Syntax	NFRame:PORT<Pt>:TIMing?
Description	This query returns the timing source.
Parameter	<Pt> = Port number
Response	<source> = <CHARACTER RESPONSE DATA>
Example	`NFR:PORT1:TIM? → INT`
Note	

3.1.5 NFRame:PORT<Pt>[:RX]:INTerface:PATTern:FOLLow

Syntax	NFRame:PORT<Pt>[:RX]:INTerface:PATTern:FOLLow <follow>
Description	This command sets the receiver to follow transmitter setup or not to follow.
Parameters	<Pt> = Port number
	<follow> = <CHARACTER PROGRAM DATA> OFF: Do not follow ON: Follow the transmitter setup *DEFault = ON*
Response	None.
Example	`NFR:PORT1:INT:PATT:FOLL ON`
Note	

Syntax	NFRame:PORT<Pt>[:RX]:INTerface:PATTern:FOLLow?
Description	This query returns if the receiver follow transmitter setup.
Parameter	<Pt> = Port number
Response	<follow> = <CHARACTER RESPONSE DATA>
Example	`NFR:PORT1:INT:PATT:FOLL? → ON`
Note	

3.1.6 NFRame:PORT<Pt>:TX:INTerface:PATTern

Syntax	NFRame:PORT<Pt>:TX:INTerface:PATTern <type>
Description	This command sets the pattern type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> PRBS7 PRBS9 PRBS15 PRBS23 PRBS31 SWAVe: Square Wave *DEFault = PRBS7*
Response	None.
Example	`NFR:PORT1:TX:INT:PATT PRBS31`
Note	This command sets all lanes when 4Lanes is selected

Syntax	NFRame:PORT<Pt>:TX:INTerface:PATTern?
Description	This query returns the pattern type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`NFR:PORT1:TX:INT:PATT?` → `PRBS31`
Note	

3.1.7 NFRame:PORT<Pt>:RX:INTerface:PATTern

Syntax	NFRame:PORT<Pt>:RX:INTerface:PATTern <type>
Description	This command sets the pattern type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> PRBS7 PRBS9 PRBS15 PRBS23 PRBS31 *DEFault = PRBS7*
Response	None.
Example	`NFR:PORT1:RX:INT:PATT PRBS31`
Note	This command sets all lanes when 4Lanes is selected

Syntax	NFRame:PORT<Pt>:RX:INTerface:PATTern?
Description	This query returns the pattern type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`NFR:PORT1:RX:INT:PATT?` → `PRBS31`
Note	

3.1.8 NFRame:PORT<Pt>:TX:INTerface:PINVersion

Syntax	NFRame:PORT<Pt>:TX:INTerface:PINVersion <inversion>
Description	This command enables or disables pattern inversion.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`NFR:PORT1:TX:INT:PINV ON`
Note	This command sets all lanes when 4Lanes is selected

Syntax	NFRame:PORT<Pt>:TX:INTerface:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the pattern.
Parameter	<Pt> = Port number
Response	<inversion> = <CHARACTER RESPONSE DATA>
Example	`NFR:PORT1:TX:INT:PINV?` → `1`
Note	

3.1.9 NFRame:PORT<Pt>:RX:INTerface:PINVersion

Syntax	NFRame:PORT<Pt>:RX:INTerface:PINVersion <inversion>
Description	This command enables or disables pattern inversion.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`NFR:PORT1:RX:INT:PINV ON`
Note	This command sets all lanes when 4Lanes is selected

Syntax	NFRame:PORT<Pt>:RX:INTerface:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the pattern.
Parameter	<Pt> = Port number
Response	<inversion> = <CHARACTER RESPONSE DATA>
Example	`NFR:PORT1:RX:INT:PINV? → 1`
Note	

3.1.10 NFRame:PORT<Pt>:TX:INTerface:LANE<Ln>:PATTern

Syntax	NFRame:PORT<Pt>:TX:INTerface:LANE<Ln>:PATTern <type>
Description	This command sets the pattern type(each lane).
Parameters	<Pt> = Port number
	<Ln> = Lane number (1-4)
	<type> = <CHARACTER PROGRAM DATA> PRBS7 PRBS9 PRBS15 PRBS23 PRBS31 *DEFault = PRBS7*
Response	None.
Example	`NFR:PORT1:TX:INT:LANE2:PATT PRBS31`
Note	This command can be used on 4Lanes

Syntax	NFRame:PORT<Pt>:TX:INTerface:LANE<Ln>:PATTern?
Description	This query returns the pattern type(each lane).
Parameter	<Pt> = Port number
	<Ln> = Lane number (1-4)
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`NFR:PORT1:TX:INT:LANE2:PATT? → PRBS31`
Note	

3.1.11 NFRame:PORT<Pt>:RX:INTerface:LANE<Ln>:PATTern

Syntax	NFRame:PORT<Pt>:RX:INTerface:LANE<Ln>:PATTern <type>
Description	This command sets the pattern type(each lane).
Parameters	<Pt> = Port number
	<Ln> = Lane number (1-4)
	<type> = <CHARACTER PROGRAM DATA> PRBS7 PRBS9 PRBS15 PRBS23 PRBS31 *DEFault = PRBS7*
Response	None.
Example	`NFR:PORT1:RX:INT:LANE2:PATT PRBS31`
Note	This command can be used on 4Lanes

Syntax	NFRame:PORT<Pt>:RX:INTerface:LANE<Ln>:PATTern?
Description	This query returns the pattern type(each lane).
Parameter	<Pt> = Port number
	<Ln> = Lane number (1-4)
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`NFR:PORT1:RX:INT:LANE2:PATT?` → `PRBS31`
Note	

3.1.12 NFRame:PORT<Pt>:TX:INTerface:LANE<Ln>:PINVersion

Syntax	NFRame:PORT<Pt>:TX:INTerface:LANE<Ln>:PINVersion <inversion>
Description	This command enables or disables pattern inversion(each lane).
Parameters	<Pt> = Port number
	<Ln> = Lane number (1-4)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`NFR:PORT1:TX:INT:LANE2:PINV ON`
Note	This command can be used on 4Lanes

Syntax	NFRame:PORT<Pt>:TX:INTerface:LANE<Ln>:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the pattern(each lane).
Parameter	<Pt> = Port number
	<Ln> = Lane number (1-4)
Response	<inversion> = <CHARACTER RESPONSE DATA>
Example	`NFR:PORT1:TX:INT:LANE2:PINV?` → `1`
Note	

3.1.13 NFRame:PORT<Pt>:RX:INTerface:LANE<Ln>:PINVersion

Syntax	NFRame:PORT<Pt>:RX:INTerface:LANE<Ln>:PINVersion <inversion>
Description	This command enables or disables pattern inversion(each lane).
Parameters	<Pt> = Port number
	<Ln> = Lane number (1-4)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`NFR:PORT1:RX:INT:LANE2:PINV ON`
Note	This command can be used on 4Lanes

Syntax	NFRame:PORT<Pt>:RX:INTerface:LANE<Ln>:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the pattern(each lane).
Parameter	<Pt> = Port number
	<Ln> = Lane number (1-4)
Response	<inversion> = <CHARACTER RESPONSE DATA>
Example	`NFR:PORT1:RX:INT:LANE2:PINV?` → `1`
Note	

3.1.14 NFRame:PORT<Pt>:TX:INTerface:PATTern:LANE:FOLLow

Syntax	NFRame:PORT<Pt>:TX:INTerface:PATTern:LANE:FOLLow <follow>
Description	This command sets the transmitter follow the pattern of lane2 to lane4 from lane1 or not to follow.
Parameters	<Pt> = Port number
	<follow> = <CHARACTER PROGRAM DATA> OFF: Do not follow ON: Follow the lane2 to lane4 from lane1 *DEFault = OFF*
Response	None.
Example	`NFR:PORT1:TX:INT:PATT:LANE:FOLL ON`
Note	

Syntax	NFRame:PORT<Pt>:TX:INTerface:PATTern:LANE:FOLLow?
Description	This query returns if the transmitter follow pattern setup.
Parameter	<Pt> = Port number
Response	<follow> = <CHARACTER RESPONSE DATA>
Example	`NFR:PORT1:TX:INT:PATT:LANE:FOLL?` → `ON`
Note	

3.1.15 NFRame:PORT<Pt>:RX:INTerface:PATTern:LANE:FOLLow

Syntax	NFRame:PORT<Pt>:RX:INTerface:PATTern:LANE:FOLLow <follow>
Description	This command sets the receiver follow the pattern of lane2 to lane4 from lane1 or not to follow.
Parameters	<Pt> = Port number
	<follow> = <CHARACTER PROGRAM DATA> OFF: Do not follow ON: Follow the lane2 to lane4 from lane1 *DEFault = OFF*
Response	None.
Example	`NFR:PORT1:RX:INT:PATT:LANE:FOLL ON`
Note	

Syntax	NFRame:PORT<Pt>:RX:INTerface:PATTern:LANE:FOLLow?
Description	This query returns if the receiver follow pattern setup.
Parameter	<Pt> = Port number
Response	<follow> = <CHARACTER RESPONSE DATA>
Example	`NFR:PORT1:RX:INT:PATT:LANE:FOLL?` → `ON`
Note	

3.2 Stimuli

3.2.1 NFRame:PORT<Pt>:STIMuli:TX:FOFFset

Syntax	NFRame:PORT<Pt>:STIMuli:TX:FOFFset <offset>
Description	This command sets the frequency offset for the clock source. Unit: ppm.
Parameters	<Pt> = Port number
	<offset> = <NUMERIC PROGRAM DATA> *MINimum=-200.0, MAXimum=200.0, DEFault = 0*
Response	None.
Example	`NFR:PORT1:STIM:TX:FOFF 10.0`
Note	

Syntax	NFRame:PORT<Pt>:STIMuli:TX:FOFFset?
Description	This query returns the frequency offset (ppm) for the clock source.
Parameter	<Pt> = Port number
Response	<offset> = <NR2 NUMERIC RESPONSE DATA>
Example	`NFR:PORT1:STIM:TX:FOFF?` → `10.0`
Note	

3.2.2 NFRame:PORT<Pt>:STIMuli:TX:LANE

Syntax	NFRame:PORT<Pt>:STIMuli:TX:LANE <lane>
Description	This command sets the bit error insertion lane.
Parameters	<Pt> = Port number
	<lane> = <NUMERIC PROGRAM DATA> 4 lanes: *MINimum=#B0, MAXimum=#B1111, DEFault=#B0* 10 lanes: *MINimum=#B0, MAXimum=#B1111111111, DEFault=#B0* 20 lanes: *MINimum=#B0, MAXimum=#B11111111111111111111, DEFault=#B0*
Response	None.
Example	`NFR:PORT1:STIM:TX:LANE #B101` This command add error into lane 0 and 2.
Note	

Syntax	NFRame:PORT<Pt>:STIMuli:TX:LANE?
Description	This query returns the bit error insertion lane.
Parameter	<Pt> = Port number
Response	<lane> = <BINARY NUMERIC RESPONSE DATA>
Example	`NFR:PORT1:STIM:TX:LANE?` → `#B1010` → `#B1010000000` → `#B10100000000000000000`
Note	

3.3 Result

3.3.1 NFRame:PORT<Pt>[:RX]:IFETch?

Syntax	NFRame:PORT<Pt>[:RX]:IFETch? <parameter>
Description	This query fetches an NoFrame interval if available.
Parameters	<Pt> = Port number
	({<parameter>} + {,}∗) = <EXPRESSION PROGRAM DATA> The response format is listed for each parameter. **Alarms** LOS: Loss of signal. Response: <Seconds>,<Ratio>

continued on next page...

. . . continued from previous page

	Errors LSSx: Loss of signal synchronization of lane x (x = 0-19). Response: <Seconds>,<Ratio> ERRPRBS: PRBS Error (total). Response: <Count>,<Ratio> ERRPRBSx: PRBS Error of lane x (x = 0-19). Response: <Count>,<Ratio> **Rx Frequency** FREQDEV: Frequency deviation. Response: <ppm> FREQDEVx: Frequency deviation of lane x (x = 0-19). Response: <ppm>
Response	{(<result>), }* = <EXPRESSION RESPONSE DATA> Expression format: Numeric List Each result is formatted according to the specification in the parameter field. Values that are not relevant or applicable for the current measurement return NaN (section 1.6.1).
Example	`NFR:PORT1:IFET? (LOS,LSS0)` → `(3,0.00532),(4,0.00709)`
Notes	This command fetches the results from the interval selected using the `MEASurement:SETup:SELect` command (see section 18.2.2). If the requested result is not available, NaN (section 1.6.1) is returned. If there is one or more results, the last ”,” is always removed.

3.4 Status

3.4.1 NFRame:STATus:PORT<Pt>[:RX]:AESummary[:EVENt]?

Syntax	NFRame:STATus:PORT<Pt>[:RX]:AESummary[:EVENt]?
Description	This query returns the NoFrame alarms and errors summary event register. The content of this event register is summarized in DB9 of the STATus:INTerface:PORT<Pt>:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 - DB8 = NOT USED DB9 (256) = Error summary DB10 - DB16 = NOT USED
Example	`NFR:STAT:PORT1:AES?` → `1`
Note	

3.4.2 NFRame:STATus:PORT<Pt>[:RX]:AESummary:CONDition?

Syntax	NFRame:STATus:PORT<Pt>[:RX]:AESummary:CONDition?
Description	This query returns the NoFrame alarms and errors summary condition register. DB1 is summary of NFRame:STATus:PORT<Pt>[:RX]:ALARm[:EVENt]? DB9 is summary of NFRame:STATus:PORT<Pt>[:RX]:ERRor[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 - DB8 = NOT USED DB9 (256) = Error summary DB10 - DB16 = NOT USED
Example	`NFR:STAT:PORT1:AES:COND?` → `2`
Note	

3.4.3 NFRame:STATus:PORT<Pt>[:RX]:ALARm[:EVENt]?

Syntax	NFRame:STATus:PORT<Pt>[:RX]:ALARm[:EVENt]?
Description	This query returns the alarms event register. The content of this register is summarized in DB1 of the NFRame:STATus:PORT<Pt>:AESummary:CONDition register.
Parameters	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = LOS, Loss of signal DB2 (2) = LSS, Loss of signal synchronization DB3 (4) = CDRL, CDR Lock status DB4 - DB16 = NOT USED
Example	`NFR:STAT:PORT1:ALAR?` → `2`
Notes	

3.4.4 NFRame:STATus:PORT<Pt>[:RX]:ALARm:CONDition?

Syntax	NFRame:STATus:PORT<Pt>[:RX]:ALARm:CONDition?
Description	This query returns the alarms condition register.
Parameters	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = LOS, Loss of signal DB2 (2) = LSS, Loss of signal synchronization DB3 (4) = CDRL, CDR Lock status DB4 - DB16 = NOT USED
Example	`NFR:STAT:PORT1:ALAR:COND?` → `4`
Notes	

3.4.5 NFRame:STATus:PORT<Pt>[:RX]:ERRor[:EVENt]?

Syntax	NFRame:STATus:PORT<Pt>[:RX]:ERRor[:EVENt]?
Description	This query returns the errors event register. The content of this register is summarized in DB9 of the NFRame:STATus:PORT<Pt>:AESummary:CONDition register.
Parameters	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = PRBS Error DB2 - DB16 = NOT USED
Example	`NFR:STAT:PORT1:ERR?` → `1`
Notes	

3.4.6 NFRame:STATus:PORT<Pt>[:RX]:ERRor:CONDition?

Syntax	NFRame:STATus:PORT<Pt>[:RX]:ERRor:CONDition?
Description	This query returns the errors condition register.
Parameters	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = PRBS Error DB2 - DB16 = NOT USED
Example	`NFR:STAT:PORT1:ERR:COND?` → `1`
Notes	

3.4.7 NFRame:STATus:PORT<Pt>[:RX]:PSLevel?

Syntax	NFRame:STATus:PORT<Pt>[:RX]:PSLevel?
Description	This query returns the physical signal level. Unit: dBm.
Parameters	<Pt> = Port number
Response	<signallevel> = <STRING RESPONSE DATA> "<power> dBm": Min: "< <min> dBm", Max: "Exceeds Level" "N/A": Module not present or not ready.
Example	`NFR:STAT:PORT1:PSL?` → `"-3 dBm"`
Note	

3.4.8 NFRame:STATus:PORT<Pt>:TX:PSLevel?

Syntax	NFRame:STATus:PORT<Pt>:TX:PSLevel?
Description	This query returns the physical signal level. Unit: dBm.
Parameters	<Pt> = Port number
Response	<signallevel> = <STRING RESPONSE DATA> "<power> dBm": Min: "< <min> dBm", Max: "Exceeds Level" "N/A": Module not present or not ready.
Example	`NFR:STAT:PORT1:TX:PSL?` → `"-3 dBm"`
Note	

3.4.9 NFRame:STATus:PORT<Pt>[:RX]:PFRequency?

Syntax	NFRame:STATus:PORT<Pt>[:RX]:PFRequency? [<lane>]
Description	This query returns the physical frequency. Unit: Hz.
Parameters	<Pt> = Port number
	<lane> = Lane number (0-9)
Response	<frequency> = <NR1 NUMERIC RESPONSE DATA>
Example	`NFR:STAT:PORT1:PFR?` → `103125000192` [1] `NFR:STAT:PORT1:PFR? 0` → `5250845696`
Note	[1] <lane> is not used if the Lane Select is 20 Lanes.

3.4.10 NFRame:STATus:PORT<Pt>[:RX]:PDEViation?

Syntax	NFRame:STATus:PORT<Pt>[:RX]:PDEViation? [<lane>]
Description	This query returns the physical frequency deviation. Unit: ppm.
Parameters	<Pt> = Port number
	<lane> = Lane number (0-9)
Response	<deviation> = <NR2 NUMERIC RESPONSE DATA>
Example	`NFR:STAT:PORT1:PDEV?` → `0.0` [1] `NFR:STAT:PORT1:PDEV? 0` → `0.0`
Note	[1] <lane> is not used if the Lane Select is 20 Lanes.

3.4.11 NFRame:STATus:PORT<Pt>[:RX]:PCDRlock?

Syntax	NFRame:STATus:PORT<Pt>[:RX]:PCDRlock? [<lane>]
Description	This query returns the physical CDR lock status.
Parameters	<Pt> = Port number
	<lane> = Lane number (0-9)
Response	<status> = <BOOLEAN RESPONSE DATA> 0: CDR Locked 1: CDR Unlocked
Example	`NFR:STAT:PORT1:PCDR?` → `0` [1] `NFR:STAT:PORT1:PCDR? 0` → `0`
Note	[1] <lane> is not used if the Lane Select is 20 Lanes.

Chapter 4

CPRI/OBSAI

4.1 Port

4.1.1 CPRI:PORT<Pt>:MODE

Syntax	CPRI:PORT<Pt>:MODE <mode>
Description	This command sets the port mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: Port off NORMal: Normal THRough: (1 Port / 2 Port) Pass Through Mode *DEFault = OFF*
Response	None.
Example	`CPRI:PORT1:MODE NORM`
Note	This command is disabled when over OTN is selected.

Syntax	CPRI:PORT<Pt>:MODE?
Description	This query returns the port mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:MODE?` → `NORM`
Note	This command is disabled when over OTN is selected.

4.1.2 CPRI:PORT<Pt>:TIMing:SOURce

Syntax	CPRI:PORT<Pt>:TIMing:SOURce <source>
Description	This command sets Timing source.
Parameters	<Pt> = Port number
	<source> = <CHARACTER PROGRAM DATA> INTernal: Internal clock EXTernal: External clock GPS: GPS signal RX: Received clock *DEFault = INTernal*
Response	None.
Example	`CPRI:PORT1:TIM:SOUR INT`
Note	

Syntax	CPRI:PORT<Pt>:TIMing:SOURce?
Description	This query returns the timing source
Parameter	<Pt> = Port number
Response	<source> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:TIM:SOUR?` → `INT`
Note	

4.1.3 CPRI:PORT<Pt>:LRATe

Syntax	CPRI:PORT<Pt>:LRATe <rate>
Description	This command set Line Rate.
Parameters	<Pt> = Port number
	<rate> = <CHARACTER PROGRAM DATA> 614M: 614.4Mbps(CPRI) 768M: 768.0Mbps(OBSAI) 1228M: 1228.8Mbps(CPRI) 1536M: 1536.0Mbps(OBSAI) 2457M: 2457.6Mbps(CPRI) 3072M: 3072.0Mbps(CPRI/OBSAI) 4915M: 4915.2Mbps(CPRI) 6144M: 6144.0Mbps(CPRI/OBSAI) 9830M: 9830.4Mbps(CPRI) 10137M: 10137.6Mbps(CPRI) 12165M: 12165.12Mbps(CPRI) 24330M: 24330.24Mbps(CPRI) *DEFault = 614M*
Response	None.
Example	`CPRI:PORT1:LRAT 614M`
Note	This command is disabled when over OTN is selected. When selecting 768M or 4915M, the contents of signal becomes unframed.

Syntax	CPRI:PORT<Pt>:LRATe?
Description	This query returns Line rate.
Parameter	<Pt> = Port number
Response	<rate> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:LRAT?` → `614M`
Note	This command is disabled when over OTN is selected.

4.1.4 CPRI:PORT<Pt>:LRATe:MODE

Syntax	CPRI:PORT<Pt>:LRATe:MODE <mode>
Description	This command set Line rate mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> CPRI: CPRI OBSAI: OBSAI *DEFault = CPRI*
Response	None.
Example	`CPRI:PORT1:LRAT:MODE OBSAI`
Note	This command is disabled when over OTN is selected. When selecting OBSAI, the contents of signal becomes unframed.

Syntax	CPRI:PORT<Pt>:LRATe:MODE?
Description	This query returns Line rate mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:LRAT:MODE?` → `OBSAI`
Note	This command is disabled when over OTN is selected.

4.1.5 CPRI:PORT<Pt>:CONTents

Syntax	CPRI:PORT<Pt>:CONTents <contents>
Description	This command sets contents.
Parameter	<Pt> = Port number
	<contents> = <CHARACTER PROGRAM DATA> UFRamed: Unframed LINK: CPRI Link OBLINK: OBSAI Link
Response	None.
Example	`CPRI:PORT1:CONT UFR`
Note	

Syntax	CPRI:PORT<Pt>:CONTents?
Description	This query returns contents.
Parameter	<Pt> = Port number
Response	<contents> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:CONT?` → `UFR`
Note	

4.1.6 CPRI:PORT<Pt>:FEC

Syntax	CPRI:PORT<Pt>:FEC <mode>
Description	The command sets the FEC mode
Parameter	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`CPRI:PORT1:FEC ON`
Note	This command is available when the line rate is 12165.12 Mbps or 24330.24 Mbps.

Syntax	CPRI:PORT<Pt>:FEC?
Description	This query returns the FEC mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:FEC?` → `1`
Note	

4.1.7 CPRI:PORT<Pt>:PATTern

Syntax	CPRI:PORT<Pt>:PATTern <type>
Description	This command sets Pattern type.
Parameter	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> PRBS15: PRBS15 pattern PRBS20: PRBS20 pattern PRBS23: PRBS23 pattern PRBS31: PRBS31 pattern USER32BIT: User Pattern (32bit) OFF: Pattern OFF
Response	None.
Example	`CPRI:PORT1:PATT PRBS15`
Note	This command supports CPRI and OBSAI mode.

Syntax	CPRI:PORT<Pt>:PATTern?
Description	This query returns pattern type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:PATT?` → `PRBS15`
Note	

4.1.8 CPRI:PORT<Pt>:PINVersion

Syntax	CPRI:PORT<Pt>:PINVersion <inversion>
Description	This command sets Inversion.
Parameter	<Pt> = Port number
	<inversion> = <CHARACTER PROGRAM DATA> NORMal: Invert Off INVerted: Invert On
Response	None.
Example	`CPRI:PORT1:PINV NORM`
Note	This command supports CPRI and OBSAI mode.

Syntax	CPRI:PORT<Pt>:PINVersion?
Description	This query returns inversion or normal
Parameter	<Pt> = Port number
Response	<inversion> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:PINV?` → `NORM`
Note	

4.1.9 CPRI:PORT<Pt>:UP32

Syntax	CPRI:PORT<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `CONTent` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters (one bit resolution).
Response	None.
Example	`CPRI:PORT1:UP32 "0110"`
Note	This command supports CPRI and OBSAI mode.

Syntax	CPRI:PORT<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`CPRI:PORT1:UP32?` → `"0110"`
Note	

4.1.10 CPRI:PORT<Pt>:PVERsion

Syntax	CPRI:PORT<Pt>:PVERsion <version>
Description	This command sets protocol version.
Parameter	<Pt> = Port number
	<version> = <CHARACTER PROGRAM DATA> PVER1: Protocol version 1 (= Scrambling OFF) PVER2: Protocol version 2 (= Scrambling ON)
Response	None.
Example	`CPRI:PORT1:PVER PVER1`
Note	This command supports CPRI and OBSAI mode.

Syntax	CPRI:PORT<Pt>:PVERsion?
Description	This query returns protocol version.
Parameter	<Pt> = Port number
Response	<version> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:PVER?` → `PVER1`
Note	

4.1.11 CPRI:PORT<Pt>:PROLe

Syntax	CPRI:PORT<Pt>:PROLe <role>
Description	This command sets port role.
Parameter	<Pt> = Port number
	<role> = <CHARACTER PROGRAM DATA> MASTer: Master SLAVe: Slave
Response	None.
Example	`CPRI:PORT1:PROL MAST`
Note	This command supports CPRI mode.

Syntax	CPRI:PORT<Pt>:PROLe?
Description	This query returns port role.
Parameter	<Pt> = Port number
Response	<role> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:PROL?` → `MAST`
Note	

4.1.12 CPRI:PORT<Pt>:SSEQence

Syntax	CPRI:PORT<Pt>:SSEQence <sequence>
Description	This command enables/disables start-up sequence.
Parameter	<Pt> = Port number
	<sequence> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`CPRI:PORT1:SSEQ OFF`
Note	This command supports CPRI mode.

Syntax	CPRI:PORT<Pt>:SSEQence?
Description	This query returns enables/disables start-up sequence.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`CPRI:PORT1:SSEQ?` → `0`
Note	This command supports CPRI mode.

4.1.13 CPRI:PORT<Pt>:CMHDlc

Syntax	CPRI:PORT<Pt>:CMHDlc <hdlc>
Description	This command sets CM HDLC.
Parameter	<Pt> = Port number
	<hdlc> = <CHARACTER PROGRAM DATA> NONE: NONE 240K: 240k 480K: 480k 960K: 960k 1920K: 1920k 2400K: 2400k HIGH: Highest Possible
Response	None.
Example	`CPRI:PORT1:CMHD NONE`
Note	This command supports CPRI mode.

Syntax	CPRI:PORT<Pt>:CMHDlc?
Description	This query returns CM HDLC.
Parameter	<Pt> = Port number
Response	<hdlc> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:CMHD?` → `NONE`
Note	

4.1.14 CPRI:PORT<Pt>:EENable

Syntax	CPRI:PORT<Pt>:EENable <enable>
Description	This command sets Ethernet Enable.
Parameter	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`CPRI:PORT1:EEN OFF`
Note	This command supports CPRI mode.

Syntax	CPRI:PORT<Pt>:EENable?
Description	This query returns Ethernet enables/disables.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`CPRI:PORT1:EEN?` → `0`
Note	

4.1.15 CPRI:PORT<Pt>:CMEPointer

Syntax	CPRI:PORT<Pt>:CMEPointer <pointer>
Description	This command sets the control manager ethernet pointer.
Parameters	<Pt> = Port number
	<pointer> = <NUMERIC PROGRAM DATA> *MINimum=20, MAXimum=63, DEFault=20*
Response	None.
Example	`CPRI:PORT1:CMEP 63`
Note	This command cannot be executed when Ethernet is disabled. This command supports CPRI mode.

Syntax	CPRI:PORT<Pt>:CMEPointer?
Description	This query returns the control manager ethernet pointer.
Parameter	<Pt> = Port number
Response	<pointer> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:CMEP?` → `20`
Note	

4.1.16 CPRI:PORT<Pt>:OBSai:FSSeed

Syntax	CPRI:PORT<Pt>:OBSai:FSSeed <enable>
Description	This command sets the forced scrambler seed.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> OFF ON
Response	None.
Example	`CPRI:PORT1:OBS:FSS ON`
Note	This command supports OBSAI mode.

Syntax	CPRI:PORT<Pt>:OBSai:FSSeed?
Description	This query returns the forced scrambler seed.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`CPRI:PORT1:OBS:FSS?` → `ON`
Note	

4.1.17 CPRI:PORT<Pt>:OBSai:TX:FIDLe

Syntax	CPRI:PORT<Pt>:OBSai:TX:FIDLe <type>
Description	This command sets the Tx forced idle.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> OFF: Idle OFF ON: Idle ON (up to 3072M) IREQ: Idle REQ (6144M) IACK: Idle ACK (6144M)
Response	None.
Example	`CPRI:PORT1:OBS:TX:FIDL ON`
Note	This command supports OBSAI mode.

Syntax	CPRI:PORT<Pt>:OBSai:TX:FIDLe?
Description	This query returns the Tx forced idle.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:OBS:TX:FIDL?` → `ON`
Note	

4.1.18 CPRI:PORT<Pt>:OBSai:TX:SSINdex

Syntax	CPRI:PORT<Pt>:OBSai:TX:SSINdex <seed>
Description	This command sets the Tx scrambler seed index.
Parameters	<Pt> = Port number
	<seed> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=17, DEFault=0*
Response	None.
Example	`CPRI:PORT1:OBS:TX:SSIN 1`
Note	This command supports OBSAI mode.

Syntax	CPRI:PORT<Pt>:OBSai:TX:SSINdex?
Description	This query returns the Tx scrambler seed index.
Parameter	<Pt> = Port number
Response	<seed> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:OBS:TX:SSIN?` → `1`
Note	

4.1.19 CPRI:PORT<Pt>:OBSai:TX:SSValue

Syntax	CPRI:PORT<Pt>:OBSai:TX:SSValue?
Description	This query returns the Tx scrambler seed value.
Parameter	<Pt> = Port number
Response	<seed> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:OBS:TX:SSV?` → `#B11`
Note	This command supports OBSAI mode.

4.1.20 CPRI:PORT<Pt>:OBSai:TX:ADDRess

Syntax	CPRI:PORT<Pt>:OBSai:TX:ADDRess <address>
Description	This command sets the Tx RP3 address.
Parameters	<Pt> = Port number
	<address> = <NUMERIC PROGRAM DATA> *MINimum=#H0000, MAXimum=#H1FFF, DEFault=#H0000*
Response	None.
Example	`CPRI:PORT1:OBS:TX:ADDR #H0001`
Note	This command supports OBSAI mode.

Syntax	CPRI:PORT<Pt>:OBSai:TX:ADDRess?
Description	This query returns Tx RP3 address.
Parameter	<Pt> = Port number
Response	<address> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:OBS:TX:ADDR?` → `#H1`
Note	

4.1.21 CPRI:PORT<Pt>:OBSai:TX:TYPE

Syntax	CPRI:PORT<Pt>:OBSai:TX:TYPE <type>
Description	This command sets the Tx RP3 type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> WCDMA: WCDMA/FDD GSM: GSM/EDGE IEEE80216: IEEE 802.16 LTE: LTE
Response	None.
Example	`CPRI:PORT1:OBS:TX:TYPE LTE`
Note	This command supports OBSAI mode.

Syntax	CPRI:PORT<Pt>:OBSai:TX:TYPE?
Description	This query returns the Tx RP3 type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:OBS:TX:TYPE?` → `LTE`
Note	

4.1.22 CPRI:PORT<Pt>:OBSai:RX:FILTer

Syntax	CPRI:PORT<Pt>:OBSai:RX:FILTer <enable>
Description	This command sets the Rx filter enable.
	<enable> = <BOOLEAN PROGRAM DATA> OFF ON
Response	None.
Example	`CPRI:PORT1:OBS:RX:FILT ON`
Note	This command supports OBSAI mode.

Syntax	CPRI:PORT<Pt>:OBSai:RX:FILTer?
Description	This query returns the Rx filter enable.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`CPRI:PORT1:OBS:RX:FILT?` → `0`
Note	

4.1.23 CPRI:PORT<Pt>:OBSai:RX:SSINdex

Syntax	CPRI:PORT<Pt>:OBSai:RX:SSINdex <seed>
Description	This command sets the Rx scrambler seed index.
Parameters	<Pt> = Port number
	<seed> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=17, DEFault=0*
Response	None.
Example	`CPRI:PORT1:OBS:RX:SSIN 1`
Note	This command supports OBSAI mode.

Syntax	CPRI:PORT<Pt>:OBSai:RX:SSINdex?
Description	This query returns the Rx scrambler seed index.
Parameter	<Pt> = Port number
Response	<seed> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:OBS:RX:SSIN? → 1`
Note	

4.1.24 CPRI:PORT<Pt>:OBSai:RX:SSValue

Syntax	CPRI:PORT<Pt>:OBSai:RX:SSValue?
Description	This query returns the Rx scrambler seed value.
Parameter	<Pt> = Port number
Response	<seed> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:OBS:RX:SSV? → #B11`
Note	

4.1.25 CPRI:PORT<Pt>:OBSai:RX:ADDRess

Syntax	CPRI:PORT<Pt>:OBSai:RX:ADDRess <address>
Description	This command sets the Rx RP3 address.
Parameters	<Pt> = Port number
	<address> = <NUMERIC PROGRAM DATA> *MINimum=#H0000, MAXimum=#H1FFF, DEFault=#H0000*
Response	None.
Example	`CPRI:PORT1:OBS:RX:ADDR #H0001`
Note	This command supports OBSAI mode.

Syntax	CPRI:PORT<Pt>:OBSai:RX:ADDRess?
Description	This query returns Rx forced idle.
Parameter	<Pt> = Port number
Response	<address> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:OBS:RX:ADDR? → #H1`
Note	

4.1.26 CPRI:PORT<Pt>:OBSai:RX:TYPE

Syntax	CPRI:PORT<Pt>:OBSai:RX:TYPE <type>
Description	This command sets the Rx RP3 type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> WCDMA: WCDMA/FDD GSM: GSM/EDGE IEEE80216: IEEE 802.16 LTE: LTE
Response	None.
Example	`CPRI:PORT1:OBS:RX:TYPE LTE`
Note	This command supports OBSAI mode.

Syntax	CPRI:PORT<Pt>:OBSai:RX:TYPE?
Description	This query returns the Rx RP3 type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:OBS:RX:TYPE?` → `LTE`
Note	

4.1.27 CPRI:PORT<Pt>:SETup:FOLLow

Syntax	CPRI:PORT<Pt>:SETup:FOLLow <follow>
Description	This command enables/disables that port2 follows port1.
Parameter	<Pt> = Port number
	<follow> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`CPRI:PORT2:SET:FOLL OFF`
Note	This command supports CPRI and OBSAI mode.

Syntax	CPRI:PORT<Pt>:SETup:FOLLow?
Description	This query returns enables/disables that port2 follows port1.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`CPRI:PORT1:SET:FOLL?` → `0`
Note	

4.2 Stimuli

4.2.1 CPRI:PORT<Pt>:STIMuli:ALARm

Syntax	CPRI:PORT<Pt>:STIMuli:ALARm <alarm>
Description	This command sets stimuli alarm.
Parameter	<Pt> = Port number
	<alarm> = <CHARACTER PROGRAM DATA> SLOS: Optical Loss of signal LOS: Loss of Signal LOF: Loss of Frame LSS: Loss of signal synchronization RLOS: Remote LOS RLOF: Remote LOF RAI: L1-RAI (Remote Alarm Indication) SDI: L1-SDI (SAP Defect Indication) RESet: L1-Reset (Master:Reset confirmation; Slave:Reset Request)
Response	None.
Example	`CPRI:PORT1:STIM:ALAR LOF`
Note	When `:CONTents` is `UNFramed` only `SLOS` and `LSS` is valid.

Syntax	CPRI:PORT<Pt>:STIMuli:ALARm?
Description	This query returns stimuli alarm.
Parameter	<Pt> = Port number
Response	<alarm> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:STIM:ALAR?` → `LOS`
Note	

4.2.2 CPRI:PORT<Pt>:STIMuli:AINSertion

Syntax	CPRI:PORT<Pt>:STIMuli:AINSertion <insertion>
Description	This command set Alarm insertion.
Parameter	<Pt> = Port number
	<insertion> = <CHARACTER RESPONSE DATA> OFF: Off PERManent: Permanent *DEFault = OFF*
Response	None.
Example	`CPRI:PORT1:STIM:AINS OFF`
Note	

Syntax	CPRI:PORT<Pt>:STIMuli:AINSertion?
Description	This query returns Alarm insertion.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:STIM:AINS?` → `OFF`
Note	

4.2.3 CPRI:PORT<Pt>:STIMuli:ERRor

Syntax	CPRI:PORT<Pt>:STIMuli:ERRor <error>
Description	This command sets stimuli Error.
Parameter	<Pt> = Port number
	<error> = <CHARACTER PROGRAM DATA> LCV: LCV SHV: SHV K307: K30.7 PE: Pattern Error FECSE: FEC Symbol Error
Response	None.
Example	`CPRI:PORT1:STIM:ERR LCV`
Note	

Syntax	CPRI:PORT<Pt>:STIMuli:ERRor?
Description	This query returns stimuli alarm.
Parameter	<Pt> = Port number
Response	<alarm> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:STIM:ERR? → LCV`
Note	

4.2.4 CPRI:PORT<Pt>:STIMuli:EINSertion

Syntax	CPRI:PORT<Pt>:STIMuli:EINSertion <insertion>
Description	This command sets the method to insert errors.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> OFF MANual B04: Burst · 1E-04 B05: Burst · 1E-05 B06: Burst · 1E-06 B07: Burst · 1E-07 B08: Burst · 1E-08 B09: Burst · 1E-09 B10: Burst · 1E-10 *DEFault = OFF*
Response	None.
Example	`CPRI:PORT1:STIM:EINS OFF`
Note	None.

Syntax	CPRI:PORT<Pt>:STIMuli:EINSertion?
Description	This query returns the error insertion method.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`CPRI:PORT1:STIM:EINS? → B04`
Note	

4.2.5 CPRI:PORT<Pt>:STIMuli:EBLength

Syntax	CPRI:PORT<Pt>:STIMuli:EBLength <length>
Description	This command sets Error Burst Length.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=256, DEFault=1*
Response	None.
Example	`CPRI:PORT1:STIM:EBL 64`
Note	This command is enabled when error insertion is Manual.

Syntax	CPRI:PORT<Pt>:STIMuli:EBLength?
Description	This query returns Error Burst Length.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:STIM:EBL` → `64`
Note	

4.2.6 CPRI:PORT<Pt>:STIMuli:FOFFset

Syntax	CPRI:PORT<Pt>:STIMuli:FOFFset <offset>
Description	This command sets the frequency offset for the clock source. Unit: ppm.
Parameters	<Pt> = Port number
	<offset> = <NUMERIC PROGRAM DATA> *MINimum=-100, MAXimum=100, DEFault=0*
Response	None.
Example	`CPRI:PORT1:STIM:FOFF 0`
Note	

Syntax	CPRI:PORT<Pt>:STIMuli:FOFFset?
Description	This query returns the frequency offset for the clock source. Unit: ppm.
Parameter	<Pt> = Port number
Response	<offset> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:STIM:FOFF` → `0`
Note	

4.3 Result

4.3.1 CPRI:PORT<Pt>:IFETch?

Syntax	CPRI:PORT<Pt>:IFETch? <parameter>
Description	This query fetches a CPRI interval if available.
Parameters	<Pt> = Port number
	({<parameter>} + {,}*) = <EXPRESSION PROGRAM DATA> The response format is listed for each parameter. **Alarms** SLOS: Optical loss of signal. Response: <Seconds>,<Ratio> LOS: Loss of signal. Response: <Seconds>,<Ratio> LOF: Loss of frame. Response: <Seconds>,<Ratio> NMES: No message. Response: <Seconds>,<Ratio> LSS: Loss of signal synchronization. Response: <Seconds>,<Ratio> **L1Remote Alarm** RLOS: Remote loss. Response: <Count>,<Ratio> RLOF: Remote loss of frame. Response: <Count>,<Ratio> RRAI: Remote alarm indication. Response: <Count>,<Ratio> RSDI: Remote SAP defect indication. Response: <Count>,<Ratio> RES: Reset. Response: <Count>,<Ratio> **Errors** LCV: LCV.Response: <Count>,<Ratio> SHV: SHV. Response: <Count>,<Ratio> K307: K30.7. Response: <Count>,<Ratio> PE: Pattern error. Response: <Count>,<Ratio> **Frame** TXHF: TX hyper frames. Response: <Count> TXCW: TX code words. Response: <Count> RXHF: RX hyper frames. Response: <Count> RXCW: RX code words. Response: <Count> TXMSGG: Tx OBSAI Message group. Response: <Count> RXMSGG: Rx OBSAI Message group. Response: <Count> **FEC** FECLOFA: Loss of FEC Alignment. Response: <Count> FECCORCW: Fec Corrected Codewords. Response: <Count> FECUCORCW: Fec Uncorrected Codewords. Response: <Count> FECSYMERR: Fec Symbol errors. Response: <Count> FECSYMERRRATE: Fec Symbol error rate. Response: <Count> **Delay** DEL: Delay. Response: <second> AVED: Average Delay. Response: <second> MIND: Maximum Delay. Response: <second> MAXD: Minimum Delay. Response: <second> DMC: Measurement Count. Response: <Count>
Response	{(<result>), }* = <EXPRESSION RESPONSE DATA> Format: Numeric List Each result is formatted according to the specification in the parameter field. Values that are not relevant or applicable for the current measurement return NaN (section 1.6.1).
Example	`CPRI:PORT1:IFET? (LOS,LOF)` → `(3,0.00532),(4,0.00709)`
Notes	This command fetches the results from the interval selected using the `MEASurement:SETup:SELect` command (see section 18.2.2). If the requested result is not available, NaN (section 1.6.1) is returned. If there is one or more results, the last "," is always removed.

4.4 Status

4.4.1 CPRI:PORT<Pt>:STATus:AESummary[:EVENt]?

Syntax	CPRI:PORT<Pt>:STATus:AESummary[:EVENt]?
Description	This query returns the alarms and errors summary event register. The content of this event register is summarized in DB13 of the STATus:INTerface:PORT<Pt>:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = CPRI Alarm DB2 (2) = CPRI Remote Alarm DB3 (4) = CPRI Error DB4 (8) = FEC Alarm/Error DB5 - DB16 = NOT USED
Example	`CPRI:PORT1:STAT:AES?` → `3`
Note	

4.4.2 CPRI:PORT<Pt>:STATus:AESummary:CONDiton?

Syntax	CPRI:PORT<Pt>:STATus:AESummary:CONDiton?
Description	This query returns alarms and errors summary condition register. DB1/DB2 is summary of CPRI:PORT<Pt>:STATus:ALARm<section>[:EVENt]? DB3 is summary of CPRI:PORT<Pt>:STATus:ERRor[:EVENt]? DB4 is summary of CPRI:PORT<Pt>:STATus:FEC[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> Same as CPRI:PORT<Pt>:STATus:AESummary[:EVENt]?
Example	`CPRI:PORT1:STAT:AES:COND?` → `3`
Note	

4.4.3 CPRI:PORT<Pt>:STATus:ALARm<section>[:EVENt]?

Syntax	CPRI:PORT<Pt>:STATus:ALARm<section>[:EVENt]?
Description	This query returns one of the alarms event register.
Parameters	<Pt> = Port number
	<section> = CPRI Alarm(1), CPRI L1Remote Alarm(2), CPRI FEC Alarm(3)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> <section> = 1: DB1 (1) = Signal Loss DB2 (2) = LOS DB3 (4) = LOF DB4 (8) = LSS DB5 (16) = No message DB6 - DB16 = NOT USED <section> = 2: DB1 (1) = Remote LOS DB2 (2) = Remote LOF DB3 (4) = RAI DB4 (8) = SDI DB5 (16) = Reset DB6 - DB16 = NOT USED
Example	`CPRI:PORT1:STAT:ALAR1?` → `1`
Note	

4.4.4 CPRI:PORT<Pt>:STATus:ALARm<section>:CONDition?

Syntax	CPRI:PORT<Pt>:STATus:ALARm<section>:CONDition?
Description	This query returns one of the alarms condition registers.
Parameters	<Pt> = Port number
	<section> = CPRI Alarm(1), CPRI L1Remote Alarm(2)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> Same as CPRI:PORT<Pt>:STATus:ALARm<section>[:EVENt]?
Example	`CPRI:PORT1:STAT:ALAR1:COND?` → `1`
Note	

4.4.5 CPRI:PORT<Pt>:STATus:ERRor[:EVENt]?

Syntax	CPRI:PORT<Pt>:STATus:ERRor[:EVENt]?
Description	This query returns the errors event register.
Parameters	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = LCV DB2 (2) = SHV DB3 (4) = K30.7 DB4 (8) = Pattern error DB5 - DB16 = NOT USED
Example	`CPRI:PORT1:STAT:ERR?` → `3`
Note	

4.4.6 CPRI:PORT<Pt>:STATus:ERRor:CONDition?

Syntax	CPRI:PORT<Pt>:STATus:ERRor:CONDition?
Description	This query returns errors condition register.
Parameters	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> Same as CPRI:PORT<Pt>:STATus:ERRor[:EVENt]?
Example	`CPRI:PORT1:STAT:ERR:COND?` → `3`
Note	

4.4.7 CPRI:PORT<Pt>:STATus:FEC[:EVENt]?

Syntax	CPRI:PORT<Pt>:STATus:FEC[:EVENt]?
Description	This query returns the FEC alarms and errors event register.
Parameters	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Loss of FEC Alignment (LOFA) DB2 (2) = FEC Corrected Codewords DB3 (4) = FEC Uncorrected Codewords DB4 (8) = FEC Symbol Errors DB5 - DB16 = NOT USED
Example	`CPRI:PORT1:STAT:FEC?` → `3`
Note	

4.4.8 CPRI:PORT<Pt>:STATus:FEC:CONDition?

Syntax	CPRI:PORT<Pt>:STATus:FEC:CONDition?
Description	This query returns FEC alarms and errors condition register.
Parameters	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> Same as CPRI:PORT<Pt>:STATus:FEC[:EVENt]?
Example	`CPRI:PORT1:STAT:FEC:COND?` → `3`
Note	

4.4.9 CPRI:PORT<Pt>:STATus:TX:PSLevel?

Syntax	CPRI:PORT<Pt>:STATus:TX:PSLevel?
Description	This query returns the TX signal level. Unit: dBm.
Parameters	<Pt> = Port number
Response	<signal-level> = <STRING RESPONSE DATA> "<signal-level> dBm": Min: "< <min> dBm", Max: "Exceeds Level" "N/A": Module not present or not ready.
Example	`CPRI:PORT1:STAT:TX:PSL? → "-3dBm"`
Note	

4.4.10 CPRI:PORT<Pt>:STATus:RX:PSLevel?

Syntax	CPRI:PORT<Pt>:STATus:RX:PSLevel?
Description	This query returns the RX signal level. Unit: dBm.
Parameters	<Pt> = Port number
Response	<signal-level> = <STRING RESPONSE DATA> "<signal-level> dBm": Min: "< <min> dBm", Max: "Exceeds Level" "N/A": Module not present or not ready.
Example	`CPRI:PORT1:STAT:RX:PSL? → "-3dBm"`
Note	

4.4.11 CPRI:PORT<Pt>:STATus:TX:PDEViation?

Syntax	CPRI:PORT<Pt>:STATus:TX:PDEViation? [<unit>]
Description	This query returns TX deviation.
Parameters	<Pt> = Port number
	<unit> = <CHARACTER PROGRAM DATA> PPM = Parts per million BPS = Bits per second
Response	<pdeviation> = <NR2 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:STAT:TX:PDEV? PPM → -3`
Note	

4.4.12 CPRI:PORT<Pt>:STATus:RX:PDEViation?

Syntax	CPRI:PORT<Pt>:STATus:RX:PDEViation? [<unit>]
Description	This query returns RX deviation.
Parameters	<Pt> = Port number
	<unit> = <CHARACTER PROGRAM DATA> PPM = Parts per million BPS = Bits per second
Response	<pdeviation> = <NR2 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:STAT:RX:PDEV? PPM → 10`
Note	

4.4.13 CPRI:PORT<Pt>:STATus:TX:PBRate?

Syntax	CPRI:PORT<Pt>:STATus:TX:PBRate?
Description	This query returns the bit rate on TX.
Parameters	<Pt> = Port number
Response	<pbrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:STAT:TX:PBR? → 614400000`
Note	

4.4.14 CPRI:PORT<Pt>:STATus:RX:PBRate?

Syntax	CPRI:PORT<Pt>:STATus:RX:PBRate?
Description	This query returns the bit rate on RX.
Parameters	<Pt> = Port number
Response	<pbrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:STAT:RX:PBR?` → `614400000`
Note	

4.4.15 CPRI:PORT<Pt>:STATus:TX:PPBRate?

Syntax	CPRI:PORT<Pt>:STATus:TX:PPBRate?
Description	This query returns the pattern bit rate on TX.
Parameters	<Pt> = Port number
Response	<ppbrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:STAT:TX:PPBR?` → `614400000`
Note	

4.4.16 CPRI:PORT<Pt>:STATus:RX:PPBRate?

Syntax	CPRI:PORT<Pt>:STATus:RX:PPBRate?
Description	This query returns the pattern bit rate on RX.
Parameters	<Pt> = Port number
Response	<ppbrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:STAT:RX:PPBR?` → `614400000`
Note	

4.4.17 CPRI:PORT<Pt>:STATus:LINK:STATe?

Syntax	CPRI:PORT<Pt>:STATus:LINK:STATe?
Description	This query returns the CPRI link State.
Parameters	<Pt> = Port number
Response	<state> = <CHARACTER PROGRAM DATA>
Example	`CPRI:PORT1:STAT:LINK:STAT?` → `"Standby"`
Note	

4.4.18 CPRI:PORT<Pt>:STATus:LINK:PROLe?

Syntax	CPRI:PORT<Pt>:STATus:LINK:PROLe?
Description	This query returns the port role of CPRI link.
Parameters	<Pt> = Port number
Response	<role> = <CHARACTER PROGRAM DATA>
Example	`CPRI:PORT1:STAT:LINK:PROL?` → `"Master"`
Note	

4.4.19 CPRI:PORT<Pt>:STATus:RX:LINK:PVERsion?

Syntax	CPRI:PORT<Pt>:STATus:RX:LINK:PVERsion?
Description	This query returns the protocol version of CPRI link.
Parameters	<Pt> = Port number
Response	<version> = <CHARACTER PROGRAM DATA>
Example	`CPRI:PORT1:STAT:RX:LINK:PVER?` → `"Version 1"`
Note	

4.4.20 CPRI:PORT<Pt>:STATus:RX:LINK:HRATe?

Syntax	CPRI:PORT<Pt>:STATus:RX:LINK:HRATe?
Description	This query returns the HDLC Rate of CPRI link.
Parameters	<Pt> = Port number
Response	<rate> = <CHARACTER PROGRAM DATA>
Example	`CPRI:PORT1:STAT:RX:LINK:HRAT?` → `"2400 kbit/s HDLC"`
Note	

4.4.21 CPRI:PORT<Pt>:STATus:RX:LINK:POINterp?

Syntax	CPRI:PORT<Pt>:STATus:RX:LINK:POINterp?
Description	This query returns the ethernet pointer of CPRI link.
Parameters	<Pt> = Port number
Response	<pointer> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:STAT:RX:LINK:POIN?` → `20`
Note	

4.4.22 CPRI:PORT<Pt>:STATus:TX:LINK:PVERsion?

Syntax	CPRI:PORT<Pt>:STATus:TX:LINK:PVERsion?
Description	This query returns the protocol version of CPRI link.
Parameters	<Pt> = Port number
Response	<version> = <CHARACTER PROGRAM DATA>
Example	`CPRI:PORT1:STAT:TX:LINK:PVER?` → `"Version 1"`
Note	

4.4.23 CPRI:PORT<Pt>:STATus:TX:LINK:HRATe?

Syntax	CPRI:PORT<Pt>:STATus:TX:LINK:HRATe?
Description	This query returns the HDLC Rate of CPRI link.
Parameters	<Pt> = Port number
Response	<rate> = <CHARACTER PROGRAM DATA>
Example	`CPRI:PORT1:STAT:TX:LINK:HRAT?` → `"2400 kbit/s HDLC"`
Note	

4.4.24 CPRI:PORT<Pt>:STATus:TX:LINK:POINterp?

Syntax	CPRI:PORT<Pt>:STATus:TX:LINK:POINterp?
Description	This query returns the ethernet pointer of CPRI link.
Parameters	<Pt> = Port number
Response	<pointer> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:STAT:TX:LINK:POIN?` → `20`
Note	

4.4.25 CPRI:PORT<Pt>:STATus:OBSai:TX:LINK:STATe?

Syntax	CPRI:PORT<Pt>:STATus:OBSai:TX:LINK:STATe?
Description	This query returns the OBSAI link Tx State.
Parameters	<Pt> = Port number
Response	<state> = <CHARACTER PROGRAM DATA>
Example	`CPRI:PORT1:STAT:OBS:TX:LINK:STAT?` → `"FRAME_TX"`
Note	

4.4.26 CPRI:PORT<Pt>:STATus:OBSai:RX:LINK:STATe?

Syntax	CPRI:PORT<Pt>:STATus:OBSai:RX:LINK:STATe?
Description	This query returns the CPRI link Rx State.
Parameters	<Pt> = Port number
Response	<state> = <CHARACTER PROGRAM DATA>
Example	`CPRI:PORT1:STAT:OBS:RX:LINK:STAT?` → `"FRAME_SYNC"`
Note	

4.5 RTD

4.5.1 CPRI:PORT<Pt>:RTD:ENABle

Syntax	CPRI:PORT<Pt>:RTD:ENABle <enable>
Description	This command enables/disables Thresholds.
Parameter	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`CPRI:PORT1:RTD:ENAB ON`
Note	

Syntax	CPRI:PORT<Pt>:RTD:ENABle?
Description	This query returns enables/disables Thresholds.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`CPRI:PORT1:RTD:ENAB?` → `1`
Note	

4.5.2 CPRI:PORT<Pt>:RTD:MLIMit

Syntax	CPRI:PORT<Pt>:RTD:MLIMit <limit>
Description	This command sets the Threshold of maximum limit. Unit: us.
Parameters	<Pt> = Port number
	<limit> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1,000,000, DEFault=1,000,000*
Response	None.
Example	`CPRI:PORT1:RTD:MLIM 1000000.00`
Note	

Syntax	CPRI:PORT<Pt>:RTD:MLIMit?
Description	This query returns the Threshold of maximum limit. Unit: us.
Parameter	<Pt> = Port number
Response	<limit> = <NR2 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:RTD:MLIM?` → `1000000.00`
Note	

4.5.3 CPRI:PORT<Pt>:RTD:NUMBer?

Syntax	CPRI:PORT<Pt>:RTD:NUMBer?
Description	This query returns the number of the RTD data.?
Parameters	<Pt> = Port number
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:RTD:NUMB?` → `1`
Note	

4.5.4 CPRI:PORT<Pt>:RTD:ATIMe?

Syntax	CPRI:PORT<Pt>:RTD:ATIMe?
Description	This query returns the Average delay. Unit:us.
Parameters	<Pt> = Port number
Response	<delay> = <NR2 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:RTD:ATIM?` → `1.46`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

4.5.5 CPRI:PORT<Pt>:RTD:MTIMe?

Syntax	CPRI:PORT<Pt>:RTD:MTIMe?
Description	This query returns the maximum delay. Unit:us.
Parameters	<Pt> = Port number
Response	<delay> = <NR2 NUMERIC RESPONSE DATA> <limit exceeded> = <NR1 NUMERIC RESPONSE DATA> Returns 1, if the reference maximum limit has been exceeded.
Example	`CPRI:PORT1:RTD:MTIM?` → `1.46`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

4.5.6 CPRI:PORT<Pt>:RTD:LTIMe?

Syntax	CPRI:PORT<Pt>:RTD:LTIMe?
Description	This query returns the minimum delay. Unit:us.
Parameters	<Pt> = Port number
Response	<delay> = <NR2 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:RTD:LTIM?` → `1.46`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

4.6 APS

4.6.1 CPRI:PORT<Pt>:APS:ENABle

Syntax	CPRI:PORT<Pt>:APS:ENABle <enable>
Description	This command enables/disables APS measurement.
Parameter	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`CPRI:PORT1:APS:ENAB ON`
Note	

Syntax	CPRI:PORT<Pt>:APS:ENABle?
Description	This query returns enables/disables Thresholds.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`CPRI:PORT1:APS:ENAB? → 1`
Note	

4.6.2 CPRI:PORT<Pt>:APS:EVENt

Syntax	CPRI:PORT<Pt>:APS:EVENt <events>
Description	This command sets the time reference event.
Parameters	<Pt> = Port number
	({<events>} ∗ {, }∗) = <EXPRESSION PROGRAM DATA> SLOS : Signal loss LOS : LOS LOF : LOF LCV : LCV SHV : SHV PE : Pattern error RLOS : Remote LOS RLOF : Remote LOF RAI : RAI SDI : SDI RESet : Reset
Response	None.
Example	`CPRI:PORT1:APS:EVEN (LOS,LOF)`
Note	

Syntax	CPRI:PORT<Pt>:APS:EVENt?
Description	This query returns the time reference event.
Parameter	<Pt> = Port number
Response	{(<events>), }* = <EXPRESSION RESPONSE DATA>
Example	`CPRI:PORT1:APS:EVEN? → (LOS,LOF)`
Note	

4.6.3 CPRI:PORT<Pt>:APS:PERiod

Syntax	CPRI:PORT<Pt>:APS:PERiod <period>
Description	This command sets the error free period.
Parameters	<Pt> = Port number
	<period> = <NUMERIC PROGRAM DATA> 1,10,20,30,40,50,60,70,80,90,100 Unit ms *DEFault = 100*
Response	None.
Example	`CPRI:PORT1:APS:PER 10`
Note	

Syntax	CPRI:PORT<Pt>:APS:PERiod?
Description	This query returns the error free period.
Parameter	<Pt> = Port number
Response	<period> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:APS:PER? → 10`
Note	

4.6.4 CPRI:PORT<Pt>:APS:MLIMit

Syntax	CPRI:PORT<Pt>:APS:MLIMit <max>
Description	This command sets the time reference maximum limit. Unit: ms.
Parameters	<Pt> = Port number
	<max> = <NUMERIC PROGRAM DATA> *MINimum = 0.000, MAXimum = 1000.000, DEFault = 50.000*
Response	None.
Example	`CPRI:PORT1:APS:MLIM 50.000`
Note	

Syntax	CPRI:PORT<Pt>:APS:MLIMit?
Description	This query returns the time reference maximum limit. Unit: ms.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:APS:MLIM? → 50.000`
Note	

4.6.5 CPRI:PORT<Pt>:APS:NUMBer?

Syntax	CPRI:PORT<Pt>:APS:NUMBer?
Description	This query returns the number of times an APS Protocol event has occurred.
Parameter	<Pt> = Port number
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:APS:NUMB? → 17`
Note	

4.6.6 CPRI:PORT<Pt>:APS:ATIMe?

Syntax	CPRI:PORT<Pt>:APS:ATIMe?
Description	This query returns the average time of the reference event occurrences. Unit: ms.
Parameter	<Pt> = Port number
Response	<avg> = <NR2 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:APS:ATIM? → 4.000`
Note	The maximum measurable time is 1000 ms. The maximum measurable time will be responded if the result exceeds 1000 ms.

4.6.7 CPRI:PORT<Pt>:APS:MTIMe?

Syntax	CPRI:PORT<Pt>:APS:MTIMe?
Description	This query returns the maximum time of the reference event occurrences. Unit: ms.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
	<limit exceeded> = <NR1 NUMERIC RESPONSE DATA> Returns 1, if the reference maximum limit has been exceeded.
Example	`CPRI:PORT1:APS:MTIM?` → `4.000,0`
Note	The maximum measurable time is 1000 ms. The maximum measurable time will be responded if the result exceeds 1000 ms.

4.6.8 CPRI:PORT<Pt>:APS:LTIMe?

Syntax	CPRI:PORT<Pt>:APS:LTIMe?
Description	This query returns the least (minimum) time of the reference event occurrences. Unit: ms.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:APS:LTIM?` → `4.000`
Note	The maximum measurable time is 1000 ms. The maximum measurable time will be responded if the result exceeds 1000 ms.

4.6.9 CPRI:PORT<Pt>:APS:CTIMe?

Syntax	CPRI:PORT<Pt>:APS:CTIMe?
Description	This query returns the current time of the reference event occurrences. Unit: ms.
Parameter	<Pt> = Port number
Response	<current> = <NR2 NUMERIC RESPONSE DATA>
Example	`CPRI:PORT1:APS:CTIM?` → `4.000`
Note	The maximum measurable time is 1000 ms. The maximum measurable time will be responded if the result exceeds 9999.999 ms.

Chapter 5

2 Mbps

5.1 Receiver

5.1.1 TMBPs:RX<Pt>[:ENABled]

Syntax	TMBPs:RX<Pt>[:ENABled] <interface>
Description	This command enables or disables the 2 Mbps receiver.
Parameters	<Pt> = Port number
	<interface> = <CHARACTER PROGRAM DATA> OFF ON *DEFault = OFF*
Response	None.
Example	`TMBP:RX1 ON`
Note	

Syntax	TMBPs:RX<Pt>[:ENABled]?
Description	This query returns the state (enabled/disabled) of the 2 Mbps receiver.
Parameter	<Pt> = Port number
Response	<interface> = <CHARACTER RESPONSE DATA>
Example	`TMBP:RX1?` → `ON` `TMBP:RX1:ENAB?` → `SDH`
Note	Returns `SDH` if E1 is over SDH or SONET.

5.1.2 TMBPs:RX<Pt>:FOLLow

Syntax	TMBPs:RX<Pt>:FOLLow <follow>
Description	This command sets the receiver to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<follow> = <CHARACTER PROGRAM DATA> NONE: Do not follow TX: Transmitter of the same port RX1: Receiver of port 1 *DEFault = NONE*
Response	None.
Example	`TMBP:RX1:FOLL TX` (i.e., RX1 follows TX1) `TMBP:RX2:FOLL RX1` (i.e., RX2 follows RX1)
Note	

Syntax	TMBPs:RX<Pt>:FOLLow?
Description	This query returns if the receiver follow another setup.
Parameter	<Pt> = Port number
Response	<follow> = <CHARACTER RESPONSE DATA>
Example	`TMBP:RX1:FOLL?` → `TX`
Note	

5.1.3 TMBPs:RX<Pt>:CONNector

Syntax	TMBPs:RX<Pt>:CONNector <type>
Description	This command sets the physical type of connector.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> BALanced: Using balanced connector UNBalanced: Using unbalanced connector *DEFault = UNBalanced*
Response	None.
Example	`TMBP:RX1:CONN BAL`
Note	

Syntax	TMBPs:RX<Pt>:CONNector?
Description	This query returns the physical type of connector.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`TMBP:RX1:CONN?` → `BAL`
Note	

5.1.4 TMBPs:RX<Pt>:MODE

Syntax	TMBPs:RX<Pt>:MODE <mode>
Description	This command sets the signal termination mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> TERMinate: Nominal impedance. Normal frequency dependent AGC. BRIDged: High impedance. MONitor: Nominal impedance. Frequency linear AGC. *DEFault = TERMinate*
Response	None.
Example	`TMBP:RX1:MODE TERM`
Note	

Syntax	TMBPs:RX<Pt>:MODE?
Description	This query returns the signal termination mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`TMBP:RX1:MODE?` → `TERM`
Note	

5.1.5 TMBPs:RX<Pt>:SENSitivity

Syntax	TMBPs:RX<Pt>:SENSitivity <sens>
Description	This command sets the input sensitivity.
Parameters	<Pt> = Port number
	<sens> = <CHARACTER PROGRAM DATA> FULL M20DB: Input sensitivity -20 dB M33DB: Input sensitivity -33 dB *DEFault = FULL*
Response	None.
Example	`TMBP:RX1:SENS FULL`
Note	

Syntax	TMBPs:RX<Pt>:SENSitivity?
Description	This query returns the input sensitivity
Parameter	<Pt> = Port number
Response	<sens> = <CHARACTER RESPONSE DATA>
Example	`TMBP:RX1:SENS?` → `FULL`
Note	

5.1.6 TMBPs:RX<Pt>:CODE

Syntax	TMBPs:RX<Pt>:CODE <code>
Description	This command sets the line code type.
Parameters	<Pt> = Port number
	<code> = <CHARACTER PROGRAM DATA> AMI HDB3 *DEFault = HDB3*
Response	None.
Example	`TMBP:RX1:CODE AMI`
Note	

Syntax	TMBPs:RX<Pt>:CODE?
Description	This query returns the line code type.
Parameter	<Pt> = Port number
Response	<code> = <CHARACTER RESPONSE DATA>
Example	`TMBP:RX1:CODE?` → `AMI`
Note	

5.1.7 TMBPs:RX<Pt>:PCMFrame

Syntax	TMBPs:RX<Pt>:PCMFrame <enable>
Description	This command enables or disables the PCM frame.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`TMBP:RX1:PCMF ON`
Note	

Syntax	TMBPs:RX<Pt>:PCMFrame?
Description	This query returns whether or not PCM frame is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:RX1:PCMF?` → `1`
Note	

5.1.8 TMBPs:RX<Pt>:CRC4

Syntax	TMBPs:RX<Pt>:CRC4 <enable>
Description	This command enables or disables the CRC4 bits.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`TMBP:RX1:CRC4 ON`
Note	

Syntax	TMBPs:RX<Pt>:CRC4?
Description	This query returns the state (enabled/disabled) of the CRC4 bits.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:RX1:CRC4?` → `1`
Note	

5.1.9 TMBPs:RX<Pt>:EBITs

Syntax	TMBPs:RX<Pt>:EBITs <enable>
Description	This command enables or disables the E-bits.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`TMBP:RX1:EBIT ON`
Note	

Syntax	TMBPs:RX<Pt>:EBITs?
Description	This query returns the state (enabled/disabled) of the E-bits.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:RX1:EBIT?` → `1`
Note	

5.1.10 TMBPs:RX<Pt>:PATTern

Syntax	TMBPs:RX<Pt>:PATTern <type>
Description	This command sets the pattern type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> OFF USER16BIT: Obsolete. For backward compatibility only. Same as USER32BIT. USER32BIT: 32 bit user defined pattern. USER2048BIT: 2048 bit user defined pattern. PRBS6 PRBS7 PRBS9 PRBS11 PRBS15 PRBS20 PRBS23 QRSS11 QRSS20 FOX FOXCMA3000 ALL0 ALL1 ALT11: Alternating 1:1 ALT13: Alternating 1:3 ALT17: Alternating 1:7 ALT324: Alternating 3:24 *DEFault = PRBS11*
Response	None.
Example	`TMBP:RX1:PATT PRBS11`
Note	

Syntax	TMBPs:RX<Pt>:PATTern?
Description	This query returns the pattern type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`TMBP:RX1:PATT?` → `PRBS11`
Note	

5.1.11 TMBPs:RX<Pt>:PINVersion

Syntax	TMBPs:RX<Pt>:PINVersion <inverted>
Description	This command enables or disables pattern inversion.
Parameters	<Pt> = Port number
	<inverted> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`TMBP:RX1:PINV ON`
Note	The following patterns can be inverted: PRBSxx, QRSSxx, ALT11, ALT13, ALT17, ALT324, USER32BIT and USER2048BIT.

Syntax	TMBPs:RX<Pt>:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the pattern.
Parameter	<Pt> = Port number
Response	<inverted> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:RX1:PINV?` → `1`
Note	

5.1.12 TMBPs:RX<Pt>:PTSLots

Syntax	TMBPs:RX<Pt>:PTSLots <slots>
Description	This command sets the pattern time slots.
Parameters	<Pt> = Port number
	<slots> = <EXPRESSION PROGRAM DATA> Format: Numeric List List consist of slot number(s) ranging from 1 to 31
Response	None.
Examples	`TMBP:RX1:PTSL (1,3,5)`
Note	

Syntax	TMBPs:RX<Pt>:PTSLots?
Description	This query returns the pattern time slots.
Parameter	<Pt> = Port number
Response	<slots> = <EXPRESSION RESPONSE DATA> Format: Numeric List
Example	`TMBP:RX1:PTSL?` → `(1,3,5)`
Note	

5.1.13 TMBPs:RX<Pt>:UP16

Syntax	TMBPs:RX<Pt>:UP16 <pattern>
Description	This command sets the 16 bit user pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Valid characters: '0' and '1' The string must consist of 1 to 16 characters.
Response	None.
Example	`TMBP:RX1:UP16 "101101"`
Note	This command is for backward compatibility only, and a query command is not available. Actually the new 32 bit user pattern is set by this command.

5.1.14 TMBPs:RX<Pt>:UP32

Syntax	TMBPs:RX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters (one bit resolution).
Response	None.
Examples	`TMBP:RX2:UP32 "01101"`
Note	

Syntax	TMBPs:RX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`TMBP:RX2:UP32?` → `"01101"`
Note	

5.1.15 TMBPs:RX<Pt>:UP2K

Syntax	TMBPs:RX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`TMBP:RX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `TMBP:RX2:PATT` is `USER2048BIT`

Syntax	TMBPs:RX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	None.
Response	<pattern> = <STRING RESPONSE DATA>
Example	`TMBP:RX1:UP2K?` → `"12DF"`
Note	

5.1.16 TMBPs:RX<Pt>:AUDio

Syntax	TMBPs:RX<Pt>:AUDio <type>
Description	This command sets the audio decoding.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> OFF ON *DEFault = OFF*
Response	None.
Example	`TMBP:RX1:AUD ON`
Note	

Syntax	TMBPs:RX<Pt>:AUDio?
Description	This query returns the audio decoding.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`TMBP:RX1:AUD? → ON`
Note	

5.1.17 TMBPs:RX<Pt>:ATSLot

Syntax	TMBPs:RX<Pt>:ATSLot <timeslotno>
Description	This command sets the audio timeslot to be monitored.
Parameters	<Pt> = Port number
	<timeslotno> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=31* *DEFault=2*
Response	None.
Example	`TMBP:RX1:ATSL 4`
Note	

Syntax	TMBPs:RX<Pt>:ATSLot?
Description	This query returns the monitored audio timeslot.
Parameter	<Pt> = Port number
Response	<timeslotno> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:RX1:ATSL? → 4`
Note	

5.1.18 TMBPs:RX<Pt>:CAS

Syntax	TMBPs:RX<Pt>:CAS <enable>
Description	This command enables or disables the channel associated signaling.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`TMBP:RX1:CAS ON`
Note	

Syntax	TMBPs:RX<Pt>:CAS?
Description	This query returns the state (enabled/disabled) of the channel associated signaling.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:RX1:CAS? → 1`
Note	

5.2 Transmitter

5.2.1 TMBPs:TX<Pt>[:ENABled]

Syntax	TMBPs:TX<Pt>[:ENABled] <interface>
Description	This command enables or disables the 2 Mbps transmitter.
Parameters	<Pt> = Port number
	<interface> = <CHARACTER PROGRAM DATA> OFF ON *DEFault = OFF*
Response	None.
Examples	`TMBP:TX1 ON` `TMBP:TX1:ENAB ON`
Note	

Syntax	TMBPs:TX<Pt>[:ENABled]?
Description	This query returns the state (enabled/disabled) of the 2 Mbps transmitter.
Parameter	<Pt> = Port number
Response	<interface> = <CHARACTER RESPONSE DATA>
Examples	`TMBP:TX1?` → `ON` `TMBP:TX1:ENAB?` → `SDH`
Note	Returns `SDH` if E1 is over SDH or SONET.

5.2.2 TMBPs:TX<Pt>:FOLLow

Syntax	TMBPs:TX<Pt>:FOLLow <follow>
Description	This command sets the transmitter setting to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<follow> = <CHARACTER PROGRAM DATA> NONE: Do not follow TX1: Follow the setting of the port 1 transmitter *DEFault = NONE*
Response	None.
Example	`TMBP:TX2:FOLL TX1` (i.e., TX2 follows TX1)
Note	This command is not valid for `:TX1`

Syntax	TMBPs:TX<Pt>:FOLLow?
Description	This query returns if the transmitter follow the TX1 settings.
Parameter	<Pt> = Port number
Response	<follow> = <CHARACTER RESPONSE DATA>
Example	`TMBP:TX2:FOLL?` → `TX1`
Note	This command is not valid for `:TX1`

5.2.3 TMBPs:TX<Pt>:CONNector

Syntax	TMBPs:TX<Pt>:CONNector <type>
Description	This command sets the physical type of connector.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> BALanced: Using balanced connector UNBalanced: Using unbalanced connector *DEFault = UNBalanced*
Response	None.
Example	`TMBP:TX1:CONN BAL`
Note	

Syntax	TMBPs:TX<Pt>:CONNector?
Description	This query returns the physical type of connector.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`TMBP:TX1:CONN?` → `BAL`
Note	

5.2.4 TMBPs:TX<Pt>:DINSert[:ENABle]

Syntax	TMBPs:TX<Pt>:DINSert[:ENABle] <enable>
Description	This command enables/disables Drop and Insert.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`TMBP:TX1:DINS OFF`
Note	

Syntax	TMBPs:TX<Pt>:DINSert[:ENABle]?
Description	This query returns whether or not Drop and Insert is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:TX1:DINS?` → `0`
Note	

5.2.5 TMBPs:TX<Pt>:TIMing

Syntax	TMBPs:TX<Pt>:TIMing <source>
Description	This command sets the timing source.
Parameters	<Pt> = Port number
	<source> = <CHARACTER PROGRAM DATA> INTernal: Internal timing source EXTernal: External timing source RX: Received signal on the same port *DEFault = INTernal*
Response	None.
Example	`TMBP:TX1:TIM INT`
Note	

Syntax	TMBPs:TX<Pt>:TIMing?
Description	This query returns the timing source.
Parameter	<Pt> = Port number
Response	<source> = <CHARACTER RESPONSE DATA>
Example	`TMBP:TX1:TIM?` → `INT`
Note	

5.2.6 TMBPs:TX<Pt>:FOFFset

Syntax	TMBPs:TX<Pt>:FOFFset <offset>
Description	This command sets the frequency offset for the clock source. Unit: ppm.
Parameters	<Pt> = Port number
	<offset> = <NUMERIC PROGRAM DATA> *MINimum=-125, MAXimum=125, DEFault=0* *Allowed suffix = ppm*
Response	None.
Example	`TMBP:TX1:FOFF 0ppm`
Note	

Syntax	TMBPs:TX<Pt>:FOFFset?
Description	This query returns the frequency offset (ppm) for the clock source.
Parameter	<Pt> = Port number
Response	<offset> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:TX1:FOFF?` → `0`
Note	

5.2.7 TMBPs:TX<Pt>:CODE

Syntax	TMBPs:TX<Pt>:CODE <code>
Description	This command sets the line code type.
Parameters	<Pt> = Port number
	<code> = <CHARACTER PROGRAM DATA> AMI HDB3 *DEFault = HDB3*
Response	None.
Example	`TMBP:TX1:CODE AMI`
Note	

Syntax	TMBPs:TX<Pt>:CODE?
Description	This query returns the line code type.
Parameter	<Pt> = Port number
Response	<code> = <CHARACTER RESPONSE DATA>
Example	`TMBP:TX1:CODE?` → `AMI`
Note	

5.2.8 TMBPs:TX<Pt>:PCMFrame

Syntax	TMBPs:TX<Pt>:PCMFrame <enable>
Description	This command enables or disables the PCM frame.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`TMBP:TX1:PCMF ON`
Note	

Syntax	TMBPs:TX<Pt>:PCMFrame?
Description	This query returns whether or not PCM frame is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:TX1:PCMF?` → `1`
Note	

5.2.9 TMBPs:TX<Pt>:CRC4

Syntax	TMBPs:TX<Pt>:CRC4 <enable>
Description	This command enables or disables the CRC4 bits.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`TMBP:TX1:CRC4 ON`
Note	

Syntax	TMBPs:TX<Pt>:CRC4?
Description	This query returns the state (enabled/disabled) of the CRC4 bits.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:TX1:CRC4?` → `1`
Note	

5.2.10 TMBPs:TX<Pt>:SABits

Syntax	TMBPs:TX<Pt>:SABits <frame1>,<frame3>,<frame5>,<frame7>
Description	This command sets the Sa-bits.
Parameters	<Pt> = Port number
	<frame1> = <NUMERIC PROGRAM DATA> *MINimum=#B00000, MAXimum=#B11111* *DEFault=#B11111*
	<frame3> = <NUMERIC PROGRAM DATA> *MINimum=#B00000, MAXimum=#B11111* *DEFault=#B11111*
	<frame5> = <NUMERIC PROGRAM DATA> *MINimum=#B00000, MAXimum=#B11111* *DEFault=#B11111*
	<frame7> = <NUMERIC PROGRAM DATA> *MINimum=#B00000, MAXimum=#B11111* *DEFault=#B11111*
Response	None.

Examples	`TMBP:TX1:SAB #B11111,#B11111,#B11111,#B11111`
	`TMBP:TX1:SAB 31,31,31,31`
	`TMBP:TX1:SAB min,max,def,0`

Note	

Syntax	TMBPs:TX<Pt>:SABits?
Description	This query returns the Sa-Bits.
Parameter	<Pt> = Port number
Response	<frame1> = <BINARY NUMERIC RESPONSE DATA>
	<frame3> = <BINARY NUMERIC RESPONSE DATA>
	<frame5> = <BINARY NUMERIC RESPONSE DATA>
	<frame7> = <BINARY NUMERIC RESPONSE DATA>
Example	`TMBP:TX1:SAB?` → `#B11111,#B11111,#B11111,#B11111`
Note	

5.2.11 TMBPs:TX<Pt>:SABMode

Syntax	TMBPs:TX<Pt>:SABMode <mode>
Description	This command sets the Sa-Bits mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> INSert BYPass *DEFault = BYPass*
Response	None.
Example	`TMBP:TX1:SABM INS`
Note	

5.2.12 TMBPs:TX<Pt>:SABMode?

Syntax	TMBPs:TX<Pt>:SABMode?
Description	This query returns the Sa-Bits mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`TMBP:TX1:SABM?` → `INS`
Note	

5.2.13 TMBPs:TX<Pt>:PATTern

Syntax	TMBPs:TX<Pt>:PATTern <type>
Description	This command sets the pattern type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> OFF USER16BIT: Obsolete. For backward compatibility only. Same as USER32BIT. USER32BIT: 32 bit user defined pattern. USER2048BIT: 2048 bit user defined pattern. PRBS6 PRBS7 PRBS9 PRBS11 PRBS15 PRBS20 PRBS23 QRSS11 QRSS20 FOX FOXCMA3000 ALL0 ALL1 ALT11: Alternating 1:1 ALT13: Alternating 1:3 ALT17: Alternating 1:7 ALT324: Alternating 3:24 *DEFault = PRBS11*
Response	None.
Example	`TMBP:TX1:PATT PRBS11`
Note	

Syntax	TMBPs:TX<Pt>:PATTern?
Description	This query returns the pattern type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`TMBP:TX1:PATT?` → `PRBS11`
Note	

5.2.14 TMBPs:TX<Pt>:PINVersion

Syntax	TMBPs:TX<Pt>:PINVersion <inversion>
Description	This command enables or disables pattern inversion.
Parameters	<Pt> = Port number
	<inversion> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`TMBP:TX1:PINV ON`
Note	The following patterns can be inverted: PRBSxx, QRSSxx, ALT11, ALT13, ALT17, ALT324, USER32BIT and USER2048BIT.

Syntax	TMBPs:TX<Pt>:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the pattern.
Parameter	<Pt> = Port number
Response	<inversion> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:TX1:PINV?` → `1`
Note	

5.2.15 TMBPs:TX<Pt>:PTSLots

Syntax	TMBPs:TX<Pt>:PTSLots <timeslot>
Description	This command sets the pattern time slots.
Parameters	<Pt> = Port number
	<timeslots> = <EXPRESSION PROGRAM DATA> Format: Numeric List List consist of slot number(s) ranging from 1 to 31
Response	None.
Examples	`TMBP:TX1:PTSL (1,3,5)`
Note	

Syntax	TMBPs:TX<Pt>:PTSLots?
Description	This query returns the pattern time slots.
Parameter	<Pt> = Port number
Response	<timeslot> = <EXPRESSION RESPONSE DATA> Format: Numeric List.
Example	`TMBP:TX1:PTSL?` → `(1,3,5)`
Note	

5.2.16 TMBPs:TX<Pt>:UP16

Syntax	TMBPs:TX<Pt>:UP16 <pattern>
Description	This command sets the 16 bit user pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Valid characters: '0' and '1' The string must consist of 1 to 16 characters.
Response	None.
Example	`TMBP:TX1:UP16 "101101"`
Note	This command is for backward compatibility only, and a query command is not available. Actually the new 32 bit user pattern is set by this command.

5.2.17 TMBPs:TX<Pt>:UP32

Syntax	TMBPs:TX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters (one bit resolution).
Response	None.
Examples	`TMBP:TX1:UP32 "01101"`
Note	

Syntax	TMBPs:TX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`TMBP:TX1:UP32?` → `"01101"`
Note	

5.2.18 TMBPs:TX<Pt>:UP2K

Syntax	TMBPs:TX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`TMBP:TX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `TMBP:TX1:PATT` is `USER2048BIT`

Syntax	TMBPs:TX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`TMBP:TX1:UP2K?` → `"12DF"`
Note	

5.2.19 TMBPs:TX<Pt>:UTSLots

Syntax	TMBPs:TX<Pt>:UTSLots <content>
Description	This command sets the content of unused time slots.
Parameters	<Pt> = Port number
	<content> = <NUMERIC PROGRAM DATA> *MINimum=#B00000000, MAXimum=#B11111111* *DEFault=#B01010101*
Response	None.
Example	`TMBP:TX1:UTSL 128`
Note	

Syntax	TMBPs:TX<Pt>:UTSLots?
Description	This query returns the content of unused time slots.
Parameter	<Pt> = Port number <content> = <BINARY NUMERIC RESPONSE DATA>
Response	None.
Example	`TMBP:TX1:UTSL?` → `#B10101010`
Note	

5.2.20 TMBPs:TX<Pt>:SCContent

Syntax	TMBPs:TX<Pt>:SCContent <content>
Description	This command sets the channel content.
Parameters	<Pt> = Port number
	<content> = <CHARACTER PROGRAM DATA> OFF TONE TRANsparent SPEech *DEFault=OFF*
Response	None.
Example	`TMBP:TX1:SCC TONE`
Note	

Syntax	TMBPs:TX<Pt>:SCContent?
Description	This query returns the channel content.
Parameter	<Pt> = Port number
Response	<content> = <CHARACTER RESPONSE DATA>
Example	`TMBP:TX1:SCC?` → `TONE`
Note	

5.2.21 TMBPs:TX<Pt>:SCTSlot

Syntax	TMBPs:TX<Pt>:SCTSlot <timeslot>
Description	This command sets the channel time slot.
Parameters	<Pt> = Port number
	<timeslot> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=31* *DEFault=2*
Response	None.
Example	`TMBP:TX1:SCTS 4`
Note	

Syntax	TMBPs:TX<Pt>:SCTSlot?
Description	This query returns the channel time slot.
Parameter	<Pt> = Port number
Response	<timeslot> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:TX1:SCTS?` → `4`
Note	

5.2.22 TMBPs:TX<Pt>:TFR

Syntax	TMBPs:TX<Pt>:TFR <frequency>
Description	This command sets the tone frequency.
Parameters	<Pt> = Port number
	<frequency> = <NUMERIC PROGRAM DATA> *MINimum = 1 hz, MAXimum = 4000 hz* *DEFault = 440* *Allowed suffixes = HZ, KHZ*
Response	None.
Example	`TMBP:TX1:TFR 500HZ`
Note	

Syntax	TMBPs:TX<Pt>:TFR?
Description	This query returns the tone frequency (Hz).
Parameter	<Pt> = Port number
Response	<frequency> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:TX1:TFR?` → `500`
Note	

5.2.23 TMBPs:TX<Pt>:TLEVel

Syntax	TMBPs:TX<Pt>:TLEVel <level>
Description	This command sets the tone level.
Parameters	<Pt> = Port number
	<level> = <NUMERIC PROGRAM DATA> *MINimum = -70, MAXimum = 3* *DEFault = -20* *Allowed suffixes = dB*
Response	None.
Example	`TMBP:TX1:TLEV 2dB`
Note	

Syntax	TMBPs:TX<Pt>:TLEVel?
Description	This query returns the tone level.
Parameter	<Pt> = Port number
Response	<level> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:TX1:TLEV?` → `2`
Note	

5.2.24 TMBPs:TX<Pt>:CAS

Syntax	TMBPs:TX<Pt>:CAS <enable>
Description	This command enables or disables the channel associated signaling.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`TMBP:TX1:CAS ON`
Note	

Syntax	TMBPs:TX<Pt>:CAS?
Description	This query returns the state (enabled/disabled) of the channel associated signaling.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:TX1:CAS?` → `1`
Note	

5.2.25 TMBPs:TX<Pt>:CASChannel

Syntax	TMBPs:TX<Pt>:CASChannel <channel>
Description	This command sets the CAS channel number.
Parameters	<Pt> = Port number
	<channel> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=30, DEFault=1*
Response	None.
Example	`TMBP:TX1:CASC 5`
Note	

Syntax	TMBPs:TX<Pt>:CASChannel?
Description	This query returns the CAS channel number.
Parameter	<Pt> = Port number
Response	<channel> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:TX1:CASC?` → `5`
Note	

5.2.26 TMBPs:TX<Pt>:CASBits

Syntax	TMBPs:TX<Pt>:CASBits <bits>
Description	This command sets the CAS channel bits.
Parameters	<Pt> = Port number
	<bits> = <NUMERIC PROGRAM DATA> *MINimum=#B0000, MAXimum=#B1111, DEFault=#B1111*
Response	None.
Example	`TMBP:TX1:CASB #B1111`
Note	

Syntax	TMBPs:TX<Pt>:CASBits?
Description	This query returns the CAS channel bits.
Parameter	<Pt> = Port number
Response	<bits> = <BINARY NUMERIC RESPONSE DATA>
Example	`TMBP:TX1:CASB?` → `#B1111`
Note	

5.2.27 TMBPs:TX<Pt>:COCBits

Syntax	TMBPs:TX<Pt>:COCBits <bits>
Description	This command sets the CAS other channel bits.
Parameters	<Pt> = Port number
	<bits> = <NUMERIC PROGRAM DATA> *MINimum=#B0000, MAXimum=#B1111, DEFault=#B1001*
Response	None.
Example	`TMBP:TX1:COCB #B0000`
Note	

Syntax	TMBPs:TX<Pt>:COCBits?
Description	This query returns the CAS other channel bits.
Parameter	<Pt> = Port number
Response	<bits> = <BINARY NUMERIC RESPONSE DATA>
Example	`TMBP:TX1:COCB?` → `#B0000`
Note	

5.3 Stimuli

5.3.1 TMBPs:STIMuli:TX<Pt>:ALARm

Syntax	TMBPs:STIMuli:TX<Pt>:ALARm <type>
Description	This command sets the alarm type to be generated.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> NALarm: No alarm NSIGnal: No signal AIS: Alarm indication signal NFRame: No frame DALarm: Distant alarm (RDI) NSYNc: No pattern sync NCAM: No CAS MFAS DMF: Distant MF alarm *DEFault = NALarm*
Response	None.
Example	`TMBP:STIM:TX1:ALAR NSIG`
Note	

Syntax	TMBPs:STIMuli:TX<Pt>:ALARm?
Description	This query returns the generated alarm type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`TMBP:STIM:TX1:ALAR? → NSIG`
Note	

5.3.2 TMBPs:STIMuli:TX<Pt>:ERRor

Syntax	TMBPs:STIMuli:TX<Pt>:ERRor <destination>
Description	This command sets the error destination.
Parameters	<Pt> = Port number
	<destination> = <CHARACTER PROGRAM DATA> FAS FNFas: FAS and NFAS FWORd: FAS word CRC4 CMFas: CRC4 MFAS CODE PATTern CAMFas: CAS MFAS EBIT: E-Bit PSLip: Pattern slip FSLip: Frame slip TRANsparent: Transparent *DEFault = CRC4*
Response	None.
Example	`TMBP:STIM:TX1:ERR FAS`
Note	

Syntax	TMBPs:STIMuli:TX<Pt>:ERRor?
Description	This query returns the error destination.
Parameter	<Pt> = Port number
Response	<destination> = <CHARACTER RESPONSE DATA>
Example	`TMBP:STIM:TX1:ERR? → FAS`
Note	

5.3.3 TMBPs:STIMuli:TX<Pt>:EINSert

Syntax	TMBPs:STIMuli:TX<Pt>:EINSert <insertion>
Description	This command sets the method to insert errors.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> OFF MANual B02: Burst · 1E-02 B03: Burst · 1E-03 B04: Burst · 1E-04 B05: Burst · 1E-05 B06: Burst · 1E-06 B07: Burst · 1E-07 ES: Errored seconds SES: Severe errored seconds *DEFault = OFF*
Response	None.
Example	`TMBP:STIM:TX1:EINS MAN`
Note	If insertion is set to MANual, errors are inserted with SYST:STIM:INS. See section 2.3.23

Syntax	TMBPs:STIMuli:TX<Pt>:EINSert?
Description	This query returns the error insertion method.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`TMBP:STIM:TX1:EINS? → MAN`
Note	

5.3.4 TMBPs:STIMuli:TX<Pt>:EBLength

Syntax	TMBPs:STIMuli:TX<Pt>:EBLength <burstlength>
Description	This command sets the error burst length to generate.
Parameters	<Pt> = Port number
	<burstlength> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=255, DEFault=1*
Response	None.
Example	`TMBP:STIM:TX1:EBL 1`
Note	For TMBP:STIM:TX<Pt>:ERR? → FWORd, *MAXimum = 16*

Syntax	TMBPs:STIMuli:TX<Pt>:EBLength?
Description	This query returns the error burst length to generate.
Parameter	<Pt> = Port number
Response	<burstlength> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:STIM:TX1:EBL? → 1`
Note	

5.4 Result

5.4.1 TMBPs:RX<Pt>:IFETch?

Syntax	TMBPs:RX<Pt>:IFETch? <parameter>
Description	This command fetches results from a 2 Mbps interval if available.
Parameters	<Pt> = Port number
	({<parameter>} + {,}∗) = <EXPRESSION PROGRAM DATA> The response format is listed for each parameter. **Alarms**

continued on next page. . .

. . . continued from previous page

	NSIGnal: No signal. Response: <Seconds>,<Ratio> AIS: Alarm indication Signal. Response: <Seconds>,<Ratio> NFRame: No frame. Response: <Seconds>,<Ratio> NCMF: No CRC4 multi frame. Response: <Seconds>,<Ratio> DALarm: Distant alarm. Response: <Seconds>,<Ratio> NSYNc: No sync. Response: <Seconds>,<Ratio> NCAM: No CAS MF. Response: <Seconds>,<Ratio> DMF: Distant MF. Response: <Seconds>,<Ratio> **Errors** FAS: Frame alignment signal. Response: <Count>,<Ratio> PATTern: Pattern. Response: <Count>,<Ratio> CRC4: CRC4. Response: <Count>,<Ratio> CRCM: CRC4 MFAS. Response: <Count>,<Ratio> EBIT: E-Bits. Response: <Count>,<Ratio> CODE: Code. Response: <Count>,<Ratio> PSLip: Pattern slip. Response: <Count>,<Ratio> PBLock: Pattern block. Response: <Count>,<Ratio> FSLip: Frame slip. Response: <Count>,<Ratio> **FAS performance errors** FES: FAS ES. Response: <Count>,<Ratio%> FSES: FAS SES. Response: <Count>,<Ratio%> FBBE: FAS BBE. Response: <Count>,<Ratio%> FALS: FAS ALS. Response: <Count>,<Ratio%> FUAT: FAS UAT. Response: <Count>,<Ratio%> FAVT: FAS AVT. Response: <Count>,<Ratio%> FEFS: FAS EFS. Response: <Count>,<Ratio%> **Pattern performance errors** PES: Pattern ES. Response: <Count>,<Ratio%> PSES: Pattern SES. Response: <Count>,<Ratio%> PBBE: Pattern BBE. Response: <Count>,<Ratio%> PALS: Pattern ALS. Response: <Count>,<Ratio%> PUAT: Pattern UAT. Response: <Count>,<Ratio%> PAVT: Pattern AVT. Response: <Count>,<Ratio%> PEFS: Pattern EFS. Response: <Count>,<Ratio%> **CRC4 performance errors** CES: CRC4 ES. Response: <Count>,<Ratio%> CSES: CRC4 SES. Response: <Count>,<Ratio%> CBBE: CRC4 BBE. Response: <Count>,<Ratio%> CALS: CRC4 ALS. Response: <Count>,<Ratio%> CUAT: CRC4 UAT. Response: <Count>,<Ratio%> CAVT: CRC4 AVT. Response: <Count>,<Ratio%> CEFS: CRC4 EFS. Response: <Count>,<Ratio%> **E-Bit performance errors** EES: E-Bit ES. Response: <Count>,<Ratio%> ESES: E-Bit SES. Response: <Count>,<Ratio%> EBBE: E-Bit BBE. Response: <Count>,<Ratio%> EALS: E-Bit ALS. Response: <Count>,<Ratio%> EUAT: E-Bit UAT. Response: <Count>,<Ratio%> EAVT: E-Bit AVT. Response: <Count>,<Ratio%> EEFS: E-Bit EFS. Response: <Count>,<Ratio%>
Response	{(<result>), }* = <EXPRESSION RESPONSE DATA> Expression format: Numeric List

continued on next page. . .

. . . continued from previous page

	Each result is formatted according to the specification in the parameter field. Values that are not relevant or applicable for the current measurement, returns NaN (section 1.6.1).
Example	`TMBP:RX1:IFET? (AIS,FES)` → `(9,0.0429),(10,0.0343)`
Note	This command fetches the results from the interval selected using the `MEASurement:SETup:SELect` command (see section 18.2.2). If the requested result is not available, NaN (section 1.6.1) is returned. If there is one or more results, the last "," is always removed.

5.5 Status

5.5.1 TMBPs:STATus:RX<Pt>:AESummary[:EVENt]?

Syntax	TMBPs:STATus:RX<Pt>:AESummary[:EVENt]?
Description	This query returns alarms and errors summary event register. The content of this event register is summarized in DB1 of the STATus:INTerface:PORT<Pt>:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 (2) = Error summary DB3 - DB16 = NOT USED
Example	`TMBP:STAT:RX1:AES?` → `1` `TMBP:STAT:RX1:AES:EVEN?` → `1`
Note	

5.5.2 TMBPs:STATus:RX<Pt>:AESummary:CONDition?

Syntax	TMBPs:STATus:RX<Pt>:AESummary:CONDition?
Description	This query returns alarms and errors summary operation register query. DB1 is summary of TMBPs:STATus:RX<Pt>:ALARm[:EVENt]? DB2 is summary of TMBPs:STATus:RX<Pt>:ERRor[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 (2) = Error summary DB3 - DB16 = NOT USED
Example	`TMBP:STAT:RX1:AES:COND?` → `1`
Note	

5.5.3 TMBPs:STATus:RX<Pt>:ALARm[:EVENt]?

Syntax	TMBPs:STATus:RX<Pt>:ALARm[:EVENt]?
Description	This query returns alarms event register query. The content of this register is summarized in DB1 of the TMBPs:STATus:RX<Pt>:AESummary:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = No sync DB2 (2) = Distant MF DB3 (4) = No CAS MF DB4 (8) = Distant DB5 (16) = No CRC4 MF DB6 (32) = No frame DB7 (64) = Alarm indication signal (AIS) DB8 (128) = No signal DB9 - DB16 = NOT USED
Example	`TMBP:STAT:RX1:ALAR?` → `64`
Note	

5.5.4 TMBPs:STATus:RX<Pt>:ALARm:CONDition?

Syntax	TMBPs:STATus:RX<Pt>:ALARm:CONDition?
Description	This query returns alarms condition register query.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = No sync DB2 (2) = Distant MF DB3 (4) = No CAS MF DB4 (8) = Distant DB5 (16) = No CRC4 MF DB6 (32) = No Frame DB7 (64) = Alarm Indication Signal (AIS) DB8 (128) = No Signal DB9 - DB16 = NOT USED
Example	`TMBP:STAT:RX1:ALAR:COND?` → `64`
Note	

5.5.5 TMBPs:STATus:RX<Pt>:ERRor[:EVENt]?

Syntax	TMBPs:STATus:RX<Pt>:ERRor[:EVENt]?
Description	This query returns errors event register. The content of this register is summarized in DB2 of the TMBPs:STATus:RX<Pt>:AESummary:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Pattern slip DB2 (2) = Pattern DB3 (4) = E-Bit DB4 (8) = CRC4 DB5 (16) = Frame slip DB6 (32) = CRC4 MFAS DB7 (64) = FAS DB8 (128) = Code DB9 (256) = Sequence DB10 - DB16 = NOT USED
Example	`TMBP:STAT:RX1:ERR?` → `64`
Note	

5.5.6 TMBPs:STATus:RX<Pt>:ERRor:CONDition?

Syntax	TMBPs:STATus:RX<Pt>:ERRor:CONDition?
Description	This query returns errors condition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Pattern slip DB2 (2) = Pattern DB3 (4) = E-Bit DB4 (8) = CRC4 DB5 (16) = Frame slip DB6 (32) = CRC4 MFAS DB7 (64) = FAS DB8 (128) = Code DB9 (256) = Sequence DB10 - DB16 = NOT USED
Example	`TMBP:STAT:RX1:ERR:COND?` → `64`
Note	

5.5.7 TMBPs:STATus:RX<Pt>:PSLevel?

Syntax	TMBPs:STATus:RX<Pt>:PSLevel?
Description	This query returns physical signal level. Unit: dB.
Parameter	<Pt> = Port number
Response	<signallevel> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:STAT:RX1:PSL?` → `0`
Note	Minimum level is -11 dB.

5.5.8 TMBPs:STATus:RX<Pt>:PDEViation?

Syntax	TMBPs:STATus:RX<Pt>:PDEViation?
Description	This query returns physical deviation. Units: ppm and bps.
Parameter	<Pt> = Port number
Response	<ppm> = <NR1 NUMERIC RESPONSE DATA>
	<bps> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:STAT:RX1:PDEV?` → `0, 0`
Note	

5.5.9 TMBPs:STATus:RX<Pt>:PBRate?

Syntax	TMBPs:STATus:RX<Pt>:PBRate?
Description	This query returns physical bit rate. Unit: bps.
Parameter	<Pt> = Port number
Response	<bitrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:STAT:RX1:PBR?` → `2048000`
Note	

5.5.10 TMBPs:STATus:RX<Pt>:PPBRate?

Syntax	TMBPs:STATus:RX<Pt>:PPBRate?
Description	This query returns payload pattern bit rate. Unit: bps.
Parameter	<Pt> = Port number
Response	<bitrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:STAT:RX1:PPBR?` → `64000`
Note	

5.5.11 TMBPs:STATus:RX<Pt>:FNFWord?

Syntax	TMBPs:STATus:RX<Pt>:FNFWord?
Description	This query returns FAS Non FAS word. Returns the content of time slot 0. Data length is always $2 \cdot 64$ bit shown as hexadecimal.
Parameter	<Pt> = Port number
Response	<hex> = <HEXADECIMAL NUMERIC RESPONSE DATA> Frame Group I, Frame 0-7
	<hex> = <HEXADECIMAL NUMERIC RESPONSE DATA> Frame Group II, Frame 8-15
Example	`TMBP:STAT:RX1:FNFW?` → `#H5F1BDF1B5F1B5F9b,#HDF1BDF1BDF1BDF9B`
Note	

5.5.12 TMBPs:STATus:RX<Pt>:FDUMp?

Syntax	TMBPs:STATus:RX<Pt>:FDUMp?
Description	This query returns a sample of a full E1 Frame (32 bytes).
Parameters	<Pt> = Port number
Response	<hex> = <EXPRESSION RESPONSE DATA> Expression format: Numeric list Contains all 32 Frame timeslots.
Example	`TMBP:STAT:RX1:FDUM?` → `#H1B,#H55,#H55,...`
Note	

5.5.13 TMBPs:STATus:RX<Pt>:ACONtent?

Syntax	TMBPs:STATus:RX<Pt>:ACONtent?
Description	This query returns audio channel content.
Parameter	<Pt> = Port number
Response	<audiocontent> = <BINARY NUMERIC RESPONSE DATA>
Example	`TMBP:STAT:RX1:ACON?` →`#B01010101`
Note	For sub-rates this field shows the content of the sub channel.

5.5.14 TMBPs:STATus:RX<Pt>:AEBContent?

Syntax	TMBPs:STATus:RX<Pt>:AEBContent?
Description	This query returns sub channel audio even bit inverting content.
Parameter	<Pt> = Port number
Response	<audiocontent> = <BINARY NUMERIC RESPONSE DATA>
Example	`TMBP:STAT:RX1:AEBC?` → `#B10000000`
Note	With A-law coded speech, it is possible to observe the A-law code words before the even bit inverting.

5.5.15 TMBPs:STATus:RX<Pt>:APPeak?

Syntax	TMBPs:STATus:RX<Pt>:APPeak?
Description	This query returns sub channel audio positive peak value.
Parameter	<Pt> = Port number
Response	<positivepeak> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:STAT:RX1:APP?` → `85`
Note	This query only applies for A-law speech and shows the largest received A-law coded value.

5.5.16 TMBPs:STATus:RX<Pt>:ANPeak?

Syntax	TMBPs:STATus:RX<Pt>:ANPeak?
Description	This query returns sub channel audio negative peak value.
Parameter	<Pt> = Port number
Response	<negativepeak> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:STAT:RX1:ANP?` → `-85`
Note	This query only applies for A-law speech and shows the smallest received A-law coded value.

5.5.17 TMBPs:STATus:RX<Pt>:ALEVel?

Syntax	TMBPs:STATus:RX<Pt>:ALEVel?
Description	This query returns sub channel audio level. Unit: dBm.
Parameter	<Pt> = Port number
Response	<audiolevel> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:STAT:RX1:ALEV?` → `-4`
Note	

5.5.18 TMBPs:STATus:RX<Pt>:AFRequency?

Syntax	TMBPs:STATus:RX<Pt>:AFRequency?
Description	This query returns sub channel audio tone frequency. Unit: Hz.
Parameter	<Pt> = Port number
Response	<audiofrequency> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:STAT:RX1:AFR?` → `366`
Note	

5.5.19 TMBPs:STATus:RX<Pt>:ACOFfset?

Syntax	TMBPs:STATus:RX<Pt>:ACOFfset?
Description	This query returns sub channel audio coder offset.
Parameter	<Pt> = Port number
Response	<coderoffset> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:STAT:RX1:ACOF?` → `0`
Note	Only available for 64 or 32 kbps audio sub channels.

5.5.20 TMBPs:STATus:RX<Pt>:CMSignal?

Syntax	TMBPs:STATus:RX<Pt>:CMSignal?
Description	This query returns CAS MFAS signal status.
Parameter	<Pt> = Port number
Response	<status> = <BINARY NUMERIC RESPONSE DATA>
Example	`TMBP:STAT:RX1:CMS?` →`#B00000101`
Note	

5.5.21 TMBPs:STATus:RX<Pt>:CBITs?

Syntax	TMBPs:STATus:RX<Pt>:CBITs?
Description	This query returns CAS bits.
Parameter	<Pt> = Port number
Response	<bits> = <EXPRESSION RESPONSE DATA> Expression format: Numeric list Contains all 30 CAS bit sets.
Example	`TMBP:STAT:RX1:CBIT?` → `(#B0101,#B0101, ..., #B0101,#B0101)`
Note	

5.6 APS

This section document commands for Automatic Protection Switching testing.

5.6.1 TMBPs:APS:STARt

Syntax	TMBPs:APS:STARt
Description	This command starts the APS (Automatic Protection Switching) measurement.
Parameter	None.
Response	None.
Example	`TMBP:APS:STAR`
Note	

5.6.2 TMBPs:APS:STOP

Syntax	TMBPs:APS:STOP
Description	This command stops the APS (Automatic Protection Switching) measurement.
Parameter	None.
Response	None.
Example	`TMBP:APS:STOP`
Note	

5.6.3 TMBPs:APS:RX<Pt>:NUMBer?

Syntax	TMBPs:APS:RX<Pt>:NUMBer?
Description	This query returns the number of times has measured.
Parameter	<Pt> = Port number
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:APS:RX1:NUMB?` → `17`
Note	

5.6.4 TMBPs:APS:RX<Pt>:ATIMe?

Syntax	TMBPs:APS:RX<Pt>:ATIMe?
Description	This query returns the average time has measured. Unit: ms.
Parameter	<Pt> = Port number
Response	<avg> = <NR2 NUMERIC RESPONSE DATA>
Example	`TMBP:APS:RX1:ATIM?` → `4.000`
Note	The maximum measurable time is 4000 ms. The maximum measurable time will be responded if the result exceeds 4000 ms.

5.6.5 TMBPs:APS:RX<Pt>:MTIMe?

Syntax	TMBPs:APS:RX<Pt>:MTIMe?
Description	This query returns the maximum time has measured. Unit: ms.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
	<limit exceeded> = <NR1 NUMERIC RESPONSE DATA> Returns 1, if the reference maximum limit has been exceeded.
Example	`TMBP:APS:RX1:MTIM?` → `29.170,0`
Note	The maximum measurable time is 4000 ms. The maximum measurable time will be responded if the result exceeds 4000 ms.

5.6.6 TMBPs:APS:RX<Pt>:LTIMe?

Syntax	TMBPs:APS:RX<Pt>:LTIMe?
Description	This query returns the minimum time of the reference event occurrences. Unit: ms.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`TMBP:APS:RX1:LTIM?` → `1.0`
Note	The maximum measurable time is 4000 ms. The maximum measurable time will be responded if the result exceeds 4000 ms.

5.6.7 TMBPs:APS:RX<Pt>:EVENt

Syntax	TMBPs:APS:RX<Pt>:EVENt <event>
Description	This command sets the Time Reference event.
Parameters	<Pt> = Port number
	<event> = <CHARACTER PROGRAM DATA> AIS = Alarm indication signal NFRame = No frame PERRor = Pattern error
Response	None.
Example	`TMBP:APS:RX1:EVEN AIS`
Note	

Syntax	TMBPs:APS:RX<Pt>:EVENt?
Description	This query returns the time reference event.
Parameter	<Pt> = Port number
Response	<event> = <CHARACTER RESPONSE DATA>
Example	`TMBP:APS:RX1:EVEN?` → `AIS`
Note	

5.6.8 TMBPs:APS:RX<Pt>:MLIMit

Syntax	TMBPs:APS:RX<Pt>:MLIMit <max>
Description	This command sets the time reference maximum limit. Unit: ms.
Parameters	<Pt> = Port number
	<max> = <NUMERIC PROGRAM DATA> *MINimum = 0.000, MAXimum = 4000.000, DEFault = 50.000*
Response	None.
Example	`TMBP:APS:RX1:MLIM 50.000`
Note	

Syntax	TMBPs:APS:RX<Pt>:MLIMit?
Description	This query returns the time reference maximum limit. Unit: ms.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
Example	`TMBP:APS:RX1:MLIM?` → `50.000`
Note	

5.7 RTD

This section document commands to retrieve Round Trip Delay measurement results.
Commands for general RTD settings are described in section 17.1 on page 953.

5.7.1 TMBPs:RTD:RX<Pt>:NUMBer?

Syntax	TMBPs:RTD:RX<Pt>:NUMBer?
Description	This query returns the number of the RTD data.
Parameter	<Pt> = Port number
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`TMBP:RTD:RX1:NUMB?` → `2`
Note	

5.7.2 TMBPs:RTD:RX<Pt>:ATIMe?

Syntax	TMBPs:RTD:RX<Pt>:ATIMe?
Description	This query returns the average time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<avg> = <NR2 NUMERIC RESPONSE DATA>
Example	`TMBP:RTD:RX1:ATIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

5.7.3 TMBPs:RTD:RX<Pt>:MTIMe?

Syntax	TMBPs:RTD:RX<Pt>:MTIMe?
Description	This query returns the maximum time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
	<limit exceeded> = <NR1 NUMERIC RESPONSE DATA> Returns 1, if the reference maximum limit has been exceeded.
Example	`TMBP:RTD:RX1:MTIM?` → `1.0,0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

5.7.4 TMBPs:RTD:RX<Pt>:LTIMe?

Syntax	TMBPs:RTD:RX<Pt>:LTIMe?
Description	This query returns the least (minimum) time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`TMBP:RTD:RX1:LTIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

Chapter 6

T1

6.1 Receiver

6.1.1 T1:RX<Pt>[:ENABled]

Syntax	T1:RX<Pt>[:ENABled] <interface>
Description	This command enables or disables the T1 receiver.
Parameters	<Pt> = Port number
	<interface> = <CHARACTER PROGRAM DATA> OFF ON *DEFault = OFF*
Response	None.
Example	`T1:RX1 ON`
Note	

Syntax	T1:RX<Pt>[:ENABled]?
Description	This query returns the state (enabled/disabled) of the T1 receiver.
Parameter	<Pt> = Port number
Response	<interface> = <CHARACTER RESPONSE DATA>
Example	`T1:RX1?` → `ON` `T1:RX1:ENAB?` → `SDH`
Note	Returns `SDH` if T1 is over SDH or SONET.

6.1.2 T1:RX<Pt>:FOLLow

Syntax	T1:RX<Pt>:FOLLow <follow>
Description	This command sets the receiver to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<follow> = <CHARACTER PROGRAM DATA> NONE: Do not follow TX: Transmitter of the same port RX1: Receiver of port 1 *DEFault = NONE*
Response	None.
Example	`T1:RX1:FOLL TX` (i.e., RX1 follows TX1) `T1:RX2:FOLL RX1` (i.e., RX2 follows RX1)
Note	

Syntax	T1:RX<Pt>:FOLLow?
Description	This query returns if the receiver follow another setup.
Parameter	<Pt> = Port number
Response	<follow> = <CHARACTER RESPONSE DATA>
Example	`T1:RX1:FOLL?` → `TX`
Note	

6.1.3 T1:RX<Pt>:MODE

Syntax	T1:RX<Pt>:MODE <mode>
Description	This command sets the signal termination mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> TERMinate: Nominal impedance. Normal frequency dependent AGC. BRIDged: High impedance. MONitor: Nominal impedance. Frequency linear AGC. *DEFault = TERMinate*
Response	None.
Example	`T1:RX1:MODE TERM`
Note	

Syntax	T1:RX<Pt>:MODE?
Description	This query returns the signal termination mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`T1:RX1:MODE?` → `TERM`
Note	

6.1.4 T1:RX<Pt>:CODE

Syntax	T1:RX<Pt>:CODE <code>
Description	This command sets the line code type.
Parameters	<Pt> = Port number
	<code> = <CHARACTER PROGRAM DATA> AMI B8ZS *DEFault = B8ZS*
Response	None.
Example	`T1:RX1:CODE AMI`
Note	

Syntax	T1:RX<Pt>:CODE?
Description	This query returns the line code type.
Parameter	<Pt> = Port number
Response	<code> = <CHARACTER RESPONSE DATA>
Example	`T1:RX1:CODE?` → `AMI`
Note	

6.1.5 T1:RX<Pt>:PCMFrame

Syntax	T1:RX<Pt>:PCMFrame <enable>
Description	This command enables or disables the PCM frame.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`T1:RX1:PCMF ON`
Note	

Syntax	T1:RX<Pt>:PCMFrame?
Description	This query returns whether or not PCM frame is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:RX1:PCMF?` → `1`
Note	

6.1.6 T1:RX<Pt>:FTYPe

Syntax	T1:RX<Pt>:FTYPe <type>
Description	This command sets the PCM frame type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> TSF: T1 SF/D4 mode TESF: T1 ESF mode JESF: J1 ESF mode *DEFault = TSF*
Response	None.
Example	`T1:RX1:FTYP TESF`
Note	

Syntax	T1:RX<Pt>:FTYPe?
Description	This query returns PCM frame type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`T1:RX1:FTYP?` → `TESF`
Note	

6.1.7 T1:RX<Pt>:PATTern

Syntax	T1:RX<Pt>:PATTern <type>
Description	This command sets the pattern type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> OFF USER16BIT: Obsolete. For backward compatibility only. Same as USER32BIT. USER32BIT: 32 bit user defined pattern. USER2048BIT: 2048 bit user defined pattern. PRBS9 PRBS11 PRBS15 PRBS20 PRBS23 PRBS29 PRBS31 QRSS20 FOX FOXCMA3000 ALL0 ALL1 ALT11: Alternating 1:1 ALT13: Alternating 1:3 ALT17: Alternating 1:7 ALT324: Alternating 3:24 *DEFault = PRBS11*
Response	None.
Example	`T1:RX1:PATT PRBS11`

Syntax	T1:RX<Pt>:PATTern?
Description	This query returns the pattern type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`T1:RX1:PATT?` → `PRBS11`
Note	

6.1.8 T1:RX<Pt>:PINVersion

Syntax	T1:RX<Pt>:PINVersion <inverted>
Description	This command enables or disables pattern inversion.
Parameters	<Pt> = Port number
	<inverted> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`T1:RX1:PINV ON`
Note	The following patterns can be inverted: PRBSxx, QRSSxx, ALT11, ALT13, ALT17, ALT324, USER32BIT and USER2048BIT.

Syntax	T1:RX<Pt>:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the pattern.
Parameter	<Pt> = Port number
Response	<inverted> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:RX1:PINV? → 1`
Note	

6.1.9 T1:RX<Pt>:PTSLots

Syntax	T1:RX<Pt>:PTSLots <slots>
Description	This command sets the pattern time slots.
Parameters	<Pt> = Port number
	<slots> = <EXPRESSION PROGRAM DATA> Format: Numeric List List consist of slot number(s) ranging from 1 to 23
Response	None.
Examples	`T1:RX1:PTSL (1,3,5)`
Note	

Syntax	T1:RX<Pt>:PTSLots?
Description	This query returns the pattern time slots.
Parameter	<Pt> = Port number
Response	<slots> = <EXPRESSION RESPONSE DATA> Format: Numeric List
Example	`T1:RX1:PTSL? → (1,3,5)`
Note	

6.1.10 T1:RX<Pt>:UP16

Syntax	T1:RX<Pt>:UP16 <pattern>
Description	This command sets the 16 bit user pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Valid characters: '0' and '1' The string must consist of 1 to 16 characters.
Response	None.
Example	`T1:RX1:UP16 "101101"`
Note	This command is for backward compatibility only, and a query command is not available. Actually the new 32 bit user pattern is set by this command.

6.1.11 T1:RX<Pt>:UP32

Syntax	T1:RX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters.
Response	None.
Examples	`T1:RX2:UP32 "01101"`
Note	

Syntax	T1:RX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`T1:RX2:UP32?` → `"01101"`
Note	

6.1.12 T1:TX<Pt>:SCContent

Syntax	T1:TX<Pt>:SCContent <content>
Description	This command sets the channel content.
Parameters	<Pt> = Port number
	<content> = <CHARACTER PROGRAM DATA> OFF TONE TRANsparent SPEech *DEFault=OFF*
Response	None.
Example	`T1:TX1:SCC TONE`
Note	

Syntax	T1:TX<Pt>:SCContent?
Description	This query returns the channel content.
Parameter	<Pt> = Port number
Response	<content> = <CHARACTER RESPONSE DATA>
Example	`T1:TX1:SCC?` → `TONE`
Note	

6.1.13 T1:RX<Pt>:AUDio

Syntax	T1:RX<Pt>:AUDio <type>
Description	This command sets the audio decoding.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> OFF ON *DEFault = OFF*
Response	None.
Example	`T1:RX1:AUD ON`
Note	

Syntax	T1:RX<Pt>:AUDio?
Description	This query returns the audio decoding.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`T1:RX1:AUD?` → `ON`
Note	

6.1.14 T1:RX<Pt>:ATSLot

Syntax	T1:RX<Pt>:ATSLot <timeslotno>
Description	This command sets the audio timeslot to be monitored.
Parameters	<Pt> = Port number
	<timeslotno> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=23, DEFault=1*
Response	None.
Example	`T1:RX1:ATSL 4`
Note	

Syntax	T1:RX<Pt>:ATSLot?
Description	This query returns the monitored audio timeslot.
Parameter	<Pt> = Port number
Response	<timeslotno> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:RX1:ATSL?` → `4`
Note	

6.1.15 T1:RX<Pt>:UP2K

Syntax	T1:RX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`T1:RX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `T1:RX2:PATT` is `USER2048BIT`

Syntax	T1:RX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`T1:RX1:UP2K?` → `"12DF"`
Note	

6.1.16 T1:RX<Pt>:CAS

Syntax	T1:RX<Pt>:CAS <enable>
Description	This command enables or disables the channel associated signaling.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`T1:RX1:CAS ON`
Note	

Syntax	T1:RX<Pt>:CAS?
Description	This query returns the state (enabled/disabled) of the channel associated signaling.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:RX1:CAS?` → `1`
Note	

6.2 Transmitter

6.2.1 T1:TX<Pt>[:ENABled]

Syntax	T1:TX<Pt>[:ENABled] <interface>
Description	This command enables or disables the T1 transmitter.
Parameters	<Pt> = Port number
	<interface> = <CHARACTER PROGRAM DATA> OFF ON *DEFault = OFF*
Response	None.
Examples	`T1:TX1 ON` `T1:TX1:ENAB ON`
Note	

Syntax	T1:TX<Pt>[:ENABled]?
Description	This query returns the state (enabled/disabled) of the T1 transmitter.
Parameter	<Pt> = Port number
Response	<interface> = <CHARACTER RESPONSE DATA>
Examples	`T1:TX1?` → `ON` `T1:TX1:ENAB?` → `SDH`
Note	Returns `SDH` if T1 is over SDH or SONET.

6.2.2 T1:TX<Pt>:FOLLow

Syntax	T1:TX<Pt>:FOLLow <follow>
Description	This command sets the transmitter setting to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<follow> = <CHARACTER PROGRAM DATA> NONE: Do not follow TX1: Follow the setting of the port 1 transmitter *DEFault = NONE*
Response	None.
Example	`T1:TX2:FOLL TX1` (i.e., TX2 follows TX1)
Note	This command is not valid for `:TX1`

Syntax	T1:TX<Pt>:FOLLow?
Description	This query returns if the transmitter follow the TX1 settings.
Parameter	<Pt> = Port number
Response	<follow> = <CHARACTER RESPONSE DATA>
Example	`T1:TX2:FOLL?` → `TX1`
Note	This command is not valid for `:TX1`

6.2.3 T1:TX<Pt>:DINSert[:ENABle]

Syntax	T1:TX<Pt>:DINSert[:ENABle] <enable>
Description	This command enables/disables Drop and Insert.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`T1:TX1:DINS OFF`
Note	

Syntax	T1:TX<Pt>:DINSert[:ENABle]?
Description	This query returns whether or not Drop and Insert is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:TX1:DINS? → 0`
Note	

6.2.4 T1:TX<Pt>:TIMing

Syntax	T1:TX<Pt>:TIMing <source>
Description	This command sets the timing source.
Parameters	<Pt> = Port number
	<source> = <CHARACTER PROGRAM DATA> INTernal: Internal timing source EXTernal: External timing source RX: Received signal on the same port *DEFault = INTernal*
Response	None.
Example	`T1:TX1:TIM INT`
Note	

Syntax	T1:TX<Pt>:TIMing?
Description	This query returns the timing source.
Parameter	<Pt> = Port number
Response	<source> = <CHARACTER RESPONSE DATA>
Example	`T1:TX1:TIM? → INT`
Note	

6.2.5 T1:TX<Pt>:FOFFset

Syntax	T1:TX<Pt>:FOFFset <offset>
Description	This command sets the frequency offset for the clock source. Unit: ppm.
Parameters	<Pt> = Port number <offset> = <NUMERIC PROGRAM DATA> *MINimum=-125, MAXimum=125, DEFault=0* *Allowed suffixes = ppm*
Response	None.
Example	`T1:TX1:FOFF 0ppm`
Note	

Syntax	T1:TX<Pt>:FOFFset?
Description	This query returns the frequency offset (ppm) for the clock source.
Parameter	<Pt> = Port number
Response	<offset> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:TX1:FOFF? → 0`
Note	

6.2.6 T1:TX<Pt>:CODE

Syntax	T1:TX<Pt>:CODE <code>
Description	This command sets the line code type.
Parameters	<Pt> = Port number
	<code> = <CHARACTER PROGRAM DATA> AMI B8ZS *DEFault = B8ZS*
Response	None.
Example	`T1:TX1:CODE AMI`
Note	

Syntax	T1:TX<Pt>:CODE?
Description	This query returns the line code type.
Parameter	<Pt> = Port number
Response	<code> = <CHARACTER RESPONSE DATA>
Example	`T1:TX1:CODE? → AMI`
Note	

6.2.7 T1:TX<Pt>:LBOut

Syntax	T1:TX<Pt>:LBOut <lbo>
Description	This command sets the line build out.
Parameters	<Pt> = Port number
	<lbo> = <CHARACTER PROGRAM DATA> 0TO133: 0 to 133 feet 133TO266: 133 tp 266 feet 266TO399: 266 to 399 feet 399TO533: 399 tp 533 feet 533TO655: 533 to 655 feet GMODe: Gain mode (Monitor) DB0: 0 db DB7: -7.5 db DB15: -15 db DB22: -22.5 db *DEFault = 0TO133*
Response	None.
Example	`T1:TX1:LBO 133TO266`
Note	

Syntax	T1:TX<Pt>:LBOut?
Description	This query returns the line build out setting.
Parameter	<Pt> = Port number
Response	<lbo> = <CHARACTER RESPONSE DATA>
Example	`T1:TX1:LBO? → 0TO133`
Note	

6.2.8 T1:TX<Pt>:PCMFrame

Syntax	T1:TX<Pt>:PCMFrame <enable>
Description	This command enables or disables the PCM frame.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`T1:TX1:PCMF ON`
Note	

Syntax	T1:TX<Pt>:PCMFrame?
Description	This query returns whether or not PCM frame is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:TX1:PCMF?` → `1`
Note	

6.2.9 T1:TX<Pt>:FTYPe

Syntax	T1:TX<Pt>:FTYPe <type>
Description	This command sets the PCM frame type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> TSF: T1 SF/D4 mode TESF: T1 ESF mode JESF: J1 ESF mode *DEFault = TSF*
Response	None.
Example	`T1:TX1:FTYP TESF`
Note	

Syntax	T1:TX<Pt>:FTYPe?
Description	This query returns PCM frame type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`T1:TX1:FTYP?` → `TESF`
Note	

6.2.10 T1:TX<Pt>:PATTern

Syntax	T1:TX<Pt>:PATTern <type>
Description	This command sets the pattern type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> OFF USER16BIT: Obsolete. For backward compatibility only. Same as USER32BIT. USER32BIT: 32 bit user defined pattern. USER2048BIT: 2048 bit user defined pattern. PRBS9 PRBS11 PRBS15 PRBS20 PRBS23 PRBS29 PRBS31 QRSS20 FOX FOXCMA3000 ALL0 ALL1 ALT11: Alternating 1:1 ALT13: Alternating 1:3 ALT17: Alternating 1:7 ALT324: Alternating 3:24 *DEFault = PRBS11*
Response	None.
Example	`T1:TX1:PATT PRBS11`

Syntax	T1:TX<Pt>:PATTern?
Description	This query returns the pattern type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`T1:TX1:PATT?` → `PRBS11`
Note	

6.2.11 T1:TX<Pt>:PINVersion

Syntax	T1:TX<Pt>:PINVersion <inversion>
Description	This command enables or disables pattern inversion.
Parameters	<Pt> = Port number
	<inversion> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`T1:TX1:PINV ON`
Note	The following patterns can be inverted: PRBSxx, QRSSxx, ALT11, ALT13, ALT17, ALT324, USER32BIT and USER2048BIT.

Syntax	T1:TX<Pt>:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the pattern.
Parameter	<Pt> = Port number
Response	<inversion> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:TX1:PINV?` → `1`
Note	

6.2.12 T1:TX<Pt>:PTSLots

Syntax	T1:TX<Pt>:PTSLots <timeslot>
Description	This command sets the pattern time slots.
Parameters	<Pt> = Port number
	<timeslots> = <EXPRESSION PROGRAM DATA> Format: Numeric List List consist of slot number(s) ranging from 1 to 23
Response	None.
Example	`T1:TX1:PTSL (1,3,5)`
Note	

Syntax	T1:TX<Pt>:PTSLots?
Description	This query returns the pattern time slots.
Parameter	<Pt> = Port number
Response	<timeslot> = <EXPRESSION RESPONSE DATA> Format: Numeric List.
Example	`T1:TX1:PTSL?` → `(1,3,5)`
Note	

6.2.13 T1:TX<Pt>:UP16

Syntax	T1:TX<Pt>:UP16 <pattern>
Description	This command sets the 16 bit user pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Valid characters: '0' and '1' The string must consist of 1 to 16 characters.
Response	None.
Example	`T1:TX1:UP16 "101101"`
Note	This command is for backward compatibility only, and a query command is not available. Actually the new 32 bit user pattern is set by this command.

6.2.14 T1:TX<Pt>:UP32

Syntax	T1:TX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters.
Response	None.
Examples	`T1:TX1:UP32 "01101"`
Note	

Syntax	T1:TX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`T1:TX1:UP32?` → `"01101"`
Note	

6.2.15 T1:TX<Pt>:UP2K

Syntax	T1:TX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`T1:TX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `T1:TX2:PATT` is `USER2048BIT`

Syntax	T1:TX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`T1:TX1:UP2K?` → `"12DF"`
Note	

6.2.16 T1:TX<Pt>:UTSLots

Syntax	T1:TX<Pt>:UTSLots <content>
Description	This command sets the content of unused time slots.
Parameters	<Pt> = Port number
	<content> = <NUMERIC PROGRAM DATA> *MINimum=#B00000000, MAXimum=#B11111111, DEFault=#B01010101*
Response	None.
Example	`T1:TX1:UTSL 128`
Note	

Syntax	T1:TX<Pt>:UTSLots?
Description	This query returns the content of unused time slots.
Parameter	<Pt> = Port number <content> = <BINARY NUMERIC RESPONSE DATA>
Response	None.
Example	`T1:TX1:UTSL? → #B10101010`
Note	

6.2.17 T1:TX<Pt>:SCTSlot

Syntax	T1:TX<Pt>:SCTSlot <timeslot>
Description	This command sets the channel time slot.
Parameters	<Pt> = Port number
	<timeslot> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=23, DEFault=1*
Response	None.
Example	`T1:TX1:SCTS 4`
Note	

Syntax	T1:TX<Pt>:SCTSlot?
Description	This query returns the sub channel time slot.
Parameter	<Pt> = Port number
Response	<timeslot> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:TX1:SCTS? → 4`
Note	

6.2.18 T1:TX<Pt>:TFR

Syntax	T1:TX<Pt>:TFR <frequency>
Description	This command sets the tone frequency.
Parameters	<Pt> = Port number
	<frequency> = <NUMERIC PROGRAM DATA> *MINimum = 1 hz, MAXimum = 4000 hz* *DEFault = 440* *Allowed suffixes = HZ, KHZ*
Response	None.
Example	`T1:TX1:TFR 500HZ`
Note	

Syntax	T1:TX<Pt>:TFR?
Description	This query returns the tone frequency (Hz).
Parameter	<Pt> = Port number
Response	<frequency> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:TX1:TFR? → 500`
Note	

6.2.19 T1:TX<Pt>:TLEVel

Syntax	T1:TX<Pt>:TLEVel <level>
Description	This command sets the tone level.
Parameters	<Pt> = Port number
	<level> = <NUMERIC PROGRAM DATA> *MINimum = -70, MAXimum = 3* *DEFault = -20* *Allowed suffixes = dB*
Response	None.
Example	`T1:TX1:TLEV 2dB`
Note	

Syntax	T1:TX<Pt>:TLEVel?
Description	This query returns the tone level.
Parameter	<Pt> = Port number
Response	<level> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:TX1:TLEV? → 2`
Note	

6.2.20 T1:TX<Pt>:CAS

Syntax	T1:TX<Pt>:CAS <enable>
Description	This command enables or disables the channel associated signaling.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`T1:TX1:CAS ON`
Note	

Syntax	T1:TX<Pt>:CAS?
Description	This query returns the state (enabled/disabled) of the channel associated signaling.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:TX1:CAS? → 1`
Note	

6.2.21 T1:TX<Pt>:CASChannel

Syntax	T1:TX<Pt>:CASChannel <channel>
Description	This command sets the CAS channel number.
Parameters	<Pt> = Port number
	<channel> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=24, DEFault=1*
Response	None.
Example	`T1:TX1:CASC 5`
Note	

Syntax	T1:TX<Pt>:CASChannel?
Description	This query returns the CAS channel number.
Parameter	<Pt> = Port number
Response	<channel> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:TX1:CASC? → 5`
Note	

6.2.22 T1:TX<Pt>:CASBits

Syntax	T1:TX<Pt>:CASBits <bits>
Description	This command sets the CAS channel bits.
Parameters	<Pt> = Port number
	<bits> = <NUMERIC PROGRAM DATA> *MINimum=#B0000, MAXimum=#B1111, DEFault=#B1111*
Response	None.
Example	`T1:TX1:CASB #B1111`
Note	

Syntax	T1:TX<Pt>:CASBits?
Description	This query returns the CAS channel bits.
Parameter	<Pt> = Port number
Response	<bits> = <BINARY NUMERIC RESPONSE DATA>
Example	`T1:TX1:CASB?` → `#B1111`
Note	

6.2.23 T1:TX<Pt>:COCBits

Syntax	T1:TX<Pt>:COCBits <bits>
Description	This command sets the CAS other channel bits.
Parameters	<Pt> = Port number
	<bits> = <NUMERIC PROGRAM DATA> *MINimum=#B0000, MAXimum=#B1111, DEFault=#B1001*
Response	None.
Example	`T1:TX1:COCB #B0000`
Note	

Syntax	T1:TX<Pt>:COCBits?
Description	This query returns the CAS other channel bits.
Parameter	<Pt> = Port number
Response	<bits> = <BINARY NUMERIC RESPONSE DATA>
Example	`T1:TX1:COCB?` → `#B0000`
Note	

6.2.24 T1:TX<Pt>:SF:CASBits

Syntax	T1:TX<Pt>:SF:CASBits <bits>
Description	This command sets the CAS channel bits for SF framing.
Parameters	<Pt> = Port number
	<bits> = <NUMERIC PROGRAM DATA> *MINimum=#B00, MAXimum=#B11, DEFault=#B11*
Response	None.
Example	`T1:TX1:SF:CASB #B11`
Note	

Syntax	T1:TX<Pt>:SF:CASBits?
Description	This query returns the CAS channel bits for SF framing.
Parameter	<Pt> = Port number
Response	<bits> = <BINARY NUMERIC RESPONSE DATA>
Example	`T1:TX1:SF:CASB?` → `#B11`
Note	

6.2.25 T1:TX<Pt>:SF:COCBits

Syntax	T1:TX<Pt>:SF:COCBits <bits>
Description	This command sets the CAS other channel bits for SF framing.
Parameters	<Pt> = Port number
	<bits> = <NUMERIC PROGRAM DATA> *MINimum=#B00, MAXimum=#B11, DEFault=#B11*
Response	None.
Example	`T1:TX1:SF:COCB #B00`
Note	

Syntax	T1:TX<Pt>:SF:COCBits?
Description	This query returns the CAS other channel bits for SF framing.
Parameter	<Pt> = Port number
Response	<bits> = <BINARY NUMERIC RESPONSE DATA>
Example	`T1:TX1:SF:COCB?` → `#B00`
Note	

6.3 Stimuli

6.3.1 T1:STIMuli:TX<Pt>:ALARm

Syntax	T1:STIMuli:TX<Pt>:ALARm <type>
Description	This command sets the alarm type to be generated.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> NALarm: No alarm OOF: Out of frame NSIGnal: No signal AIS: Alarm indication signal RAI: Remote Alarm Indication NSYNc: No pattern sync (LSS) *DEFault = NALarm*
Response	None.
Example	`T1:STIM:TX1:ALAR NSIG`
Note	

Syntax	T1:STIMuli:TX<Pt>:ALARm?
Description	This query returns the generated alarm type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`T1:STIM:TX1:ALAR?` → `NSIG`
Note	

6.3.2 T1:STIMuli:TX<Pt>:ERRor

Syntax	T1:STIMuli:TX<Pt>:ERRor <destination>
Description	This command sets the error destination.
Parameters	<Pt> = Port number
	<destination> = <CHARACTER PROGRAM DATA> OFF CRC6: CRC-6 PATTern: Pattern Error FBIT: F-Bit SBIT: S-Bit BPV: Bipolar Violation PSLip: Pattern slip EXZ: Excessive Zeroes *DEFault = OFF*
Response	None.
Example	`T1:STIM:TX1:ERR CODE`
Note	

Syntax	T1:STIMuli:TX<Pt>:ERRor?
Description	This query returns the error destination.
Parameter	<Pt> = Port number
Response	<destination> = <CHARACTER RESPONSE DATA>
Example	`T1:STIM:TX1:ERR? → CODE`
Note	

6.3.3 T1:STIMuli:TX<Pt>:EINSert

Syntax	T1:STIMuli:TX<Pt>:EINSert <insertion>
Description	This command sets the method to insert errors.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> OFF MANual B02: Burst · 1E-02 B03: Burst · 1E-03 B04: Burst · 1E-04 B05: Burst · 1E-05 B06: Burst · 1E-06 B07: Burst · 1E-07 ES: Errored seconds SES: Severe errored seconds *DEFault = OFF*
Response	None.
Example	`T1:STIM:TX1:EINS MAN`
Note	If insertion is set to MANual, errors are inserted with SYST:STIM:INS. See section 2.3.23

Syntax	T1:STIMuli:TX<Pt>:EINSert?
Description	This query returns the error insertion method.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`T1:STIM:TX1:EINS? → MAN`
Note	

6.3.4 T1:STIMuli:TX<Pt>:EBLength

Syntax	T1:STIMuli:TX<Pt>:EBLength <burstlength>
Description	This command sets the error burst length to generate.
Parameters	<Pt> = Port number
	<burstlength> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=255, DEFault=1*
Response	None.
Example	`T1:STIM:TX1:EBL 1`
Note	For T1:STIM:TX<Pt>:ERR? → CODE, *MAXimum = 1*

Syntax	T1:STIMuli:TX<Pt>:EBLength?
Description	This query returns the error burst length to generate.
Parameter	<Pt> = Port number
Response	<burstlength> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:STIM:TX1:EBL? → 1`
Note	

6.3.5 T1:STIMuli:TX<Pt>:FDLink

Syntax	T1:STIMuli:TX<Pt>:FDLink <code>
Description	This command sets the transmitted FDL- or in-bound code.
Parameters	<Pt> = Port number
	<code> = <CHARACTER PROGRAM DATA> OFF: LLA: LLD: PLA: PLD: ULB: NLA: USER: Use the user defined FDL value ACS: DCS: AN1: DN1: AN2: DN2: 100K: UINBand: Use the user defined in-band coed. *DEFault = OFF*
Response	None.
Example	`T1:STIM:TX1:FDL LLA`
Note	

Syntax	T1:STIMuli:TX<Pt>:FDLink?
Description	This query returns the transmitted FDL- or inbound code.
Parameter	<Pt> = Port number
Response	<code> = <CHARACTER RESPONSE DATA>
Example	`T1:STIM:TX1:FDL? → LLA`
Note	

6.3.6 T1:STIMuli:TX<Pt>:FDLink:UFDL

Syntax	T1:STIMuli:TX<Pt>:FDLink:UFDL <value>
Description	This command sets the user defined FDL code.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=63, DEFault=0*
Response	None.
Example	`T1:STIM:TX1:FDL:UFDL #B111111`
Note	

Syntax	T1:STIMuli:TX<Pt>:FDLink:UFDL?
Description	This query returns the user defined FDL code.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:STIM:TX1:FDL:UFDL? → 63`
Note	

6.3.7 T1:STIMuli:TX<Pt>:FDLink:IBCode

Syntax	T1:STIMuli:TX<Pt>:FDLink:IBCode <value>
Description	This command sets the user defined in-bound code.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=16777215, DEFault=0*
Response	None.
Example	`T1:STIM:TX1:FDL:IBC #HFFFFFE`
Note	

Syntax	T1:STIMuli:TX<Pt>:FDLink:IBCode?
Description	This query returns the user defined in-bound code.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:STIM:TX1:FDL:IBC? → 16777214`
Note	

6.4 Result

6.4.1 T1:RX<Pt>:IFETch?

Syntax	T1:RX<Pt>:IFETch? <parameter>
Description	This command fetches results from a T1 interval if available.
Parameters	<Pt> = Port number
	({<parameter>} + {,}*) = <EXPRESSION PROGRAM DATA> The response format is listed for each parameter. **Alarms** NSIGnal: No signal. Response: <Seconds>,<Ratio> AIS: Alarm indication Signal. Response: <Seconds>,<Ratio> OOF: Out of frame. Response: <Seconds>,<Ratio> NSYNc: No sync (LSS). Response: <Seconds>,<Ratio> RAI: Remote Alarm Indication. Response: <Seconds>,<Ratio> **Errors** PATTern: Pattern. Response: <Count>,<Ratio> PSLip: Pattern slip. Response: <Count>,<Ratio> PBLock: Pattern block. Response: <Count>,<Ratio> CODE: Bipolar Violation (BPV). Response: <Count>,<Ratio> FBIT: F-bit. Response: <Count>,<Ratio> SBIT: S-bit. Response: <Count>,<Ratio> CRC6: CRC-6. Response: <Count>,<Ratio> EZERro: Excessive Zeroes. Response: <Count>,<Ratio> **FAS performance errors** FES: FAS ES. Response: <Count>,<Ratio%> FSES: FAS SES. Response: <Count>,<Ratio%> FBBE: FAS BBE. Response: <Count>,<Ratio%> FALS: FAS ALS. Response: <Count>,<Ratio%> FUAT: FAS UAT. Response: <Count>,<Ratio%> FAVT: FAS AVT. Response: <Count>,<Ratio%> FEFS: FAS EFS. Response: <Count>,<Ratio%> **Pattern performance errors** PES: Pattern ES. Response: <Count>,<Ratio%> PSES: Pattern SES. Response: <Count>,<Ratio%> PBBE: Pattern BBE. Response: <Count>,<Ratio%> PALS: Pattern ALS. Response: <Count>,<Ratio%> PUAT: Pattern UAT. Response: <Count>,<Ratio%> PAVT: Pattern AVT. Response: <Count>,<Ratio%> PEFS: Pattern EFS. Response: <Count>,<Ratio%> **CRC6 performance errors** CES: CRC6 ES. Response: <Count>,<Ratio%> CSES: CRC6 SES. Response: <Count>,<Ratio%> CBBE: CRC6 BBE. Response: <Count>,<Ratio%> CALS: CRC6 ALS. Response: <Count>,<Ratio%> CUAT: CRC6 UAT. Response: <Count>,<Ratio%> CAVT: CRC6 AVT. Response: <Count>,<Ratio%> CEFS: CRC6 EFS. Response: <Count>,<Ratio%>
Response	{(<result>),}* = <EXPRESSION RESPONSE DATA> Expression format: Numeric List Each result is formatted according to the specification in the parameter field. Values that are not relevant or applicable for the current measurement, returns NaN (section 1.6.1).
Example	`T1:RX1:IFET? (AIS,PATT)` → `(9,0.0429),(10,0.0343)`
Note	This command fetches the results from the interval selected using the `MEASurement:SETup:SELect` command (see section 18.2.2). If the requested result is not available, NaN (section 1.6.1) is returned. If there is one or more results, the last "," is always removed.

6.5 Status

6.5.1 T1:STATus:RX<Pt>:AESummary[:EVENt]?

Syntax	T1:STATus:RX<Pt>:AESummary[:EVENt]?
Description	This query returns alarms and errors summary event register. The content of this event register is summarized in DB7 of the STATus:INTerface:PORT<Pt>:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 (2) = Error summary DB3 - DB16 = NOT USED
Example	`T1:STAT:RX1:AES?` → `1` `T1:STAT:RX1:AES:EVEN?` → `1`
Note	

6.5.2 T1:STATus:RX<Pt>:AESummary:CONDition?

Syntax	T1:STATus:RX<Pt>:AESummary:CONDition?
Description	This query returns alarms and errors summary operation register query. DB1 is summary of T1:STATus:RX<Pt>:ALARm[:EVENt]? DB2 is summary of T1:STATus:RX<Pt>:ERRor[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 (2) = Error summary DB3 - DB16 = NOT USED
Example	`T1:STAT:RX1:AES:COND?` → `1`
Note	

6.5.3 T1:STATus:RX<Pt>:ALARm[:EVENt]?

Syntax	T1:STATus:RX<Pt>:ALARm[:EVENt]?
Description	This query returns alarms event register query. The content of this register is summarized in DB1 of the T1:STATus:RX<Pt>:AESummary:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = No sync (LSS) DB2 (2) = NOT USED DB3 (4) = CRC6err DB4 (8) = Remote Alarm Indication (RAI) DB5 (16) = NOT USED DB6 (32) = Out of frame (OOF) DB7 (64) = Alarm indication signal (AIS) DB8 (128) = No signal DB9 - DB16 = NOT USED
Example	`T1:STAT:RX1:ALAR?` → `64`
Note	

6.5.4 T1:STATus:RX<Pt>:ALARm:CONDition?

Syntax	T1:STATus:RX<Pt>:ALARm:CONDition?
Description	This query returns alarms condition register query.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = No sync (LSS) DB2 (2) = NOT USED DB3 (4) = CRC6err DB4 (8) = Remote Alarm Indication (RAI) DB5 (16) = NOT USED DB6 (32) = Out of frame (OOF) DB7 (64) = Alarm indication signal (AIS) DB8 (128) = No signal DB9 - DB16 = NOT USED
Example	`T1:STAT:RX1:ALAR:COND?` → `64`
Note	

6.5.5 T1:STATus:RX<Pt>:ERRor[:EVENt]?

Syntax	T1:STATus:RX<Pt>:ERRor[:EVENt]?
Description	This query returns errors event register. The content of this register is summarized in DB2 of the T1:STATus:RX<Pt>:AESummary:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Pattern slip DB2 (2) = Pattern DB3 (4) = S-Bit DB4 (8) = CRC-6 DB5 (16) = NOT USED DB6 (32) = Excessive Zeroes (EXZ) DB7 (64) = F-Bit DB8 (128) = Bipolar Violation (BPV) DB9 - DB16 = NOT USED
Example	`T1:STAT:RX1:ERR?` → `64`
Note	

6.5.6 T1:STATus:RX<Pt>:ERRor:CONDition?

Syntax	T1:STATus:RX<Pt>:ERRor:CONDition?
Description	This query returns errors condition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Pattern slip DB2 (2) = Pattern DB3 (4) = S-Bit DB4 (8) = CRC-6 DB5 (16) = NOT USED DB6 (32) = Excessive Zeroes (EXZ) DB7 (64) = F-Bit DB8 (128) = Bipolar Violation (BPV) DB9 - DB16 = NOT USED
Example	`T1:STAT:RX1:ERR:COND?` → `64`
Note	

6.5.7 T1:STATus:RX<Pt>:PSLevel?

Syntax	T1:STATus:RX<Pt>:PSLevel?
Description	This query returns physical signal level. Unit: dB.
Parameter	<Pt> = Port number
Response	<signallevel> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:PSL?` → `0`
Note	Minimum level is -11 dB.

6.5.8 T1:STATus:RX<Pt>:PDEViation?

Syntax	T1:STATus:RX<Pt>:PDEViation?
Description	This query returns physical deviation. Units: ppm and bps.
Parameter	<Pt> = Port number
Response	<ppm> = <NR1 NUMERIC RESPONSE DATA>
	<bps> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:PDEV?` → `0, 0`
Note	

6.5.9 T1:STATus:RX<Pt>:PBRate?

Syntax	T1:STATus:RX<Pt>:PBRate?
Description	This query returns physical bit rate. Unit: bps.
Parameter	<Pt> = Port number
Response	<bitrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:PBR?` → `1544000`
Note	

6.5.10 T1:STATus:RX<Pt>:FDUMp?

Syntax	T1:STATus:RX<Pt>:FDUMp?
Description	This query returns a sample of a full DS1 Frame (24 bytes).
Parameters	<Pt> = Port number
Response	<hex> = <EXPRESSION RESPONSE DATA> Expression format: Numeric list Contains all 24 Frame timeslots.
Example	`T1:STAT:RX1:FDUM?` → `#H55,#H55,#H55,...`
Note	

6.5.11 T1:STATus:RX<Pt>:ACONtent?

Syntax	T1:STATus:RX<Pt>:ACONtent?
Description	This query returns audio channel content.
Parameter	<Pt> = Port number
Response	<audiocontent> = <BINARY NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:ACON?` →`#B01010101`
Note	For sub-rates this field shows the content of the sub channel.

6.5.12 T1:STATus:RX<Pt>:AIBContent?

Syntax	T1:STATus:RX<Pt>:AIBContent?
Description	This query returns sub channel audio all bit inverting content.
Parameter	<Pt> = Port number
Response	<audiocontent> = <BINARY NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:AIBC?` → `#B10000000`
Note	With u-law coded speech, it is possible to observe the u-law code words before the bit inverting.

6.5.13 T1:STATus:RX<Pt>:APPeak?

Syntax	T1:STATus:RX<Pt>:APPeak?
Description	This query returns sub channel audio positive peak value.
Parameter	<Pt> = Port number
Response	<positivepeak> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:APP?` → `85`
Note	This query only applies for u-law speech and shows the largest received u-law coded value.

6.5.14 T1:STATus:RX<Pt>:ANPeak?

Syntax	T1:STATus:RX<Pt>:ANPeak?
Description	This query returns sub channel audio negative peak value.
Parameter	<Pt> = Port number
Response	<negativepeak> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:ANP?` → `-85`
Note	This query only applies for u-law speech and shows the smallest received u-law coded value.

6.5.15 T1:STATus:RX<Pt>:ALEVel?

Syntax	T1:STATus:RX<Pt>:ALEVel?
Description	This query returns sub channel audio level. Unit: dBm.
Parameter	<Pt> = Port number
Response	<audiolevel> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:ALEV?` → `-4`
Note	

6.5.16 T1:STATus:RX<Pt>:AFRequency?

Syntax	T1:STATus:RX<Pt>:AFRequency?
Description	This query returns sub channel audio tone frequency. Unit: Hz.
Parameter	<Pt> = Port number
Response	<audiofrequency> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:AFR?` → `366`
Note	

6.5.17 T1:STATus:RX<Pt>:ACOFfset?

Syntax	T1:STATus:RX<Pt>:ACOFfset?
Description	This query returns sub channel audio coder offset.
Parameter	<Pt> = Port number
Response	<coderoffset> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:ACOF?` → `0`
Note	Only available for 64 kbps audio channels.

6.5.18 T1:STATus:RX<Pt>:PPBRate?

Syntax	T1:STATus:RX<Pt>:PPBRate?
Description	This query returns payload pattern bit rate. Unit: bps.
Parameter	<Pt> = Port number
Response	<bitrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:PPBR?` → `64000`
Note	

6.5.19 T1:STATus:RX<Pt>:FBIT?

Syntax	T1:STATus:RX<Pt>:FBIT?
Description	This query returns FAS bits. (Frame Alignment Signal bits). Data is shown as binary.
Parameter	<Pt> = Port number
Response	<bits> = <BINARY NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:FBIT?` → `#B101010`
Note	

6.5.20 T1:STATus:RX<Pt>:SBIT?

Syntax	T1:STATus:RX<Pt>:SBIT?
Description	This query returns S-Bits (Multi Frame Alignment Signal bits). Data is shown as binary.
Parameter	<Pt> = Port number
Response	<bits> = <BINARY NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:SBIT?` → `#B001110`
Note	

6.5.21 T1:STATus:RX<Pt>:MBIT?

Syntax	T1:STATus:RX<Pt>:MBIT?
Description	This query returns M-Bits (Data Link Message bits). Data is shown as binary.
Parameter	<Pt> = Port number
Response	<bits> = <BINARY NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:MBIT?` → `#B0000000000000000`
Note	

6.5.22 T1:STATus:RX<Pt>:FDL:MODE

Syntax	T1:STATus:RX<Pt>:FDL:MODE <interface>
Description	This command selects the FDL mode. This can be MBIT or INBand.
Parameters	<Pt> = Port number
	<interface> = <CHARACTER PROGRAM DATA> MBIT INBand *DEFault = MBIT*
Response	None.
Example	`T1:STAT:RX1:FDL:MODE INB`
Note	

Syntax	T1:STATus:RX<Pt>:FDL:MODE?
Description	This query returns the state of the FDL mode.
Parameter	<Pt> = Port number
Response	<interface> = <CHARACTER RESPONSE DATA>
Example	`T1:STAT:RX1:FDL:MODE?` → `MBIT`
Note	

6.5.23 T1:STATus:RX<Pt>:FDL:MBIT:CODE?

Syntax	T1:STATus:RX<Pt>:FDL:MBIT:CODE?
Description	This query returns MBIT FDL Code (Data Link Message mnemonic).
Parameter	<Pt> = Port number
Response	<bits> = <BINARY NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:FDL:MBIT:CODE?` → `PLA`
Note	

6.5.24 T1:STATus:RX<Pt>:FDL:INBand:CODE?

Syntax	T1:STATus:RX<Pt>:FDL:INBand:CODE?
Description	This query returns Inband FDL Code (Data Link Message mnemonic).
Parameter	<Pt> = Port number
Response	<bits> = <BINARY NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:FDL:INB:CODE?` → `AN1`
Note	

6.5.25 T1:STATus:RX<Pt>:FDL:UDEFined

Syntax	T1:STATus:RX<Pt>:FDL:UDEFined <enable>
Description	This command enables or disables the User Defined check mark.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`T1:STAT:RX1:FDL:UDEF 1`
Note	

Syntax	T1:STATus:RX<Pt>:FDL:UDEFined?
Description	This query returns the state of the User Defined check mark.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`T1:STAT:RX1:FDL:UDEF?` → `1`
Note	

6.5.26 T1:STATus:RX<Pt>:FDL:UCODe

Syntax	T1:STATus:RX<Pt>:FDL:UCODe <max>
Description	This command sets the FDL User Code.
Parameters	<Pt> = Port number
	<max> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 16777215, DEFault = 0*
Response	None.
Example	`T1:STAT:RX1:FDL:UCOD 123456`
Note	

Syntax	T1:STATus:RX<Pt>:FDL:UCODe?
Description	This query returns the FDL User Code.
Parameter	<Pt> = Port number
Response	<max> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:STAT:RX1:FDL:UCOD?` → `123456`
Note	

6.5.27 T1:STATus:RX<Pt>:FDL:TRIGger

Syntax	T1:STATus:RX<Pt>:FDL:TRIGger <interface>
Description	This command enables or disables the FDL trigger.
Parameters	<Pt> = Port number
	<interface> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`T1:STAT:RX1:FDL:TRIG ON`
Note	

Syntax	T1:STATus:RX<Pt>:FDL:TRIGger?
Description	This query returns the state of the FDL trigger.
Parameter	<Pt> = Port number
Response	<interface> = <BOOLEAN RESPONSE DATA>
Example	`T1:STAT:RX1:FDL:TRIG?` → `1`
Note	

6.5.28 T1:STATus:RX<Pt>:CBITs?

Syntax	T1:STATus:RX<Pt>:CBITs?
Description	This query returns CAS bits.
Parameter	<Pt> = Port number
Response	<bits> = <EXPRESSION RESPONSE DATA> Expression format: Numeric list Contains all 24 CAS bit sets.
Example	`T1:STAT:RX1:CBIT?` → `(#B0101,#B0101, ..., #B0101,#B0101)`
Note	

6.6 APS

6.6.1 T1:APS:STARt

Syntax	T1:APS:STARt
Description	This command starts the APS (Automatic Protection Switching) measurement.
Parameter	None.
Response	None.
Example	`T1:APS:STAR`
Note	

6.6.2 T1:APS:STOP

Syntax	T1:APS:STOP
Description	This command stops the APS (Automatic Protection Switching) measurement command.
Parameter	None.
Response	None.
Example	`T1:APS:STOP`
Note	

6.6.3 T1:APS:RX<Pt>:NUMBer?

Syntax	T1:APS:RX<Pt>:NUMBer?
Description	This query returns the number of times has measured.
Parameter	<Pt> = Port number
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:APS:RX1:NUMB?` → `17`
Note	

6.6.4 T1:APS:RX<Pt>:ATIMe?

Syntax	T1:APS:RX<Pt>:ATIMe?
Description	This query returns the average time has measured. Unit: ms.
Parameter	<Pt> = Port number
Response	<avg> = <NR2 NUMERIC RESPONSE DATA>
Example	`T1:APS:RX1:ATIM?` → `4.000`
Note	The maximum measurable time is 4000 ms. The maximum measurable time will be responded if the result exceeds 4000 ms.

6.6.5 T1:APS:RX<Pt>:MTIMe?

Syntax	T1:APS:RX<Pt>:MTIMe?
Description	This query returns the maximum time has measured. Unit: ms.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
	<limit exceeded> = <NR1 NUMERIC RESPONSE DATA> Returns 1, if the reference maximum limit has been exceeded.
Example	`T1:APS:RX1:MTIM?` → `29.170,0`
Note	The maximum measurable time is 4000 ms. The maximum measurable time will be responded if the result exceeds 4000 ms.

6.6.6 T1:APS:RX<Pt>:LTIMe?

Syntax	T1:APS:RX<Pt>:LTIMe?
Description	This query returns the minimum time has measured. Unit: ms.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`T1:APS:RX1:LTIM? → 1.0`
Note	The maximum measurable time is 4000 ms. The maximum measurable time will be responded if the result exceeds 4000 ms.

6.6.7 T1:APS:RX<Pt>:EVENt

Syntax	T1:APS:RX<Pt>:EVENt <event>
Description	This command sets the Time Reference event.
Parameters	<Pt> = Port number
	<event> = <CHARACTER PROGRAM DATA> AIS = Alarm indication signal OOF = Out of frame PERRor = Pattern error
Response	None.
Example	`T1:APS:RX1:EVEN AIS`
Note	

Syntax	T1:APS:RX<Pt>:EVENt?
Description	This query returns the time reference event.
Parameter	<Pt> = Port number
Response	<event> = <CHARACTER RESPONSE DATA>
Example	`T1:APS:RX1:EVEN? → AIS`
Note	

6.6.8 T1:APS:RX<Pt>:MLIMit

Syntax	T1:APS:RX<Pt>:MLIMit <max>
Description	This command sets the time reference maximum limit. Unit: ms.
Parameters	<Pt> = Port number
	<max> = <NUMERIC PROGRAM DATA> *MINimum = 0.000, MAXimum = 4000.000, DEFault = 50.000*
Response	None.
Example	`T1:APS:RX1:MLIM 50.000`
Note	

Syntax	T1:APS:RX<Pt>:MLIMit?
Description	This query returns the time reference maximum limit. Unit: ms.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
Example	`T1:APS:RX1:MLIM? → 50.000`
Note	

6.7 RTD

This section document commands to retrieve Round Trip Delay measurement results.
Commands for general RTD settings are described in section 17.1 on page 953.

6.7.1 T1:RTD:RX<Pt>:NUMBer?

Syntax	T1:RTD:RX<Pt>:NUMBer?
Description	This query returns the number of the RTD data.
Parameter	<Pt> = Port number
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`T1:RTD:RX1:NUMB?` → `2`
Note	

6.7.2 T1:RTD:RX<Pt>:ATIMe?

Syntax	T1:RTD:RX<Pt>:ATIMe?
Description	This query returns the average time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<avg> = <NR2 NUMERIC RESPONSE DATA>
Example	`T1:RTD:RX1:ATIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

6.7.3 T1:RTD:RX<Pt>:MTIMe?

Syntax	T1:RTD:RX<Pt>:MTIMe?
Description	This query returns the maximum time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
	<limit exceeded> = <NR1 NUMERIC RESPONSE DATA> Returns 1, if the reference maximum limit has been exceeded.
Example	`T1:RTD:RX1:MTIM?` → `1.0,0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

6.7.4 T1:RTD:RX<Pt>:LTIMe?

Syntax	T1:RTD:RX<Pt>:LTIMe?
Description	This query returns the least (minimum) time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`T1:RTD:RX1:LTIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

Chapter 7

E3

7.1 Receiver

7.1.1 E3:RX<Pt>[:ENABled]

Syntax	E3:RX<Pt>[:ENABled] <state>
Description	This command enables/disables the receiver.
Parameters	<Pt> = Port number
	<state> = <CHARACTER PROGRAM DATA> OFF ON *DEFault = OFF*
Response	None.
Examples	`E3:RX1 ON` `E3:RX1:ENAB ON`
Note	

Syntax	E3:RX<Pt>[:ENABled]?
Description	This query returns the state (enabled/disabled) of the receiver.
Parameter	<Pt> = Port number
Response	<state> = <CHARACTER RESPONSE DATA>
Examples	`E3:RX1?` → `ON` `E3:RX1:ENAB?` → `SDH`
Note	Returns `SDH` if E3 is over SDH or SONET.

7.1.2 E3:RX<Pt>:MODE

Syntax	E3:RX<Pt>:MODE <mode>
Description	This command sets the signal termination mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> TERMinate: Nominal impedance. Normal frequency dependent AGC. MONitor: Nominal impedance. Frequency linear AGC. *DEFault = TERMinate*
Response	None.
Example	`E3:RX1:MODE TERM`
Note	

Syntax	E3:RX<Pt>:MODE?
Description	This query returns the signal termination mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`E3:RX1:MODE?` → `TERM`
Note	

7.1.3 E3:RX<Pt>:FOLLow

Syntax	E3:RX<Pt>:FOLLow <follow>
Description	This command sets the receiver to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<follow> = <CHARACTER PROGRAM DATA> NONE: Do not follow TX: Transmitter of the same port RX1: Receiver of port 1 *DEFault = NONE*
Response	None.
Example	`E3:RX1:FOLL TX`
Note	

Syntax	E3:RX<Pt>:FOLLow?
Description	This query returns if the receiver follow another setup.
Parameter	<Pt> = Port number
Response	<follow> = <CHARACTER RESPONSE DATA>
Example	`E3:RX1:FOLL?` → `TX`
Note	

7.1.4 E3:RX<Pt>:PCMFrame

Syntax	E3:RX<Pt>:PCMFrame <enable>
Description	This command enables or disables the PCM frame.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`E3:RX1:PCMF ON`
Note	

Syntax	E3:RX<Pt>:PCMFrame?
Description	This query returns whether or not PCM frame is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`E3:RX1:PCMF?` → `1`
Note	

7.1.5 E3:RX<Pt>:PATTern

Syntax	E3:RX<Pt>:PATTern <type>
Description	This command sets the pattern type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> OFF USER16BIT: Obsolete. For backward compatibility only. Same as USER32BIT. USER32BIT: 32 bit user defined pattern. USER2048BIT: 2048 bit user defined pattern. PRBS9 PRBS11 PRBS15 PRBS20 PRBS23 PRBS31 FOX FOXCMA3000 ALL0 ALL1 ALT11: Alternating 1:1 ALT13: Alternating 1:3 ALT17: Alternating 1:7 ALT324: Alternating 3:24 *DEFault = PRBS23*
Response	None.
Example	`E3:RX1:PATT PRBS11`
Note	

Syntax	E3:RX<Pt>:PATTern?
Description	This query returns the pattern type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`E3:RX1:PATT? → PRBS11`
Note	

7.1.6 E3:RX<Pt>:PINVersion

Syntax	E3:RX<Pt>:PINVersion <inverted>
Description	This command enables or disables pattern inversion.
Parameters	<Pt> = Port number
	<inverted> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`E3:RX1:PINV ON`
Note	The following patterns can be inverted: PRBSxx, FOX, ALL1, ALL0, ALT11, ALT13, ALT17, ALT324, UP16 and UP2K.

Syntax	E3:RX<Pt>:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the pattern.
Parameter	<Pt> = Port number
Response	<inverted> = <NR1 NUMERIC RESPONSE DATA>
Example	`E3:RX1:PINV? → 1`
Note	

7.1.7 E3:RX<Pt>:UP16

Syntax	E3:RX<Pt>:UP16 <pattern>
Description	This command sets 16 bit user pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Valid characters: '0' and '1' The string must consist of 1 to 16 characters.
Response	None.
Example	`E3:RX1:UP16 "101101"`
Note	This command is for backward compatibility only, and a query command is not available. Actually the new 32 bit user pattern is set by this command.

7.1.8 E3:RX<Pt>:UP32

Syntax	E3:RX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters (one bit resolution).
Response	None.
Example	`E3:RX2:UP32 "01101"`
Note	

Syntax	E3:RX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`E3:RX2:UP32?` → `"01101"`
Note	

7.1.9 E3:RX<Pt>:UP2K

Syntax	E3:RX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`E3:RX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `E3:RX2:PATT` is `USER2048BIT`

Syntax	E3:RX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`E3:RX1:UP2K?` → `"12DF"`
Note	

7.2 Transmitter

7.2.1 E3:TX<Pt>[:ENABled]

Syntax	E3:TX<Pt>[:ENABled] <state>
Description	This command enables/disables the transmitter.
Parameters	<Pt> = Port number
	<state> = <CHARACTER PROGRAM DATA> OFF ON *DEFault = OFF*
Response	None.
Examples	`E3:TX1 ON` `E3:TX1:ENAB ON`
Note	The ENABled command is the default node for E3:TX<Pt>.

Syntax	E3:TX<Pt>[:ENABled]?
Description	This query returns the state (enabled/disabled) of the transmitter.
Parameter	<Pt> = Port number
Response	<state> = <CHARACTER RESPONSE DATA>
Examples	`E3:TX1?` → `ON` `E3:TX1:ENAB?` → `SDH`
Note	Returns `SDH` if E3 is over SDH or SONET.

7.2.2 E3:TX<Pt>:FOLLow

Syntax	E3:TX<Pt>:FOLLow <follow>
Description	This command sets the transmitter setting to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<follow> = <CHARACTER PROGRAM DATA> NONE: Do not follow TX1: Follow the setting of the port 1 transmitter *DEFault = NONE*
Response	None.
Example	`E3:TX2:FOLL TX1`
Note	This command is not valid for `:TX1`

Syntax	E3:TX<Pt>:FOLLow?
Description	This query returns if the transmitter follow the TX1 settings.
Parameter	<Pt> = Port number
Response	<follow> = <CHARACTER RESPONSE DATA>
Example	`E3:TX2:FOLL?` → `TX1`
Note	This command is not valid for `:TX1`

7.2.3 E3:TX<Pt>:TIMing

Syntax	E3:TX<Pt>:TIMing <source>
Description	This command sets the timing source.
Parameters	<Pt> = Port number
	<source> = <CHARACTER PROGRAM DATA> INTernal: Internal timing source EXTernal: External timing source RX: Received signal on the same port *DEFault = INTernal*
Response	None.
Example	`E3:TX1:TIM INT`
Note	

Syntax	E3:TX<Pt>:TIMing?
Description	This query returns the timing source.
Parameter	<Pt> = Port number
Response	<source> = <CHARACTER RESPONSE DATA>
Example	`E3:TX1:TIM?` → `INT`
Note	

7.2.4 E3:TX<Pt>:FOFFset

Syntax	E3:TX<Pt>:FOFFset <offset>
Description	This command sets the frequency offset for the clock source. Unit: ppm.
Parameters	<Pt> = Port number
	<offset> = <NUMERIC PROGRAM DATA> *MINimum=-125, MAXimum=125, DEFault=0* *Allowed suffix = ppm*
Response	None.
Example	`E3:TX1:FOFF -25ppm`
Note	

Syntax	E3:TX<Pt>:FOFFset?
Description	This query returns the frequency offset for the clock source. Unit: ppm.
Parameter	<Pt> = Port number
Response	<offset> = <NR1 NUMERIC RESPONSE DATA>
Example	`E3:TX1:FOFF?` → `-25`
Note	

7.2.5 E3:TX<Pt>:PCMFrame

Syntax	E3:TX<Pt>:PCMFrame <enable>
Description	This command enables or disables the PCM frame.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`E3:TX1:PCMF ON`
Note	

Syntax	E3:TX<Pt>:PCMFrame?
Description	This query returns whether or not PCM frame is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`E3:TX1:PCMF?` → `1`
Note	

7.2.6 E3:TX<Pt>:PATTern

Syntax	E3:TX<Pt>:PATTern <type>
Description	This command sets the pattern type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> OFF USER16BIT: Obsolete. For backward compatibility only. Same as USER32BIT. USER32BIT: 32 bit user defined pattern. USER2048BIT: 2048 bit user defined pattern. PRBS9 PRBS11 PRBS15 PRBS20 PRBS23 PRBS31 FOX FOXCMA3000 ALL0 ALL1 ALT11: Alternating 1:1 ALT13: Alternating 1:3 ALT17: Alternating 1:7 ALT324: Alternating 3:24 *DEFault = PRBS23*
Response	None.
Example	`E3:TX1:PATT PRBS11`
Note	

Syntax	E3:TX<Pt>:PATTern?
Description	This query returns the pattern type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`E3:TX1:PATT?` → `PRBS11`
Note	

7.2.7 E3:TX<Pt>:PINVersion

Syntax	E3:TX<Pt>:PINVersion <inverted>
Description	This command sets pattern inversion.
Parameters	<Pt> = Port number
	<inverted> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`E3:TX1:PINV ON`
Note	

Syntax	E3:TX<Pt>:PINVersion?
Description	This query returns pattern inversion.
Parameter	<Pt> = Port number
Response	<inverted> = <NR1 NUMERIC RESPONSE DATA>
Example	`E3:TX1:PINV?` → `1`
Note	

7.2.8 E3:TX<Pt>:UP16

Syntax	E3:TX<Pt>:UP16 <pattern>
Description	This command sets 16 bit user pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Valid characters: '0' and '1' The string must consist of 1 to 16 characters.
Response	None.
Example	`E3:TX1:UP16 "101101"`
Note	This command is for backward compatibility only, and a query command is not available. Actually the new 32 bit user pattern is set by this command.

7.2.9 E3:TX<Pt>:UP32

Syntax	E3:TX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters (one bit resolution).
Response	None.
Example	`E3:TX1:UP32 "01101"`
Note	

Syntax	E3:TX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`E3:TX1:UP32?` → `"01101"`
Note	

7.2.10 E3:TX<Pt>:UP2K

Syntax	E3:TX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`E3:TX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `E3:TX2:PATT` is `USER2048BIT`

Syntax	E3:TX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`E3:TX1:UP2K?` → `"12DF"`
Note	

7.3 Stimuli

7.3.1 E3:STIMuli:TX<Pt>:ALARm

Syntax	E3:STIMuli:TX<Pt>:ALARm <alarmtype>
Description	This command sets the alarm type to generate.
Parameters	<Pt> = Port number
	<alarmtype> = <CHARACTER PROGRAM DATA> NALarm: No alarm NSIGnal: No signal AIS: Alarm Indication Signal NFRame: No frame DALarm: Distant alarm (RDI) NSYNc: No pattern sync *DEFault = NALarm*
Response	None.
Example	`E3:STIM:TX1:ALAR NALarm`
Note	

Syntax	E3:STIMuli:TX<Pt>:ALARm?
Description	This query returns the stimuli alarm type.
Parameter	<Pt> = Port number
Response	<alarmtype> = <CHARACTER RESPONSE DATA>
Example	`E3:STIM:TX1:ALAR?` → `NSIG`
Note	

7.3.2 E3:STIMuli:TX<Pt>:ERRor

Syntax	E3:STIMuli:TX<Pt>:ERRor <errordestination>
Description	This command sets the error destination.
Parameters	<Pt> = Port number
	<errordestination> = <CHARACTER PROGRAM DATA> FRAMe: Frame CODE: Code PATTern: Pattern PSLip: Pattern slip *DEFault = FRAMe*
Response	None.
Example	`E3:STIM:TX1:ERR CODE`
Note	

Syntax	E3:STIMuli:TX<Pt>:ERRor?
Description	This query returns the error destination.
Parameter	<Pt> = Port number
Response	<errordestination> = <CHARACTER RESPONSE DATA>
Example	`E3:STIM:TX1:ERR?` → `CODE`
Note	

7.3.3 E3:STIMuli:TX<Pt>:EINSert

Syntax	E3:STIMuli:TX<Pt>:EINSert <insertion>
Description	This command sets the method to insert errors.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> OFF MANual B02: Burst · 1E-02 B03: Burst · 1E-03 B04: Burst · 1E-04 B05: Burst · 1E-05 B06: Burst · 1E-06 B07: Burst · 1E-07 *DEFault = OFF*
Response	None.
Example	`E3:STIM:TX1:EINS MAN`
Note	If insertion is set to MANual, errors are inserted with SYST:STIM:INS. See section 2.3.23

Syntax	E3:STIMuli:TX<Pt>:EINSert?
Description	This query returns the error insertion method.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`E3:STIM:TX1:EINS? → MAN`
Note	

7.3.4 E3:STIMuli:TX<Pt>:EBLength

Syntax	E3:STIMuli:TX<Pt>:EBLength <burstlength>
Description	This command sets the error burst length to generate.
Parameters	<Pt> = Port number
	<burstlength> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=255, DEFault=1*
Response	None.
Example	`E3:STIM:TX1:EBL 1`
Note	

Syntax	E3:STIMuli:TX<Pt>:EBLength?
Description	This query returns the error burst length to generate.
Parameter	<Pt> = Port number
Response	<burstlength> = <NR1 NUMERIC RESPONSE DATA>
Example	`E3:STIM:TX1:EBL? → 1`
Note	

7.4 Results

7.4.1 E3:RX<Pt>:IFETch?

Syntax	E3:RX<Pt>:IFETch? <parameters>
Description	This query fetches an interval if available.
Parameters	<Pt> = Port number
	({<parameter>} + {,}∗) = <EXPRESSION PROGRAM DATA> The response format is listed for each parameter. **Alarms** NSIG: No signal. Response: <Seconds>,<Ratio> AIS: Alarm indication signal. Response: <Seconds>,<Ratio> NFR: No frame. Response: <Seconds>,<Ratio> NSYN: No sync. Response: <Seconds>,<Ratio> DAL: Distant Alarm. Response: <Seconds>,<Ratio> **Errors** FASW: FAS words. Response: <Count>,<Ratio> CODE: Code. Response: <Count>,<Ratio> PATT: Pattern. Response: <Count>,<Ratio> PSL: Pattern slip. Response: <Count>,<Ratio> PBL: Pattern block. Response: <Count>,<Ratio> **Rx frequency** DEV: Frequency deviation. Response: <ppm> **FAS performance errors** FES: FAS ES. Response: <Count>,<Ratio%> FSES: FAS SES. Response: <Count>,<Ratio%> FBBE: FAS BBE. Response: <Count>,<Ratio%> FALS: FAS ALS. Response: <Count>,<Ratio%> FUAT: FAS UAT. Response: <Count>,<Ratio%> FAVT: FAS AVT. Response: <Count>,<Ratio%> FEFS: FAS EFS. Response: <Count>,<Ratio%> **Pattern performance errors** PES: Pattern ES. Response: <Count>,<Ratio%> PSES: Pattern SES. Response: <Count>,<Ratio%> PBBE: Pattern BBE. Response: <Count>,<Ratio%> PALS: Pattern ALS. Response: <Count>,<Ratio%> PUAT: Pattern UAT. Response: <Count>,<Ratio%> PAVT: Pattern AVT. Response: <Count>,<Ratio%> PEFS: Pattern EFS. Response: <Count>,<Ratio%>
Response	{(<result>),}* = <EXPRESSION RESPONSE DATA> Expression format: Numeric List Each result is formatted according to the specification in the parameter field. Values that are not relevant or applicable for the current measurement return NaN (section 1.6.1).
Example	`E3:RX1:IFET? (FES,PATT)` → `(2,0.5),(4,0.25)`
Notes	This command fetches the result from the interval selected by the `MEASurement:SETup:SELect` command (see section 18.2.2). NFR, DAL, FASW, FASB return NaN (section 1.6.1), if the PCM frame is disabled (E3:RX<Pt>:PCMFrame? → 0). If requested result is not available, NaN (section 1.6.1) is returned. If there is one or more results, the last ”,” is always removed.

7.5 Status

7.5.1 E3:STATus:RX<Pt>:PSLevel?

Syntax	E3:STATus:RX<Pt>:PSLevel?
Description	This query returns physical signal level. Unit: dB.
Parameter	<Pt> = Port number
Response	<signallevel> = <NR1 NUMERIC RESPONSE DATA>
Example	`E3:STAT:RX1:PSL? → 0`
Note	Minimum level is -48 dB.

7.5.2 E3:STATus:RX<Pt>:PDEViation?

Syntax	E3:STATus:RX<Pt>:PDEViation?
Description	This query returns physical deviation. Units: ppm and bps.
Parameter	<Pt> = Port number
Response	<ppm> = <NR1 NUMERIC RESPONSE DATA>
	<bps> = <NR1 NUMERIC RESPONSE DATA>
Example	`E3:STAT:RX1:PDEV? → 0, 0`
Note	

7.5.3 E3:STATus:RX<Pt>:PBRate?

Syntax	E3:STATus:RX<Pt>:PBRate?
Description	This query returns physical bit rate. Unit: bps.
Parameter	<Pt> = Port number
Response	<bitrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`E3:STAT:RX1:PBR? → 34368000`
Note	

7.5.4 E3:STATus:RX<Pt>:PPBRate?

Syntax	E3:STATus:RX<Pt>:PPBRate?
Description	This query returns payload pattern bit rate. Unit: bps.
Parameter	<Pt> = Port number
Response	<bitrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`E3:STAT:RX1:PPBR? → 34099504`
Note	

7.5.5 E3:STATus:RX<Pt>:AESummary[:EVENt]?

Syntax	E3:STATus:RX<Pt>:AESummary[:EVENt]?
Description	This query returns the E3 alarms and errors summary event register. The content of this event register is summarized in DB3 of the STATus:INTerface:PORT<Pt>:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 (2) = Error summary DB3 - DB16 = NOT USED
Example	`E3:STAT:RX1:AES? → 1`
Note	

7.5.6 E3:STATus:RX<Pt>:AESummary:CONDition?

Syntax	E3:STATus:RX<Pt>:AESummary:CONDition?
Description	This query returns the E3 alarms and errors summary condition register. DB1 is summary of E3:STATus:RX<Pt>:ALARm[:EVENt]? DB2 is summary of E3:STATus:RX<Pt>:ERRor[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 (2) = Error summary DB3 - DB16 = NOT USED
Example	`E3:STAT:RX1:AES:COND?` → `2`
Note	

7.5.7 E3:STATus:RX<Pt>:ALARm[:EVENt]?

Syntax	E3:STATus:RX<Pt>:ALARm[:EVENt]?
Description	This query returns the alarms event register. The content of this register is summarized in DB1 of the E3:STATus:RX<Pt>:AESummary:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = No signal DB2 (2) = Alarm Indication Signal (AIS) DB3 (4) = No frame DB4 (8) = Distant DB5 (16) = No sync DB6 - DB16 = NOT USED
Example	`E3:STAT:RX1:ALAR?` → `2`
Note	The No frame and Distant are only valid when the PCM frame is enabled (E3:RX<Pt>:PCMFrame? → 1).

7.5.8 E3:STATus:RX<Pt>:ALARm:CONDition?

Syntax	E3:STATus:RX<Pt>:ALARm:CONDition?
Description	This query returns the alarms condition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = No signal DB2 (2) = Alarm Indication Signal (AIS) DB3 (4) = No frame DB4 (8) = Distant DB5 (16) = No sync DB6 - DB16 = NOT USED
Example	`E3:STAT:RX1:ALAR:COND?` → `4`
Note	The No frame and Distant are only valid when the PCM frame is enabled (E3:RX<Pt>:PCMFrame? → 1).

7.5.9 E3:STATus:RX<Pt>:ERRor[:EVENt]?

Syntax	E3:STATus:RX<Pt>:ERRor[:EVENt]?
Description	This query returns the errors event register. The content of this register is summarized in DB2 of the E3:STATus:RX<Pt>:AESummary:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = FAS words DB2 (2) = NOT USED DB3 (4) = Pattern DB4 (8) = Code DB5 (16) = Pattern slip (PSL) DB6 - DB16 = NOT USED
Example	`E3:STAT:RX1:ERR?` → `8`
Notes	The FAS words and FAS bits are only valid when the PCM frame is enabled (E3:RX<Pt>:PCMFrame? → 1). The Code is not valid when using E3 over SDH.

7.5.10 E3:STATus:RX<Pt>:ERRor:CONDition?

Syntax	E3:STATus:RX<Pt>:ERRor:CONDition?
Description	This query returns the errors condition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = FAS words DB2 (2) = NOT USED DB3 (4) = Pattern DB4 (8) = Code DB5 (16) = Pattern slip (PSL) DB6 - DB16 = NOT USED
Example	`E3:STAT:RX1:ERR:COND?` → `8`
Note	The FAS words and FAS bits are only valid when the PCM frame is enabled (E3:RX<Pt>:PCMFrame? → 1). The Code is not valid when using E3 over SDH.

7.6 RTD

This section document commands to retrieve Round Trip Delay measurement results.
Commands for general RTD settings are described in section 17.1 on page 953.

7.6.1 E3:RTD:RX<Pt>:NUMBer?

Syntax	E3:RTD:RX<Pt>:NUMBer?
Description	This query returns the number of the RTD data.
Parameter	<Pt> = Port number
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`E3:RTD:RX1:NUMB?` → `2`
Note	

7.6.2 E3:RTD:RX<Pt>:ATIMe?

Syntax	E3:RTD:RX<Pt>:ATIMe?
Description	This query returns the average time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<avg> = <NR2 NUMERIC RESPONSE DATA>
Example	`E3:RTD:RX1:ATIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

7.6.3 E3:RTD:RX<Pt>:MTIMe?

Syntax	E3:RTD:RX<Pt>:MTIMe?
Description	This query returns the maximum time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
	<limit exceeded> = <NR1 NUMERIC RESPONSE DATA>
Example	`E3:RTD:RX1:MTIM?` → `1.0,0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

7.6.4 E3:RTD:RX<Pt>:LTIMe?

Syntax	E3:RTD:RX<Pt>:LTIMe?
Description	This query returns the least (minimum) time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`E3:RTD:RX1:LTIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

Chapter 8

T3

8.1 Receiver

8.1.1 T3:RX<Pt>[:ENABled]

Syntax	T3:RX<Pt>[:ENABled] <state>
Description	This command enables/disables the receiver.
Parameters	<Pt> = Port number
	<state> = <CHARACTER PROGRAM DATA> OFF ON *DEFault = OFF*
Response	None.
Examples	`T3:RX1 ON` `T3:RX1:ENAB ON`
Note	

Syntax	T3:RX<Pt>[:ENABled]?
Description	This query returns the state (enabled/disabled) of the receiver.
Parameter	<Pt> = Port number
Response	<state> = <CHARACTER RESPONSE DATA>
Examples	`T3:RX1?` → `ON` `T3:RX1:ENAB?` → `SDH`
Note	Returns `SDH` if T3 is over SDH or SONET.

8.1.2 T3:RX<Pt>:MODE

Syntax	T3:RX<Pt>:MODE <mode>
Description	This command sets the signal termination mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> TERMinate: Nominal impedance. Normal frequency dependent AGC. MONitor: Nominal impedance. Frequency linear AGC. *DEFault = TERMinate*
Response	None.
Example	`T3:RX1:MODE TERM`
Note	

Syntax	T3:RX<Pt>:MODE?
Description	This query returns the signal termination mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`T3:RX1:MODE?` → `TERM`
Note	

8.1.3 T3:RX<Pt>:FOLLow

Syntax	T3:RX<Pt>:FOLLow <follow>
Description	This command sets the receiver to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<follow> = <CHARACTER PROGRAM DATA> NONE: Do not follow TX: Transmitter of the same port RX1: Receiver of port 1 *DEFault = NONE*
Response	None.
Example	`T3:RX1:FOLL TX`
Note	

Syntax	T3:RX<Pt>:FOLLow?
Description	This query returns if the receiver follow another setup.
Parameter	<Pt> = Port number
Response	<follow> = <CHARACTER RESPONSE DATA>
Example	`T3:RX1:FOLL?` → `TX`
Note	

8.1.4 T3:RX<Pt>:PCMFrame

Syntax	T3:RX<Pt>:PCMFrame <enable>
Description	This command enables or disables the PCM frame.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`T3:RX1:PCMF ON`
Note	

Syntax	T3:RX<Pt>:PCMFrame?
Description	This query returns whether or not PCM frame is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`T3:RX1:PCMF?` → `1`
Note	

8.1.5 T3:RX<Pt>:FTYPe

Syntax	T3:RX<Pt>:FTYPe <type>
Description	This command sets the PCM frame type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> M13: M13 mode CBIT: C-bit mode *DEFault = M13*
Response	None.
Example	`T3:RX1:FTYP CBIT`
Note	

Syntax	T3:RX<Pt>:FTYPe?
Description	This query returns PCM frame type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`T3:RX1:FTYP?` → `CBIT`
Note	

8.1.6 T3:RX<Pt>:PATTern

Syntax	T3:RX<Pt>:PATTern <type>
Description	This command sets the pattern type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> OFF USER16BIT: Obsolete. For backward compatibility only. Same as USER32BIT. USER32BIT: 32 bit user defined pattern. USER2048BIT: 2048 bit user defined pattern. PRBS9 PRBS11 PRBS15 PRBS20 PRBS23 PRBS29 PRBS31 QRSS20 FOX FOXCMA3000 ALL0 ALL1 ALT11: Alternating 1:1 ALT13: Alternating 1:3 ALT17: Alternating 1:7 ALT324: Alternating 3:24 *DEFault = PRBS23*
Response	None.
Example	`T3:RX1:PATT PRBS11`
Note	

Syntax	T3:RX<Pt>:PATTern?
Description	This query returns the pattern type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`T3:RX1:PATT?` → `PRBS11`
Note	

8.1.7 T3:RX<Pt>:PINVersion

Syntax	T3:RX<Pt>:PINVersion <inverted>
Description	This command enables or disables pattern inversion.
Parameters	<Pt> = Port number
	<inverted> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`T3:RX1:PINV ON`
Note	The following patterns can be inverted: PRBSxx, FOX, ALL1, ALL0, ALT11, ALT13, ALT17, ALT324, UP16 and UP2K.

Syntax	T3:RX<Pt>:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the pattern.
Parameter	<Pt> = Port number
Response	<inverted> = <NR1 NUMERIC RESPONSE DATA>
Example	`T3:RX1:PINV?` → `1`
Note	

8.1.8 T3:RX<Pt>:UP16

Syntax	T3:RX<Pt>:UP16 <pattern>
Description	This command sets 16 bit user pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Valid characters: '0' and '1' The string must consist of 1 to 16 characters.
Response	None.
Example	`T3:RX1:UP16 "101101"`
Note	This command is for backward compatibility only, and a query command is not available. Actually the new 32 bit user pattern is set by this command.

8.1.9 T3:RX<Pt>:UP32

Syntax	T3:RX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters.
Response	None.
Example	`T3:RX2:UP32 "01101"`
Note	

Syntax	T3:RX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`T3:RX2:UP32?` → `"01101"`
Note	

8.1.10 T3:RX<Pt>:UP2K

Syntax	T3:RX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`T3:RX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `T3:RX2:PATT` is `USER2048BIT`

Syntax	T3:RX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`T3:RX1:UP2K?` → `"12DF"`
Note	

8.2 Transmitter

8.2.1 T3:TX<Pt>[:ENABled]

Syntax	T3:TX<Pt>[:ENABled] <state>
Description	This command enables/disables the transmitter.
Parameters	<Pt> = Port number
	<state> = <CHARACTER PROGRAM DATA> OFF ON *DEFault = OFF*
Response	None.
Examples	`T3:TX1 ON` `T3:TX1:ENAB ON`
Note	The ENABled command is the default node for T3:TX<Pt>.

Syntax	T3:TX<Pt>[:ENABled]?
Description	This query returns the state (enabled/disabled) of the transmitter.
Parameter	<Pt> = Port number
Response	<state> = <CHARACTER RESPONSE DATA>
Examples	`T3:TX1?` → `ON` `T3:TX1:ENAB?` → `SDH`
Note	Returns `SDH` if T3 is over SDH or SONET.

8.2.2 T3:TX<Pt>:FOLLow

Syntax	T3:TX<Pt>:FOLLow <follow>
Description	This command sets the transmitter setting to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<follow> = <CHARACTER PROGRAM DATA> NONE: Do not follow TX1: Follow the setting of the port 1 transmitter *DEFault = NONE*
Response	None.
Example	`T3:TX2:FOLL TX1`
Note	This command is not valid for `:TX1`

Syntax	T3:TX<Pt>:FOLLow?
Description	This query returns if the transmitter follow the TX1 settings.
Parameter	<Pt> = Port number
Response	<follow> = <CHARACTER RESPONSE DATA>
Example	`T3:TX2:FOLL?` → `TX1`
Note	This command is not valid for `:TX1`

8.2.3 T3:TX<Pt>:TIMing

Syntax	T3:TX<Pt>:TIMing <source>
Description	This command sets the timing source.
Parameters	<Pt> = Port number
	<source> = <CHARACTER PROGRAM DATA> INTernal: Internal timing source EXTernal: External timing source RX: Received signal on the same port *DEFault = INTernal*
Response	None.
Example	`T3:TX1:TIM INT`
Note	

Syntax	T3:TX<Pt>:TIMing?
Description	This query returns the timing source.
Parameter	<Pt> = Port number
Response	<source> = <CHARACTER RESPONSE DATA>
Example	`T3:TX1:TIM?` → `INT`
Note	

8.2.4 T3:TX<Pt>:FOFFset

Syntax	T3:TX<Pt>:FOFFset <offset>
Description	This command sets the frequency offset for the clock source. Unit: ppm.
Parameters	<Pt> = Port number
	<offset> = <NUMERIC PROGRAM DATA> *MINimum=-125, MAXimum=125, DEFault=0* *Allowed suffix = ppm*
Response	None.
Example	`T3:TX1:FOFF -25ppm`
Note	

Syntax	T3:TX<Pt>:FOFFset?
Description	This query returns the frequency offset for the clock source. Unit: ppm.
Parameter	<Pt> = Port number
Response	<offset> = <NR1 NUMERIC RESPONSE DATA>
Example	`T3:TX1:FOFF?` → `-25`
Note	

8.2.5 T3:TX<Pt>:PCMFrame

Syntax	T3:TX<Pt>:PCMFrame <enable>
Description	This command enables or disables the PCM frame.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`T3:TX1:PCMF ON`
Note	

Syntax	T3:TX<Pt>:PCMFrame?
Description	This query returns whether or not PCM frame is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`T3:TX1:PCMF?` → `1`
Note	

8.2.6 T3:TX<Pt>:FTYPe

Syntax	T3:TX<Pt>:FTYPe <type>
Description	This command sets the PCM frame type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> M13: M13 mode CBIT: C-bit mode *DEFault = M13*
Response	None.
Example	`T3:TX1:FTYP CBIT`
Note	

Syntax	T3:TX<Pt>:FTYPe?
Description	This query returns PCM frame type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`T3:TX1:FTYP?` → `CBIT`
Note	

8.2.7 T3:TX<Pt>:LBOut

Syntax	T3:TX<Pt>:LBOut <lbo>
Description	This command sets the line build out.
Parameters	<Pt> = Port number
	<lbo> = <CHARACTER PROGRAM DATA> HIGH: High-0 ft DSX: DSX-450 ft *DEFault = HIGH*
Response	None.
Example	`T3:TX1:LBO DSX`
Note	

Syntax	T3:TX<Pt>:LBOut?
Description	This query returns the line build out setting.
Parameter	<Pt> = Port number
Response	<lbo> = <CHARACTER RESPONSE DATA>
Example	`T3:TX1:LBO?` → `DSX`
Note	

8.2.8 T3:TX<Pt>:PATTern

Syntax	T3:TX<Pt>:PATTern <type>
Description	This command sets the pattern type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> OFF USER16BIT: Obsolete. For backward compatibility only. Same as USER32BIT. USER32BIT: 32 bit user defined pattern. USER2048BIT: 2048 bit user defined pattern. PRBS9 PRBS11 PRBS15 PRBS20 PRBS23 PRBS29 PRBS31 QRSS20 FOX FOXCMA3000 ALL0 ALL1 ALT11: Alternating 1:1 ALT13: Alternating 1:3 ALT17: Alternating 1:7 ALT324: Alternating 3:24 *DEFault = PRBS23*
Response	None.
Example	`T3:TX1:PATT PRBS11`
Note	

Syntax	T3:TX<Pt>:PATTern?
Description	This query returns the pattern type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`T3:TX1:PATT?` → `PRBS11`
Note	

8.2.9 T3:TX<Pt>:PINVersion

Syntax	T3:TX<Pt>:PINVersion <inverted>
Description	This command sets pattern inversion.
Parameters	<Pt> = Port number
	<inverted> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`T3:TX1:PINV ON`
Note	

Syntax	T3:TX<Pt>:PINVersion?
Description	This query returns pattern inversion.
Parameter	<Pt> = Port number
Response	<inverted> = <NR1 NUMERIC RESPONSE DATA>
Example	`T3:TX1:PINV?` → `1`
Note	

8.2.10 T3:TX<Pt>:UP16

Syntax	T3:TX<Pt>:UP16 <pattern>
Description	This command sets 16 bit user pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Valid characters: '0' and '1' The string must consist of 1 to 16 characters.
Response	None.
Example	`T3:TX1:UP16 "101101"`
Note	This command is for backward compatibility only, and a query command is not available. Actually the new 32 bit user pattern is set by this command.

8.2.11 T3:TX<Pt>:UP32

Syntax	T3:TX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters.
Response	None.
Examples	`T3:TX1:UP32 "01101"`
Note	

Syntax	T3:TX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`T3:TX1:UP32?` → `"01101"`
Note	

8.2.12 T3:TX<Pt>:UP2K

Syntax	T3:TX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`T3:TX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `T3:TX2:PATT` is `USER2048BIT`

Syntax	T3:TX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`T3:TX1:UP2K?` → `"12DF"`
Note	

8.3 Stimuli

8.3.1 T3:STIMuli:TX<Pt>:ALARm

Syntax	T3:STIMuli:TX<Pt>:ALARm <alarmtype>
Description	This command sets the alarm type to generate.
Parameters	<Pt> = Port number
	<alarmtype> = <CHARACTER PROGRAM DATA> NALarm: No alarm NSIGnal: No signal AIS: Alarm Indication Signal RAI: Remote Alarm Indication IDLE: Idle NFRame: No frame (LOF) NSYNc: No pattern sync (LSS) *DEFault = NALarm*
Response	None.
Example	`T3:STIM:TX1:ALAR NALarm`
Note	

Syntax	T3:STIMuli:TX<Pt>:ALARm?
Description	This query returns the stimuli alarm type.
Parameter	<Pt> = Port number
Response	<alarmtype> = <CHARACTER RESPONSE DATA>
Example	`T3:STIM:TX1:ALAR? → NSIG`
Note	

8.3.2 T3:STIMuli:TX<Pt>:ERRor

Syntax	T3:STIMuli:TX<Pt>:ERRor <errordestination>
Description	This command sets the error destination.
Parameters	<Pt> = Port number
	<errordestination> = <CHARACTER PROGRAM DATA> OFF CBIT: C-Bit FBIT: F-Bit PBIT: P-Bit FEBE: PATTern: Pattern Error PSLip: Pattern slip BPV: *DEFault = FBIT*
Response	None.
Example	`T3:STIM:TX1:ERR FEBE`
Note	

Syntax	T3:STIMuli:TX<Pt>:ERRor?
Description	This query returns the error destination.
Parameter	<Pt> = Port number
Response	<errordestination> = <CHARACTER RESPONSE DATA>
Example	`T3:STIM:TX1:ERR? → FEBE`
Note	

8.3.3 T3:STIMuli:TX<Pt>:EINSert

Syntax	T3:STIMuli:TX<Pt>:EINSert <insertion>
Description	This command sets the method to insert errors.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> OFF MANual B02: Burst · 1E-02 B03: Burst · 1E-03 B04: Burst · 1E-04 B05: Burst · 1E-05 B06: Burst · 1E-06 B07: Burst · 1E-07 *DEFault = OFF*
Response	None.
Example	`T3:STIM:TX1:EINS MAN`
Note	If insertion is set to MANual, errors are inserted with SYST:STIM:INS. See section 2.3.23

Syntax	T3:STIMuli:TX<Pt>:EINSert?
Description	This query returns the error insertion method.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`T3:STIM:TX1:EINS? → MAN`
Note	

8.3.4 T3:STIMuli:TX<Pt>:EBLength

Syntax	T3:STIMuli:TX<Pt>:EBLength <burstlength>
Description	This command sets the error burst length to generate.
Parameters	<Pt> = Port number
	<burstlength> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=255, DEFault=1*
Response	None.
Example	`T3:STIM:TX1:EBL 1`
Note	

Syntax	T3:STIMuli:TX<Pt>:EBLength?
Description	This query returns the error burst length to generate.
Parameter	<Pt> = Port number
Response	<burstlength> = <NR1 NUMERIC RESPONSE DATA>
Example	`T3:STIM:TX1:EBL? → 1`
Note	

8.4 Results

8.4.1 T3:RX<Pt>:IFETch?

Syntax	T3:RX<Pt>:IFETch? <parameters>
Description	This query fetches an interval if available.
Parameters	<Pt> = Port number
	({<parameter>} + {,}*) = <EXPRESSION PROGRAM DATA> The response format is listed for each parameter. **Alarms** NSIG: No signal. Response: <Seconds>,<Ratio> AIS: Alarm indication signal. Response: <Seconds>,<Ratio> NFR: No frame. Response: <Seconds>,<Ratio> NSYN: No sync (LSS). Response: <Seconds>,<Ratio> DAL: Distant Alarm (RAI). Response: <Seconds>,<Ratio> IDLE: Idle alarm. Response: <Seconds>,<Ratio> **Errors** CODE: Code (BPV). Response: <Count>,<Ratio> FBIT: F-bit. Response: <Count>,<Ratio> CBIT: C-bit. Response: <Count>,<Ratio> PATT: Pattern. Response: <Count>,<Ratio> PSL: Pattern slip. Response: <Count>,<Ratio> PBL: Pattern block. Response: <Count>,<Ratio> PAR: Parity. Response: <Count>,<Ratio> FEBE: FEBE. Response: <Count>,<Ratio> **Rx frequency** DEV: Frequency deviation. Response: <ppm> **FAS performance errors** FES: FAS ES. Response: <Count>,<Ratio%> FSES: FAS SES. Response: <Count>,<Ratio%> FBBE: FAS BBE. Response: <Count>,<Ratio%> FALS: FAS ALS. Response: <Count>,<Ratio%> FUAT: FAS UAT. Response: <Count>,<Ratio%> FAVT: FAS AVT. Response: <Count>,<Ratio%> FEFS: FAS EFS. Response: <Count>,<Ratio%> **Pattern performance errors** PES: Pattern ES. Response: <Count>,<Ratio%> PSES: Pattern SES. Response: <Count>,<Ratio%> PBBE: Pattern BBE. Response: <Count>,<Ratio%> PALS: Pattern ALS. Response: <Count>,<Ratio%> PUAT: Pattern UAT. Response: <Count>,<Ratio%> PAVT: Pattern AVT. Response: <Count>,<Ratio%> PEFS: Pattern EFS. Response: <Count>,<Ratio%>
Response	{(<result>),}* = <EXPRESSION RESPONSE DATA> Expression format: Numeric List Each result is formatted according to the specification in the parameter field. Values that are not relevant or applicable for the current measurement return NaN.
Example	`T3:RX1:IFET? (FES,PATT)` → `(2,0.5),(4,0.25)`
Notes	This command fetches the result from the interval selected by the `MEASurement:SETup:SELect` command (see section 18.2.2). NFR, DAL, FASW, FASB return NaN, if the PCM frame is disabled (T3:RX<Pt>:PCMFrame? → 0). If requested result is not available, NaN is returned. If there is one or more results, the last ”,” is always removed.

8.5 Status

8.5.1 T3:STATus:RX<Pt>:PSLevel?

Syntax	T3:STATus:RX<Pt>:PSLevel?
Description	This query returns physical signal level. Unit: dB.
Parameter	<Pt> = Port number
Response	<signallevel> = <NR1 NUMERIC RESPONSE DATA>
Example	`T3:STAT:RX1:PSL?` → `0`
Note	Minimum level is -48 dB.

8.5.2 T3:STATus:RX<Pt>:PDEViation?

Syntax	T3:STATus:RX<Pt>:PDEViation?
Description	This query returns physical deviation. Units: ppm and bps.
Parameter	<Pt> = Port number
Response	<ppm> = <NR1 NUMERIC RESPONSE DATA>
	<bps> = <NR1 NUMERIC RESPONSE DATA>
Example	`T3:STAT:RX1:PDEV?` → `0, 0`
Note	

8.5.3 T3:STATus:RX<Pt>:PBRate?

Syntax	T3:STATus:RX<Pt>:PBRate?
Description	This query returns physical bit rate. Unit: bps.
Parameter	<Pt> = Port number
Response	<bitrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`T3:STAT:RX1:PBR?` → `44736000`
Note	

8.5.4 T3:STATus:RX<Pt>:PPBRate?

Syntax	T3:STATus:RX<Pt>:PPBRate?
Description	This query returns payload pattern bit rate. Unit: bps.
Parameter	<Pt> = Port number
Response	<bitrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`T3:STAT:RX1:PPBR?` → `44209704`
Note	

8.5.5 T3:STATus:RX<Pt>:AESummary[:EVENt]?

Syntax	T3:STATus:RX<Pt>:AESummary[:EVENt]?
Description	This query returns the T3 alarms and errors summary event register. The content of this event register is summarized in DB11 of the STATus:INTerface:PORT<Pt>:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 (2) = Error summary DB3 - DB16 = NOT USED
Example	`T3:STAT:RX1:AES?` → `1`
Note	

8.5.6 T3:STATus:RX<Pt>:AESummary:CONDition?

Syntax	T3:STATus:RX<Pt>:AESummary:CONDition?
Description	This query returns the T3 alarms and errors summary condition register. DB1 is summary of T3:STATus:RX<Pt>:ALARm[:EVENt]? DB2 is summary of T3:STATus:RX<Pt>:ERRor[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 (2) = Error summary DB3 - DB16 = NOT USED
Example	`T3:STAT:RX1:AES:COND?` → `2`
Note	

8.5.7 T3:STATus:RX<Pt>:ALARm[:EVENt]?

Syntax	T3:STATus:RX<Pt>:ALARm[:EVENt]?
Description	This query returns the alarms event register. The content of this register is summarized in DB1 of the T3:STATus:RX<Pt>:AESummary:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = No signal DB2 (2) = NOT USED DB3 (4) = Alarm Indication Signal (AIS) DB4 (8) = No frame (LOF) DB5 (16) = NOT USED DB6 (32) = Distant (RAI) DB7 (64) = No sync (LSS) DB8 (128) = Idle DB9 - DB16 = NOT USED
Example	`T3:STAT:RX1:ALAR?` → `2`
Note	The No frame and Distant are only valid when the PCM frame is enabled (T3:RX<Pt>:PCMFrame? → 1).

8.5.8 T3:STATus:RX<Pt>:ALARm:CONDition?

Syntax	T3:STATus:RX<Pt>:ALARm:CONDition?
Description	This query returns the alarms condition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = No signal DB2 (2) = NOT USED DB3 (4) = Alarm Indication Signal (AIS) DB4 (8) = No frame (LOF) DB5 (16) = NOT USED DB6 (32) = Distant (RAI) DB7 (64) = No sync (LSS) DB8 (128) = Idle DB9 - DB16 = NOT USED
Example	`T3:STAT:RX1:ALAR:COND?` → `4`
Note	The No frame and Distant are only valid when the PCM frame is enabled (T3:RX<Pt>:PCMFrame? → 1).

8.5.9 T3:STATus:RX<Pt>:ERRor[:EVENt]?

Syntax	T3:STATus:RX<Pt>:ERRor[:EVENt]?
Description	This query returns the errors event register. The content of this register is summarized in DB2 of the T3:STATus:RX<Pt>:AESummary:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = F-bit) DB2 (2) = NOT USED DB3 (4) = Pattern DB4 (8) = Code (BPV) DB5 (16) = Pattern slip (PSL) DB6 (32) = Parity DB7 (64) = C-bit DB8 (128) = FEBE DB9 - DB16 = NOT USED
Example	`T3:STAT:RX1:ERR?` → `8`
Notes	The FAS words and FAS bits are only valid when the PCM frame is enabled (T3:RX<Pt>:PCMFrame? → 1). The Code is not valid when using T3 over SDH.

8.5.10 T3:STATus:RX<Pt>:ERRor:CONDition?

Syntax	T3:STATus:RX<Pt>:ERRor:CONDition?
Description	This query returns the errors condition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = F-bit) DB2 (2) = NOT USED DB3 (4) = Pattern DB4 (8) = Code (BPV) DB5 (16) = Pattern slip (PSL) DB6 (32) = Parity DB7 (64) = C-bit DB8 (128) = FEBE DB9 - DB16 = NOT USED
Example	`T3:STAT:RX1:ERR:COND?` → `8`
Note	The FAS words and FAS bits are only valid when the PCM frame is enabled (T3:RX<Pt>:PCMFrame? → 1). The Code is not valid when using T3 over SDH.

8.6 RTD

This section document commands to retrieve Round Trip Delay measurement results.
Commands for general RTD settings are described in section 17.1 on page 953.

8.6.1 T3:RTD:RX<Pt>:NUMBer?

Syntax	T3:RTD:RX<Pt>:NUMBer?
Description	This query returns the number of the RTD data.
Parameter	<Pt> = Port number
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`T3:RTD:RX1:NUMB?` → `2`
Note	

8.6.2 T3:RTD:RX<Pt>:ATIMe?

Syntax	T3:RTD:RX<Pt>:ATIMe?
Description	This query returns the average time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<avg> = <NR2 NUMERIC RESPONSE DATA>
Example	`T3:RTD:RX1:ATIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

8.6.3 T3:RTD:RX<Pt>:MTIMe?

Syntax	T3:RTD:RX<Pt>:MTIMe?
Description	This query returns the maximum time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
	<limit exceeded> = <NR1 NUMERIC RESPONSE DATA> Returns 1, if the reference maximum limit has been exceeded.
Example	`T3:RTD:RX1:MTIM?` → `1.0,0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

8.6.4 T3:RTD:RX<Pt>:LTIMe?

Syntax	T3:RTD:RX<Pt>:LTIMe?
Description	This query returns the least (minimum) time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`T3:RTD:RX1:LTIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

Chapter 9

E4

9.1 Receiver

9.1.1 E4:RX<Pt>[:ENABled]

Syntax	E4:RX<Pt>[:ENABled] <state>
Description	This command enables/disables the receiver.
Parameters	<Pt> = Port number
	<state> = <CHARACTER PROGRAM DATA> OFF ON *DEFault = OFF*
Response	None.
Examples	`E4:RX1 ON` `E4:RX1:ENAB ON`
Note	ENABled is the default node for E4:RX<Pt>.

Syntax	E4:RX<Pt>[:ENABled]?
Description	This query returns the state (enabled/disabled) of the receiver.
Parameter	<Pt> = Port number
Response	<state> = <CHARACTER RESPONSE DATA>
Examples	`E4:RX1? → ON` `E4:RX1:ENAB? → SDH`
Note	Returns `SDH` if E4 is over SDH or SONET.

9.1.2 E4:RX<Pt>:GAIN

Syntax	E4:RX<Pt>:GAIN <gain>
Description	This command sets the gain for the receiver.
Parameters	<Pt> = Port number
	<gain> = <CHARACTER PROGRAM DATA> TERMinate: Nominal impedance. Normal frequency dependent AGC. MONitor: Nominal impedance. Frequency linear AGC. *DEFault = TERMinate*
Response	None.
Example	`E4:RX1:GAIN MON`
Note	

Syntax	E4:RX<Pt>:GAIN?
Description	This query returns the gain of the receiver.
Parameter	<Pt> = Port number
Response	<gain> = <CHARACTER RESPONSE DATA>
Example	`E4:RX1:GAIN? → MON`
Note	

9.1.3 E4:RX<Pt>:FOLLow

Syntax	E4:RX<Pt>:FOLLow <follow>
Description	This command sets the receiver to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<follow> = <CHARACTER PROGRAM DATA> NONE: Do not follow TX: Transmitter of the same port RX1: Receiver of port 1 *DEFault = NONE*
Response	None.
Example	`E4:RX1:FOLL TX`
Note	

Syntax	E4:RX<Pt>:FOLLow?
Description	This query returns if the receiver follow another setup.
Parameter	<Pt> = Port number
Response	<follow> = <CHARACTER RESPONSE DATA>
Example	`E4:RX1:FOLL?` → `TX`
Note	

9.1.4 E4:RX<Pt>:PCMFrame

Syntax	E4:RX<Pt>:PCMFrame <enable>
Description	This command enables or disables the PCM frame.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`E4:RX1:PCMF ON`
Note	

Syntax	E4:RX<Pt>:PCMFrame?
Description	This query returns whether or not PCM frame is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`E4:RX1:PCMF?` → `1`
Note	

9.1.5 E4:RX<Pt>:PATTern

Syntax	E4:RX<Pt>:PATTern <type>
Description	This command sets the pattern type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> OFF USER16BIT: Obsolete. For backward compatibility only. Same as USER32BIT. USER32BIT: 32 bit user defined pattern. USER2048BIT: 2048 bit user defined pattern. PRBS9 PRBS11 PRBS15 PRBS20 PRBS23 PRBS29 PRBS31 QRSS20 ALL0 ALL1 ALT11: Alternating 1:1 ALT13: Alternating 1:3 ALT17: Alternating 1:7 ALT324: Alternating 3:24 *DEFault = PRBS23*
Response	None.
Example	`E4:RX1:PATT PRBS23`
Note	

Syntax	E4:RX<Pt>:PATTern?
Description	This query returns the pattern type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`E4:RX1:PATT? → PRBS11`
Note	

9.1.6 E4:RX<Pt>:PINVersion

Syntax	E4:RX<Pt>:PINVersion <inverted>
Description	This command enables or disables pattern inversion.
Parameters	<Pt> = Port number
	<inverted> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`E4:RX1:PINV ON`
Note	The following patterns can be inverted: PRBSxx, QRSSxx, ALL1, ALL0, ALT11, ALT13, ALT17 and UP16.

Syntax	E4:RX<Pt>:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the pattern.
Parameter	<Pt> = Port number
Response	<inverted> = <NR1 NUMERIC RESPONSE DATA>
Example	`E4:RX1:PINV? → 1`
Note	

9.1.7 E4:RX<Pt>:UP16

Syntax	E4:RX<Pt>:UP16 <pattern>
Description	This command sets 16 bit user pattern.
Parameters	<Pt> = Port number
	<pattern> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=65535* *DEFault=0*
Response	None.
Example	`E4:RX1:UP16 #B1111000011110000`
Note	This command is for backward compatibility only, and a query command is not available. Actually the new 32 bit user pattern is set by this command.

9.1.8 E4:RX<Pt>:UP32

Syntax	E4:RX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters (one bit resolution).
Response	None.
Examples	`E4:RX1:UP32 "01101"`
Note	

Syntax	E4:RX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`E4:RX1:UP32?` → `"01101"`
Note	

9.1.9 E4:RX<Pt>:UP2K

Syntax	E4:RX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`E4:RX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `E4:RX2:PATT` is `USER2048BIT`

Syntax	E4:RX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`E4:RX1:UP2K?` → `"12DF"`
Note	

9.2 Transmitter

9.2.1 E4:TX<Pt>[:ENABled]

Syntax	E4:TX<Pt>[:ENABled] <state>
Description	This command enables/disables the transmitter.
Parameters	<Pt> = Port number
	<state> = <CHARACTER PROGRAM DATA> OFF ON *DEFault = OFF*
Response	None.
Examples	`E4:TX ON` `E4:TX1:ENAB ON`
Note	The ENABled command is the default node for E4:TX.

Syntax	E4:TX<Pt>[:ENABled]?
Description	This query returns the state (enabled/disabled) of the transmitter.
Parameter	<Pt> = Port number
Response	<state> = <CHARACTER RESPONSE DATA>
Examples	`E4:TX?` → `ON` `E4:TX1:ENAB?` → `SDH`
Note	Returns `SDH` if E4 is over SDH or SONET.

9.2.2 E4:TX<Pt>:FOLLow

Syntax	E4:TX<Pt>:FOLLow <follow>
Description	This command sets the transmitter setting to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<follow> = <CHARACTER PROGRAM DATA> NONE: Do not follow TX1: Follow the setting of the port 1 transmitter *DEFault = NONE*
Response	None.
Example	`E4:TX2:FOLL TX1`
Note	This command is not valid for `:TX1`

Syntax	E4:TX<Pt>:FOLLow?
Description	This query returns if the transmitter follow the TX1 settings.
Parameter	<Pt> = Port number
Response	<follow> = <CHARACTER RESPONSE DATA>
Example	`E4:TX2:FOLL?` → `TX1`
Note	This command is not valid for `:TX1`

9.2.3 E4:TX<Pt>:TIMing

Syntax	E4:TX<Pt>:TIMing <source>
Description	This command sets the timing source.
Parameters	<Pt> = Port number
	<source> = <CHARACTER PROGRAM DATA> INTernal: Internal timing source EXTernal: External timing source RX: Received signal on the same port *DEFault = INTernal*
Response	None.
Example	`E4:TX1:TIM INT`
Note	

Syntax	E4:TX<Pt>:TIMing?
Description	This query returns the timing source.
Parameter	<Pt> = Port number
Response	<source> = <CHARACTER RESPONSE DATA>
Example	`E4:TX1:TIM?` → `INT`
Note	

9.2.4 E4:TX<Pt>:FOFFset

Syntax	E4:TX<Pt>:FOFFset <offset>
Description	This command sets the frequency offset for the clock source. Unit: ppm.
Parameters	<Pt> = Port number
	<offset> = <NUMERIC PROGRAM DATA> *MINimum=-125, MAXimum=125, DEFault=0* *Allowed suffix = ppm*
Response	None.
Example	`E4:TX1:FOFF 0ppm`
Note	

Syntax	E4:TX<Pt>:FOFFset?
Description	This query returns the frequency offset for the clock source.
Parameter	<Pt> = Port number
Response	<offset> = <NR1 NUMERIC RESPONSE DATA>
Example	`E4:TX1:FOFF?` → `0`
Note	Returned value is in ppm.

9.2.5 E4:TX<Pt>:PCMFrame

Syntax	E4:TX<Pt>:PCMFrame <enable>
Description	This command enables or disables the PCM frame.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`E4:TX1:PCMF ON`
Note	

Syntax	E4:TX<Pt>:PCMFrame?
Description	This query returns whether or not PCM frame is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`E4:TX1:PCMF?` → `1`
Note	

9.2.6 E4:TX<Pt>:PATTern

Syntax	E4:TX<Pt>:PATTern <type>
Description	This command sets the pattern type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> OFF USER16BIT: Obsolete. For backward compatibility only. Same as USER32BIT. USER32BIT: 32 bit user defined pattern. USER2048BIT: 2048 bit user defined pattern. PRBS9 PRBS11 PRBS15 PRBS20 PRBS23 PRBS29 PRBS31 QRSS20 ALL0 ALL1 ALT11: Alternating 1:1 ALT13: Alternating 1:3 ALT17: Alternating 1:7 ALT324: Alternating 3:24 *DEFault = PRBS23*
Response	None.
Example	`E4:TX1:PATT PRBS23`
Note	

Syntax	E4:TX<Pt>:PATTern?
Description	This query returns the pattern type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`E4:TX1:PATT?` → `PRBS11`
Note	

9.2.7 E4:TX<Pt>:PINVersion

Syntax	E4:TX<Pt>:PINVersion <inverted>
Description	This command sets pattern inversion.
Parameters	<Pt> = Port number
	<inverted> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`E4:TX1:PINV ON`
Note	

Syntax	E4:TX<Pt>:PINVersion?
Description	This query returns pattern inversion.
Parameter	<Pt> = Port number
Response	<inverted> = <NR1 NUMERIC RESPONSE DATA>
Example	`E4:TX1:PINV?` → `1`
Note	

9.2.8 E4:TX<Pt>:UP16

Syntax	E4:TX<Pt>:UP16 <pattern>
Description	This command sets 16 bit user pattern.
Parameters	<Pt> = Port number
	<pattern> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=65535* *DEFault=0*
Response	None.
Example	`E4:TX1:UP16 #B1111000011110000`
Note	This command is for backward compatibility only, and a query command is not available. Actually the new 32 bit user pattern is set by this command.

9.2.9 E4:TX<Pt>:UP32

Syntax	E4:TX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters (one bit resolution).
Response	None.
Examples	`E4:TX1:UP32 "01101"`
Note	

Syntax	E4:TX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`E4:TX1:UP32?` → `"01101"`
Note	

9.2.10 E4:TX<Pt>:UP2K

Syntax	E4:TX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`E4:TX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `E4:TX2:PATT` is `USER2048BIT`

Syntax	E4:TX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`E4:TX1:UP2K?` → `"12DF"`
Note	

9.3 Stimuli

9.3.1 E4:STIMuli:TX<Pt>:ALARm

Syntax	E4:STIMuli:TX<Pt>:ALARm <alarmtype>
Description	This command sets the alarm type to generate.
Parameters	<Pt> = Port number
	<alarmtype> = <CHARACTER PROGRAM DATA> NALarm: No alarm NSIGnal: No signal AIS: Alarm Indication Signal NFRame: No frame DALarm: Distant alarm (RDI) NSYNc: No pattern sync *DEFault = NALarm*
Response	None.
Example	`E4:STIM:TX1:ALAR NSIG`
Note	

Syntax	E4:STIMuli:TX<Pt>:ALARm?
Description	This query returns the stimuli alarm type.
Parameter	<Pt> = Port number
Response	<alarmtype> = <CHARACTER RESPONSE DATA>
Example	`E4:STIM:TX1:ALAR?` → `NSIG`
Note	

9.3.2 E4:STIMuli:TX<Pt>:ERRor

Syntax	E4:STIMuli:TX<Pt>:ERRor <errordestination>
Description	This command sets the error destination.
Parameters	<Pt> = Port number
	<errordestination> = <CHARACTER PROGRAM DATA> FRAMe: Frame PATTern: Pattern PSLip: Pattern slip *DEFault = FRAMe*
Response	None.
Example	`E4:STIM:TX1:ERR PATT`
Note	

Syntax	E4:STIMuli:TX<Pt>:ERRor?
Description	This query returns the error destination.
Parameter	<Pt> = Port number
Response	<errordestination> = <CHARACTER RESPONSE DATA>
Example	`E4:STIM:TX1:ERR?` → `PATT`
Note	

9.3.3 E4:STIMuli:TX<Pt>:EINSert

Syntax	E4:STIMuli:TX<Pt>:EINSert <insertion>
Description	This command sets the method to insert errors.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> OFF MANual B02: Burst · 1E-02 B03: Burst · 1E-03 B04: Burst · 1E-04 B05: Burst · 1E-05 B06: Burst · 1E-06 B07: Burst · 1E-07 *DEFault = OFF*
Response	None.
Example	`E4:STIM:TX1:EINS MAN`
Note	If insertion is set to MANual, errors are inserted with SYST:STIM:INS. See section 2.3.23

Syntax	E4:STIMuli:TX<Pt>:EINSert?
Description	This query returns the error insertion method.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`E4:STIM:TX1:EINS? → MAN`
Note	

9.3.4 E4:STIMuli:TX<Pt>:EBLength

Syntax	E4:STIMuli:TX<Pt>:EBLength <burstlength>
Description	This command sets the error burst length to generate.
Parameters	<Pt> = Port number
	<burstlength> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=255, DEFault=1*
Response	None.
Example	`E4:STIM:TX1:EBL 1`
Note	

Syntax	E4:STIMuli:TX<Pt>:EBLength?
Description	This query returns the error burst length to generate.
Parameter	<Pt> = Port number
Response	<burstlength> = <NR1 NUMERIC RESPONSE DATA>
Example	`E4:STIM:TX1:EBL? → 1`
Note	

9.4 Status

9.4.1 E4:STATus:RX<Pt>:PSLevel?

Syntax	E4:STATus:RX<Pt>:PSLevel?
Description	This query returns physical signal level. Unit: dB.
Parameter	<Pt> = Port number
Response	<signallevel> = <STRING RESPONSE DATA>
Example	`E4:STAT:RX1:PSL?` → `"-0.5"`
Notes	If signal levels is too high, the response will be "TOO HIGH".

9.4.2 E4:STATus:RX<Pt>:PDEViation?

Syntax	E4:STATus:RX<Pt>:PDEViation?
Description	This query returns physical deviation. Units: ppm and bps.
Parameter	<Pt> = Port number
Response	<ppm> = <NR1 NUMERIC RESPONSE DATA>
	<bps> = <NR1 NUMERIC RESPONSE DATA>
Example	`E4:STAT:RX1:PDEV?` → `1,136`
Note	

9.4.3 E4:STATus:RX<Pt>:PBRate?

Syntax	E4:STATus:RX<Pt>:PBRate?
Description	This query returns physical bit rate. Unit: bps.
Parameter	<Pt> = Port number
Response	<bitrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`E4:STAT:RX1:PBR?` → `139264000`
Note	

9.4.4 E4:STATus:RX<Pt>:PPBRate?

Syntax	E4:STATus:RX<Pt>:PPBRate?
Description	This query returns payload pattern bit rate. Unit: bps.
Parameter	<Pt> = Port number
Response	<bitrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`E4:STAT:RX1:PPBR?` → `137361488`
Note	

9.4.5 E4:STATus:RX<Pt>:AESummary[:EVENt]?

Syntax	E4:STATus:RX<Pt>:AESummary[:EVENt]?
Description	This query returns the E4 alarms and errors summary event register. The content of this event register is summarized in DB4 of the STATus:INTerface:PORT<Pt>:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 (2) = Error summary DB3 - DB16 = NOT USED
Example	`E4:STAT:RX1:AES?` → `1`
Note	

9.4.6 E4:STATus:RX<Pt>:AESummary:CONDition?

Syntax	E4:STATus:RX<Pt>:AESummary:CONDition?
Description	This query returns the E4 alarms and errors summary condition register. DB1 is summary of E4:STATus:RX<Pt>:ALARm[:EVENt]? DB2 is summary of E4:STATus:RX<Pt>:ERRor[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 (2) = Error summary DB3 - DB16 = NOT USED
Example	`E4:STAT:RX1:AES:COND?` → `2`
Note	

9.4.7 E4:STATus:RX<Pt>:ALARm[:EVENt]?

Syntax	E4:STATus:RX<Pt>:ALARm[:EVENt]?
Description	This query returns the alarms event register. The content of this register is summarized in DB1 of the E4:STATus:RX<Pt>:AESummary:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = No signal DB2 (2) = Alarm Indication Signal (AIS) DB3 (4) = No frame DB4 (8) = Distant DB5 (16) = No sync DB6 - DB16 = NOT USED
Example	`E4:STAT:RX1:ALAR?` → `2`
Note	The No frame and Distant are only valid when the PCM frame is enabled (E4:RX<Pt>:PCMFrame? → 1).

9.4.8 E4:STATus:RX<Pt>:ALARm:CONDition?

Syntax	E4:STATus:RX<Pt>:ALARm:CONDition?
Description	This query returns the alarms condition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = No signal DB2 (2) = Alarm Indication Signal (AIS) DB3 (4) = No frame DB4 (8) = Distant DB5 (16) = No sync DB6 - DB16 = NOT USED
Example	`E4:STAT:RX1:ALAR:COND?` → `4`
Note	The No frame and Distant are only valid when the PCM frame is enabled (E4:RX<Pt>:PCMFrame? → 1).

9.4.9 E4:STATus:RX<Pt>:ERRor[:EVENt]?

Syntax	E4:STATus:RX<Pt>:ERRor[:EVENt]?
Description	This query returns the errors event register. The content of this register is summarized in DB2 of the E4:STATus:RX<Pt>:AESummary:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = FAS words DB2 - DB3 = NOT USED DB4 (8) = Pattern DB5 (16) = Pattern slip DB6 - DB16 = NOT USED
Example	`E4:STAT:RX1:ERR?` → `8`
Note	The FAS words is only valid when PCM frame is enabled (E4:RX<Pt>:PCMFrame? → 1).

9.4.10 E4:STATus:RX<Pt>:ERRor:CONDition?

Syntax	E4:STATus:RX<Pt>:ERRor:CONDition?
Description	This query returns the errors condition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = FAS words DB2 - DB3 = NOT USED DB4 (8) = Pattern DB5 (16) = Pattern slip DB6 - DB16 = NOT USED
Example	`E4:STAT:RX1:ERR:COND?` → `8`
Notes	The FAS words is only valid when the PCM frame is enabled (E4:RX<Pt>:PCMFrame? → 1).

9.5 Results

9.5.1 E4:RX<Pt>:IFETch?

Syntax	E4:RX<Pt>:IFETch? <parameters>
Description	This query fetches an interval if available.
Parameters	<Pt> = Port number
	({<parameter>} + {,}∗) = <EXPRESSION PROGRAM DATA> The response format is listed for each parameter. **Alarms** NSIG: No signal. Response: <Seconds>,<Ratio> AIS: Alarm indication signal. Response: <Seconds>,<Ratio> NFR: No frame. Response: <Seconds>,<Ratio> DAL: Distant alarm. Response: <Seconds>,<Ratio> NSYN: No sync. Response: <Seconds>,<Ratio> **Errors** FASW: FAS words. Response: <Count>,<Ratio> PATT: PAttern. Response: <Count>,<Ratio> PSL: Pattern Slip. Response: <Count>,<Ratio> PBL: Pattern block. Response: <Count>,<Ratio> **Rx frequency** DEV: Frequency deviation. Response: <ppm> DIFF: Frequency difference (RX1-RX2). Response: <ppm> (Only for RX1, RX2 is always NaN (section 1.6.1)) **FAS performance errors** FES: FAS ES. Response: <Count>,<Ratio%> FSES: FAS SES. Response: <Count>,<Ratio%> FBBE: FAS BBE. Response: <Count>,<Ratio%> FALS: FAS ALS. Response: <Count>,<Ratio%> FUAT: FAS UAT. Response: <Count>,<Ratio%> FAVT: FAS AVT. Response: <Count>,<Ratio%> FEFS: FAS EFS. Response: <Count>,<Ratio%> **Pattern performance errors** PES: Pattern ES. Response: <Count>,<Ratio%> PSES: Pattern SES. Response: <Count>,<Ratio%> PBBE: Pattern BBE. Response: <Count>,<Ratio%> PALS: Pattern ALS. Response: <Count>,<Ratio%> PUAT: Pattern UAT. Response: <Count>,<Ratio%> PAVT: Pattern AVT. Response: <Count>,<Ratio%> PEFS: Pattern EFS. Response: <Count>,<Ratio%>
Response	{(<result>), }* = <EXPRESSION RESPONSE DATA> Expression format: Numeric List Each result is formatted according to the specification in the parameter field. Values that are not relevant or applicable for the current measurement return NaN (section 1.6.1).
Example	`E4:RX1:IFET? (FES,PATT)` → `(2,0.5),(4,0.25)`
Notes	This command fetches the result from the interval selected by the `MEASurement:SETup:SELect` command (see section 18.2.2). NFR, DAL, FASW, FASB will only return NaN (section 1.6.1), if the PCM frame is disabled (E4:RX<Pt>:PCMFrame? → 0). If requested result is not available, NaN (section 1.6.1) is returned. If there is one or more results, the last ”,” is always removed.

9.6 RTD

This section document commands to retrieve Round Trip Delay measurement results.
Commands for general RTD settings are described in section 17.1 on page 953.

9.6.1 E4:RTD:RX<Pt>:NUMBer?

Syntax	E4:RTD:RX<Pt>:NUMBer?
Description	This query returns the number of the RTD data.
Parameter	<Pt> = Port number
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`E4:RTD:RX1:NUMB?` → `2`
Note	

9.6.2 E4:RTD:RX<Pt>:ATIMe?

Syntax	E4:RTD:RX<Pt>:ATIMe?
Description	This query returns the average time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<avg> = <NR2 NUMERIC RESPONSE DATA>
Example	`E4:RTD:RX1:ATIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

9.6.3 E4:RTD:RX<Pt>:MTIMe?

Syntax	E4:RTD:RX<Pt>:MTIMe?
Description	This query returns the maximum time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
	<limit exceeded> = <NR1 NUMERIC RESPONSE DATA>
Example	`E4:RTD:RX1:MTIM?` → `1.0,0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

9.6.4 E4:RTD:RX<Pt>:LTIMe?

Syntax	E4:RTD:RX<Pt>:LTIMe?
Description	This query returns the least (minimum) time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`E4:RTD:RX1:LTIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

Chapter 10

SDH

Use `MEAS:SET:PORT<Pt>:TERM` to change the SDH/SONET terminology (see section 18.2.1).

10.1 Receiver

10.1.1 SDH:RX<Pt>:INTerface

Syntax	SDH:RX<Pt>:INTerface <mode>
Description	This command sets the used interface or switches off the receiver.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: Disables receiver ELECtrical: Electrical interface SFP: SFP/SFP+ optical interface CFP: CFP optical interface OPTical: Obsolete. For CMA 3000 backward compatibility only. Same as SFP *DEFault = OFF*
Response	None.
Example	`SDH:RX1:INT OFF`
Note	

Syntax	SDH:RX<Pt>:INTerface?
Description	This query returns the used interface.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SDH:RX1:INT?` → `OFF`
Note	Returns OTN when SDH over OTN.

10.1.2 SDH:RX<Pt>:STMLevel

Syntax	SDH:RX<Pt>:STMLevel <level>
Description	This command sets the STM level of the SDH receiver.
Parameters	<Pt> = Port number
	<level> = <CHARACTER PROGRAM DATA> 1: STM-1 signal. 4: STM-4 signal. 16: STM-16 signal. 64: STM-64 signal. 256: STM-256 signal. PRBSTRANS: Trans.(PRBS). NULLTRANS: Trans.(NULL). *DEFault = 1*
Response	None.
Example	`SDH:RX1:STML 1`
Note	Setting the STM level may change the VC-4 concatenation level.

Syntax	SDH:RX<Pt>:STMLevel?
Description	This query returns the STM level of the SDH receiver.
Parameter	<Pt> = Port number
Response	<level> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:RX1:STML?` → `1`
Note	

10.1.3 SDH:RX<Pt>:AUGRoup

Syntax	SDH:RX<Pt>:AUGRoup <aug>
Description	This command sets the AU Group(s) to be used in the signal structure.
Parameters	<Pt> = Port number
	<aug> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 768, DEFault = 1*
Response	None.
Example	`SDH:RX1:AUGR 1`
Note	The specified AUG cannot exceed the STM level divided by the VC-4 concatenation level.

Syntax	SDH:RX<Pt>:AUGRoup?
Description	This query returns AU Group(s) used in the signal structure.
Parameter	<Pt> = Port number
Response	<aug> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:RX1:AUGR?` → `1`
Note	

10.1.4 SDH:RX<Pt>:CONCatenation

Syntax	SDH:RX<Pt>:CONCatenation <level>
Description	This command sets the VC concatenation level.
Parameters	<Pt> = Port number
	<level> = <CHARACTER PROGRAM DATA> VC3 VC4 4C 16C 64C 256C *DEFault = VC4*
Response	None.
Example	`SDH:RX1:CONC VC4`
Note	Setting the VC concatenation level may change the STM level.

Syntax	SDH:RX<Pt>:CONCatenation?
Description	This query returns the VC concatenation level.
Parameter	<Pt> = Port number
Response	<level> = <CHARACTER RESPONSE DATA>
Example	`SDH:RX1:CONC?` → `VC4`
Note	

10.1.5 SDH:RX<Pt>:CONTainer

Syntax	SDH:RX<Pt>:CONTainer <type>
Description	This command sets the container type (container-n).
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> C3 C4 C11 C12 *DEFault = C4*
Response	None.
Example	`SDH:RX1:CONT C4`
Note	Setting the container type.

Syntax	SDH:RX<Pt>:CONTainer?
Description	This query returns the container type (container-n).
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`SDH:RX1:CONT? → 4C`
Note	

10.1.6 SDH:RX<Pt>:SSTRucture

Syntax	SDH:RX<Pt>:SSTRucture <structure>
Description	This command sets the expected SDH signal structure.
Parameters	<Pt> = Port number
	<structure> = <CHARACTER PROGRAM DATA> BULK: Bulk data in the VC-4. E1: E1 signal in a VC-12 in one or all AU-4's E3: E3 signal in a VC-3 in one or all AU-4's DS1: DS1 signal in a VC-11 in one or all AU-4's DS3: DS3 signal in a VC-3 in one or all AU-4's E4: E4 signal in a VC-4 in one or all AU-4's *DEFault = BULK*
Response	None.
Example	`SDH:RX1:SSTR BULK`
Note	Choosing any of the interfaces above requires the corresponding option to be present.

Syntax	SDH:RX<Pt>:SSTRucture?
Description	This query returns the expected SDH signal structure.
Parameter	<Pt> = Port number
Response	<structure> = <CHARACTER RESPONSE DATA>
Example	`SDH:RX1:SSTR? → BULK`
Note	

10.1.7 SDH:RX<Pt>:PATTern

Syntax	SDH:RX<Pt>:PATTern <pattern>
Description	This command sets the pattern to be expected as payload when SSTRucture is BULK.
Parameters	<Pt> = Port number
	<pattern> = <CHARACTER PROGRAM DATA> OFF USER32BIT: 32 bit user defined pattern. USER2048BIT: 2048 bit user defined pattern. PRBS9: PRBS-9 pattern. PRBS11: PRBS-11 pattern. PRBS15: PRBS-15 pattern. PRBS20: PRBS-20 pattern. PRBS23: PRBS23 pattern. PRBS29: PRBS-29 pattern. PRBS31: PRBS-31 pattern. ALL0: Pattern of all zeros. ALL1: Pattern of all ones. ALT11: Alternating 1:1 ALT13: Alternating 1:3 ALT17: Alternating 1:7 B2IN8: Random pattern with two ones for every eight bits. *DEFault = PRBS23*
Response	None.
Example	`SDH:RX1:PATT PRBS23`
Note	

Syntax	SDH:RX<Pt>:PATTern?
Description	This query returns the pattern to be expected as payload when SSTRucture is BULK.
Parameter	<Pt> = Port number
Response	<pattern> = <CHARACTER RESPONSE DATA>
Example	`SDH:RX1:PATT?` → `PRBS23`
Note	

10.1.8 SDH:RX<Pt>:UP16

Syntax	SDH:RX<Pt>:UP16 <pattern>
Description	This command sets the 16 bit user pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Valid characters: '0' and '1' The string must consist of 1 to 16 characters.
Response	None.
Example	`SDH:RX1:UP16 "101101"`
Note	This command is for backward compatibility only, and a query command is not available. Actually the new 32 bit user pattern is set by this command.

10.1.9 SDH:RX<Pt>:UP32

Syntax	SDH:RX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters (one bit resolution).
Response	None.
Examples	`SDH:RX1:UP32 "01101"`
Note	

Syntax	SDH:RX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`SDH:RX1:UP32?` → `"01101"`
Note	

10.1.10 SDH:RX<Pt>:UP2K

Syntax	SDH:RX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`SDH:RX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `SDH:RX1:PATT` is `USER2048BIT`

Syntax	SDH:RX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`SDH:RX1:UP2K?` → `"12DF"`
Note	

10.1.11 SDH:RX<Pt>:PINVersion

Syntax	SDH:RX<Pt>:PINVersion <inverted>
Description	This command enables or disables PRBS pattern inversion (when SSTRucture is BULK).
Parameters	<Pt> = Port number
	<inverted> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`SDH:RX1:PINV OFF`
Note	

Syntax	SDH:RX<Pt>:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the PRBS pattern.
Parameter	<Pt> = Port number
Response	<inverted> = <BOOLEAN RESPONSE DATA>
Example	`SDH:RX1:PINV?` → `0`
Note	

10.1.12 SDH:RX<Pt>:TCM

Syntax	SDH:RX<Pt>:TCM <mode>
Description	This command sets the Tandem Connection Monitoring mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: Off VC4: N1 (VC-4) VC3: N1 (VC-3) VC12: N2 (VC-12) VC11: N2 (VC-12) *DEFault = OFF*
Response	None.
Example	`SDH:RX1:TCM OFF`
Note	This command requires the TCM option to be present.

Syntax	SDH:RX<Pt>:TCM?
Description	This query returns Tandem Connection Monitoring mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SDH:RX1:TCM? → OFF`
Note	

10.1.13 SDH:RX<Pt>:TUG3

Syntax	SDH:RX<Pt>:TUG3 <id>
Description	This command sets the TUG-3 number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: TUG-3 number one. 2: TUG-3 number two. 3: TUG-3 number three. *MINimum = 1, MAXimum = 3, DEFault = 1*
Response	None.
Example	`SDH:RX1:TUG3 1`
Note	This value influences the channel number.

Syntax	SDH:RX<Pt>:TUG3?
Description	This query returns TUG3 number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:RX1:TUG3? → 1`
Note	

10.1.14 SDH:RX<Pt>:TUG2

Syntax	SDH:RX<Pt>:TUG2 <id>
Description	This command sets the TUG-2 number for signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: TUG-2 number one. 2: TUG-2 number two. ... 7: TUG-2 number seven. *MINimum = 1, MAXimum = 7, DEFault = 1*
Response	None.
Example	`SDH:RX1:TUG2 1`
Note	This value influences the channel number.

Syntax	SDH:RX<Pt>:TUG2?
Description	This query returns TUG2 number for signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:RX1:TUG2?` → `1`
Note	

10.1.15 SDH:RX<Pt>:TU12

Syntax	SDH:RX<Pt>:TU12 <id>
Description	This command sets the TU-12 number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: TU-12 number one. 2: TU-12 number two. 3: TU-12 number three. *MINimum = 1, MAXimum = 3, DEFault = 1*
Response	None.
Example	`SDH:RX1:TU12 1`
Note	This value influences the channel number.

Syntax	SDH:RX<Pt>:TU12?
Description	This query returns TU-12 number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:RX1:TU12?` → `1`
Note	

10.1.16 SDH:RX<Pt>:TU11

Syntax	SDH:RX<Pt>:TU11 <id>
Description	This command sets the TU-11 number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: TU-11 number one. 2: TU-11 number two. 3: TU-11 number three. 4: TU-11 number four. *MINimum = 1, MAXimum = 4, DEFault = 1*
Response	None.
Example	`SDH:RX1:TU11 1`
Note	This value influences the channel number.

Syntax	SDH:RX<Pt>:TU11?
Description	This query returns TU-11 number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:RX1:TU11?` → `1`
Note	

10.1.17 SDH:RX<Pt>:CHANnel

Syntax	SDH:RX<Pt>:CHANnel <id>
Description	This command sets the channel number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=84, DEFault = 1*
Response	None.
Example	`SDH:RX1:CHAN 1`
Note	This value influences the TUG-3, TUG-2, TU-12 and TU-11 numbers.

Syntax	SDH:RX<Pt>:CHANnel?
Description	This query returns the channel number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:RX1:CHAN? → 1`
Note	

10.1.18 SDH:RX<Pt>:MAPPing

Syntax	SDH:RX<Pt>:MAPPing <type>
Description	This command sets the mapping type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> SYNChronous: Synchronous mapping. ASYNchronous: Asynchronous mapping. *DEFault = SYNC*
Response	None.
Example	`SDH:RX1:MAPP SYNC`
Note	

Syntax	SDH:RX<Pt>:MAPPing?
Description	This query returns returns the mapping type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`SDH:RX1:MAPP? → SYNC`
Note	

10.1.19 SDH:RX<Pt>:GAIN

Syntax	SDH:RX<Pt>:GAIN <type>
Description	This command sets the receiver gain.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> TERMinate: Frequency dependent AGC. MONitor: Frequency linear AGC. *DEFault = TERMinate*
Response	None.
Example	`SDH:RX1:GAIN TERM`
Note	

Syntax	SDH:RX<Pt>:GAIN?
Description	This query returns the receiver gain.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`SDH:RX1:GAIN? → TERM`
Note	

10.1.20 SDH:RX<Pt>:FOLLow

Syntax	SDH:RX<Pt>:FOLLow <mode>
Description	This command sets the receiver setup to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> NONE: Do not follow. TX: Follows setup of the Tx port. RX1: Follows setup of the Rx port1. *DEFault = NONE*
Response	None.
Example	`SDH:RX1:FOLL NONE`
Note	

Syntax	SDH:RX<Pt>:FOLLow?
Description	This query returns the receiver setup to follow another setup or not to follow.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SDH:RX1:FOLL?` → `NONE`
Note	

10.1.21 SDH:RX<Pt>:MEASurement:HPLM

Syntax	SDH:RX<Pt>:MEASurement:HPLM <enable>
Description	This command enables or disables HP-PLM measurement.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SDH:RX1:MEAS:HPLM ON`
Note	

Syntax	SDH:RX<Pt>:MEASurement:HPLM?
Description	This query returns whether or not HP-PLM measurement is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`SDH:RX1:MEAS:HPLM?` → `1`
Note	

10.1.22 SDH:RX<Pt>:MEASurement:LPLM

Syntax	SDH:RX<Pt>:MEASurement:LPLM <enable>
Description	This command enables or disables LP-PLM measurement.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SDH:RX1:MEAS:LPLM ON`
Note	

Syntax	SDH:RX<Pt>:MEASurement:LPLM?
Description	This query returns whether or not LP-PLM measurement is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`SDH:RX1:MEAS:LPLM?` → `1`
Note	

10.2 Transmitter

10.2.1 SDH:TX<Pt>:INTerface

Syntax	SDH:TX<Pt>:INTerface <mode>
Description	This command sets the used interface of the transmitter.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> ELECtrical: Electrical interface SFP: SFP/SFP+ optical interface CFP: CFP optical interface *DEFault = ELEC*
Response	None.
Example	`SDH:TX1:INT SFP`
Note	

Syntax	SDH:TX<Pt>:INTerface?
Description	This query returns the used interface.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SDH:TX1:INT?` → `SFP`
Note	Returns OTN when SDH over OTN.

10.2.2 SDH:TX<Pt>[:ENABle]

Syntax	SDH:TX<Pt>[:ENABle] <mode>
Description	This command sets the mode of the transmitter.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: Turn off the transmitter. NORMal: Normal transmission mode using the configured signal. THRough: Through mode. The signal from the receiver is transmitted. OTHRough: OH overwrite through mode. THA: Obsolete. For CMA 3000 backward compatibility only. Same as THRough. *DEFault = OFF*
Response	None.
Example	`SDH:TX1 NORM`
Note	

Syntax	SDH:TX<Pt>[:ENABle]?
Description	This query returns the mode of the transmitter.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SDH:TX1?` → `NORM`
Note	

10.2.3 SDH:TX<Pt>:OPTical

Syntax	SDH:TX<Pt>:OPTical <mode>
Description	Obsolete. For CMA 3000 backward compatibility only. Same as SDH:TX<Pt>:INT SFP and SDH:TX<Pt>
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA>
Response	None.
Example	`SDH:TX1:OPT NORM`
Note	Compared to CMA 3000 there is no longer a suffix on the last node.

Syntax	SDH:TX<Pt>:OPTical?
Description	Obsolete. For CMA 3000 backward compatibility only. Same as SDH:TX<Pt>?
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SDH:TX1:OPT?` → `NORM`
Note	

10.2.4 SDH:TX<Pt>:ELECtrical

Syntax	SDH:TX<Pt>:ELECtrical <mode>
Description	Obsolete. For CMA 3000 backward compatibility only. Same as SDH:TX<Pt>:INT ELEC and SDH:TX<Pt>
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA>
Response	None.
Example	`SDH:TX1:ELEC NORM`
Note	Compared to CMA 3000 mode=THB is no longer supported.

Syntax	SDH:TX<Pt>:ELECtrical?
Description	Obsolete. For CMA 3000 backward compatibility only. Same as SDH:TX<Pt>?
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SDH:TX1:ELEC?` → `NORM`
Note	

10.2.5 SDH:TX<Pt>:STMLevel

Syntax	SDH:TX<Pt>:STMLevel <level>
Description	This command sets the STM level of the transmitted SDH signal.
Parameters	<Pt> = Port number
	<level> = <CHARACTER PROGRAM DATA> 1: STM-1 signal. 4: STM-4 signal. 16: STM-16 signal. 64: STM-64 signal. 256: STM-256 signal. PRBSTRANS: Trans.(PRBS). NULLTRANS: Trans.(NULL). *DEFault = 1*
Response	None.
Example	`SDH:TX1:STML 1`
Note	Setting the STM level may change the VC-4 concatenation level.

Syntax	SDH:TX<Pt>:STMLevel?
Description	This query returns the STM level of the transmitted SDH signal.
Parameter	<Pt> = Port number
Response	<level> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:TX1:STML?` → `1`
Note	

10.2.6 SDH:TX<Pt>:AUGRoup

Syntax	SDH:TX<Pt>:AUGRoup <value>
Description	This command sets the AUG where the test signal is inserted.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> ALL 1 to 768 *DEFault = ALL*
Response	None.
Example	`SDH:TX1:AUGR ALL`
Note	The specified AUG cannot exceed the STM level divided by the VC-4 concatenation level.

Syntax	SDH:TX<Pt>:AUGRoup?
Description	This query returns the AUG where the test signal has been inserted.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:TX1:AUGR? → ALL`
Note	

10.2.7 SDH:TX<Pt>:CONCatenation

Syntax	SDH:TX<Pt>:CONCatenation <level>
Description	This command sets the VC concatenation level.
Parameters	<Pt> = Port number
	<level> = <CHARACTER PROGRAM DATA> VC3 VC4 4C 16C 64C 256C *DEFault = VC4*
Response	None.
Example	`SDH:TX1:CONC 4C`
Note	Setting the VC concatenation level may change the STM level.

Syntax	SDH:TX<Pt>:CONCatenation?
Description	This query returns the VC concatenation level.
Parameter	<Pt> = Port number
Response	<level> = <CHARACTER RESPONSE DATA>
Example	`SDH:TX1:CONC? → 4C`
Note	

10.2.8 SDH:TX<Pt>:CONTainer

Syntax	SDH:TX<Pt>:CONTainer <type>
Description	This command sets the container type (container-n).
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> C3 C4 C11 C12 *DEFault = C4*
Response	None.
Example	`SDH:TX1:CONT C4`
Note	Setting the container type.

Syntax	SDH:TX<Pt>:CONTainer?
Description	This query returns the container type (container-n).
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`SDH:TX1:CONT? → 4C`
Note	

10.2.9 SDH:TX<Pt>:SSTRucture

Syntax	SDH:TX<Pt>:SSTRucture <structure>
Description	This command sets the SDH signal structure.
Parameters	<Pt> = Port number
	<structure> = <CHARACTER PROGRAM DATA> BULK: Bulk data in the VC-4. E1: E1 signal in a VC-12 in one or all AU-4's E3: E3 signal in a VC-3 in one or all AU-4's DS1: DS1 signal in a VC-11 in one or all AU-4's DS3: DS3 signal in a VC-3 in one or all AU-4's E4: E4 signal in a VC-4 in one or all AU-4's *DEFault = BULK*
Response	None.
Example	`SDH:TX1:SSTR BULK`
Note	Choosing any of the interfaces above requires the corresponding option to be present.

Syntax	SDH:TX<Pt>:SSTRucture?
Description	This query returns the SDH signal structure.
Parameter	<Pt> = Port number
Response	<structure> = <CHARACTER RESPONSE DATA>
Example	`SDH:TX1:SSTR? → BULK`
Note	

10.2.10 SDH:TX<Pt>:PATTern

Syntax	SDH:TX<Pt>:PATTern <pattern>
Description	This command sets the pattern to be used as payload when SSTRucture is BULK.
Parameters	<Pt> = Port number
	<pattern> = <CHARACTER PROGRAM DATA> OFF USER32BIT: 32 bit user defined pattern. USER2048BIT: 2048 bit user defined pattern. PRBS9: PRBS-9 pattern. PRBS11: PRBS-11 pattern. PRBS15: PRBS-15 pattern. PRBS20: PRBS-20 pattern. PRBS23: PRBS23 pattern. PRBS29: PRBS-29 pattern. PRBS31: PRBS-31 pattern. ALL0: Pattern of all zeros. ALL1: Pattern of all ones. ALT11: Alternating 1:1 ALT13: Alternating 1:3 ALT17: Alternating 1:7 B2IN8: Random pattern with two ones for every eight bits. *DEFault = PRBS23*
Response	None.
Example	`SDH:TX1:PATT PRBS23`
Note	

Syntax	SDH:TX<Pt>:PATTern?
Description	Queries the pattern to be used as payload when SSTRucture is BULK.
Parameter	<Pt> = Port number
Response	<pattern> = <CHARACTER RESPONSE DATA>
Example	`SDH:TX1:PATT?` → `PRBS23`
Note	

10.2.11 SDH:TX<Pt>:UP16

Syntax	SDH:TX<Pt>:UP16 <pattern>
Description	This command sets the 16 bit user pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Valid characters: '0' and '1' The string must consist of 1 to 16 characters.
Response	None.
Example	`SDH:TX1:UP16 "101101"`
Note	This command is for backward compatibility only, and a query command is not available. Actually the new 32 bit user pattern is set by this command.

10.2.12 SDH:TX<Pt>:UP32

Syntax	SDH:TX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters (one bit resolution).
Response	None.
Examples	`SDH:TX1:UP32 "01101"`
Note	

Syntax	SDH:TX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`SDH:TX1:UP32?` → `"01101"`
Note	

10.2.13 SDH:TX<Pt>:UP2K

Syntax	SDH:TX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`SDH:TX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `SDH:TX1:PATT` is `USER2048BIT`

Syntax	SDH:TX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`SDH:TX1:UP2K?` → `"12DF"`
Note	

10.2.14 SDH:TX<Pt>:PINVersion

Syntax	SDH:TX<Pt>:PINVersion <inverted>
Description	This command enables or disables PRBS pattern inversion (when SSTRucture is BULK).
Parameters	<Pt> = Port number
	<inverted> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`SDH:TX1:PINV OFF`
Note	

Syntax	SDH:TX<Pt>:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the PRBS pattern.
Parameter	<Pt> = Port number
Response	<inverted> = <BOOLEAN RESPONSE DATA>
Example	`SDH:TX1:PINV?` → `0`
Note	

10.2.15 SDH:TX<Pt>:TIMing

Syntax	SDH:TX<Pt>:TIMing <source>
Description	This command sets the timing source.
Parameters	<Pt> = Port number
	<source> = <CHARACTER PROGRAM DATA> INTernal: Internal clock. EXTernal: External clock. RX: Received Rx signal clock. *DEFault = INT*
Response	None.
Example	`SDH:TX1:TIM INT`
Note	

Syntax	SDH:TX<Pt>:TIMing?
Description	This query returns the timing source.
Parameter	<Pt> = Port number
Response	<source> = <CHARACTER RESPONSE DATA>
Example	`SDH:TX1:TIM? → INT`
Note	

10.2.16 SDH:TX<Pt>:TCM

Syntax	SDH:TX<Pt>:TCM <mode>
Description	This command sets the TCM mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: Off VC4: N1 (VC-4) VC3: N1 (VC-3) VC12: N2 (VC-12) VC11: N2 (VC-12) *DEFault = OFF*
Response	None.
Example	`SDH:TX1:TCM OFF`
Note	

Syntax	SDH:TX<Pt>:TCM?
Description	Queries the TCM mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SDH:TX1:TCM? → OFF`
Note	

10.2.17 SDH:TX<Pt>:TUG3

Syntax	SDH:TX<Pt>:TUG3 <id>
Description	This command sets the TUG-3 number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: TUG-3 number one. 2: TUG-3 number two. 3: TUG-3 number three. *MINimum = 1, MAXimum = 3, DEFault = 1*
Response	None.
Example	`SDH:TX1:TUG3 1`
Note	This value influences the channel number.

Syntax	SDH:TX<Pt>:TUG3?
Description	This query returns TUG3 number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:TX1:TUG3?` → `1`
Note	

10.2.18 SDH:TX<Pt>:TUG2

Syntax	SDH:TX<Pt>:TUG2 <id>
Description	This command sets the TUG-2 number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: TUG-2 number one. 2: TUG-2 number two. ... 7: TUG-2 number seven. *MINimum = 1, MAXimum = 7, DEFault = 1*
Response	None.
Example	`SDH:TX1:TUG2 1`
Note	This value influences the channel number.

Syntax	SDH:TX<Pt>:TUG2?
Description	This query returns TUG2 number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:TX1:TUG2?` → `1`
Note	

10.2.19 SDH:TX<Pt>:TU12

Syntax	SDH:TX<Pt>:TU12 <id>
Description	This command sets the TU-12 number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: TU-12 number one. 2: TU-12 number two. 3: TU-12 number three. *MINimum = 1, MAXimum = 3, DEFault = 1*
Response	None.
Example	`SDH:TX1:TU12 1`
Note	This value influences the channel number.

Syntax	SDH:TX<Pt>:TU12?
Description	This query returns TU-12 number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:TX1:TU12?` → `1`
Note	

10.2.20 SDH:TX<Pt>:TU11

Syntax	SDH:TX<Pt>:TU11 <id>
Description	This command sets the TU-11 number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: TU-11 number one. 2: TU-11 number two. 3: TU-11 number three. 4: TU-11 number four. *MINimum = 1, MAXimum = 4, DEFault = 1*
Response	None.
Example	`SDH:TX1:TU11 1`
Note	This value influences the channel number.

Syntax	SDH:TX<Pt>:TU11?
Description	This query returns TU-11 number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:TX1:TU11? → 1`
Note	

10.2.21 SDH:TX<Pt>:CHANnel

Syntax	SDH:TX<Pt>:CHANnel <id>
Description	This command sets the channel number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=84, DEFault = 1*
Response	None.
Example	`SDH:TX1:CHAN 1`
Note	The value influences the TUG-3, TUG-2 and TU-12 numbers.

Syntax	SDH:TX<Pt>:CHANnel?
Description	This query returns the channel number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:TX1:CHAN? → 1`
Note	

10.2.22 SDH:TX<Pt>:MAPPing

Syntax	SDH:TX<Pt>:MAPPing <type>
Description	This command sets the mapping type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> SYNChronous: Synchronous mapping. ASYNchronous: Asynchronous mapping. *DEFault = SYNC*
Response	None.
Example	`SDH:TX1:MAPP SYNC`
Note	

Syntax	SDH:TX<Pt>:MAPPing?
Description	This query returns the mapping type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`SDH:TX1:MAPP?` → `SYNC`
Note	

10.2.23 SDH:TX<Pt>:SOH:DEFault

Syntax	SDH:TX<Pt>:SOH:DEFault
Description	This command sets all STM SOH bytes to their default value.
Parameter	<Pt> = Port number
Response	None.
Example	`SDH:TX1:SOH:DEF`
Note	There is no query version of this command.

10.2.24 SDH:TX<Pt>:SOH:TRACe

Syntax	SDH:TX<Pt>:SOH:TRACe <string>[,<idlechar>]
Description	This command sets the SOH trace (J0) to the specified string.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: The string to be used as section trace string. *DEFault = "Message_Test_J0"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SDH:TX1:SOH:TRAC "Message_Test_J0",#H20`
Note	If the entered trace string is more than 15 characters long, the string will be truncated.

Syntax	SDH:TX<Pt>:SOH:TRACe?
Description	This query returns the SOH trace for string and idle char.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:TX1:SOH:TRAC?` → `"Message_Test_J0",#H20`
Note	

10.2.25 SDH:TX<Pt>:SOH:TRACe:CRC

Syntax	SDH:TX<Pt>:SOH:TRACe:CRC <mode>
Description	This command sets the SOH trace (J0) CRC mode (OFF/ON).
Parameters	<Pt> = Port number
	<mode> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SDH:TX1:SOH:TRAC:CRC ON`
Note	

Syntax	SDH:TX<Pt>:SOH:TRACe:CRC?
Description	This query returns the mode of the SOH trace (J0) CRC.
Parameter	<Pt> = Port number
Response	<mode> = <BOOLEAN RESPONSE DATA>
Example	`SDH:TX1:SOH:TRAC:CRC?` → `1`
Note	

10.2.26 SDH:TX<Pt>:SOH

Syntax	SDH:TX<Pt>:SOH <SOH-byte>,<value1>[,<value2>[,<value3>]]
Description	This command sets the value of the specified bytes in the SOH.
Parameters	<Pt> = Port number
	<SOH-byte> = <CHARACTER PROGRAM DATA> A1: 3 bytes. A2: 3 bytes. J0: 3 bytes. Note: The defined trace string will be disabled. B1: 2 bytes. <value3> is ignored. E1: 3 bytes. F1: 3 bytes. D1: 3 bytes. D2: 3 bytes. D3: 3 bytes. K1: 3 bytes. K2: 3 bytes. D4: 3 bytes. D5: 3 bytes. D6: 3 bytes. D7: 3 bytes. D8: 3 bytes. D9: 3 bytes. D10: 3 bytes. D11: 3 bytes. D12: 3 bytes. S1: 3 bytes. Z2: 3 byte. E2: 3 bytes. M0: 1 byte. <value2> and <value3> are ignored. [1] M1: 1 byte. <value2> and <value3> are ignored.
	<value1> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
	<value2> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
	<value3> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
Response	None.

Examples	`SDH:TX1:SOH A1,0,0,0`
	`SDH:TX1:SOH D7,253,32,26`
	`SDH:TX1:SOH B1,4,5`
	`SDH:TX1:SOH M1,0`

Note	[1] Only valid for SDH:TX<Pt>:STMLevel = 64 or 256.

Syntax	SDH:TX<Pt>:SOH? <SOH-byte>
Description	This query returns the value of the specified bytes in the SOH.
Parameters	<Pt> = Port number
	<SOH-byte> = <CHARACTER PROGRAM DATA>
Response	<value1> = <HEXADECIMAL NUMERIC RESPONSE DATA> [,<value2> = <HEXADECIMAL NUMERIC RESPONSE DATA> [,<value3> = <HEXADECIMAL NUMERIC RESPONSE DATA>]]

Examples	`SDH:TX1:SOH? A1` → `#H00,#H00,#H00`
	`SDH:TX1:SOH? D7` → `#HFD,#H20,#H1A`
	`SDH:TX1:SOH? B1` → `#H04,#H05`
	`SDH:TX1:SOH? M1` → `#H00`

Note	

10.2.27 SDH:TX<Pt>:VC4:POH:DEFault

Syntax	SDH:TX<Pt>:VC4:POH:DEFault
Description	This command sets all VC-4 path overhead bytes to their default value.
Parameter	<Pt> = Port number
Response	None.
Example	`SDH:TX1:VC4:POH:DEF`
Note	There is no query version of this command.

10.2.28 SDH:TX<Pt>:VC4:POH:TRACe

Syntax	SDH:TX<Pt>:VC4:POH:TRACe <string>[,<idlechar>]
Description	This command sets the VC-4 path trace (J1) to the specified string and its idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: The string to be used as path trace string. *DEFault = "Message_Test_J1"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SDH:TX1:VC4:POH:TRAC "Message_Test_J1",#H20`
Note	If the entered trace string is more than 15 characters long, the string will be truncated.

Syntax	SDH:TX<Pt>:VC4:POH:TRACe?
Description	This query returns the VC-4 path trace text and idle char.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:TX1:VC4:POH:TRAC?` → `"Message_Test_J1",#H20`
Note	

10.2.29 SDH:TX<Pt>:VC4:POH:TRACe:CRC

Syntax	SDH:TX<Pt>:VC4:POH:TRACe:CRC <mode>
Description	This command sets the section overhead trace (J1) CRC mode (OFF/ON).
Parameters	<Pt> = Port number
	<mode> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SDH:TX1:VC4:POH:TRAC:CRC ON`
Note	

Syntax	SDH:TX<Pt>:VC4:POH:TRACe:CRC?
Description	This query returns the mode of the overhead trace (J1) CRC.
Parameter	<Pt> = Port number
Response	<mode> = <BOOLEAN RESPONSE DATA>
Example	`SDH:TX1:VC4:POH:TRAC:CRC?` → `1`
Note	

10.2.30 SDH:TX<Pt>:VC4:POH:TTCM

Syntax	SDH:TX<Pt>:VC4:POH:TTCM <string>[,<idlechar>]
Description	This command sets the VC-4 TCM trace (N1) to the specified string and its idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Apid_TCM_N1"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SDH:TX1:VC4:POH:TTCM "Apid_TCM_N1",#H20`
Note	

Syntax	SDH:TX<Pt>:VC4:POH:TTCM?
Description	This query returns the VC-4 TCM trace text and idle char.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:TX1:VC4:POH:TTCM?` → `"Apid_TCM_N1",#H20`
Note	

10.2.31 SDH:TX<Pt>:VC4:POH

Syntax	SDH:TX<Pt>:VC4:POH <POH-byte>,<value>
Description	This command sets the value of the specified byte in the VC-4 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA> C2: C2 byte. G1: G1 byte. F2: F2 byte. H4: H4 byte. F3: F3 byte. K3: K3 byte. N1: N1 byte.
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
Response	None.
Example	`SDH:TX1:VC4:POH C2,0`
Note	

Syntax	SDH:TX<Pt>:VC4:POH? <POH-byte>
Description	This query returns the value of the specified byte in the VC-4 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA>
Response	<value> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Examples	`SDH:TX1:VC4:POH? C2` → `#H00` `SDH:TX1:VC4:POH? H4` → `#HFF`
Note	

10.2.32 SDH:TX<Pt>:VC3:POH:DEFault

Syntax	SDH:TX<Pt>:VC3:POH:DEFault
Description	This command sets all VC-3 path overhead bytes to their default value.
Parameter	<Pt> = Port number
Response	None.
Example	`SDH:TX1:VC3:POH:DEF`
Note	There is no query version of this command.

10.2.33 SDH:TX<Pt>:VC3:POH:TRACe

Syntax	SDH:TX<Pt>:VC3:POH:TRACe <string>[,<idlechar>]
Description	This command sets the VC-3 path trace (J1) to the specified string.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Message_Test_J1"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SDH:TX1:VC3:POH:TRAC "Message_Test_J1",#H20`
Note	

Syntax	SDH:TX<Pt>:VC3:POH:TRACe?
Description	This query returns the VC-3 path trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:TX1:VC3:POH:TRAC?` → `"Message_Test_J1",#H20`
Note	

10.2.34 SDH:TX<Pt>:VC3:POH:TRACe:CRC

Syntax	SDH:TX<Pt>:VC3:POH:TRACe:CRC <mode>
Description	This command sets the section overhead trace (J1) CRC mode (OFF/ON).
Parameters	<Pt> = Port number
	<mode> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SDH:TX1:VC3:POH:TRAC:CRC ON`
Note	

Syntax	SDH:TX<Pt>:VC3:POH:TRACe:CRC?
Description	This query returns the mode of the overhead trace (J1) CRC.
Parameter	<Pt> = Port number
Response	<mode> = <BOOLEAN RESPONSE DATA>
Example	`SDH:TX1:VC3:POH:TRAC:CRC?` → `1`
Note	

10.2.35 SDH:TX<Pt>:VC3:POH:TTCM

Syntax	SDH:TX<Pt>:VC3:POH:TTCM <string>[,<idlechar>]
Description	This command sets the VC-3 TCM trace (N1) to the specified string, and the idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Apid_TCM_N1"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SDH:TX1:VC3:POH:TTCM "Apid_TCM_N1",#H20`
Note	

Syntax	SDH:TX<Pt>:VC3:POH:TTCM?
Description	This query returns the VC-3 TCM trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:TX1:VC3:POH:TTCM? → "Apid_TCM_N1",#H20`
Note	

10.2.36 SDH:TX<Pt>:VC3:POH

Syntax	SDH:TX<Pt>:VC3:POH <POH-byte>,<value>
Description	This command sets the value of the specified byte in the VC-3 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA> C2: C2 byte. G1: G1 byte. F2: F2 byte. H4: H4 byte. F3: F3 byte. K3: K3 byte. N1: N1 byte. Only available when there is no TCM.
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
Response	None.
Example	`SDH:TX1:VC3:POH C2,0`
Note	

Syntax	SDH:TX<Pt>:VC3:POH? <POH-byte>
Description	This query returns the value of the specified byte in the VC-3 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA>
Response	<value> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:TX1:VC3:POH? C2 → #H00`
Note	

10.2.37 SDH:TX<Pt>:VC12:POH:DEFault

Syntax	SDH:TX<Pt>:VC12:POH:DEFault
Description	This command sets all VC-12 path overhead bytes to their default value.
Parameter	<Pt> = Port number
Response	None.
Example	`SDH:TX1:VC12:POH:DEF`
Note	There is no query version of this command.

10.2.38 SDH:TX<Pt>:VC12:POH:TRACe

Syntax	SDH:TX<Pt>:VC12:POH:TRACe <string>[,<idlechar>]
Description	This command sets the VC-12 path trace (J2) to the specified string and the idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Message_Test_J2"*
	<idlechar> = <NUMERIC PROGRAM DATA>
Response	None.
Example	`SDH:TX1:VC12:POH:TRAC "Message_Test_J2",#H20`
Note	

Syntax	SDH:TX<Pt>:VC12:POH:TRACe?
Description	This query returns the VC-12 path trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:TX1:VC12:POH:TRAC?` → `"Message_Test_J2",#H20`
Note	

10.2.39 SDH:TX<Pt>:VC12:POH:TRACe:CRC

Syntax	SDH:TX<Pt>:VC12:POH:TRACe:CRC <mode>
Description	This command sets the section overhead trace (J2) CRC mode (OFF/ON).
Parameters	<Pt> = Port number
	<mode> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SDH:TX1:VC12:POH:TRAC:CRC ON`
Note	

Syntax	SDH:TX<Pt>:VC12:POH:TRACe:CRC?
Description	This query returns the mode of the overhead trace (J2) CRC.
Parameter	<Pt> = Port number
Response	<mode> = <BOOLEAN RESPONSE DATA>
Example	`SDH:TX1:VC12:POH:TRAC:CRC?` → `1`
Note	

10.2.40 SDH:TX<Pt>:VC12:POH:TTCM

Syntax	SDH:TX<Pt>:VC12:POH:TTCM <string>[,<idlechar>]
Description	This command sets the VC-12 TCM trace (N2) to the specified string and the idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Apid_TCM_N2"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SDH:TX1:VC12:POH:TTCM "Apid_TCM_N2",#H20`
Note	

Syntax	SDH:TX<Pt>:VC12:POH:TTCM?
Description	This query returns the VC-12 trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:TX1:VC12:POH:TTCM?` → `"Apid_TCM_N2",#H20`
Note	

10.2.41 SDH:TX<Pt>:VC12:POH

Syntax	SDH:TX<Pt>:VC12:POH <POH-byte>,<value>
Description	This command sets the value of the specified byte in the VC-12 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA> V5: V5 byte. N2: N2 byte. K4: K4 byte.
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
Response	None.
Example	`SDH:TX1:VC12:POH V5,#H04`
Note	

Syntax	SDH:TX<Pt>:VC12:POH? <POH-byte>
Description	This query returns the value of the specified byte in the VC-11 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA>
Response	<value> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:TX1:VC12:POH? V5` → `#H04`
Note	

10.2.42 SDH:TX<Pt>:VC11:POH:DEFault

Syntax	SDH:TX<Pt>:VC11:POH:DEFault
Description	This command sets all VC-11 path overhead bytes to their default value.
Parameter	<Pt> = Port number
Response	None.
Example	`SDH:TX1:VC11:POH:DEF`
Note	There is no query version of this command.

10.2.43 SDH:TX<Pt>:VC11:POH:TRACe

Syntax	SDH:TX<Pt>:VC11:POH:TRACe <string>[,<idlechar>]
Description	This command sets the VC-11 path trace (J2) to the specified string and the idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Message_Test_J2"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SDH:TX1:VC11:POH:TRAC "Message_Test_J2",#H20`
Note	

Syntax	SDH:TX<Pt>:VC11:POH:TRACe?
Description	This query returns the VC-11 path trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:TX1:VC11:POH:TRAC?` → `"Message_Test_J2",#H20`
Note	

10.2.44 SDH:TX<Pt>:VC11:POH:TRACe:CRC

Syntax	SDH:TX<Pt>:VC11:POH:TRACe:CRC <mode>
Description	This command sets the section overhead trace (J2) CRC mode (OFF/ON).
Parameters	<Pt> = Port number
	<mode> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SDH:TX1:VC11:POH:TRAC:CRC ON`
Note	

Syntax	SDH:TX<Pt>:VC11:POH:TRACe:CRC?
Description	This query returns the mode of the overhead trace (J2) CRC.
Parameter	<Pt> = Port number
Response	<mode> = <BOOLEAN RESPONSE DATA>
Example	`SDH:TX1:VC11:POH:TRAC:CRC?` → `1`
Note	

10.2.45 SDH:TX<Pt>:VC11:POH:TTCM

Syntax	SDH:TX<Pt>:VC11:POH:TTCM <string>[,<idlechar>]
Description	This command sets the VC-11 TCM trace (N2) to the specified string and the idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Apid_TCM_N2"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SDH:TX1:VC11:POH:TTCM "Apid_TCM_N2",#H20`
Note	

Syntax	SDH:TX<Pt>:VC11:POH:TTCM?
Description	This query returns the VC-11 trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:TX1:VC11:POH:TTCM?` → `"Apid_TCM_N2",#H20`
Note	

10.2.46 SDH:TX<Pt>:VC11:POH

Syntax	SDH:TX<Pt>:VC11:POH <POH-byte>,<value>
Description	This command sets the value of the specified byte in the VC-11 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA> V5: V5 byte. N2: N2 byte. K4: K4 byte.
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
Response	None.
Example	`SDH:TX1:VC11:POH V5,#H04`
Note	

Syntax	SDH:TX<Pt>:VC11:POH? <POH-byte>
Description	This query returns the value of the specified byte in the VC-11 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA>
Response	<value> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:TX1:VC11:POH? V5` → `#H04`
Note	

10.2.47 SDH:TX<Pt>:AU3:VC3:POH:DEFault

Syntax	SDH:TX<Pt>:AU3:VC3:POH:DEFault
Description	This command sets all VC-3 path overhead bytes to their default value.
Parameter	<Pt> = Port number
Response	None.
Example	`SDH:TX1:AU3:VC3:POH:DEF`
Note	There is no query version of this command.

10.2.48 SDH:TX<Pt>:AU3:VC3:POH:TRACe

Syntax	SDH:TX<Pt>:AU3:VC3:POH:TRACe <string>[,<idlechar>]
Description	This command sets the VC-3 path trace (J1) to the specified string.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Message_Test_J1"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SDH:TX1:AU3:VC3:POH:TRAC "Message_Test_J1",#H20`
Note	

Syntax	SDH:TX<Pt>:AU3:VC3:POH:TRACe?
Description	This query returns the VC-3 path trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:TX1:AU3:VC3:POH:TRAC?` → `"Message_Test_J1",#H20`
Note	

10.2.49 SDH:TX<Pt>:AU3:VC3:POH:TRACe:CRC

Syntax	SDH:TX<Pt>:AU3:VC3:POH:TRACe:CRC <mode>
Description	This command sets the section overhead trace (J1) CRC mode (OFF/ON).
Parameters	<Pt> = Port number
	<mode> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SDH:TX1:AU3:VC3:POH:TRAC:CRC ON`
Note	

Syntax	SDH:TX<Pt>:AU3:VC3:POH:TRACe:CRC?
Description	This query returns the mode of the overhead trace (J1) CRC.
Parameter	<Pt> = Port number
Response	<mode> = <BOOLEAN RESPONSE DATA>
Example	`SDH:TX1:AU3:VC3:POH:TRAC:CRC? → 1`
Note	

10.2.50 SDH:TX<Pt>:AU3:VC3:POH:TTCM

Syntax	SDH:TX<Pt>:AU3:VC3:POH:TTCM <string>[,<idlechar>]
Description	This command sets the VC-3 TCM trace (N1) to the specified string, and the idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Apid_TCM_N1"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SDH:TX1:AU3:VC3:POH:TTCM "Apid_TCM_N1",#H20`
Note	

Syntax	SDH:TX<Pt>:AU3:VC3:POH:TTCM?
Description	This query returns the VC-3 TCM trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:TX1:AU3:VC3:POH:TTCM? → "Apid_TCM_N1",#H20`
Note	

10.2.51 SDH:TX<Pt>:AU3:VC3:POH

Syntax	SDH:TX<Pt>:AU3:VC3:POH <POH-byte>,<value>
Description	This command sets the value of the specified byte in the VC-3 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA> C2: C2 byte. G1: G1 byte. F2: F2 byte. H4: H4 byte. F3: F3 byte. K3: K3 byte. N1: N1 byte. Only available when there is no TCM.
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
Response	None.
Example	`SDH:TX1:AU3:VC3:POH C2,0`
Note	

Syntax	SDH:TX<Pt>:AU3:VC3:POH? <POH-byte>
Description	This query returns the value of the specified byte in the VC-3 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA>
Response	<value> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:TX1:AU3:VC3:POH? C2` → `#H00`
Note	

10.2.52 SDH:TX<Pt>:FOLLow

Syntax	SDH:TX<Pt>:FOLLow <mode>
Description	This command sets the transmitter setup to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> NONE: Do not follow. TX1: Follows setup of the Tx port1. *DEFault = NONE*
Response	None.
Example	`SDH:TX1:FOLL NONE`
Note	

Syntax	SDH:TX<Pt>:FOLLow?
Description	This query returns the transmitter setup to follow another setup or not to follow.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SDH:TX1:FOLL?` → `NONE`
Note	

10.2.53 SDH:TX<Pt>:OTHRough

Syntax	SDH:TX<Pt>:OTHRough <mode>
Description	This command sets the OH Overwrite Position.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> SOH: SOH Data. A1A2: A1/A2 byte. K1K2: K1/K2 byte. S1: S1 byte. DCC1TO3: DCC1-3 byte. DCC4TO12: DCC4-12 byte. J0: J0 byte. SOH1BYTE: 1 byte of SOH.(selectable) *DEFault = SOH*
Response	None.
Example	`SDH:TX1:OTHR A1A2`
Note	

Syntax	SDH:TX<Pt>:OTHRough?
Description	This query returns the OH Overwrite Position.
Parameter	<Pt> = Port number
Response	<position> = <CHARACTER RESPONSE DATA>
Example	`SDH:TX1:OTHR?` → `A1A2`
Note	

10.2.54 SDH:TX<Pt>:OTHRough:BYTE

Syntax	SDH:TX<Pt>:OTHRough:BYTE <byte>
Description	This command sets the OH Overwrite Position.
Parameters	<Pt> = Port number
	<byte> = <CHARACTER PROGRAM DATA> A11-A13: A1 bytes. A21-A23: A2 bytes. J0: J0 byte. E1: E1 byte. F1: F1 byte. D1: D1 byte. D2: D2 byte. D3: D3 byte. K1: K1 byte. K2: K2 byte. D4: D4 byte. D5: D5 byte. D6: D6 byte. D7: D7 byte. D8: D8 byte. D9: D9 byte. D10: D10 byte. D11: D11 byte. D12: D12 byte. S1: S1 byte. M0: M0 byte. M1: M1 byte. E2: E2 byte. X18-X99: X<Line><Column> *DEFault = A11*
Response	None.
Example	`SDH:TX1:OTHR:BYTE A11`
Note	

Syntax	SDH:TX<Pt>:OTHRough:BYTE?
Description	This query returns the OH Overwrite Position.
Parameter	<Pt> = Port number
Response	<byte> = <CHARACTER RESPONSE DATA>
Example	`SDH:TX1:OTHR:BYTE?` → `A11`
Note	

10.2.55 SDH:TX<Pt>:STL:MMAPing:LANE

Syntax	SDH:TX<Pt>:STL:MMAPing:LANE <value>
Description	This command sets the STL lane marker assignment.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List List consist of the value of the lane marker ranging from 0 to 3.
Response	None.
Example	`SDH:TX1:STL:MMAP:LANE (0,1,2,3)`
Note	This command can be used on 40G

Syntax	SDH:TX<Pt>:STL:MMAPing:LANE?
Description	This query returns the STL lane marker assignment.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION RESPONSE DATA> Format: Numeric List
Example	`SDH:TX1:STL:MMAP:LANE?` → `(0,1,2,3)`
Note	This command can be used on 40G

10.3 Stimuli

10.3.1 SDH:STIMuli:TX<Pt>:FOFFset

Syntax	SDH:STIMuli:TX<Pt>:FOFFset <offset>
Description	This command sets the frequency offset for the clock source. Unit: ppm.
Parameters	<Pt> = Port number
	<offset> = <NUMERIC PROGRAM DATA> MT1000A: *MINimum=-50, MAXimum=50, DEFault=0* MT1100A: *MINimum=-200.0, MAXimum=200.0, DEFault = 0*
Response	None.
Example	`SDH:STIM:TX1:FOFF 0`
Note	

Syntax	SDH:STIMuli:TX<Pt>:FOFFset?
Description	This query returns the frequency offset for the clock source. Unit: ppm.
Parameter	<Pt> = Port number
Response	MT1000A: <offset> = <NR1 NUMERIC RESPONSE DATA> MT1100A: <offset> = <NR2 NUMERIC RESPONSE DATA>
Example	MT1000A: `SDH:STIM:TX1:FOFF?` → `0` MT1100A: `SDH:STIM:TX1:FOFF?` → `0.0`
Note	

10.3.2 SDH:STIMuli:TX<Pt>:ALARm

Syntax	SDH:STIMuli:TX<Pt>:ALARm <alarmtype>
Description	This command sets the type of alarm to be generated.
Parameters	<Pt> = Port number
	<alarmtype> = <CHARACTER PROGRAM DATA> LOS: Loss of signal LOF: Loss of frame OOF: Out of frame MSAIS: MS alarm indication signal MSRDI: MS remote defect indicator AUAIS: AU alarm indication signal AULOP: AU loss of pointer HPTIM: HP trace identifier mismatch HPPLM: HP payload label mismatch HPUNEQ: HP unequipped HPRDI: HP remote defect indicator TUAIS: TU alarm indication signal TULOP: TU loss of pointer TULOM: TU loss of multiframe LPTIM: LP trace identifier mismatch LPUNEQ: LP unequipped LPRDI: LP remote defect indicator LPPLM: LP payload label mismatch LSS: Loss of signal synchronization TCUNEQ: TC unequipped TCLTC: TC loss of tandem connection TCTIM: TC trace identifier mismatch TCAIS: TC alarm indication signal TCRDI: TC remote defect indicator TCODI: TC outgoing defect indicator *DEFault = LOS*
Response	None.
Example	`SDH:STIM:TX1:ALAR LOS`
Note	

Syntax	SDH:STIMuli:TX<Pt>:ALARm?
Description	This query returns the type of alarm to be generated.
Parameter	<Pt> = Port number
Response	<alarmtype> = <CHARACTER RESPONSE DATA>
Example	`SDH:STIM:TX1:ALAR?` → `LOS`
Note	

10.3.3 SDH:STIMuli:TX<Pt>:AINSert

Syntax	SDH:STIMuli:TX<Pt>:AINSert <insertion>
Description	This command sets the method to insert alarm.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> OFF PERManent ALTernate *DEFault = OFF*
Response	None.
Example	`SDH:STIM:TX1:AINS PERM`
Note	

Syntax	SDH:STIMuli:TX<Pt>:AINSert?
Description	This query returns the alarm insertion method.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`SDH:STIM:TX1:AINS?` → `PERM`
Note	

10.3.4 SDH:STIMuli:TX<Pt>:ANLength

Syntax	SDH:STIMuli:TX<Pt>:ANLength <frames>
Description	This command sets the alternate normal length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1* Maximum depends on the error insert method, see `SDH:STIM:TX<Pt>:AINS`
Response	None.
Example	`SDH:STIM:TX1:ANL 1`
Note	

Syntax	SDH:STIMuli:TX<Pt>:ANLength?
Description	This query returns the alternate normal length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:STIM:TX1:ANL?` → `1`
Note	

10.3.5 SDH:STIMuli:TX<Pt>:AALength

Syntax	SDH:STIMuli:TX<Pt>:AALength <frames>
Description	This command sets the alternate alarm length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1* Maximum depends on the error insert method, see `SDH:STIM:TX<Pt>:AINS`
Response	None.
Example	`SDH:STIM:TX1:AAL 1`
Note	

Syntax	SDH:STIMuli:TX<Pt>:AALength?
Description	This query returns the alternate alarm length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:STIM:TX1:AAL?` → `1`
Note	

10.3.6 SDH:STIMuli:TX<Pt>:ERRor

Syntax	SDH:STIMuli:TX<Pt>:ERRor <errordestination>
Description	This command sets the error destination.
Parameters	<Pt> = Port number
	<errordestination> = <CHARACTER PROGRAM DATA> A1A2: Frame alignment word B1: B1 checksum byte B2: B2 checksum byte MSREI: MS remote error indication B3: B3 checksum byte HPREI: HP Remote error indication V5LPB3: B3 checksum of the low-order path LPREI: LP remote error indication PRBS: Pattern error ETRans: ERR-TRANS TCIEC: Tandem Connection incoming error count TCREI: Tandem Connection remote error indication TCOEI: Tandem Connection outgoing error indication TCBIP2: 2-bit Bit Interleaved Parity for Tandem Connection *DEFault = A1A2*
Response	None.
Example	`SDH:STIM:TX1:ERR MSREI`
Note	Some errors are depended on type of content, setting them without the correct type of content will set EINSert to OFF. Changing the error may also change the insert method. See section 10.3.7.

Syntax	SDH:STIMuli:TX<Pt>:ERRor?
Description	This query returns the error destination.
Parameter	<Pt> = Port number
Response	<errordestination> = <CHARACTER RESPONSE DATA>
Example	`SDH:STIM:TX1:ERR?` → `MSREI`
Note	

10.3.7 SDH:STIMuli:TX<Pt>:EINSert

Syntax	SDH:STIMuli:TX<Pt>:EINSert <insertion>
Description	This command sets the method to insert errors.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> OFF MANual B03: Burst · 1E-03 [1] B04: Burst · 1E-04 [1] B05: Burst · 1E-05 [1] B06: Burst · 1E-06 [2] B07: Burst · 1E-07 B08: Burst · 1E-08 B09: Burst · 1E-09 B10: Burst · 1E-10 ALTernate *DEFault = OFF*
Response	None.
Example	`SDH:STIM:TX1:EINS MAN`
Note	If insertion is set to MANual, errors are inserted with SYST:STIM:INS. See section 2.3.23. [1] Is available for PRBS and ETRans (B03 is only available for ETRans if the STM Level is 1). [2] Is available for PRBS, ETRans, B2, V5LPB3, LPREI, TCIEC, TCBIP2, TCREI and TCOEI. Selecting an unsupported burst rate will reset the rate to the closest possible.

Syntax	SDH:STIMuli:TX<Pt>:EINSert?
Description	This query returns the error insertion method.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`SDH:STIM:TX1:EINS?` → `MAN`
Note	

10.3.8 SDH:STIMuli:TX<Pt>:EBLength

Syntax	SDH:STIMuli:TX<Pt>:EBLength <burstlength>
Description	This command sets the error burst length to generate.
Parameters	<Pt> = Port number
	<burstlength> = <NUMERIC PROGRAM DATA> *MINimum = 1, DEFault = 1* Maximum depends on the error insert method, see `SDH:STIM:TX<Pt>:EINS` B03 - B10: Maximum = 1 MANual: Maximum = 8000[1]
Response	None.
Example	`SDH:STIM:TX1:EBL 1`
Note	[1] If VC12 is active the maximum value for LPREI, V5LPB3, TCIEC, TCBIP2, TCREI and TCOEI is 2000. The maximum value is 4000 when :STIM:TX<Pt>:ERR is PRBS (Pattern error).

Syntax	SDH:STIMuli:TX<Pt>:EBLength?
Description	This query returns the error burst length to generate.
Parameter	<Pt> = Port number
Response	<burstlength> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:STIM:TX1:EBL?` → `1`
Note	

10.3.9 SDH:STIMuli:TX<Pt>:ENLength

Syntax	SDH:STIMuli:TX<Pt>:ENLength <frames>
Description	This command sets the alternate normal length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1* Maximum depends on the error insert method, see `SDH:STIM:TX<Pt>:EINS`
Response	None.
Example	`SDH:STIM:TX1:ENL 1`
Note	

Syntax	SDH:STIMuli:TX<Pt>:ENLength?
Description	This query returns the alternate normal length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:STIM:TX1:ENL? → 1`
Note	

10.3.10 SDH:STIMuli:TX<Pt>:EELength

Syntax	SDH:STIMuli:TX<Pt>:EELength <frames>
Description	This command sets the alternate error length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1* Maximum depends on the error insert method, see `SDH:STIM:TX<Pt>:EINS`
Response	None.
Example	`SDH:STIM:TX1:EEL 1`
Note	

Syntax	SDH:STIMuli:TX<Pt>:EELength?
Description	This query returns the alternate error length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:STIM:TX1:EEL? → 1`
Note	

10.3.11 SDH:STIMuli:TX<Pt>:PTSequence

Syntax	SDH:STIMuli:TX<Pt>:PTSequence <type>,<sequence>
Description	This command sets the SDH pointer test sequence.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> AU4: AU-4 pointer AU3: AU-3 pointer TU3: TU-3 pointer TU12: TU-12 pointer TU11: TU-11 pointer
	<sequence> = <CHARACTER PROGRAM DATA> NONE: No test sequence SALTernating: Single alternating RDOuble: Regular + double RMISsing: Regular + missing DALTernating: Double alternating
Response	None.
Example	`SDH:STIM:TX1:PTS AU4,NONE`
Note	

Syntax	SDH:STIMuli:TX<Pt>:PTSequence? <type>
Description	This query returns the SDH pointer test sequence.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> AU4: AU-4 pointer AU3: AU-3 pointer TU3: TU-3 pointer TU12: TU-12 pointer TU11: TU-11 pointer
Response	<sequence> = <CHARACTER RESPONSE DATA>
Example	`SDH:STIM:TX1:PTS? AU4` → `NONE`
Note	

10.3.12 SDH:STIMuli:TX<Pt>:PMOVe

Syntax	SDH:STIMuli:TX<Pt>:PMOVe <type>,<value>
Description	This command sets the SDH pointer movement sequence. The value specifies the number of pointer increments.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> AU4: AU-4 pointer AU3: AU-3 pointer TU3: TU-3 pointer TU12: TU-12 pointer TU11: TU-11 pointer
	<value> = <NUMERIC PROGRAM DATA> *MINimum = -100, MAXimum = 100*
Response	None.
Example	`SDH:STIM:TX1:PMOV AU4,9`
Notes	There is no query version of this command. A value of zero has no effect. A negative value results in pointer decrements.

10.3.13 SDH:STIMuli:TX<Pt>:PJUMp

Syntax	SDH:STIMuli:TX<Pt>:PJUMp <type>,<value>[,<ndf>]
Description	This command sets the SDH pointer jump. The value specifies the new pointer value.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> AU4: AU-4 pointer AU3: AU-3 pointer TU3: TU-3 pointer TU12: TU-12 pointer TU11: TU-11 pointer
	<value> = <NUMERIC PROGRAM DATA> The valid range depends on SDH:STIMuli:TX<Pt>:PTYPe AU4: 0 to 782 AU3: 0 to 782 TU3: 0 to 764 TU12: 0 to 139 TU11: 0 to 103
	<ndf> = <CHARACTER PROGRAM DATA> WITH: With new data flag (NDF) WOUT: Without new data flag (NDF) *DEFault = WITH*
Response	None.
Example	`SDH:STIM:TX1:PJUM AU4,300`
Notes	There is no query version of this command. A negative value disables new data flag (NDF).

10.3.14 SDH:STIMuli:TX<Pt>:STL:AERRor:LANE

Syntax	SDH:STIMuli:TX<Pt>:STL:AERRor:LANE <lane>
Description	This command sets the lane of the multi lane alarm/error to be inserted.
Parameters	<Pt> = Port number
	<lane> = <NUMERIC PROGRAM DATA> *MINimum=#B0000, MAXimum=#B1111, DEFault=#B1000*
Response	None.

Example	`SDH:STIM:TX1:STL:AERR:LANE #B01`
	`SDH:STIM:TX1:STL:AERR:LANE #B0100`
	These commands add error into lane 1.

Note	This command can be used on 40G

Syntax	SDH:STIMuli:TX<Pt>:STL:AERRor:LANE?
Description	This query returns the lane of the multi lane alarm/error to be inserted.
Parameters	<Pt> = Port number
Response	<lane> = <BINARY NUMERIC RESPONSE DATA>
Example	`SDH:STIM:TX1:STL:AERR:LANE?` → `#B0100`
Note	This command can be used on 40G

10.3.15 SDH:STIMuli:TX<Pt>:STL:ALARm

Syntax	SDH:STIMuli:TX<Pt>:STL:ALARm <alarmtype>
Description	This command sets the type of alarm to be inserted.
Parameters	<Pt> = Port number
	<alarmtype> = <CHARACTER PROGRAM DATA> LOFOOF: LOF/OOF-STL LOROOR: LOR/OOR-STL *DEFault = LOFOOF*
Response	None.
Example	`SDH:STIM:TX1:STL:ALAR LOFOOF`
Note	This command can be used on 40G

Syntax	SDH:STIMuli:TX<Pt>:STL:ALARm?
Description	This query returns the type of alarm to be inserted.
Parameter	<Pt> = Port number
Response	<alarmtype> = <CHARACTER RESPONSE DATA>
Example	`SDH:STIM:TX1:STL:ALAR?` → `LOFOOF`
Note	This command can be used on 40G

10.3.16 SDH:STIMuli:TX<Pt>:STL:AINSert

Syntax	SDH:STIMuli:TX<Pt>:STL:AINSert <insertion>
Description	This command sets the method to insert alarm.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> OFF MANual ALTernate *DEFault = OFF*
Response	None.
Example	`SDH:STIM:TX1:STL:AINS MAN`
Note	This command can be used on 40G

Syntax	SDH:STIMuli:TX<Pt>:STL:AINSert?
Description	This query returns the alarm insertion method.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`SDH:STIM:TX1:STL:AINS?` → `MAN`
Note	This command can be used on 40G

10.3.17 SDH:STIMuli:TX<Pt>:STL:ABLength

Syntax	SDH:STIMuli:TX<Pt>:STL:ABLength <frames>
Description	This command sets the alarm burst length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1*
Response	None.
Example	`SDH:STIM:TX1:STL:ABL 8000`
Note	This command can be used on 40G

Syntax	SDH:STIMuli:TX<Pt>:STL:ABLength?
Description	This query returns the alarm burst length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:STIM:TX1:STL:ABL?` → `8000`
Note	This command can be used on 40G

10.3.18 SDH:STIMuli:TX<Pt>:STL:AALength

Syntax	SDH:STIMuli:TX<Pt>:STL:AALength <frames>
Description	This command sets the alternate alarm length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1*
Response	None.
Example	`SDH:STIM:TX1:STL:AAL 8000`
Note	This command can be used on 40G

Syntax	SDH:STIMuli:TX<Pt>:STL:AALength?
Description	This query returns the alternate alarm length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:STIM:TX1:STL:AAL?` → `8000`
Note	This command can be used on 40G

10.3.19 SDH:STIMuli:TX<Pt>:STL:ANLength

Syntax	SDH:STIMuli:TX<Pt>:STL:ANLength <frames>
Description	This command sets the alternate normal length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1*
Response	None.
Example	`SDH:STIM:TX1:STL:ANL 8000`
Note	This command can be used on 40G

Syntax	SDH:STIMuli:TX<Pt>:STL:ANLength?
Description	This query returns the alternate normal length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:STIM:TX1:STL:ANL?` → `8000`
Note	This command can be used on 40G

10.3.20 SDH:STIMuli:TX<Pt>:STL:ERRor

Syntax	SDH:STIMuli:TX<Pt>:STL:ERRor <errortype>
Description	This command sets the type of error to be inserted.
Parameters	<Pt> = Port number
	<errortype> = <CHARACTER PROGRAM DATA> A1A2: A1A2-STL *DEFault = A1A2*
Response	None.
Example	`SDH:STIM:TX1:STL:ERR A1A2`
Note	This command can be used on 40G

Syntax	SDH:STIMuli:TX<Pt>:STL:ERRor?
Description	This query returns the type of error to be inserted.
Parameter	<Pt> = Port number
Response	<errortype> = <CHARACTER RESPONSE DATA>
Example	`SDH:STIM:TX1:STL:ERR? → A1A2`
Note	This command can be used on 40G

10.3.21 SDH:STIMuli:TX<Pt>:STL:EINSert

Syntax	SDH:STIMuli:TX<Pt>:STL:EINSert <insertion>
Description	This command sets the method to insert errors.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> OFF ALTernate *DEFault = OFF*
Response	None.
Example	`SDH:STIM:TX1:STL:EINS ALT`
Note	This command can be used on 40G

Syntax	SDH:STIMuli:TX<Pt>:STL:EINSert?
Description	This query returns the error insertion method.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`SDH:STIM:TX1:STL:EINS? → ALT`
Note	This command can be used on 40G

10.3.22 SDH:STIMuli:TX<Pt>:STL:EELength

Syntax	SDH:STIMuli:TX<Pt>:STL:EELength <frames>
Description	This command sets the alternate error length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1*
Response	None.
Example	`SDH:STIM:TX1:STL:EEL 8000`
Note	This command can be used on 40G

Syntax	SDH:STIMuli:TX<Pt>:STL:EELength?
Description	This query returns the alternate error length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:STIM:TX1:STL:EEL? → 8000`
Note	This command can be used on 40G

10.3.23 SDH:STIMuli:TX<Pt>:STL:ENLength

Syntax	SDH:STIMuli:TX<Pt>:STL:ENLength <frames>
Description	This command sets the alternate normal length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1*
Response	None.
Example	`SDH:STIM:TX1:STL:ENL 8000`
Note	This command can be used on 40G

Syntax	SDH:STIMuli:TX<Pt>:STL:ENLength?
Description	This query returns the alternate normal length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:STIM:TX1:STL:ENL?` → `8000`
Note	This command can be used on 40G

10.3.24 SDH:STIMuli:TX<Pt>:STL:SKEW:BIT

Syntax	SDH:STIMuli:TX<Pt>:STL:SKEW:BIT <bits>
Description	This command sets the bits of the skew .
Parameters	<Pt> = Port number
	<bits> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 138240, DEFault=1*
Response	None.
Example	`SDH:STIM:TX1:STL:SKEW:BIT 1000`
Note	This command can be used on 40G

Syntax	SDH:STIMuli:TX<Pt>:STL:SKEW:BIT?
Description	This query returns the bits of the skew .
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:STIM:TX1:STL:SKEW:BIT?` → `1000`
Note	This command can be used on 40G

10.3.25 SDH:STIMuli:TX<Pt>:STL:SKEW:NS?

Syntax	SDH:STIMuli:TX<Pt>:STL:SKEW:NS?
Description	This query returns the time of the skew to be inserted. Unit: ns
Parameter	<Pt> = Port number
Response	<skew> = <NR2 NUMERIC RESPONSE DATA>
Example	`SDH:STIM:TX1:STL:SKEW:NS?` → `100.469`
Note	This command can be used on 40G

10.3.26 SDH:STIMuli:TX<Pt>:STL:SKEW:LANE

Syntax	SDH:STIMuli:TX<Pt>:STL:SKEW:LANE <lane>
Description	This command sets the lane of the skew to be inserted.
Parameters	<Pt> = Port number
	<lane> = <NUMERIC PROGRAM DATA> *MINimum=#B0000, MAXimum=#B1111, DEFault=#B1000*
Response	None.

Example	`SDH:STIM:TX1:STL:SKEW:LANE #B01`
	`SDH:STIM:TX1:STL:SKEW:LANE #B0100`
	These commands add skew into lane 1.

Note	This command can be used on 40G

Syntax	SDH:STIMuli:TX<Pt>:STL:SKEW:LANE?
Description	This query returns the lane of the skew to be inserted.
Parameters	<Pt> = Port number
Response	<lane> = <BINARY NUMERIC RESPONSE DATA>
Example	`SDH:STIM:TX1:STL:SKEW:LANE?` → `#B0100`
Note	This command can be used on 40G

10.4 Result

10.4.1 SDH:RX<Pt>:IFETch?

Syntax	SDH:RX<Pt>:IFETch? <parameter>
Description	This query fetches a SDH interval if available.
Parameters	<Pt> = Port number
	({<parameter>} + {,}∗) = <EXPRESSION PROGRAM DATA> The response format is listed for each parameter. **Alarms** LOS: Loss of signal. Response: <Seconds>,<Ratio> GAIS: Generic alarm indication signal. Response: <Seconds>,<Ratio> LOF: Loss of frame. Response: <Seconds>,<Ratio> OOF: Out of frame. Response: <Seconds>,<Ratio> MSAIS: Multiplex section alarm indication signal. Response: <Seconds>,<Ratio> MSRDI: Multiplex section remote defect indicator. Response: <Seconds>,<Ratio> AUAIS: Administrative unit - alarm indication signal. Response: <Seconds>,<Ratio> AULOP: Administrative unit - loss of pointer. Response: <Seconds>,<Ratio> HPTIM: High-order path trace identifier mismatch. Response: <Seconds>,<Ratio> HPPLM: Payload label mismatch. Response: <Seconds>,<Ratio> HPUNEQ: High-order path unequipped. Response: <Seconds>,<Ratio> HPRDI: High-order path remote defect indicator. Response: <Seconds>,<Ratio> TUAIS: Tributary unit alarm indication signal[1]. Response: <Seconds>,<Ratio> TULOP: Tributary unit loss of pointer[1]. Response: <Seconds>,<Ratio> TULOM: Tributary unit loss of multi frame[1]. Response: <Seconds>,<Ratio> LPTIM: Low-order path trace identifier mismatch[1]. Response: <Seconds>,<Ratio> LPUNEQ: Low-order path unequipped[1]. Response: <Seconds>,<Ratio> LPRDI: Low-order path remote defect indicator[1]. Response: <Seconds>,<Ratio> LPPLM: Low-order path payload label mismatch[1]. Response: <Seconds>,<Ratio> LSS: Loss of signal synchronization[1]. Response: <Seconds>,<Ratio> **Errors** A1A2: Response: <Count>,<Ratio> B1: Response: <Count>,<Ratio> B2: Response: <Count>,<Ratio> MSREI: Response: <Count>,<Ratio> B3: Response: <Count>,<Ratio> HPREI: Response: <Count>,<Ratio> V5LPB3: [1]Response: <Count>,<Ratio> ERRPRBS: Pattern errors[1]. Response: <Count>,<Ratio> ERRPRBSBLK: Pattern block errors[1]. Response: <Count>,<Ratio> LPREI: Response: <Count>,<Ratio> AUNDF: Response: <Count>,<Ratio> TUNDF: Response: <Count>,<Ratio> SAPS: Switch APS. Response: <Count>,<Ratio> **Rx frequency** FREQ: Frequency deviation. Response: <ppm> FREQDIF: Frequency difference (RX1-RX2). Response: <ppm> (Only for RX1, RX2 is always NaN (section 1.6.1)) **Mux quality** MFES: Forward ES. Response: <Count>,<Ratio%> MFSES: Forward SES. Response: <Count>,<Ratio%> MFUNAV: Forward UNAV. Response: <Count>,<Ratio%> MBES: Backward ES. Response: <Count>,<Ratio%> MBSES: Backward SES. Response: <Count>,<Ratio%> MBUNAV: Backward UNAV. Response: <Count>,<Ratio%> **VC-4 quality** VC4FES: Forward ES. Response: <Count>,<Ratio%> VC4FSES: Forward SES. Response: <Count>,<Ratio%>

continued on next page...

. . . continued from previous page

	VC4FUNAV: Forward UNAV. Response: <Count>,<Ratio%> VC4BES: Backward ES. Response: <Count>,<Ratio%> VC4BSES: Backward SES. Response: <Count>,<Ratio%> VC4BUNAV: Backward UNAV. Response: <Count>,<Ratio%> **VC-3 quality**[1] VC3FES: Forward ES. Response: <Count>,<Ratio%> VC3FSES: Forward SES. Response: <Count>,<Ratio%> VC3FUNAV: Forward UNAV. Response: <Count>,<Ratio%> VC3BES: Backward ES. Response: <Count>,<Ratio%> VC3BSES: Backward SES. Response: <Count>,<Ratio%> VC3BUNAV: Backward UNAV. Response: <Count>,<Ratio%> **VC-12 quality**[1] VC12FES: Forward ES. Response: <Count>,<Ratio%> VC12FSES: Forward SES. Response: <Count>,<Ratio%> VC12FUNAV: Forward UNAV. Response: <Count>,<Ratio%> VC12BES: Backward ES. Response: <Count>,<Ratio%> VC12BSES: Backward SES. Response: <Count>,<Ratio%> VC12BUNAV: Backward UNAV. Response: <Count>,<Ratio%> **VC-11 quality**[1] VC11FES: Forward ES. Response: <Count>,<Ratio%> VC11FSES: Forward SES. Response: <Count>,<Ratio%> VC11FUNAV: Forward UNAV. Response: <Count>,<Ratio%> VC11BES: Backward ES. Response: <Count>,<Ratio%> VC11BSES: Backward SES. Response: <Count>,<Ratio%> VC11BUNAV: Backward UNAV. Response: <Count>,<Ratio%> **Bulk quality** ES: ES. Response: <Count>,<Ratio%> SES: SES. Response: <Count>,<Ratio%> UNAV: UNAV. Response: <Count>,<Ratio%> **AU-4 pointer** AUPOINT: AU pointer. Response: <Count> AUNEG: Negative. Response: <Count> AUPOS: Positive. Response: <Count> **TU-3 or TU-12 or TU-11 pointer**[1] TUPOINT: TU pointer. Response: <Count> TUNEG: Negative. Response: <Count> TUPOS: Positive. Response: <Count> **Justification**[1] JNEG: Negative justification. Response: <Count> JPOS: Positive justification. Response: <Count> **MUX performance**[1] PMPES: Performance MUX PO limit ES. Response: <Count>,<Ratio%> PMPSES: Performance MUX PO limit SES. Response: <Count>,<Ratio%> PMPBBE: Performance MUX PO limit BBE. Response: <Count>,<Ratio%> PMBS1ES: Performance MUX BIS limit S1ES. Response: <Count> PMBS2ES: Performance MUX BIS limit S2ES. Response: <Count> PMBS1SES: Performance MUX BIS limit S1SES. Response: <Count> PMBS2SES: Performance MUX BIS limit S2SES. Response: <Count> PMFSTAT: Performance MUX forward status. Response: <STRING RESPONSE DATA> PMFES: Performance MUX forward ES. Response: <Count>,<Ratio%> PMFSES: Performance MUX forward SES. Response: <Count>,<Ratio%> PMFUNAV: Performance MUX forward UNAV. Response: <Count>,<Ratio%> PMFBBE: Performance MUX forward BBE. Response: <Count>,<Ratio%> PMBSTAT: Performance MUX backward status. Response: <STRING RESPONSE DATA> PMBES: Performance MUX backward ES. Response: <Count>,<Ratio%>

continued on next page. . .

. . . continued from previous page

	PMBSES: Performance MUX backward SES. Response: <Count>,<Ratio%> PMBUNAV: Performance MUX backward UNAV. Response: <Count>,<Ratio%> PMBBBE: Performance MUX backward BBE. Response: <Count>,<Ratio%> **VC-4 performance**[1] PVC4PES: Performance VC4 PO limit ES. Response: <Count>,<Ratio%> PVC4PSES: Performance VC4 PO limit SES. Response: <Count>,<Ratio%> PVC4PBBE: Performance VC4 PO limit BBE. Response: <Count>,<Ratio%> PVC4BS1ES: Performance VC4 BIS limit S1ES. Response: <Count> PVC4BS2ES: Performance VC4 BIS limit S2ES. Response: <Count> PVC4BS1SES: Performance VC4 BIS limit S1SES. Response: <Count> PVC4BS2SES: Performance VC4 BIS limit S2SES. Response: <Count> PVC4FSTAT: Performance VC4 forward status. Response: <STRING RESPONSE DATA> PVC4FES: Performance VC4 forward ES. Response: <Count>,<Ratio%> PVC4FSES: Performance VC4 forward SES. Response: <Count>,<Ratio%> PVC4FUNAV: Performance VC4 forward UNAV. Response: <Count>,<Ratio%> PVC4FBBE: Performance VC4 forward BBE. Response: <Count>,<Ratio%> PVC4BSTAT: Performance VC4 backward status. Response: <STRING RESPONSE DATA> PVC4BES: Performance VC4 backward ES. Response: <Count>,<Ratio%> PVC4BSES: Performance VC4 backward SES. Response: <Count>,<Ratio%> PVC4BUNAV: Performance VC4 backward UNAV. Response: <Count>,<Ratio%> PVC4BBBE : Performance VC4 backward BBE. Response: <Count>,<Ratio%> **VC-12 performance**[1] PVC12PES: Performance VC12 PO limit ES. Response: <Count>,<Ratio%> PVC12PSES: Performance VC12 PO limit SES. Response: <Count>,<Ratio%> PVC12PBBE: Performance VC12 PO limit BBE. Response: <Count>,<Ratio%> PVC12BS1ES: Performance VC12 BIS limit S1ES. Response: <Count> PVC12BS2ES: Performance VC12 BIS limit S2ES. Response: <Count> PVC12BS1SES: Performance VC12 BIS limit S1SES. Response: <Count> PVC12BS2SES: Performance VC12 BIS limit S2SES. Response: <Count> PVC12FSTAT: Performance VC12 forward status. Response: <STRING RESPONSE DATA> PVC12FES: Performance VC12 forward ES. Response: <Count>,<Ratio%> PVC12FSES: Performance VC12 forward SES. Response: <Count>,<Ratio%> PVC12FUNAV: Performance VC12 forward UNAV. Response: <Count>,<Ratio%> PVC12FBBE: Performance VC12 forward BBE. Response: <Count>,<Ratio%> PVC12BSTAT: Performance VC12 backward status. Response: <STRING RESPONSE DATA> PVC12BES: Performance VC12 backward ES. Response: <Count>,<Ratio%> PVC12BSES: Performance VC12 backward SES. Response: <Count>,<Ratio%> PVC12BUNAV: Performance VC12 backward UNAV. Response: <Count>,<Ratio%> PVC12BBBE: Performance VC12 backward BBE. Response: <Count>,<Ratio%> **VC-11 performance**[1] PVC11PES: Performance VC11 PO limit ES. Response: <Count>,<Ratio%> PVC11PSES: Performance VC11 PO limit SES. Response: <Count>,<Ratio%> PVC11PBBE: Performance VC11 PO limit BBE. Response: <Count>,<Ratio%> PVC11BS1ES: Performance VC11 BIS limit S1ES. Response: <Count> PVC11BS2ES: Performance VC11 BIS limit S2ES. Response: <Count> PVC11BS1SES: Performance VC11 BIS limit S1SES. Response: <Count> PVC11BS2SES: Performance VC11 BIS limit S2SES. Response: <Count> PVC11FSTAT: Performance VC11 forward status. Response: <STRING RESPONSE DATA> PVC11FES: Performance VC11 forward ES. Response: <Count>,<Ratio%> PVC11FSES: Performance VC11 forward SES. Response: <Count>,<Ratio%> PVC11FUNAV: Performance VC11 forward UNAV. Response: <Count>,<Ratio%> PVC11FBBE: Performance VC11 forward BBE. Response: <Count>,<Ratio%>

continued on next page. . .

. . . continued from previous page

	PVC11BSTAT: Performance VC11 backward status. Response: <STRING RESPONSE DATA> PVC11BES: Performance VC11 backward ES. Response: <Count>,<Ratio%> PVC11BSES: Performance VC11 backward SES. Response: <Count>,<Ratio%> PVC11BUNAV: Performance VC11 backward UNAV. Response: <Count>,<Ratio%> PVC11BBBE: Performance VC11 backward BBE. Response: <Count>,<Ratio%> **VC-3 performance**[1] PVC3PES: Performance VC3 PO limit ES. Response: <Count>,<Ratio%> PVC3PSES: Performance VC3 PO limit SES. Response: <Count>,<Ratio%> PVC3PBBE: Performance VC3 PO limit BBE. Response: <Count>,<Ratio%> PVC3BS1ES: Performance VC3 BIS limit S1ES. Response: <Count> PVC3BS2ES: Performance VC3 BIS limit S2ES. Response: <Count> PVC3BS1SES: Performance VC3 BIS limit S1SES. Response: <Count> PVC3BS2SES: Performance VC3 BIS limit S2SES. Response: <Count> PVC3FSTAT: Performance VC3 forward status. Response: <STRING RESPONSE DATA> PVC3FES: Performance VC3 forward ES. Response: <Count>,<Ratio%> PVC3FSES: Performance VC3 forward SES. Response: <Count>,<Ratio%> PVC3FUNAV: Performance VC3 forward UNAV. Response: <Count>,<Ratio%> PVC3FBBE: Performance VC3 forward BBE. Response: <Count>,<Ratio%> PVC3BSTAT: Performance VC3 backward status. Response: <STRING RESPONSE DATA> PVC3BES: Performance VC3 backward ES. Response: <Count>,<Ratio%> PVC3BSES: Performance VC3 backward SES. Response: <Count>,<Ratio%> PVC3BUNAV: Performance VC3 backward UNAV. Response: <Count>,<Ratio%> PVC3BBBE: Performance VC3 backward BBE. Response: <Count>,<Ratio%> **Frequency** DEV: Deviation(ppm). Response: <Min>,<Max>,<Avg> **SDH TCM**[1] TCUNEQ: Tandem connection unequipped. response: <Count>,<Ratio%> TCLTC: Response: <Count>,<Ratio%> TCTIM: Tandem connection trace identifier mismatch. Response: <Count>,<Ratio%> TCAIS: Tandem connection alarm indication signal. Response: <Count>,<Ratio%> TCRDI: Tandem connection remote defect indicator. Response: <Count>,<Ratio%> TCODI: Tandem Connection Outgoing Defect Indicator. Response: <Count>,<Ratio%> TCIEC: Response: <Count>,<Ratio%> TCBIP2: Response: <Count>,<Ratio%> TCREI: Response: <Count>,<Ratio%> TCOEI: Response: <Count>,<Ratio%> TAPID: TCM APID. Response N1 or N2 TCM ID: <id> **STL** LOFSTL: STL LOF. Response: <Seconds> x 4 lanes LORSTL: STL LOR. Response: <Seconds> x 4 lanes OOFSTL: STL OOF. Response: <Frames> x 4 lanes OORSTL: STL OOR. Response: <Frames> x 4 lanes A1A2STL: A1A2-STL. Response: <Count> x 4 lanes OLA: OLA. Response: <Seconds>,<Ratio> RSKEW: Relative Skew. Response: <Nanoseconds> x 4 lanes MMAP: Marker Map. Response: <NR1> x 4 lanes
Response	{(<result>), }* = <EXPRESSION RESPONSE DATA> Format: Numeric List Each result is formatted according to the specification in the parameter field. Values that are not relevant or applicable for the current measurement return NaN (section 1.6.1).
Example	`SDH:RX1:IFET? (LOS,LOF,OOF)` → `(3,0.00532),(4,0.00709),(5,0.00887)`
Notes	This command fetches the results from the interval selected using the `MEASurement:SETup:SELect` command (see section 18.2.2).

continued on next page. . .

. . . continued from previous page

	[1] Requires the current interface/application is active in the measurement. If the requested result is not available, NaN (section 1.6.1) is returned. If there is one or more results, the last ”,” is always removed.

10.4.2 SDH:PMOVement:RX<Pt>:FETCh?

Syntax	SDH:PMOVement:RX<Pt>:FETCh? <pointer>,<min_time>,<max_time>
Description	This query fetches the pointer values.
Parameters	<pointer> = <CHARACTER PROGRAM DATA> AU: AU-4 or AU-3 pointer movements TU: TU pointer movements. TU-3 or TU-12 or TU-11 depending on SDH interface setup.
	<min_time> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=31536000* Beginning of the interval in seconds.
	<max_time> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=31536000* End of the interval in seconds.
Response	{<pointermovements>}* = <EXPRESSION RESPONSE DATA> The response is all pointer movements taking place in the requested time interval. Format: (pointerChange0,timestamp0),(pointerChange1,timestamp1),. . .
Example	`SDH:PMOV:RX1:FETC? AU,0,3600` → `(1,10.000),(-1,15.002)`
Notes	The interval is closed, i.e., it includes both endpoints. The displayed resolution of the timestamps is 1/1000 second = 0.001s.

10.5 Status

10.5.1 SDH:STATus:RX<Pt>:AESummary[:EVENt]?

Syntax	SDH:STATus:RX<Pt>:AESummary[:EVENt]?
Description	This query returns the alarms and errors summary event register. The content of this event register is summarized in DB2 of the STATus:INTerface:PORT<Pt>:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Physical and high-order alarm summary DB2 (2) = Low-order path and tandem connection alarm summary DB3 (4) = Error section 1 summary DB4 (8) = Error section 2 summary DB5 (16) = STL alarm summary DB6 (32) = STL error summary DB7 - DB16 = NOT USED
Example	`SDH:STAT:RX1:AES?` → `3`
Note	SDH can embed E1, E3 or E4 so if a PDH signal is embedded it is a good idea to send the following query STAT:INT:PORT<Pt>[:EVENt]? to see if there is alarms or error at the embedded signal.

10.5.2 SDH:STATus:RX<Pt>:AESummary:CONDition?

Syntax	SDH:STATus:RX<Pt>:AESummary:CONDition?
Description	This query returns alarms and errors summary condition register. DB1/DB2/DB5 is summary of SDH:STATus:RX<Pt>:ALARm<section>[:EVENt]? DB3/DB4/DB6 is summary of SDH:STATus:RX<Pt>:ERRor<section>[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> Same as SDH:STATus:RX<Pt>:AESummary[:EVENt]?
Example	`SDH:STAT:RX1:AES:COND?` → `3`
Note	

10.5.3 SDH:STATus:RX<Pt>:ALARm<section>[:EVENt]?

Syntax	SDH:STATus:RX<Pt>:ALARm<section>[:EVENt]?
Description	This query returns one of the alarms event register. These registers are summarized in DB1/DB2/DB5 of the SDH:STATus:RX<Pt>:AESummary:CONDition register.
Parameters	<Pt> = Port number
	<section> = Physical- and high-order path alarms(1), Low-order path and tandem connection alarms(2) or STL alarms(3)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> <section> = 1: DB1 (1) = LOS, Loss of signal DB2 (2) = LOF, Loss of frame DB3 (4) = OOF, Out of frame DB4 (8) = MS-AIS, Multiplex section - alarm indication signal DB5 (16) = MS-RDI, Multiplex section - remote defect indicator DB6 (32) = AU-AIS, Administrative unit - alarm indication signal DB7 (64) = AU-LOP, Administrative unit - loss of pointer DB8 (128) = HP-TIM, High-order path - trace identifier mismatch DB9 (256) = HP-PLM, High-order path - payload label mismatch DB10 (512) = HP-UNEQ, High-order path - unequipped DB11 (1024) = HP-RDI, High-order path - remote defect indicator DB12 (2048) = TU-AIS, Tributary unit - alarm indication signal DB13 (4096) = TU-LOP, Tributary unit - loss of pointer DB14 (8192) = TU-LOM, Tributary unit - loss of multi frame DB15 (16384) = G-AIS, Generic alarm indication signal DB16 = NOT USED <section> = 2: DB1 (1) = LP-TIM, Low-order path - trace identifier mismatch DB2 (2) = LP-UNEQ, Low-order path - unequipped DB3 (4) = LP-RDI, Low-order path - remote defect indicator DB4 (8) = LSS, Loss of signal synchronization DB5 = NOT USED DB6 (32) = LP-PLM, Low-order path - payload label mismatch DB7 (64) = TC-UNEQ, Tandem connection - unequipped DB8 (128) = TC-LTC, Tandem connection - loss of tandem connection DB9 (256) = TC-TIM, Tandem connection - trace identifier mismatch DB10 (512) = TC-AIS, Tandem connection - alarm indication signal DB11 (1024) = TC-RDI, Tandem connection - remote defect indicator DB12 (2048) = TC-ODI, Tandem connection - outgoing defect indicator DB13 - DB16 = NOT USED <section> = 3: DB1 (1) = STL LOF DB2 (2) = STL OOF DB3 (4) = STL LOR DB4 (8) = STL OOR DB5 (16) = OLA DB6 - DB16 = NOT USED
Example	`SDH:STAT:RX1:ALAR1?` → `1`
Note	

10.5.4 SDH:STATus:RX<Pt>:ALARm<section>:CONDition?

Syntax	SDH:STATus:RX<Pt>:ALARm<section>:CONDition?
Description	This query returns one of the alarms condition registers.
Parameters	<Pt> = Port number
	<section> = Physical- and high-order path alarms(1), Low-order path and tandem connection alarms(2) or STL alarms(3)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> Same as SDH:STATus:RX<Pt>:ALARm<section>[:EVENt]?
Example	`SDH:STAT:RX1:ALAR1:COND?` → `1`
Note	

10.5.5 SDH:STATus:RX<Pt>:ERRor<section>[:EVENt]?

Syntax	SDH:STATus:RX<Pt>:ERRor<section>[:EVENt]?
Description	This query returns the errors event register. The content of this register is summarized in DB3/DB4/DB6 of the SDH:STATus:RX<Pt>:AESummary:CONDition register.
Parameters	<Pt> = Port number
	<section> = (1-3)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> <section> = 1: DB1 (1) = A1A2 DB2 (2) = B1 DB3 (4) = B2 DB4 (8) = MS-REI DB5 (16) = B3 DB6 (32) = HP-REI DB7 (64) = V5/LP-B3 DB8 (128) = LP-REI DB9 (256) = PRBS DB10 (512) = TU-NDF DB11 (1024) = AU-NDF DB12 (2048) = APS DB13 - DB16 = NOT USED <section> = 2: DB1 (1) = TU-NEG DB2 (2) = TU-POS DB3 (4) = AU-NEG DB4 (8) = AU-POS DB5 not used DB6 (32) = TC-IEC DB7 (64) = TC-BIP-2 DB8 (128) = TC-REI DB9 (256) = TC-OEI DB10 - DB16 = NOT USED <section> = 3: DB1 (1) = A1A2-STL DB2 - DB16 = NOT USED
Example	`SDH:STAT:RX1:ERR1?` → `3`
Note	

10.5.6 SDH:STATus:RX<Pt>:ERRor<section>:CONDition?

Syntax	SDH:STATus:RX<Pt>:ERRor<section>:CONDition?
Description	This query returns errors condition register.
Parameters	<Pt> = Port number
	<section> = (1-3)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> Same as SDH:STATus:RX<Pt>:ERRor<section>[:EVENt]?
Example	`SDH:STAT:RX1:ERR1:COND?` → `3`
Note	

10.5.7 SDH:STATus:RX<Pt>:PSLevel?

Syntax	SDH:STATus:RX<Pt>:PSLevel?
Description	This query returns the physical signal level. Unit: dBm.
Parameter	<Pt> = Port number
Response	<signallevel> = <STRING RESPONSE DATA> "N/A": Module not present or not ready. - Electrical(STM1) - "<power> dBm": Min: "< -48 dBm", Max: "Exceeds Level" - Optical(Both module types) - "<power> dBm": Min: "< -27 dBm", Max: "Exceeds Level"
Example	`SDH:STAT:RX1:PSL?` → `"-3 dBm"`
Note	

10.5.8 SDH:STATus:TX<Pt>:PSLevel?

Syntax	SDH:STATus:TX<Pt>:PSLevel?
Description	This query returns the physical signal level. Unit: dBm.
Parameter	<Pt> = Port number
Response	<signallevel> = <STRING RESPONSE DATA> "<power> dBm": Min: "< -27 dBm", Max: "Exceeds Level" "N/A": Module not present or not ready.
Example	`SDH:STAT:TX1:PSL?` → `"-3 dBm"`
Note	Only available for Optical.

10.5.9 SDH:STATus:RX<Pt>:PDEViation?

Syntax	SDH:STATus:RX<Pt>:PDEViation? [<unit>]
Description	This query returns the physical deviation.
Parameters	<Pt> = Port number
	<unit> = <CHARACTER PROGRAM DATA> PPM = Parts per million BPS = Bits per second *DEFault = PPM*
Response	<deviation> = <NR2 NUMERIC RESPONSE DATA>
Example	`SDH:STAT:RX1:PDEV? PPM` → `0`
Note	

10.5.10 SDH:STATus:RX<Pt>:PBRate?

Syntax	SDH:STATus:RX<Pt>:PBRate?
Description	This query returns the physical bit rate. Unit: bps.
Parameter	<Pt> = Port number
Response	<bitrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:STAT:RX1:PBR?` → `155520000`
Note	

10.5.11 SDH:STATus:RX<Pt>:DIFFerence?

Syntax	SDH:STATus:RX<Pt>:DIFFerence?
Description	This query returns physical bit rate difference between port A and port B (RX1 - RX2). Units: ppm, bps and bits.
Parameter	<Pt> = Port number
Response	<ppm> = <NR1 NUMERIC RESPONSE DATA>
	<bps> = <NR1 NUMERIC RESPONSE DATA>
	<acc> = <NR1 NUMERIC RESPONSE DATA> Accumulated difference in bits.
Example	`SDH:STAT:RX1:DIFF?` → `-1,-4,-324`
Note	A valid response is only available if both RX1 and RX2 are on. Using either RX1 or RX2 will give the same results.

10.5.12 SDH:STATus:RX<Pt>:RACCumulated

Syntax	SDH:STATus:RX<Pt>:RACCumulated
Description	This command resets the accumulated difference.
Parameter	<Pt> = Port number
Response	None.
Example	`SDH:STAT:RX1:RACC`
Note	Using either RX1 or RX2 will give the same results.

10.5.13 SDH:STATus:RX<Pt>:PPBRate?

Syntax	SDH:STATus:RX<Pt>:PPBRate?
Description	This query returns physical pattern bit rate. Unit: bps.
Parameter	<Pt> = Port number
Response	<bitrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:STAT:RX1:PPBR?` → `149760000`
Note	

10.5.14 SDH:STATus:RX<Pt>:CAPTure:SOH:TRACe?

Syntax	SDH:STATus:RX<Pt>:CAPTure:SOH:TRACe?
Description	This query returns the SOH trace (J0) for the latest captured frames.
Parameter	<Pt> = Port number
Response	<J0_trace> = <STRING RESPONSE DATA>
Example	`SDH:STAT:RX1:CAPT:SOH:TRAC?` → `"Message_Test_J0"`
Note	If one of the alarms LOS or LOF is present, an empty string is returned.

10.5.15 SDH:STATus:RX<Pt>:CAPTure<Frame>:SOH?

Syntax	SDH:STATus:RX<Pt>:CAPTure<Frame>:SOH? <SOH-byte>
Description	This query returns the SOH bytes from the selected frame. 256 new frames are captured every second.
Parameters	<Pt> = Port number
	<Frame> = Frame number (1-64)
	<SOH-byte> = <CHARACTER PROGRAM DATA> A1: Returns 3 bytes. A2: Returns 3 bytes. J0: Returns 3 bytes. B1: Returns 3 bytes. E1: Returns 3 bytes. F1: Returns 3 bytes. D1: Returns 3 bytes. D2: Returns 3 bytes. D3: Returns 3 bytes. H1: Returns 3 bytes. H2: Returns 3 bytes. H3: Returns 3 bytes. B2: Returns 3 bytes. K1: Returns 3 bytes. K2: Returns 3 bytes. D4: Returns 3 bytes. D5: Returns 3 bytes. D6: Returns 3 bytes. D7: Returns 3 bytes. D8: Returns 3 bytes. D9: Returns 3 bytes. D10: Returns 3 bytes. D11: Returns 3 bytes. D12: Returns 3 bytes. S1: Returns 3 bytes. Z2: Returns 3 bytes. E2: Returns 3 bytes. M0: Returns 1 byte. [1] M1: Returns 1 byte.
Response	<byte1>[,<byte2>[,<byte3>]] = <HEXADECIMAL NUMERIC RESPONSE DATA> Refer to <SOH-byte> parameter description above to see how many bytes this command returns.

Examples	`SDH:STAT:RX1:CAPT64:SOH? A1` → `#HF6,#HF6,#HF6`
	`SDH:STAT:RX2:CAPT23:SOH? H1` → `#H69,#H93,#H93`
	`SDH:STAT:RX1:CAPT1:SOH? M1` → `#H00`

Note	If one of the alarms LOS or LOF is present NaN (section 1.6.1) is returned. [1] Only valid for SDH:RX<Pt>:STMLevel = 64 or 256.

10.5.16 SDH:STATus:RX<Pt>:CAPTure:VC4:POH:TRACe?

Syntax	SDH:STATus:RX<Pt>:CAPTure:VC4:POH:TRACe?
Description	This query returns the VC4 path overhead trace (J1) for the latest captured frames. 64 new frames are captured every second.
Parameter	<Pt> = Port number
Response	<J1_trace> = <STRING RESPONSE DATA>
Example	`SDH:STAT:RX1:CAPT:VC4:POH:TRAC?` → `"Message_Test_J1"`
Note	If one of the alarms LOS, LOF, HPUNEQ or AULOP is present, an empty string is returned.

10.5.17 SDH:STATus:RX<Pt>:CAPTure<Frame>:VC4:POH?

Syntax	SDH:STATus:RX<Pt>:CAPTure<Frame>:VC4:POH? <POH-byte>
Description	This query returns the VC4 path overhead bytes from the selected frame. 256 new frames are captured every second.
Parameters	<Pt> = Port number
	<Frame> = Frame number (1-64)
	<POH-byte> = <CHARACTER PROGRAM DATA> J1: J1 byte. B3: B3 byte. C2: C2 byte. G1: G1 byte. F2: F2 byte. H4: H4 byte. F3: F3 byte. K3: K3 byte. N1: N1 byte.
Response	<byte> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:STAT:RX1:CAPT1:VC4:POH? H4` → `#HFF`
Note	If one of the alarms LOS, LOF, HPUNEQ or AULOP is present NaN (section 1.6.1) is returned.

10.5.18 SDH:STATus:RX<Pt>:CAPTure:VC3:POH:TRACe?

Syntax	SDH:STATus:RX<Pt>:CAPTure:VC3:POH:TRACe?
Description	This command queries the VC3 path overhead trace (J1) for the latest captured frames. 64 new frames are captured every second.
Parameter	<Pt> = Port number
Response	<J1_trace> = <STRING RESPONSE DATA>
Example	`SDH:STAT:RX1:CAPT:VC3:POH:TRAC?` → `"Message_Test_J1"`
Note	If one of the alarms LOS, LOF, HPUNEQ or LPUNEQ is present, an empty string is returned.

10.5.19 SDH:STATus:RX<Pt>:CAPTure<Frame>:VC3:POH?

Syntax	SDH:STATus:RX<Pt>:CAPTure<Frame>:VC3:POH? <POH-byte>
Description	This query returns the VC3 path overhead bytes from the selected frame. 64 new frames are captured every second.
Parameters	<Pt> = Port number
	<Frame> = Frame number (1-64)
	<POH-byte> = <CHARACTER PROGRAM DATA> J1: J1 byte. B3: B3 byte. C2: C2 byte. G1: G1 byte. F2: F2 byte. H4: H4 byte. F3: F3 byte. K3: K3 byte. N1: N1 byte.
Response	<byte> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:STAT:RX1:CAPT1:VC3:POH? C2` → `#HA3`
Note	If one of the alarms LOS, LOF, HPUNEQ or LPUNEQ is present NaN (section 1.6.1) is returned. Use the SDH:RX<Pt>:VC3:POH:CMASk command to selected which bytes to bee captured.

10.5.20 SDH:STATus:RX<Pt>:CAPTure:VC12:POH:TRACe?

Syntax	SDH:STATus:RX<Pt>:CAPTure:VC12:POH:TRACe?
Description	This query returns VC12 path overhead trace (J2) for the latest captured frames. 64 new frames are captured every second.
Parameter	<Pt> = Port number
Response	<J2_trace> = <STRING RESPONSE DATA>
Example	`SDH:STAT:RX1:CAPT:VC12:POH:TRAC?` → `"Message_Test_J2"`
Note	If one of the alarms LOS, LOF, HPUNEQ or LPUNEQ is present, an empty string is returned.

10.5.21 SDH:STATus:RX<Pt>:CAPTure<Frame>:VC12:POH?

Syntax	SDH:STATus:RX<Pt>:CAPTure<Frame>:VC12:POH? <POH-byte>
Description	This query returns the VC12 path overhead bytes from the selected frame. 64 new frames are captured every second.
Parameters	<Pt> = Port number
	<Frame> = Frame number (1-64)
	<POH-byte> = <CHARACTER PROGRAM DATA> V5: V5-SL byte. J2: J2 byte. N2: N2 byte. K4: K4 byte.
Response	<byte> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:STAT:RX1:CAPT1:VC12:POH? V5` → `#H04`
Note	If one of the alarms LOS, LOF, HPUNEQ or LPUNEQ is present NaN (section 1.6.1) is returned.

10.5.22 SDH:STATus:RX<Pt>:CAPTure:VC11:POH:TRACe?

Syntax	SDH:STATus:RX<Pt>:CAPTure:VC11:POH:TRACe?
Description	This query returns VC11 path overhead trace (J2) for the latest captured frames. 64 new frames are captured every second.
Parameter	<Pt> = Port number
Response	<J2_trace> = <STRING RESPONSE DATA>
Example	`SDH:STAT:RX1:CAPT:VC11:POH:TRAC?` → `"Message_Test_J2"`
Note	If one of the alarms LOS, LOF, HPUNEQ or LPUNEQ is present, an empty string is returned.

10.5.23 SDH:STATus:RX<Pt>:CAPTure<Frame>:VC11:POH?

Syntax	SDH:STATus:RX<Pt>:CAPTure<Frame>:VC11:POH? <POH-byte>
Description	This query returns the VC11 path overhead bytes from the selected frame. 64 new frames are captured every second.
Parameters	<Pt> = Port number
	<Frame> = Frame number (1-64)
	<POH-byte> = <CHARACTER PROGRAM DATA> V5: V5-SL byte. J2: J2 byte. N2: N2 byte. K4: K4 byte.
Response	<byte> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:STAT:RX1:CAPT1:VC11:POH? V5` → `#H04`
Note	If one of the alarms LOS, LOF, HPUNEQ or LPUNEQ is present NaN (section 1.6.1) is returned.

10.5.24 SDH:STATus:RX<Pt>:CAPTure:AU3:VC3:POH:TRACe?

Syntax	SDH:STATus:RX<Pt>:CAPTure:AU3:VC3:POH:TRACe?
Description	This query returns the VC3 path overhead trace (J1) for the latest captured frames. 64 new frames are captured every second.
Parameters	<Pt> = Port number
Response	<J1_trace> = <STRING RESPONSE DATA>
Example	`SDH:STAT:RX1:CAPT:AU3:VC3:POH:TRAC?` → `"Message_Test_J1"`
Note	If one of the alarms LOS, LOF, HPUNEQ or AULOP is present, an empty string is returned.

10.5.25 SDH:STATus:RX<Pt>:CAPTure<Frame>:AU3:VC3:POH?

Syntax	SDH:STATus:RX<Pt>:CAPTure<Frame>:AU3:VC3:POH? <POH-byte>
Description	This query returns the VC3 path overhead bytes from the selected frame. 64 new frames are captured every second.
Parameters	<Pt> = Port number
	<Frame> = Frame number (1-64)
	<POH-byte> = <CHARACTER PROGRAM DATA> J1: J1 byte. B3: B3 byte. C2: C2 byte. G1: G1 byte. F2: F2 byte. H4: H4 byte. F3: F3 byte. K3: K3 byte. N1: N1 byte.
Response	<byte> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SDH:STAT:RX1:CAPT1:AU3:VC3:POH? H4` → `#HFF`
Note	If one of the alarms LOS, LOF, HPUNEQ or AULOP is present NaN (section 1.6.1) is returned.

10.6 APS

10.6.1 SDH:APS:STARt

Syntax	SDH:APS:STARt
Description	This command starts the APS (Automatic Protection Switching) measurement.
Parameter	None.
Response	None.
Example	`SDH:APS:STAR`
Note	

10.6.2 SDH:APS:STOP

Syntax	SDH:APS:STOP
Description	This command stops the APS (Automatic Protection Switching) measurement.
Parameter	None.
Response	None.
Example	`SDH:APS:STOP`
Note	

10.6.3 SDH:APS:RX<Pt>:PINTerpret?

Syntax	SDH:APS:RX<Pt>:PINTerpret?
Description	This query returns the protocol interpretation.
Parameter	<Pt> = Port number
Response	<interpretation> = <STRING RESPONSE DATA>

Example	`SDH:APS:RX1:PINT? →`
	`"00:00:19 | Number 0 | Time:  0.000ms`
	`----------`
	`k1:  Signal degrade (protection)`
	` Destination Node (K1) 2`
	`k2:  Short`
	` Source Node (K2) 7"`

Note	

10.6.4 SDH:APS:RX<Pt>:NUMBer?

Syntax	SDH:APS:RX<Pt>:NUMBer?
Description	This query returns the number of times has measured.
Parameter	<Pt> = Port number
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:APS:RX1:NUMB? → 17`
Note	

10.6.5 SDH:APS:RX<Pt>:ATIMe?

Syntax	SDH:APS:RX<Pt>:ATIMe?
Description	This query returns the average time has measured. Unit: ms.
Parameter	<Pt> = Port number
Response	<avg> = <NR2 NUMERIC RESPONSE DATA>
Example	`SDH:APS:RX1:ATIM? → 4.000`
Note	The maximum measurable time is 10000 ms. The maximum measurable time will be responded if the result exceeds 10000 ms.

10.6.6 SDH:APS:RX<Pt>:MTIMe?

Syntax	SDH:APS:RX<Pt>:MTIMe?
Description	This query returns the maximum time has measured. Unit: ms.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
	<limit exceeded> = <NR1 NUMERIC RESPONSE DATA> Returns 1, if the reference maximum limit has been exceeded.
Example	`SDH:APS:RX1:MTIM?` → `4.000,0`
Note	The maximum measurable time is 10000 ms. The maximum measurable time will be responded if the result exceeds 10000 ms.

10.6.7 SDH:APS:RX<Pt>:LTIMe?

Syntax	SDH:APS:RX<Pt>:LTIMe?
Description	This query returns the least (minimum) time has measured. Unit: ms.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`SDH:APS:RX1:LTIM?` → `4.000`
Note	The maximum measurable time is 10000 ms. The maximum measurable time will be responded if the result exceeds 10000 ms.

10.6.8 SDH:APS:RX<Pt>:CTIMe?

Syntax	SDH:APS:RX<Pt>:CTIMe?
Description	This query returns the current time has measured. Unit: ms.
Parameter	<Pt> = Port number
Response	<current> = <NR2 NUMERIC RESPONSE DATA>
Example	`SDH:APS:RX1:CTIM?` → `4.000`
Note	The maximum measurable time is 10000 ms. The maximum measurable time will be responded if the result exceeds 99999.999 ms.

10.6.9 SDH:APS:RX<Pt>:EVENt

Syntax	SDH:APS:RX<Pt>:EVENt <event>
Description	This command sets the time reference event.
Parameters	<Pt> = Port number
	<event> = <CHARACTER PROGRAM DATA> LOS = Loss of signal LOF = Loss of frame OOF = Out of frame MSAIS = MS alarm indication signal MSRDI = MS remote defect indicator APSS = APS switch-over AUAIS = AU alarm indication signal AULOP = AU loss of pointer HPTIM = HP trace identifier mismatch HPPLM = HP payload label mismatch HPUNEQ = HP unequipped TULOM = TU loss of multiframe TUAIS = TU alarm indication signal TULOP = TU loss of pointer LPTIM = LP trace identifier mismatch LPPLM = LP payload label mismatch LPUNEQ = LP unequipped A1A2 = Frame alignment word error B1 = B1 checksum byte error B2 = B2 checksum byte error MSREI = MS remote error indication B3 = B3 checksum byte error V5 = B3 checksum of the low-order path PERRor = Pattern error ANYERROR = Any errors *DEFault = LOS*
Response	None.
Example	`SDH:APS:RX1:EVEN LOF`
Note	

Syntax	SDH:APS:RX<Pt>:EVENt?
Description	This query returns the time reference event.
Parameter	<Pt> = Port number
Response	<event> = <CHARACTER RESPONSE DATA>
Example	`SDH:APS:RX1:EVEN?` → `LOF`
Note	

10.6.10 SDH:APS:RX<Pt>:MLIMit

Syntax	SDH:APS:RX<Pt>:MLIMit <max>
Description	This command sets the time reference maximum limit. Unit: ms.
Parameters	<Pt> = Port number
	<max> = <NUMERIC PROGRAM DATA> *MINimum = 0.000, MAXimum = 10000.000, DEFault = 50.000*
Response	None.
Example	`SDH:APS:RX1:MLIM 50.000`
Note	

Syntax	SDH:APS:RX<Pt>:MLIMit?
Description	This query returns the time reference maximum limit. Unit: ms.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
Example	`SDH:APS:RX1:MLIM?` → `50.000`
Note	

10.6.11 SDH:APS:RX<Pt>:PERiod

Syntax	SDH:APS:RX<Pt>:PERiod <period>
Description	This command sets the error free period.
Parameters	<Pt> = Port number
	<period> = <NUMERIC PROGRAM DATA> 1,10,20,30,40,50,60,70,80,90,100 Unit ms *DEFault = 1*
Response	None
Example	`SDH:APS:RX1:PER 1`
Note	

Syntax	SDH:APS:RX<Pt>:PERiod?
Description	This query returns the error free period.
Parameter	<Pt> = Port number
Response	<period> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:APS:RX1:PER?` → `1`
Note	

10.6.12 SDH:APS:RX<Pt>:CONFig

Syntax	SDH:APS:RX<Pt>:CONFig <protection>,<path>
Description	This command sets the protection type and the path or architecture.
Parameters	<Pt> = Port number
	<protection> = <CHARACTER PROGRAM DATA> RING = Ring protection LINear = Linear *DEFault = RING*
	<path> = <CHARACTER PROGRAM DATA> Ring Path: SHRT = Short path LONG = Long path *DEFault = SHRT* Linear Architecture: 1P1 = 1+1 Architecture 1N = 1:n Architecture
Response	None.
Example	`SDH:APS:RX1:CONF RING,SHRT`
Notes	

Syntax	SDH:APS:RX<Pt>:CONFig?
Description	This query returns the protection type and the path or architecture.
Parameter	<Pt> = Port number
Response	<protection> = <CHARACTER RESPONSE DATA>
	<path> = <CHARACTER PROGRAM DATA>
Example	`SDH:APS:RX1:CONF?` → `RING,SHRT`
Note	

10.6.13 SDH:APS:RX<Pt>:RTYPe

Syntax	SDH:APS:RX<Pt>:RTYPe <type>
Description	This command sets the request type for the transmitter.
Parameter	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> Ring Request Types: for `SDH:APS:RX<Pt>:CONFig?` → `RING,....` RLOP = Lockout of protection (span) RFSS = Forced switch (span) RFSR = Forced switch (ring) RSFS = Signal fail (span) RSFR = Signal fail (ring) RSDP = Signal degrade (protection) RSDS = Signal degrade (span) RSDR = Signal degrade (ring) RMSS = Manual switch (span) RMSR = Manual switch (ring) RWTR = Wait to restore REXS = Exercise (span) REXR = Exercise (ring) RRRS = Reverse request (span) RRRR = Reverse request (ring) RNRQ = No request *DEFault = RLOP* Linear Request Types: for `SDH:APS:RX<Pt>:CONFig?` → `LIN,....` LLOP = Lockout of protection LFSW = Forced switch LSFH = Signal fail (high priority) LSFL = Signal fail (low priority) LSDH = Signal degrade (high priority) LSDL = Signal degrade (low priority) LMSW = Manual switch (ring) LWTR = Wait to restore LEXC = Exercise LRRQ = Reverse request LDNR = Do not revert LNRQ = No request *DEFault = LLOP*
Response	None.
Example	`SDH:APS:RX1:RTYP RLOP`
Note	Use the SDH:APS:RX<Pt>:APPLy command to apply the request to the transmitter.

Syntax	SDH:APS:RX<Pt>:RTYPe?
Description	This query returns the request type for the transmitter.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`SDH:APS:RX1:RTYP?` → `RLOP`
Note	

10.6.14 SDH:APS:RX<Pt>:K1

Syntax	SDH:APS:RX<Pt>:K1 <value>
Description	This command sets the destination node/source channel (K1).
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=15, DEFault = 0*
Response	None.
Example	`SDH:APS:RX1:K1 3`
Notes	Cannot be changed if the SDH:APS:CONFig command is set to LIN, 1P1. Changes to SDH:APS:CONFig will reset this value to 0.

Syntax	SDH:APS:RX<Pt>:K1?
Description	This query returns the destination node/source channel (K1).
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:APS:RX1:K1?` → `3`
Note	

10.6.15 SDH:APS:RX<Pt>:K2

Syntax	SDH:APS:RX<Pt>:K2 <value>
Description	This command sets the source node/bridged channel (K2).
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=15, DEFault = 0*
Response	None.
Example	`SDH:APS:RX1:K2 3`
Notes	Cannot be changed if the SDH:APS:CONFig command is set to LIN, 1P1. Changes to SDH:APS:CONFig will reset this value to 0.

Syntax	SDH:APS:RX<Pt>:K2?
Description	This query returns the source node/bridged channel (K2).
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:APS:RX1:K2?` → `3`
Note	

10.6.16 SDH:APS:RX<Pt>:APPLy

Syntax	SDH:APS:RX<Pt>:APPLy
Description	This command applies the K1/K2 request setup to the transmitter.
Parameter	<Pt> = Port number
Response	None.
Example	`SDH:APS:RX1:APPL`
Notes	

10.7 Tributary Scan

10.7.1 SDH:TSCan:STARt

Syntax	SDH:TSCan:STARt
Description	This command starts the tributary scan test.
Parameter	None.
Response	None.
Example	`SDH:TSC:STAR`
Note	It is possible to run only one test or measurement at a time. The SDH interface must be active for at least one of the receivers.

10.7.2 SDH:TSCan:STOP

Syntax	SDH:TSCan:STOP
Description	This command stops the tributary scan test.
Parameter	None.
Response	None.
Example	`SDH:TSC:STOP`
Note	

10.7.3 SDH:TSCan:RX<Pt>:NHOContainer?

Syntax	SDH:TSCan:RX<Pt>:NHOContainer?
Description	This query returns the number of High-order containers (VC-4/3)
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:TSC:RX1:NHOC?` → `4`
Note	If '0' is returned, no High-order containers were found. To get more information, use the SDH:TSCan:RX<Pt>:MUX? query.

10.7.4 SDH:TSCan:RX<Pt>:GHOContainer?

Syntax	SDH:TSCan:RX<Pt>:GHOContainer?
Description	This query returns the state of High-order containers (VC-4/3)
Parameter	<Pt> = Port number
Response	({<value>,}+) = <NR1 NUMERIC RESPONSE DATA> The values are presented in ascending order, meaning that VC-4 #1 is the first on the list. 0 = No alarms or errors. 1 = Alarms or errors present.
Example	`SDH:TSC:RX1:GHOC?` → `(0,0,1,0)`
Note	If there is one or more results, the last "," is always removed.

10.7.5 SDH:TSCan:RX<Pt>:SHOContainer

Syntax	SDH:TSCan:RX<Pt>:SHOContainer <VC4>
Description	This command sets the High-order container (VC-4/3) for scanning.
Parameters	<Pt> = Port number
	<VC4> = High-order container number *MINimum=1, MAXimum=768*
Response	None.
Example	`SDH:TSC:RX1:SHOC 2`
Note	

10.7.6 SDH:TSCan:RX<Pt>:SHOContainer?

Syntax	SDH:TSCan:RX<Pt>:SHOContainer?
Description	This query returns the High-order container (VC-4/3) set for scanning.
Parameter	<Pt> = Port number
Response	<VC4> = High-order container number
Example	`SDH:TSC:RX1:SHOC?` → `2`
Note	

10.7.7 SDH:TSCan:RX<Pt>:DHOContainer?

Syntax	SDH:TSCan:RX<Pt>:DHOContainer? <VC4/3>
Description	This query returns the detailed alarm and error information from a High-order container (VC-4/3).
Parameters	<Pt> = Port number
	<VC4> = High-order container number *MINimum=1, MAXimum=768*
Response	<selected>, = <STRING RESPONSE DATA>. {RXn}␣{High-order}, separated by one space character. RXn = RX1 or RX2 High-order = VC-4#0 . . . VC-4#64 or VC-3#0 . . . VC-3#192
	({<alarmserrors>,}∗) = <STRING RESPONSE DATA> List of alarms and errors.
Example	`SDH:TSC:RX1:DHOC? 1` → `"RX1 VC-4#1",("B1","A1A2")`
Note	If there is one or more responses, the last ”,” is always removed.

10.7.8 SDH:TSCan:RX<Pt>:NLOContainer?

Syntax	SDH:TSCan:RX<Pt>:NLOContainer?
Description	This query returns the number of Low-order containers (VC-3/12/11).
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:TSC:RX1:NLOC?` → `3`
Note	

10.7.9 SDH:TSCan:RX<Pt>:GLOContainer?

Syntax	SDH:TSCan:RX<Pt>:GLOContainer?
Description	This query returns the state of Low-order containers (VC-3/12/11).
Parameter	<Pt> = Port number
Response	({<value>,}∗) = <NR1 NUMERIC RESPONSE DATA> The values are presented in ascending order, meaning that VC-4 #1 is the first in the list. 0 = No alarms or errors. 1 = Alarms or errors present.
Example	`SDH:TSC:RX1:GLOC?` → `(0,1,1)`
Note	If there is one or more responses, the last ”,” is always removed.

10.7.10 SDH:TSCan:RX<Pt>:DLOContainer?

Syntax	SDH:TSCan:RX<Pt>:DLOContainer? <VC3/12/11>
Description	This query returns the detailed alarm and error information from a Low-order container (VC-3/12/11).
Parameters	<Pt> = Port number
	<VC3/12/11> = Low-order container number *MINimum=1, MAXimum=84*
Response	<selected>, = <STRING RESPONSE DATA>. {RXn}␣{High-order}:{Low-order}, First separated by one space character and next with a colon char. RXn = RX1 or RX2 High-order = VC-4#0 . . . VC-4#64 or VC-3#0 . . . VC-3#192 Low-order = VC-3#1 . . . VC-3#3 or VC-12#1 . . . VC-12#63 or VC-11#1 . . . VC-11#84
	({<alarmserrors>, }∗) = <STRING RESPONSE DATA> List of alarms and errors.
Example	`SDH:TSC:RX1:DLOC? 1` → `"RX1 VC-4#1:VC-3#1",("B1","A1A2")`
Notes	To select the High-order container, use the SDH:TSCan:RX<rx>:SHOContainer command. If there is one or more responses, the last ”,” is always removed.

10.7.11 SDH:TSCan:RX<Pt>:DMUX?

Syntax	SDH:TSCan:RX<Pt>:DMUX?
Description	This query returns the detailed alarm and error information from the MUX.
Parameter	<Pt> = Port number
Response	<selected> = <STRING RESPONSE DATA>. {Rx} MUX
	({<alarmserrors>, }∗) = <STRING RESPONSE DATA> List of alarms and errors.
Example	`SDH:TSC:RX1:DMUX?` → `"Rx1 MUX",("LOS","LOF","OOF")`
Notes	This query is only available when no High-order containers are present. To determine if MUX is available, use the SDH:TSCan:RX<Pt>:NHOContainer? query. If there is one or more responses, the last ”,” is always removed.

10.8 RTD

This section document commands for the Round Trip Delay application.
Commands for general RTD settings are described in section 17.1 on page 953.

10.8.1 SDH:RTD:RX<Pt>:MLIMit

Syntax	SDH:RTD:RX<Pt>:MLIMit <max>
Description	This command sets the time reference maximum limit. Unit: us.
Parameters	<Pt> = Port number
	<max> = <NUMERIC PROGRAM DATA> *MINimum = 0.0, MAXimum = 10000000.0, DEFault = MAXimum*
Response	None.
Example	`SDH:RTD:RX1:MLIM 0.0`
Note	

Syntax	SDH:RTD:RX<Pt>:MLIMit?
Description	This query returns the time reference maximum limit. Unit: us.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
Example	`SDH:RTD:RX1:MLIM? → 0.0`
Note	

10.8.2 SDH:RTD:RX<Pt>:NUMBer?

Syntax	SDH:RTD:RX<Pt>:NUMBer?
Description	This query returns the number of the RTD data.
Parameter	<Pt> = Port number
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`SDH:RTD:RX1:NUMB? → 2`
Note	

10.8.3 SDH:RTD:RX<Pt>:ATIMe?

Syntax	SDH:RTD:RX<Pt>:ATIMe?
Description	This query returns the average time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<avg> = <NR2 NUMERIC RESPONSE DATA>
Example	`SDH:RTD:RX1:ATIM? → 1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

10.8.4 SDH:RTD:RX<Pt>:MTIMe?

Syntax	SDH:RTD:RX<Pt>:MTIMe?
Description	This query returns the maximum time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
	<limit exceeded> = <NR1 NUMERIC RESPONSE DATA> Returns 1, if the reference maximum limit has been exceeded.
Example	`SDH:RTD:RX1:MTIM? → 1.0,0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

10.8.5 SDH:RTD:RX<Pt>:LTIMe?

Syntax	SDH:RTD:RX<Pt>:LTIMe?
Description	This query returns the least (minimum) time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`SDH:RTD:RX1:LTIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

10.8.6 SDH:RTD:RX<Pt>:CTIMe?

Syntax	SDH:RTD:RX<Pt>:CTIMe?
Description	This query returns the current time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`SDH:RTD:RX1:CTIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds 99999999.9 us.

Chapter 11

SONET

Use `MEAS:SET:PORT<Pt>:TERM` to change the SDH/SONET terminology (see section 18.2.1).

11.1 Receiver

11.1.1 SONet:RX<Pt>:INTerface

Syntax	SONet:RX<Pt>:INTerface <mode>
Description	This command sets the used interface or switches off the receiver.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: Disables receiver ELECtrical: Electrical interface SFP: SFP/SFP+ optical interface CFP: CFP optical interface OPTical: Obsolete. For CMA 3000 backward compatibility only. Same as SFP *DEFault = OFF*
Response	None.
Example	`SON:RX1:INT OFF`
Note	

Syntax	SONet:RX<Pt>:INTerface?
Description	This query returns the used interface.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SON:RX1:INT?` → `OFF`
Note	Returns OTN when SONET over OTN.

11.1.2 SONet:RX<Pt>:OCLevel

Syntax	SONet:RX<Pt>:OCLevel <level>
Description	This command sets the OC level of the SONET receiver.
Parameters	<Pt> = Port number
	<level> = <CHARACTER PROGRAM DATA> 3: OC-3 signal. 12: OC-12 signal. 48: OC-48 signal. 192: OC-192 signal. 768: OC-768 signal. *DEFault = 3*
Response	None.
Example	`SON:RX1:OCL 3`
Note	Setting the OC level may change the STS-3cSPE concatenation level.

Syntax	SONet:RX<Pt>:OCLevel?
Description	This query returns the OC level of the SONET receiver.
Parameter	<Pt> = Port number
Response	<level> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:RX1:OCL?` → `3`
Note	

11.1.3 SONet:RX<Pt>:STSLevel

Syntax	SONet:RX<Pt>:STSLevel <level>
Description	This command sets the STS level of the SONET receiver.
Parameters	<Pt> = Port number
	<level> = <CHARACTER PROGRAM DATA> 3: STS-3 signal. 12: STS-12 signal. 48: STS-48 signal. 192: STS-192 signal. 768: STS-768 signal. *DEFault = 3*
Response	None.
Example	`SON:RX1:STSL 3`
Note	Setting the STS level may change the STS-3cSPE concatenation level.

Syntax	SONet:RX<Pt>:STSLevel?
Description	This query returns the STS level of the SONET receiver.
Parameter	<Pt> = Port number
Response	<level> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:RX1:STSL?` → `3`
Note	

11.1.4 SONet:RX<Pt>:STS3c

Syntax	SONet:RX<Pt>:STS3c <sts>
Description	This command sets the STS Group(s) to be used in the signal structure.
Parameters	<Pt> = Port number
	<sts> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 768, DEFault = 1*
Response	None.
Example	`SON:RX1:STS3 1`
Note	The specified STS3c cannot exceed the OC level divided by the STS-3cSPE concatenation level.

Syntax	SONet:RX<Pt>:STS3c?
Description	This query returns STS Group(s) used in the signal structure.
Parameter	<Pt> = Port number
Response	<sts> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:RX1:STS3?` → `1`
Note	

11.1.5 SONet:RX<Pt>:CONCatenation

Syntax	SONet:RX<Pt>:CONCatenation <level>
Description	This command sets the STS concatenation level.
Parameters	<Pt> = Port number
	<level> = <CHARACTER PROGRAM DATA> STS1SPE STS3CSPE STS12C STS48C STS192C STS768C *DEFault = STS3CSPE*
Response	None.
Example	`SON:RX1:CONC STS3CSPE`
Note	Setting the STS concatenation level may change the OC level.

Syntax	SONet:RX<Pt>:CONCatenation?
Description	This query returns the STS concatenation level.
Parameter	<Pt> = Port number
Response	<level> = <CHARACTER RESPONSE DATA>
Example	`SON:RX1:CONC?` → `STS3CSPE`
Note	

11.1.6 SONet:RX<Pt>:CONTainer

Syntax	SONet:RX<Pt>:CONTainer <type>
Description	This command sets the container type (container-n).
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> C3 C4 C11 C12 *DEFault = C4*
Response	None.
Example	`SON:RX1:CONT C4`
Note	Setting the container type.

Syntax	SONet:RX<Pt>:CONTainer?
Description	This query returns the container type (container-n).
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`SON:RX1:CONT?` → `C4`
Note	

11.1.7 SONet:RX<Pt>:SSTRucture

Syntax	SONet:RX<Pt>:SSTRucture <structure>
Description	This command sets the expected SONET signal structure.
Parameters	<Pt> = Port number
	<structure> = <CHARACTER PROGRAM DATA> BULK: Bulk data in the STS3. E1: E1 signal in a VT2 in one or all STS's E3: E3 signal in a STS1 in one or all STS's DS1: DS1 signal in a VT15 in one or all STS's DS3: DS3 signal in a STS1 in one or all STS's E4: E4 signal in a STS3 in one or all STS's *DEFault = BULK*
Response	None.
Example	`SON:RX1:SSTR BULK`
Note	

Syntax	SONet:RX<Pt>:SSTRucture?
Description	This query returns the expected SONET signal structure.
Parameter	<Pt> = Port number
Response	<structure> = <CHARACTER RESPONSE DATA>
Example	`SON:RX1:SSTR? → BULK`
Note	

11.1.8 SONet:RX<Pt>:PATTern

Syntax	SONet:RX<Pt>:PATTern <pattern>
Description	This command sets the pattern to be expected as payload when SSTRucture is BULK.
Parameters	<Pt> = Port number
	<pattern> = <CHARACTER PROGRAM DATA> OFF USER32BIT: 32 bit user defined pattern. USER2048BIT: 2048 bit user defined pattern. PRBS9: PRBS-9 pattern. PRBS11: PRBS-11 pattern. PRBS15: PRBS-15 pattern. PRBS20: PRBS-20 pattern. PRBS23: PRBS23 pattern. PRBS29: PRBS-29 pattern. PRBS31: PRBS-31 pattern. ALL0: Pattern of all zeros. ALL1: Pattern of all ones. ALT11: Alternating 1:1 ALT13: Alternating 1:3 ALT17: Alternating 1:7 B2IN8: Random pattern with two ones for every eight bits. *DEFault = PRBS23*
Response	None.
Example	`SON:RX1:PATT PRBS23`
Note	

Syntax	SONet:RX<Pt>:PATTern?
Description	This query returns the pattern to be expected as payload when SSTRucture is BULK.
Parameter	<Pt> = Port number
Response	<pattern> = <CHARACTER RESPONSE DATA>
Example	`SON:RX1:PATT? → PRBS23`
Note	

11.1.9 SONet:RX<Pt>:UP32

Syntax	SONet:RX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters (one bit resolution).
Response	None.
Examples	`SON:RX1:UP32 "01101"`
Note	

Syntax	SONet:RX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`SON:RX1:UP32?` → `"01101"`
Note	

11.1.10 SONet:RX<Pt>:UP2K

Syntax	SONet:RX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`SON:RX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `SON:RX1:PATT` is `USER2048BIT`

Syntax	SONet:RX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`SON:RX1:UP2K?` → `"12DF"`
Note	

11.1.11 SONet:RX<Pt>:PINVersion

Syntax	SONet:RX<Pt>:PINVersion <inverted>
Description	This command enables or disables PRBS pattern inversion (when SSTRucture is BULK).
Parameters	<Pt> = Port number
	<inverted> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`SON:RX1:PINV OFF`
Note	

Syntax	SONet:RX<Pt>:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the PRBS pattern.
Parameter	<Pt> = Port number
Response	<inverted> = <BOOLEAN RESPONSE DATA>
Example	`SON:RX1:PINV?` → `0`
Note	

11.1.12 SONet:RX<Pt>:TCM

Syntax	SONet:RX<Pt>:TCM <mode>
Description	This command sets the Tandem Connection Monitoring mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: Off STS3: Z5 (STS3) STS1: Z5 (STS1) VT2: Z6 (VT2) VT15: Z6 (VT1.5) *DEFault = OFF*
Response	None.
Example	`SON:RX1:TCM OFF`
Note	

Syntax	SONet:RX<Pt>:TCM?
Description	This query returns Tandem Connection Monitoring mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SON:RX1:TCM?` → `OFF`
Note	

11.1.13 SONet:RX<Pt>:TUG3

Syntax	SONet:RX<Pt>:TUG3 <id>
Description	This command sets the TUG-3 number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: TUG-3 number one. 2: TUG-3 number two. 3: TUG-3 number three. *MINimum = 1, MAXimum = 3, DEFault = 1*
Response	None.
Example	`SON:RX1:TUG3 1`
Note	This value influences the channel number.

Syntax	SONet:RX<Pt>:TUG3?
Description	This query returns TUG3 number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:RX1:TUG3?` → `1`
Note	

11.1.14 SONet:RX<Pt>:VTG

Syntax	SONet:RX<Pt>:VTG <id>
Description	This command sets the VTG number for signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: VTG number one. 2: VTG number two. ... 7: VTG number seven. *MINimum = 1, MAXimum = 7, DEFault = 1*
Response	None.
Example	`SON:RX1:VTG 1`
Note	This value influences the channel number.

Syntax	SONet:RX<Pt>:VTG?
Description	This query returns VTG number for signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:RX1:VTG?` → `1`
Note	

11.1.15 SONet:RX<Pt>:VT2

Syntax	SONet:RX<Pt>:VT2 <id>
Description	This command sets the VT2 number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: VT2 number one. 2: VT2 number two. 3: VT2 number three. *MINimum = 1, MAXimum = 3, DEFault = 1*
Response	None.
Example	`SON:RX1:VT2 1`
Note	This value influences the channel number.

Syntax	SONet:RX<Pt>:VT2?
Description	This query returns VT2 number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:RX1:VT2?` → `1`
Note	

11.1.16 SONet:RX<Pt>:VT15

Syntax	SONet:RX<Pt>:VT15 <id>
Description	This command sets the VT1.5 number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: VT1.5 number one. 2: VT1.5 number two. 3: VT1.5 number three. 4: VT1.5 number four. *MINimum = 1, MAXimum = 4, DEFault = 1*
Response	None.
Example	`SON:RX1:VT15 1`
Note	This value influences the channel number.

Syntax	SONet:RX<Pt>:VT15?
Description	This query returns VT1.5 number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:RX1:VT15?` → `1`
Note	

11.1.17 SONet:RX<Pt>:CHANnel

Syntax	SONet:RX<Pt>:CHANnel <id>
Description	This command sets the channel number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=84, DEFault = 1*
Response	None.
Example	`SON:RX1:CHAN 1`
Note	This value influences the TUG-3, VTG, VT2 and VT1.5 numbers.

Syntax	SONet:RX<Pt>:CHANnel?
Description	This query returns the channel number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:RX1:CHAN? → 1`
Note	

11.1.18 SONet:RX<Pt>:MAPPing

Syntax	SONet:RX<Pt>:MAPPing <type>
Description	This command sets the mapping type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> SYNChronous: Synchronous mapping. ASYNchronous: Asynchronous mapping. *DEFault = SYNC*
Response	None.
Example	`SON:RX1:MAPP SYNC`
Note	

Syntax	SONet:RX<Pt>:MAPPing?
Description	This query returns returns the mapping type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`SON:RX1:MAPP? → SYNC`
Note	

11.1.19 SONet:RX<Pt>:GAIN

Syntax	SONet:RX<Pt>:GAIN <type>
Description	This command sets the receiver gain.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> TERMinate: Frequency dependent AGC. MONitor: Frequency linear AGC. *DEFault = TERMinate*
Response	None.
Example	`SON:RX1:GAIN TERM`
Note	

Syntax	SONet:RX<Pt>:GAIN?
Description	This query returns the receiver gain.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`SON:RX1:GAIN? → TERM`
Note	

11.1.20 SONet:RX<Pt>:FOLLow

Syntax	SONet:RX<Pt>:FOLLow <mode>
Description	This command sets the receiver setup to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> NONE: Do not follow. TX: Follows setup of the Tx port. RX1: Follows setup of the Rx port1. *DEFault = NONE*
Response	None.
Example	`SON:RX1:FOLL NONE`
Note	

Syntax	SONet:RX<Pt>:FOLLow?
Description	This query returns the receiver setup to follow another setup or not to follow.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SON:RX1:FOLL?` → `NONE`
Note	

11.1.21 SONet:RX<Pt>:MEASurement:PLMP

Syntax	SONet:RX<Pt>:MEASurement:PLMP <detected>
Description	This command enables or disables PLM-P measurement.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SON:RX1:MEAS:PLMP ON`
Note	

Syntax	SONet:RX<Pt>:MEASurement:PLMP?
Description	This query returns whether or not PLM-P measurement is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`SON:RX1:MEAS:PLMP?` → `1`
Note	

11.1.22 SONet:RX<Pt>:MEASurement:PLMV

Syntax	SONet:RX<Pt>:MEASurement:PLMV <detected>
Description	This command enables or disables PLM-V measurement.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SON:RX1:MEAS:PLMV ON`
Note	

Syntax	SONet:RX<Pt>:MEASurement:PLMV?
Description	This query returns whether or not PLM-V measurement is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`SON:RX1:MEAS:PLMV?` → `1`
Note	

11.2 Transmitter

11.2.1 SONet:TX<Pt>:INTerface

Syntax	SONet:TX<Pt>:INTerface <mode>
Description	This command sets the used interface of the transmitter.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> ELECtrical: Electrical interface SFP: SFP/SFP+ optical interface CFP: CFP optical interface *DEFault = ELEC*
Response	None.
Example	`SON:TX1:INT SFP`
Note	

Syntax	SONet:TX<Pt>:INTerface?
Description	This query returns the used interface.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SON:TX1:INT?` → `SFP`
Note	Returns OTN when SONET over OTN.

11.2.2 SONet:TX<Pt>[:ENABle]

Syntax	SONet:TX<Pt>[:ENABle] <mode>
Description	This command sets the mode of the transmitter.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: Turn off the transmitter. NORMal: Normal transmission mode using the configured signal. THRough: Through mode. The signal from the receiver is transmitted. OTHRough: OH overwrite through mode. THA: Obsolete. For CMA 3000 backward compatibility only. Same as THRough. *DEFault = OFF*
Response	None.
Example	`SON:TX1 NORM`
Note	

Syntax	SONet:TX<Pt>[:ENABle]?
Description	This query returns the mode of the transmitter.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SON:TX1?` → `NORM`
Note	

11.2.3 SONet:TX<Pt>:OPTical

Syntax	SONet:TX<Pt>:OPTical <mode>
Description	Obsolete. For CMA 3000 backward compatibility only. Same as SON:TX<Pt>:INT SFP and SON:TX<Pt>
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA>
Response	None.
Example	`SON:TX1:OPT NORM`
Note	Compared to CMA 3000 there is no longer a suffix on the last node.

Syntax	SONet:TX<Pt>:OPTical?
Description	Obsolete. For CMA 3000 backward compatibility only. Same as SON:TX<Pt>?
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SON:TX1:OPT?` → `NORM`
Note	

11.2.4 SONet:TX<Pt>:ELECtrical

Syntax	SONet:TX<Pt>:ELECtrical <mode>
Description	Obsolete. For CMA 3000 backward compatibility only. Same as SON:TX<Pt>:INT ELEC and SON:TX<Pt>
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA>
Response	None.
Example	`SON:TX1:ELEC NORM`
Note	Compared to CMA 3000 mode=THB is no longer supported.

Syntax	SONet:TX<Pt>:ELECtrical?
Description	Obsolete. For CMA 3000 backward compatibility only. Same as SON:TX<Pt>?
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SON:TX1:ELEC?` → `NORM`
Note	

11.2.5 SONet:TX<Pt>:OCLevel

Syntax	SONet:TX<Pt>:OCLevel <level>
Description	This command sets the OC level of the transmitted SONET signal.
Parameters	<Pt> = Port number
	<level> = <CHARACTER PROGRAM DATA> 3: OC-3 signal. 12: OC-12 signal. 48: OC-48 signal. 192: OC-192 signal. 768: OC-768 signal. *DEFault = 3*
Response	None.
Example	`SON:TX1:OCL 3`
Notes	Setting the OC level may change the STS-3cSPE concatenation level.

Syntax	SONet:TX<Pt>:OCLevel?
Description	This query returns the OC level of the transmitted SONET signal.
Parameter	<Pt> = Port number
Response	<level> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:TX1:OCL?` → `1`
Note	

11.2.6 SONet:TX<Pt>:STSLevel

Syntax	SONet:TX<Pt>:STSLevel <level>
Description	This command sets the STS level of the SONET receiver.
Parameters	<Pt> = Port number
	<level> = <CHARACTER PROGRAM DATA> 3: STS-3 signal. 12: STS-12 signal. 48: STS-48 signal. 192: STS-192 signal. 768: STS-768 signal. *DEFault = 3*
Response	None.
Example	`SON:TX1:STSL 3`
Note	Setting the STS level may change the STS-3cSPE concatenation level.

Syntax	SONet:TX<Pt>:STSLevel?
Description	This query returns the STS level of the SONET receiver.
Parameter	<Pt> = Port number
Response	<level> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:TX1:STSL?` → `3`
Note	

11.2.7 SONet:TX<Pt>:STS3c

Syntax	SONet:TX<Pt>:STS3c <value>
Description	This command sets the STS where the test signal is inserted.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> ALL 1 to 768 *DEFault = ALL*
Response	None.
Example	`SON:TX1:STS3 ALL`
Note	The specified STS3c cannot exceed the OC level divided by the STS-3cSPE concatenation level.

Syntax	SONet:TX<Pt>:STS3c?
Description	This query returns the STS where the test signal has been inserted.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:TX1:STS3?` → `ALL`
Note	

11.2.8 SONet:TX<Pt>:CONCatenation

Syntax	SONet:TX<Pt>:CONCatenation <level>
Description	This command sets the STS concatenation level.
Parameters	<Pt> = Port number
	<level> = <CHARACTER PROGRAM DATA> STS1SPE STS3CSPE STS12C STS48C STS192C STS768C *DEFault = STS3CSPE*
Response	None.
Example	`SON:TX1:CONC STS3CSPE`
Note	Setting the STS concatenation level may change the STM level.

Syntax	SONet:TX<Pt>:CONCatenation?
Description	This query returns the STS concatenation level.
Parameter	<Pt> = Port number
Response	<level> = <CHARACTER RESPONSE DATA>
Example	`SON:TX1:CONC?` → `STS3CSPE`
Note	

11.2.9 SONet:TX<Pt>:CONTainer

Syntax	SONet:TX<Pt>:CONTainer <type>
Description	This command sets the container type (container-n).
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> C3 C4 C11 C12 *DEFault = C4*
Response	None.
Example	`SON:TX1:CONT C4`
Note	Setting the container type.

Syntax	SONet:TX<Pt>:CONTainer?
Description	This query returns the container type (container-n).
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`SON:TX1:CONT?` → `C4`
Note	

11.2.10 SONet:TX<Pt>:SSTRucture

Syntax	SONet:TX<Pt>:SSTRucture <structure>
Description	This command sets the SONET signal structure.
Parameters	<Pt> = Port number
	<structure> = <CHARACTER PROGRAM DATA> BULK: Bulk data in the STS3. E1: E1 signal in a VC-12 in one or all STS's E3: E3 signal in a VC-3 in one or all STS's DS1: DS1 signal in a VC-11 in one or all STS's DS3: DS3 signal in a VC-3 in one or all STS's E4: E4 signal in a VC-4 in one or all STS's *DEFault = BULK*
Response	None.
Example	`SON:TX1:SSTR BULK`
Note	

Syntax	SONet:TX<Pt>:SSTRucture?
Description	This query returns the SONET signal structure.
Parameter	<Pt> = Port number
Response	<structure> = <CHARACTER RESPONSE DATA>
Example	`SON:TX1:SSTR?` → `BULK`
Note	

11.2.11 SONet:TX<Pt>:PATTern

Syntax	SONet:TX<Pt>:PATTern <pattern>
Description	This command sets the pattern to be used as payload when SSTRucture is BULK.
Parameters	<Pt> = Port number
	<pattern> = <CHARACTER PROGRAM DATA> OFF USER32BIT: 32 bit user defined pattern. USER2048BIT: 2048 bit user defined pattern. PRBS9: PRBS-9 pattern. PRBS11: PRBS-11 pattern. PRBS15: PRBS-15 pattern. PRBS20: PRBS-20 pattern. PRBS23: PRBS23 pattern. PRBS29: PRBS-29 pattern. PRBS31: PRBS-31 pattern. ALL0: Pattern of all zeros. ALL1: Pattern of all ones. ALT11: Alternating 1:1 ALT13: Alternating 1:3 ALT17: Alternating 1:7 B2IN8: Random pattern with two ones for every eight bits. *DEFault = PRBS23*
Response	None.
Example	`SON:TX1:PATT PRBS23`
Note	

Syntax	SONet:TX<Pt>:PATTern?
Description	Queries the pattern to be used as payload when SSTRucture is BULK.
Parameter	<Pt> = Port number
Response	<pattern> = <CHARACTER RESPONSE DATA>
Example	`SON:TX1:PATT?` → `PRBS23`
Note	

11.2.12 SONet:TX<Pt>:UP32

Syntax	SONet:TX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters (one bit resolution).
Response	None.
Examples	`SON:TX1:UP32 "01101"`
Note	

Syntax	SONet:TX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`SON:TX1:UP32?` → `"01101"`
Note	

11.2.13 SONet:TX<Pt>:UP2K

Syntax	SONet:TX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`SON:TX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `SON:TX1:PATT` is `USER2048BIT`

Syntax	SONet:TX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`SON:TX1:UP2K?` → `"12DF"`
Note	

11.2.14 SONet:TX<Pt>:PINVersion

Syntax	SONet:TX<Pt>:PINVersion <inverted>
Description	This command enables or disables PRBS pattern inversion (when SSTRucture is BULK).
Parameters	<Pt> = Port number
	<inverted> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`SON:TX1:PINV OFF`
Note	

Syntax	SONet:TX<Pt>:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the PRBS pattern.
Parameter	<Pt> = Port number
Response	<inverted> = <BOOLEAN RESPONSE DATA>
Example	`SON:TX1:PINV?` → `0`
Note	

11.2.15 SONet:TX<Pt>:TIMing

Syntax	SONet:TX<Pt>:TIMing <source>
Description	This command sets the timing source.
Parameters	<Pt> = Port number
	<source> = <CHARACTER PROGRAM DATA> INTernal: Internal clock. EXTernal: External clock. RX: Received Rx signal clock. *DEFault = INT*
Response	None.
Example	`SON:TX1:TIM INT`
Note	

Syntax	SONet:TX<Pt>:TIMing?
Description	This query returns the timing source.
Parameter	<Pt> = Port number
Response	<source> = <CHARACTER RESPONSE DATA>
Example	`SON:TX1:TIM? → INT`
Note	

11.2.16 SONet:TX<Pt>:TCM

Syntax	SONet:TX<Pt>:TCM <mode>
Description	This command sets the TCM mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: Off STS3: Z5 (STS3) STS1: Z5 (STS1) VT2: Z6 (VT2) VT15: Z6 (VT1.5) *DEFault = OFF*
Response	None.
Example	`SON:TX1:TCM OFF`
Note	

Syntax	SONet:TX<Pt>:TCM?
Description	Queries the TCM mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SON:TX1:TCM? → OFF`
Note	

11.2.17 SONet:TX<Pt>:TUG3

Syntax	SONet:TX<Pt>:TUG3 <id>
Description	This command sets the TUG-3 number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: TUG-3 number one. 2: TUG-3 number two. 3: TUG-3 number three. *MINimum = 1, MAXimum = 3, DEFault = 1*
Response	None.
Example	`SON:TX1:TUG3 1`
Note	This value influences the channel number.

Syntax	SONet:TX<Pt>:TUG3?
Description	This query returns TUG3 number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:TX1:TUG3?` → `1`
Note	

11.2.18 SONet:TX<Pt>:VTG

Syntax	SONet:TX<Pt>:VTG <id>
Description	This command sets the VTG number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: VTG number one. 2: VTG number two. ... 7: VTG number seven. *MINimum = 1, MAXimum = 7, DEFault = 1*
Response	None.
Example	`SON:TX1:VTG 1`
Note	This value influences the channel number.

Syntax	SONet:TX<Pt>:VTG?
Description	This query returns VTG number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:TX1:VTG?` → `1`
Note	

11.2.19 SONet:TX<Pt>:VT2

Syntax	SONet:TX<Pt>:VT2 <id>
Description	This command sets the VT2 number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: VT2 number one. 2: VT2 number two. 3: VT2 number three. *MINimum = 1, MAXimum = 3, DEFault = 1*
Response	None.
Example	`SON:TX1:VT2 1`
Note	This value influences the channel number.

Syntax	SONet:TX<Pt>:VT2?
Description	This query returns VT2 number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:TX1:VT2?` → `1`
Note	

11.2.20 SONet:TX<Pt>:VT15

Syntax	SONet:TX<Pt>:VT15 <id>
Description	This command sets the VT1.5 number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> 1: VT1.5 number one. 2: VT1.5 number two. 3: VT1.5 number three. 4: VT1.5 number four. *MINimum = 1, MAXimum = 4, DEFault = 1*
Response	None.
Example	`SON:TX1:VT15 1`
Note	This value influences the channel number.

Syntax	SONet:TX<Pt>:VT15?
Description	This query returns VT1.5 number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:TX1:VT15?` → `1`
Note	

11.2.21 SONet:TX<Pt>:CHANnel

Syntax	SONet:TX<Pt>:CHANnel <id>
Description	This command sets the channel number for the signal structure.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=84, DEFault = 1*
Response	None.
Example	`SON:TX1:CHAN 1`
Note	The value influences the TUG-3, TUG-2 and TU-12 numbers.

Syntax	SONet:TX<Pt>:CHANnel?
Description	This query returns the channel number for the signal structure.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:TX1:CHAN?` → `1`
Note	

11.2.22 SONet:TX<Pt>:MAPPing

Syntax	SONet:TX<Pt>:MAPPing <type>
Description	This command sets the mapping type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> SYNChronous: Synchronous mapping. ASYNchronous: Asynchronous mapping. *DEFault = SYNC*
Response	None.
Example	`SON:TX1:MAPP SYNC`
Note	

Syntax	SONet:TX<Pt>:MAPPing?
Description	This query returns the mapping type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`SON:TX1:MAPP?` → `SYNC`
Note	

11.2.23 SONet:TX<Pt>:TOH:DEFault

Syntax	SONet:TX<Pt>:TOH:DEFault
Description	This command sets all OC TOH bytes to their default value.
Parameter	<Pt> = Port number
Response	None.
Example	`SON:TX1:TOH:DEF`
Note	There is no query version of this command.

11.2.24 SONet:TX<Pt>:TOH:TRACe

Syntax	SONet:TX<Pt>:TOH:TRACe <string>[,<idlechar>]
Description	This command sets the TOH trace (J0) to the specified string.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: The string to be used as section trace string. *DEFault = "Message_Test_J0"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SON:TX1:TOH:TRAC "Message_Test_J0",#H20`
Note	If the entered trace string is more than 15 characters long, the string will be truncated.

Syntax	SONet:TX<Pt>:TOH:TRACe?
Description	This query returns the TOH trace for string and idle char.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:TX1:TOH:TRAC?` → `"Message_Test_J0",#H20`
Note	

11.2.25 SONet:TX<Pt>:TOH:TRACe:CRC

Syntax	SONet:TX<Pt>:TOH:TRACe:CRC <mode>
Description	This command sets the TOH trace (J0) CRC mode (OFF/ON).
Parameters	<Pt> = Port number
	<mode> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SON:TX1:TOH:TRAC:CRC ON`
Note	

Syntax	SONet:TX<Pt>:TOH:TRACe:CRC?
Description	This query returns the mode of the TOH trace (J0) CRC.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SON:TX1:TOH:TRAC:CRC?` → `1`
Note	

11.2.26 SONet:TX<Pt>:TOH

Syntax	SONet:TX<Pt>:TOH <TOH-byte>,<value1>[,<value2>[,<value3>]]
Description	This command sets the value of the specified bytes in the TOH.
Parameters	<Pt> = Port number
	<TOH-byte> = <CHARACTER PROGRAM DATA> A1: 3 bytes. A2: 3 bytes. J0: 3 bytes. Note: The defined trace string will be disabled. B1: 2 bytes. <value3> is ignored. E1: 3 bytes. F1: 3 bytes. D1: 3 bytes. D2: 3 bytes. D3: 3 bytes. K1: 3 bytes. K2: 3 bytes. D4: 3 bytes. D5: 3 bytes. D6: 3 bytes. D7: 3 bytes. D8: 3 bytes. D9: 3 bytes. D10: 3 bytes. D11: 3 bytes. D12: 3 bytes. S1: 3 bytes. Z2: 3 bytes. E2: 3 bytes. M0: 1 byte. <value2> and <value3> are ignored. [1] M1: 1 byte. <value2> and <value3> are ignored.
	<value1> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
	<value2> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
	<value3> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
Response	None.

Examples	`SON:TX1:TOH A1,0,0,0`
	`SON:TX1:TOH D7,253,32,26`
	`SON:TX1:TOH B1,4,5`
	`SON:TX1:TOH M1,0`

Note	[1] Only valid for SONet:TX<Pt>:OCLevel = 192 or 768.

Syntax	SONet:TX<Pt>:TOH? <TOH-byte>
Description	This query returns the value of the specified bytes in the TOH.
Parameters	<Pt> = Port number
	<SOH-byte> = <CHARACTER PROGRAM DATA>
Response	<value1> = <HEXADECIMAL NUMERIC RESPONSE DATA> [,<value2> = <HEXADECIMAL NUMERIC RESPONSE DATA> [,<value3> = <HEXADECIMAL NUMERIC RESPONSE DATA>]]

Examples	`SON:TX1:TOH? A1` → `#H00,#H00,#H00`
	`SON:TX1:TOH? D7` → `#HFD,#H20,#H1A`
	`SON:TX1:TOH? B1` → `#H04,#H05`
	`SON:TX1:TOH? M1` → `#H00`

Note	

11.2.27 SONet:TX<Pt>:STS3:POH:DEFault

Syntax	SONet:TX<Pt>:STS3:POH:DEFault
Description	This command sets all STS3 path overhead bytes to their default value.
Parameter	<Pt> = Port number
Response	None.
Example	`SON:TX1:STS3:POH:DEF`
Note	There is no query version of this command.

11.2.28 SONet:TX<Pt>:STS3:POH:TRACe

Syntax	SONet:TX<Pt>:STS3:POH:TRACe <string>[,<idlechar>]
Description	This command sets the STS3 path trace (J1) to the specified string and its idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: The string to be used as path trace string. *DEFault = "Message_Test_J1"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SON:TX1:STS3:POH:TRAC "Message_Test_J1",#H20`
Note	If the entered trace string is more than 15 characters long, the string will be truncated.

Syntax	SONet:TX<Pt>:STS3:POH:TRACe?
Description	This query returns the STS3 path trace text and idle char.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:TX1:STS3:POH:TRAC?` → `"Message_Test_J1",#H20`
Note	

11.2.29 SONet:TX<Pt>:STS3:POH:TRACe:CRC

Syntax	SONet:TX<Pt>:STS3:POH:TRACe:CRC <mode>
Description	This command sets the section overhead trace (J1) CRC mode (OFF/ON).
Parameters	<Pt> = Port number
	<mode> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SON:TX1:STS3:POH:TRAC:CRC ON`
Note	

Syntax	SONet:TX<Pt>:STS3:POH:TRACe:CRC?
Description	This query returns the mode of the overhead trace (J1) CRC.
Parameter	<Pt> = Port number
Response	<mode> = <BOOLEAN RESPONSE DATA>
Example	`SON:TX1:STS3:POH:TRAC:CRC? → 1`
Note	

11.2.30 SONet:TX<Pt>:STS3:POH:TTCM

Syntax	SONet:TX<Pt>:STS3:POH:TTCM <string>[,<idlechar>]
Description	This command sets the STS3 TCM trace (Z5) to the specified string and its idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Apid_TCM_Z5"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SON:TX1:STS3:POH:TTCM "Apid_TCM_Z5",#H20`
Note	

Syntax	SONet:TX<Pt>:STS3:POH:TTCM?
Description	This query returns the STS3 TCM trace text and idle char.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:TX1:STS3:POH:TTCM? → "Apid_TCM_Z5",#H20`
Note	

11.2.31 SONet:TX<Pt>:STS3:POH

Syntax	SONet:TX<Pt>:STS3:POH <POH-byte>,<value>
Description	This command sets the value of the specified byte in the STS3 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA> C2: C2 byte. G1: G1 byte. F2: F2 byte. H4: H4 byte. Z3: Z3 byte. Z4: Z4 byte. Z5: Z5 byte. Only available when there is no TCM.
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
Response	None.
Example	`SON:TX1:STS3:POH C2,0`
Note	

Syntax	SONet:TX<Pt>:STS3:POH? <POH-byte>
Description	This query returns the value of the specified byte in the STS3 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA>
Response	<value> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Examples	`SON:TX1:STS3:POH? C2 → #H00` `SON:TX1:STS3:POH? H4 → #HFF`
Note	

11.2.32 SONet:TX<Pt>:STS1:POH:DEFault

Syntax	SONet:TX<Pt>:STS1:POH:DEFault
Description	This command sets all STS1 path overhead bytes to their default value.
Parameter	<Pt> = Port number
Response	None.
Example	`SON:TX1:STS1:POH:DEF`
Note	There is no query version of this command.

11.2.33 SONet:TX<Pt>:STS1:POH:TRACe

Syntax	SONet:TX<Pt>:STS1:POH:TRACe <string>[,<idlechar>]
Description	This command sets the STS1 path trace (J1) to the specified string.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Message_Test_J1"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SON:TX1:STS1:POH:TRAC "Message_Test_J1",#H20`
Note	

Syntax	SONet:TX<Pt>:STS1:POH:TRACe?
Description	This query returns the STS1 path trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:TX1:STS1:POH:TRAC?` → `"Message_Test_J1",#H20`
Note	

11.2.34 SONet:TX<Pt>:STS1:POH:TRACe:CRC

Syntax	SONet:TX<Pt>:STS1:POH:TRACe:CRC <mode>
Description	This command sets the section overhead trace (J1) CRC mode (OFF/ON).
Parameters	<Pt> = Port number
	<mode> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SON:TX1:STS1:POH:TRAC:CRC ON`
Note	

Syntax	SONet:TX<Pt>:STS1:POH:TRACe:CRC?
Description	This query returns the mode of the overhead trace (J1) CRC.
Parameter	<Pt> = Port number
Response	<mode> = <BOOLEAN RESPONSE DATA>
Example	`SON:TX1:STS1:POH:TRAC:CRC?` → `1`
Note	

11.2.35 SONet:TX<Pt>:STS1:POH:TTCM

Syntax	SONet:TX<Pt>:STS1:POH:TTCM <string>[,<idlechar>]
Description	This command sets the STS1 TCM trace (Z5) to the specified string, and the idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Apid_TCM_Z5"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SON:TX1:STS1:POH:TTCM "Apid_TCM_Z5",#H20`
Note	

Syntax	SONet:TX<Pt>:STS1:POH:TTCM?
Description	This query returns the STS1 TCM trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:TX1:STS1:POH:TTCM? → "Apid_TCM_Z5",#H20`
Note	

11.2.36 SONet:TX<Pt>:STS1:POH

Syntax	SONet:TX<Pt>:STS1:POH <POH-byte>,<value>
Description	This command sets the value of the specified byte in the STS1 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA> C2: C2 byte. G1: G1 byte. F2: F2 byte. H4: H4 byte. Z3: Z3 byte. Z4: Z4 byte. Z5: Z5 byte. Only available when there is no TCM.
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
Response	None.
Example	`SON:TX1:STS1:POH C2,0`
Note	

Syntax	SONet:TX<Pt>:STS1:POH? <POH-byte>
Description	This query returns the value of the specified byte in the STS1 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA>
Response	<value> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:TX1:STS1:POH? C2 → #H00`
Note	

11.2.37 SONet:TX<Pt>:VT2:POH:DEFault

Syntax	SONet:TX<Pt>:VT2:POH:DEFault
Description	This command sets all VT2 path overhead bytes to their default value.
Parameter	<Pt> = Port number
Response	None.
Example	`SON:TX1:VT2:POH:DEF`
Note	There is no query version of this command.

11.2.38 SONet:TX<Pt>:VT2:POH:TRACe

Syntax	SONet:TX<Pt>:VT2:POH:TRACe <string>[,<idlechar>]
Description	This command sets the VT2 path trace (J2) to the specified string and the idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Message_Test_J2"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SON:TX1:VT2:POH:TRAC "Message_Test_J2",#H20`
Note	

Syntax	SONet:TX<Pt>:VT2:POH:TRACe?
Description	This query returns the VT2 path trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:TX1:VT2:POH:TRAC? → "Message Test J2",#H20`
Note	

11.2.39 SONet:TX<Pt>:VT2:POH:TRACe:CRC

Syntax	SONet:TX<Pt>:VT2:POH:TRACe:CRC <mode>
Description	This command sets the section overhead trace (J2) CRC mode (OFF/ON).
Parameters	<Pt> = Port number
	<mode> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SON:TX1:VT2:POH:TRAC:CRC ON`
Note	

Syntax	SONet:TX<Pt>:VT2:POH:TRACe:CRC?
Description	This query returns the mode of the overhead trace (J2) CRC.
Parameter	<Pt> = Port number
Response	<mode> = <BOOLEAN RESPONSE DATA>
Example	`SON:TX1:VT2:POH:TRAC:CRC? → 1`
Note	

11.2.40 SONet:TX<Pt>:VT2:POH:TTCM

Syntax	SONet:TX<Pt>:VT2:POH:TTCM <string>[,<idlechar>]
Description	This command sets the VT2 TCM trace (Z6) to the specified string and the idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Apid_TCM_Z6"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SON:TX1:VT2:POH:TTCM "Apid_TCM_Z6",#H20`
Note	

Syntax	SONet:TX<Pt>:VT2:POH:TTCM?
Description	This query returns the VT2 trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:TX1:VT2:POH:TTCM?` → `"Apid_TCM_Z6",#H20`
Note	

11.2.41 SONet:TX<Pt>:VT2:POH

Syntax	SONet:TX<Pt>:VT2:POH <POH-byte>,<value>
Description	This command sets the value of the specified byte in the VT2 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA> V5: V5 byte. Z6: Z6 byte. Z7: Z7 byte.
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
Response	None.
Example	`SON:TX1:VT2:POH V5,#H04`
Note	

Syntax	SONet:TX<Pt>:VT2:POH? <POH-byte>
Description	This query returns the value of the specified byte in the VT2 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA>
Response	<value> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:TX1:VT2:POH? V5` → `#H04`
Note	

11.2.42 SONet:TX<Pt>:VT15:POH:DEFault

Syntax	SONet:TX<Pt>:VT15:POH:DEFault
Description	This command sets all VT1.5 path overhead bytes to their default value.
Parameter	<Pt> = Port number
Response	None.
Example	`SON:TX1:VT15:POH:DEF`
Note	There is no query version of this command.

11.2.43 SONet:TX<Pt>:VT15:POH:TRACe

Syntax	SONet:TX<Pt>:VT15:POH:TRACe <string>[,<idlechar>]
Description	This command sets the VT1.5 path trace (J2) to the specified string and the idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Message_Test_J2"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SON:TX1:VT15:POH:TRAC "Message_Test_J2",#H20`
Note	

Syntax	SONet:TX<Pt>:VT15:POH:TRACe?
Description	This query returns the VT1.5 path trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:TX1:VT15:POH:TRAC? → "Message_Test_J2",#H20`
Note	

11.2.44 SONet:TX<Pt>:VT15:POH:TRACe:CRC

Syntax	SONet:TX<Pt>:VT15:POH:TRACe:CRC <mode>
Description	This command sets the section overhead trace (J2) CRC mode (OFF/ON).
Parameters	<Pt> = Port number
	<mode> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SON:TX1:VT15:POH:TRAC:CRC ON`
Note	

Syntax	SONet:TX<Pt>:VT15:POH:TRACe:CRC?
Description	This query returns the mode of the overhead trace (J2) CRC.
Parameter	<Pt> = Port number
Response	<mode> = <BOOLEAN RESPONSE DATA>
Example	`SON:TX1:VT15:POH:TRAC:CRC? → 1`
Note	

11.2.45 SONet:TX<Pt>:VT15:POH:TTCM

Syntax	SONet:TX<Pt>:VT15:POH:TTCM <string>[,<idlechar>]
Description	This command sets the VT1.5 TCM trace (Z6) to the specified string and the idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Apid_TCM_Z6"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SON:TX1:VT15:POH:TTCM "Apid_TCM_Z6",#H20`
Note	

Syntax	SONet:TX<Pt>:VT15:POH:TTCM?
Description	This query returns the VT1.5 trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:TX1:VT15:POH:TTCM? → "Apid_TCM_Z6",#H20`
Note	

11.2.46 SONet:TX<Pt>:VT15:POH

Syntax	SONet:TX<Pt>:VT15:POH <POH-byte>,<value>
Description	This command sets the value of the specified byte in the VT1.5 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA> V5: V5 byte. Z6: Z6 byte. Z7: Z7 byte.
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
Response	None.
Example	`SON:TX1:VT15:POH V5,#H04`
Note	

Syntax	SONet:TX<Pt>:VT15:POH? <POH-byte>
Description	This query returns the value of the specified byte in the VT1.5 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA>
Response	<value> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:TX1:VT15:POH? V5` → `#H04`
Note	

11.2.47 SONet:TX<Pt>:STS1:STS1:POH:DEFault

Syntax	SONet:TX<Pt>:STS1:STS1:POH:DEFault
Description	This command sets all STS1 path overhead bytes to their default value.
Parameter	<Pt> = Port number
Response	None.
Example	`SON:TX1:STS1:STS1:POH:DEF`
Note	There is no query version of this command.

11.2.48 SONet:TX<Pt>:STS1:STS1:POH:TRACe

Syntax	SONet:TX<Pt>:STS1:STS1:POH:TRACe <string>[,<idlechar>]
Description	This command sets the STS1 path trace (J1) to the specified string.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Message_Test_J1"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SON:TX1:STS1:STS1:POH:TRAC "Message_Test_J1",#H20`
Note	

Syntax	SONet:TX<Pt>:STS1:STS1:POH:TRACe?
Description	This query returns the STS1 path trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:TX1:STS1:STS1:POH:TRAC?` → `"Message_Test_J1",#H20`
Note	

11.2.49 SONet:TX<Pt>:STS1:STS1:POH:TRACe:CRC

Syntax	SONet:TX<Pt>:STS1:STS1:POH:TRACe:CRC <mode>
Description	This command sets the section overhead trace (J1) CRC mode (OFF/ON).
Parameters	<Pt> = Port number
	<mode> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`SON:TX1:STS1:STS1:POH:TRAC:CRC ON`
Note	

Syntax	SONet:TX<Pt>:STS1:STS1:POH:TRACe:CRC?
Description	This query returns the mode of the overhead trace (J1) CRC.
Parameter	<Pt> = Port number
Response	<mode> = <BOOLEAN RESPONSE DATA>
Example	`SON:TX1:STS1:STS1:POH:TRAC:CRC? → 1`
Note	

11.2.50 SONet:TX<Pt>:STS1:STS1:POH:TTCM

Syntax	SONet:TX<Pt>:STS1:STS1:POH:TTCM <string>[,<idlechar>]
Description	This command sets the STS1 TCM trace (Z5) to the specified string, and the idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> Quoted string: Path trace string. *DEFault = "Apid_TCM_Z5"*
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`SON:TX1:STS1:STS1:POH:TTCM "Apid_TCM_Z5",#H20`
Note	

Syntax	SONet:TX<Pt>:STS1:STS1:POH:TTCM?
Description	This query returns the STS1 TCM trace.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>, <idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:TX1:STS1:STS1:POH:TTCM? → "Apid_TCM_Z5",#H20`
Note	

11.2.51 SONet:TX<Pt>:STS1:STS1:POH

Syntax	SONet:TX<Pt>:STS1:STS1:POH <POH-byte>,<value>
Description	This command sets the value of the specified byte in the STS1 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA> C2: C2 byte. G1: G1 byte. F2: F2 byte. H4: H4 byte. Z3: Z3 byte. Z4: Z4 byte. Z5: Z5 byte. Only available when there is no TCM.
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
Response	None.
Example	`SON:TX1:STS1:STS1:POH C2,0`
Note	

Syntax	SONet:TX<Pt>:STS1:STS1:POH? <POH-byte>
Description	This query returns the value of the specified byte in the STS1 path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA>
Response	<value> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:TX1:STS1:STS1:POH? C2` → `#H00`
Note	

11.2.52 SONet:TX<Pt>:FOLLow

Syntax	SONet:TX<Pt>:FOLLow <mode>
Description	This command sets the transmitter setup to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> NONE: Do not follow. TX1: Follows setup of the Tx port1. *DEFault = NONE*
Response	None.
Example	`SON:TX1:FOLL NONE`
Note	

Syntax	SONet:TX<Pt>:FOLLow?
Description	This query returns the transmitter setup to follow another setup or not to follow.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`SON:TX1:FOLL?` → `NONE`
Note	

11.2.53 SONet:TX<Pt>:OTHRough

Syntax	SONet:TX<Pt>:OTHRough <mode>
Description	This command sets the OH Overwrite Position.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> TOH: SOH Data. A1A2: A1/A2 byte. K1K2: K1/K2 byte. S1: S1 byte. DCC1TO3: DCC1-3 byte. DCC4TO12: DCC4-12 byte. J0: J0 byte. TOH1BYTE: 1 byte of TOH.(selectable) *DEFault = TOH*
Response	None.
Example	`SON:TX1:OTHR A1A2`
Note	

Syntax	SONet:TX<Pt>:OTHRough?
Description	This query returns the OH Overwrite Position.
Parameter	<Pt> = Port number
Response	<position> = <CHARACTER RESPONSE DATA>
Example	`SON:TX1:OTHR?` → `A1A2`
Note	

11.2.54 SONet:TX<Pt>:OTHRough:BYTE

Syntax	SONet:TX<Pt>:OTHRough:BYTE <byte>
Description	This command sets the OH Overwrite Position.
Parameters	<Pt> = Port number
	<byte> = <CHARACTER PROGRAM DATA> A11-A13: A1 bytes. A21-A23: A2 bytes. J0: J0 byte. E1: E1 byte. F1: F1 byte. D1: D1 byte. D2: D2 byte. D3: D3 byte. K1: K1 byte. K2: K2 byte. D4: D4 byte. D5: D5 byte. D6: D6 byte. D7: D7 byte. D8: D8 byte. D9: D9 byte. D10: D10 byte. D11: D11 byte. D12: D12 byte. S1: S1 byte. M0: M0 byte. M1: M1 byte. E2: E2 byte. X18-X99: X<Line><Column> *DEFault = A11*
Response	None.
Example	`SON:TX1:OTHR:BYTE A11`
Note	

Syntax	SONet:TX<Pt>:OTHRough:BYTE?
Description	This query returns the OH Overwrite Position.
Parameter	<Pt> = Port number
Response	<byte> = <CHARACTER RESPONSE DATA>
Example	`SON:TX1:OTHR:BYTE?` → `A11`
Note	

11.2.55 SONet:TX<Pt>:STL:MMAPing:LANE

Syntax	SONet:TX<Pt>:STL:MMAPing:LANE <lanes>
Description	This command sets the STL lane marker assignment.
Parameters	<Pt> = Port number
	{(<lanes>),}* = <EXPRESSION PROGRAM DATA> Format: Numeric List List consist of the value of the lane marker ranging from 0 to 3.
Response	None.
Example	`SON:TX1:STL:MMAP:LANE (0,1,2,3)`
Note	This command can be used on 40G

Syntax	SONet:TX<Pt>:STL:MMAPing:LANE?
Description	This query returns the STL lane marker assignment.
Parameter	<Pt> = Port number
Response	{(<lanes>),}* = <EXPRESSION RESPONSE DATA> Format: Numeric List
Example	`SON:TX1:STL:MMAP:LANE?` → `(0,1,2,3)`
Note	This command can be used on 40G

11.3 Stimuli

11.3.1 SONet:STIMuli:TX<Pt>:FOFFset

Syntax	SONet:STIMuli:TX<Pt>:FOFFset <offset>
Description	This command sets the frequency offset for the clock source. Unit: ppm.
Parameters	<Pt> = Port number
	<offset> = <NUMERIC PROGRAM DATA> MT1000A: *MINimum=-50, MAXimum=50, DEFault=0* MT1100A: *MINimum=-200.0, MAXimum=200.0, DEFault = 0*
Response	None.
Example	`SON:STIM:TX1:FOFF 0`
Note	

Syntax	SONet:STIMuli:TX<Pt>:FOFFset?
Description	This query returns the frequency offset for the clock source. Unit: ppm.
Parameter	<Pt> = Port number
Response	MT1000A: <offset> = <NR1 NUMERIC RESPONSE DATA> MT1100A: <offset> = <NR2 NUMERIC RESPONSE DATA>
Example	MT1000A: `SON:STIM:TX1:FOFF?` → `0` MT1100A: `SON:STIM:TX1:FOFF?` → `0.0`
Note	

11.3.2 SONet:STIMuli:TX<Pt>:ALARm

Syntax	SONet:STIMuli:TX<Pt>:ALARm <alarmtype>
Description	This command sets the type of alarm to be generated.
Parameters	<Pt> = Port number
	<alarmtype> = <CHARACTER PROGRAM DATA> LOS: Loss of signal LOF: Loss of frame OOF: Out of frame AISL(MSAIS): MS alarm indication signal RDIL(MSRDI): MS remote defect indicator AISP(AUAIS): AU alarm indication signal LOPP(AULOP): AU loss of pointer TIMP(HPTIM): HP trace identifier mismatch PLMP(HPPLM): HP payload label mismatch UNEQP(HPUNEQ): HP unequipped RDIP(HPRDI): HP remote defect indicator AISV(TUAIS): TU alarm indication signal LOPV(TULOP): TU loss of pointer LOMV(TULOM): TU loss of multiframe TIMV(LPTIM): LP trace identifier mismatch UNEQV(LPUNEQ): LP unequipped RDIV(LPRDI): LP remote defect indicator PLMV(LPPLM): LP payload label mismatch LSS: Loss of signal synchronization TCUNEQ: TC unequipped TCLTC: TC loss of tandem connection TCTIM: TC trace identifier mismatch TCAIS: TC alarm indication signal TCRDI: TC remote defect indicator TCODI: TC outgoing defect indicator *DEFault = LOS*
Response	None.
Example	`SON:STIM:TX1:ALAR LOS`
Note	

Syntax	SONet:STIMuli:TX<Pt>:ALARm?
Description	This query returns the type of alarm to be generated.
Parameter	<Pt> = Port number
Response	<alarmtype> = <CHARACTER RESPONSE DATA>
Example	`SON:STIM:TX1:ALAR?` → `LOS`
Note	

11.3.3 SONet:STIMuli:TX<Pt>:AINSert

Syntax	SONet:STIMuli:TX<Pt>:AINSert <insertion>
Description	This command sets the method to insert alarm.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> OFF PERManent ALTernate *DEFault = OFF*
Response	None.
Example	`SON:STIM:TX1:AINS PERM`
Note	

Syntax	SONet:STIMuli:TX<Pt>:AINSert?
Description	This query returns the alarm insertion method.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`SON:STIM:TX1:AINS?` → `PERM`
Note	

11.3.4 SONet:STIMuli:TX<Pt>:ANLength

Syntax	SONet:STIMuli:TX<Pt>:ANLength <frames>
Description	This command sets the alternate normal length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1* Maximum depends on the error insert method, see `SON:STIM:TX<Pt>:AINS`
Response	None.
Example	`SON:STIM:TX1:ANL 1`
Note	

Syntax	SONet:STIMuli:TX<Pt>:ANLength?
Description	This query returns the alternate normal length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:STIM:TX1:ANL?` → `1`
Note	

11.3.5 SONet:STIMuli:TX<Pt>:AALength

Syntax	SONet:STIMuli:TX<Pt>:AALength <frames>
Description	This command sets the alternate alarm length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1* Maximum depends on the error insert method, see `SON:STIM:TX<Pt>:AINS`
Response	None.
Example	`SON:STIM:TX1:AAL 1`
Note	

Syntax	SONet:STIMuli:TX<Pt>:AALength?
Description	This query returns the alternate alarm length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:STIM:TX1:AAL?` → `1`
Note	

11.3.6 SONet:STIMuli:TX<Pt>:ERRor

Syntax	SONet:STIMuli:TX<Pt>:ERRor <errordestination>
Description	This command sets the error destination.
Parameters	<Pt> = Port number
	<errordestination> = <CHARACTER PROGRAM DATA> A1A2: Frame alignment word B1: B1 checksum byte B2: B2 checksum byte REIL(MSREI): MS remote error indication B3: B3 checksum byte REIP(HPREI): HP Remote error indication V5LPB3: B3 checksum of the low-order path REIV(LPREI): LP remote error indication PRBS: Pattern error ETRans: ERR-TRANS TCIEC: Tandem Connection incoming error count TCREI: Tandem Connection remote error indication TCOEI: Tandem Connection outgoing error indication TCBIP2: 2-bit Bit Interleaved Parity for Tandem Connection *DEFault = A1A2*
Response	None.
Example	`SON:STIM:TX1:ERR MSREI`
Note	Some errors are depended on type of content, setting them without the correct type of content will set EINSert to OFF. Changing the error may also change the insert method. See section 11.3.7.

Syntax	SONet:STIMuli:TX<Pt>:ERRor?
Description	This query returns the error destination.
Parameter	<Pt> = Port number
Response	<errordestination> = <CHARACTER RESPONSE DATA>
Example	`SON:STIM:TX1:ERR?` → `MSREI`
Note	

11.3.7 SONet:STIMuli:TX<Pt>:EINSert

Syntax	SONet:STIMuli:TX<Pt>:EINSert <insertion>
Description	This command sets the method to insert errors.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> OFF MANual B03: Burst · 1E-03 [1] B04: Burst · 1E-04 [1] B05: Burst · 1E-05 [1] B06: Burst · 1E-06 [2] B07: Burst · 1E-07 B08: Burst · 1E-08 B09: Burst · 1E-09 B10: Burst · 1E-10 ALTernate *DEFault = OFF*
Response	None.
Example	`SON:STIM:TX1:EINS MAN`
Note	If insertion is set to MANual, errors are inserted with SYST:STIM:INS. See section 2.3.23. [1] Is available for PRBS and ETRans (B03 is only available for ETRans if the STM Level is 1). [2] Is available for PRBS, ETRans, B2, V5LPB3, REIL, TCIEC, TCBIP2, TCREI and TCOEI. Selecting an unsupported burst rate will reset the rate to the closest possible.

Syntax	SONet:STIMuli:TX<Pt>:EINSert?
Description	This query returns the error insertion method.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`SON:STIM:TX1:EINS?` → `MAN`
Note	

11.3.8 SONet:STIMuli:TX<Pt>:EBLength

Syntax	SONet:STIMuli:TX<Pt>:EBLength <burstlength>
Description	This command sets the error burst length to generate.
Parameters	<Pt> = Port number
	<burstlength> = <NUMERIC PROGRAM DATA> *MINimum = 1, DEFault = 1* Maximum depends on the error insert method, see `SON:STIM:TX<Pt>:EINS` B03 - B10: Maximum = 1 MANual: Maximum = 8000[1]
Response	None.
Example	`SON:STIM:TX1:EBL 1`
Note	[1] If VC12 is active the maximum value for REIL, V5LPB3, TCIEC, TCBIP2, TCREI and TCOEI is 2000. The maximum value is 4000 when :STIM:TX<Pt>:ERR is PRBS (Pattern error).

Syntax	SONet:STIMuli:TX<Pt>:EBLength?
Description	This query returns the error burst length to generate.
Parameter	<Pt> = Port number
Response	<burstlength> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:STIM:TX1:EBL?` → `1`
Note	

11.3.9 SONet:STIMuli:TX<Pt>:ENLength

Syntax	SONet:STIMuli:TX<Pt>:ENLength <frames>
Description	This command sets the alternate normal length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1* Maximum depends on the error insert method, see `SON:STIM:TX<Pt>:EINS`
Response	None.
Example	`SON:STIM:TX1:ENL 1`
Note	

Syntax	SONet:STIMuli:TX<Pt>:ENLength?
Description	This query returns the alternate normal length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:STIM:TX1:ENL? → 1`
Note	

11.3.10 SONet:STIMuli:TX<Pt>:EELength

Syntax	SONet:STIMuli:TX<Pt>:EELength <frames>
Description	This command sets the alternate error length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1* Maximum depends on the error insert method, see `SON:STIM:TX<Pt>:EINS`
Response	None.
Example	`SON:STIM:TX1:EEL 1`
Note	

Syntax	SONet:STIMuli:TX<Pt>:EELength?
Description	This query returns the alternate error length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:STIM:TX1:EEL? → 1`
Note	

11.3.11 SONet:STIMuli:TX<Pt>:PTSequence

Syntax	SONet:STIMuli:TX<Pt>:PTSequence <type>,<sequence>
Description	This command sets the SONET pointer test sequence.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> STS3: STS3 pointer STS1: STS1 pointer TU3: TU-3 pointer VT2: VT2 pointer VT15: VT1.5 pointer
	<sequence> = <CHARACTER PROGRAM DATA> NONE: No test sequence SALTernating: Single alternating RDOuble: Regular + double RMISsing: Regular + missing DALTernating: Double alternating
Response	None.
Example	`SON:STIM:TX1:PTS STS3,NONE`
Note	

Syntax	SONet:STIMuli:TX<Pt>:PTSequence? <type>
Description	This query returns the SONET pointer test sequence.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> STS3: STS3 pointer STS1: STS1 pointer TU3: TU-3 pointer VT2: VT2 pointer VT15: VT1.5 pointer
Response	<sequence> = <CHARACTER RESPONSE DATA>
Example	`SON:STIM:TX1:PTS? STS3` → `NONE`
Note	

11.3.12 SONet:STIMuli:TX<Pt>:PMOVe

Syntax	SONet:STIMuli:TX<Pt>:PMOVe <type>,<value>
Description	This command sets the SONET pointer movement sequence. The value specifies the number of pointer increments.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> STS3: STS3 pointer STS1: STS1 pointer TU3: TU-3 pointer VT2: VT2 pointer VT15: VT1.5 pointer
	<value> = <NUMERIC PROGRAM DATA> *MINimum = -100, MAXimum = 100*
Response	None.
Example	`SON:STIM:TX1:PMOV AU4,9`
Notes	There is no query version of this command. A value of zero has no effect. A negative value results in pointer decrements.

11.3.13 SONet:STIMuli:TX<Pt>:PJUMp

Syntax	SONet:STIMuli:TX<Pt>:PJUMp <type>,<value>[,<ndf>]
Description	This command sets the SONET pointer jump. The value specifies the new pointer value.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> STS3: STS3 pointer STS1: STS1 pointer TU3: TU-3 pointer VT2: VT2 pointer VT15: VT1.5 pointer
	<value> = <NUMERIC PROGRAM DATA> The valid range depends on SONet:STIMuli:TX<Pt>:PTYPe STS3: 0 to 782 STS1: 0 to 782 TU3: 0 to 764 VT2: 0 to 139 VT15: 0 to 103
	<ndf> = <CHARACTER PROGRAM DATA> WITH: With new data flag (NDF) WOUT: Without new data flag (NDF) *DEFault = WITH*
Response	None.
Example	`SON:STIM:TX1:PJUM AU4,300`
Notes	There is no query version of this command. A negative value disables new data flag (NDF).

11.3.14 SONet:STIMuli:TX<Pt>:STL:AERRor:LANE

Syntax	SONet:STIMuli:TX<Pt>:STL:AERRor:LANE <lane>
Description	This command sets the lane of the multi lane alarm/error to be inserted.
Parameters	<Pt> = Port number
	<lane> = <NUMERIC PROGRAM DATA> *MINimum=#B0000, MAXimum=#B1111, DEFault=#B1000*
Response	None.

Example	`SON:STIM:TX1:STL:AERR:LANE #B01`
	`SON:STIM:TX1:STL:AERR:LANE #B0100`
	These commands add error into lane 1.

Note	This command can be used on 40G

Syntax	SONet:STIMuli:TX<Pt>:STL:AERRor:LANE?
Description	This query returns the lane of the multi lane alarm/error to be inserted.
Parameter	<Pt> = Port number
Response	<lane> = <BINARY NUMERIC RESPONSE DATA>
Example	`SON:STIM:TX1:STL:AERR:LANE?` → `#B0100`
Note	This command can be used on 40G

11.3.15 SONet:STIMuli:TX<Pt>:STL:ALARm

Syntax	SONet:STIMuli:TX<Pt>:STL:ALARm <alarmtype>
Description	This command sets the type of alarm to be inserted.
Parameters	<Pt> = Port number
	<alarmtype> = <CHARACTER PROGRAM DATA> LOFOOF: LOF/OOF-STL LOROOR: LOR/OOR-STL *DEFault = LOFOOF*
Response	None.
Example	`SON:STIM:TX1:STL:ALAR LOF`
Note	This command can be used on 40G

Syntax	SONet:STIMuli:TX<Pt>:STL:ALARm?
Description	This query returns the type of alarm to be inserted.
Parameter	<Pt> = Port number
Response	<alarmtype> = <CHARACTER RESPONSE DATA>
Example	`SON:STIM:TX1:STL:ALAR?` → `LOF`
Note	This command can be used on 40G

11.3.16 SONet:STIMuli:TX<Pt>:STL:AINSert

Syntax	SONet:STIMuli:TX<Pt>:STL:AINSert <insertion>
Description	This command sets the method to insert alarm.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> OFF MANual ALTernate *DEFault = OFF*
Response	None.
Example	`SON:STIM:TX1:STL:AINS MAN`
Note	This command can be used on 40G

Syntax	SONet:STIMuli:TX<Pt>:STL:AINSert?
Description	This query returns the alarm insertion method.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`SON:STIM:TX1:STL:AINS?` → `MAN`
Note	This command can be used on 40G

11.3.17 SONet:STIMuli:TX<Pt>:STL:ABLength

Syntax	SONet:STIMuli:TX<Pt>:STL:ABLength <frames>
Description	This command sets the alarm burst length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1*
Response	None.
Example	`SON:STIM:TX1:STL:ABL 8000`
Note	This command can be used on 40G

Syntax	SONet:STIMuli:TX<Pt>:STL:ABLength?
Description	This query returns the alarm burst length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:STIM:TX1:STL:ABL?` → `8000`
Note	This command can be used on 40G

11.3.18 SONet:STIMuli:TX<Pt>:STL:AALength

Syntax	SONet:STIMuli:TX<Pt>:STL:AALength <frames>
Description	This command sets the alternate alarm length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1*
Response	None.
Example	`SON:STIM:TX1:STL:AAL 8000`
Note	This command can be used on 40G

Syntax	SONet:STIMuli:TX<Pt>:STL:AALength?
Description	This query returns the alternate alarm length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:STIM:TX1:STL:AAL?` → `8000`
Note	This command can be used on 40G

11.3.19 SONet:STIMuli:TX<Pt>:STL:ANLength

Syntax	SONet:STIMuli:TX<Pt>:STL:ANLength <frames>
Description	This command sets the alternate normal length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1*
Response	None.
Example	`SON:STIM:TX1:STL:ANL 8000`
Note	This command can be used on 40G

Syntax	SONet:STIMuli:TX<Pt>:STL:ANLength?
Description	This query returns the alternate normal length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:STIM:TX1:STL:ANL?` → `8000`
Note	This command can be used on 40G

11.3.20 SONet:STIMuli:TX<Pt>:STL:ERRor

Syntax	SONet:STIMuli:TX<Pt>:STL:ERRor <errortype>
Description	This command sets the type of error to be inserted.
Parameters	<Pt> = Port number
	<errortype> = <CHARACTER PROGRAM DATA> A1A2: A1A2-STL *DEFault = A1A2*
Response	None.
Example	`SON:STIM:TX1:STL:ERR A1A2`
Note	This command can be used on 40G

Syntax	SONet:STIMuli:TX<Pt>:STL:ERRor?
Description	This query returns the type of error to be inserted.
Parameter	<Pt> = Port number
Response	<errortype> = <CHARACTER RESPONSE DATA>
Example	`SON:STIM:TX1:STL:ERR? → A1A2`
Note	This command can be used on 40G

11.3.21 SONet:STIMuli:TX<Pt>:STL:EINSert

Syntax	SONet:STIMuli:TX<Pt>:STL:EINSert <insertion>
Description	This command sets the method to insert errors.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> OFF ALTernate *DEFault = OFF*
Response	None.
Example	`SON:STIM:TX1:STL:EINS ALT`
Note	This command can be used on 40G

Syntax	SONet:STIMuli:TX<Pt>:STL:EINSert?
Description	This query returns the error insertion method.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`SON:STIM:TX1:STL:EINS? → ALT`
Note	This command can be used on 40G

11.3.22 SONet:STIMuli:TX<Pt>:STL:EELength

Syntax	SONet:STIMuli:TX<Pt>:STL:EELength <frames>
Description	This command sets the alternate error length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1*
Response	None.
Example	`SON:STIM:TX1:STL:EEL 8000`
Note	This command can be used on 40G

Syntax	SONet:STIMuli:TX<Pt>:STL:EELength?
Description	This query returns the alternate error length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:STIM:TX1:STL:EEL? → 8000`
Note	This command can be used on 40G

11.3.23 SONet:STIMuli:TX<Pt>:STL:ENLength

Syntax	SONet:STIMuli:TX<Pt>:STL:ENLength <frames>
Description	This command sets the alternate normal length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault=1*
Response	None.
Example	`SON:STIM:TX1:STL:ENL 8000`
Note	This command can be used on 40G

Syntax	SONet:STIMuli:TX<Pt>:STL:ENLength?
Description	This query returns the alternate normal length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:STIM:TX1:STL:ENL?` → `8000`
Note	This command can be used on 40G

11.3.24 SONet:STIMuli:TX<Pt>:STL:SKEW:BIT

Syntax	SONet:STIMuli:TX<Pt>:STL:SKEW:BIT <bits>
Description	This command sets the bits of the skew .
Parameters	<Pt> = Port number
	<bits> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 138240, DEFault=1*
Response	None.
Example	`SON:STIM:TX1:STL:SKEW:BIT 1000`
Note	This command can be used on 40G

Syntax	SONet:STIMuli:TX<Pt>:STL:SKEW:BIT?
Description	This query returns the bits of the skew .
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:STIM:TX1:STL:SKEW:BIT?` → `1000`
Note	This command can be used on 40G

11.3.25 SONet:STIMuli:TX<Pt>:STL:SKEW:NS?

Syntax	SONet:STIMuli:TX<Pt>:STL:SKEW:NS?
Description	This query returns the time of the skew to be inserted. Unit: ns
Parameter	<Pt> = Port number
Response	<skew> = <NR2 NUMERIC RESPONSE DATA>
Example	`SON:STIM:TX1:STL:SKEW:NS?` → `100.469`
Note	This command can be used on 40G

11.3.26 SONet:STIMuli:TX<Pt>:STL:SKEW:LANE

Syntax	SONet:STIMuli:TX<Pt>:STL:SKEW:LANE <lane>
Description	This command sets the lane of the skew to be inserted.
Parameters	<Pt> = Port number
	<lane> = <NUMERIC PROGRAM DATA> *MINimum=#B0000, MAXimum=#B1111, DEFault=#B1000*
Response	None.

Example	`SON:STIM:TX1:STL:SKEW:LANE #B01`
	`SON:STIM:TX1:STL:SKEW:LANE #B0100`
	These commands add skew into lane 1.

Note	This command can be used on 40G

11.3.27 SONet:STIMuli:TX<Pt>:STL:SKEW:LANE?

Syntax	SONet:STIMuli:TX<Pt>:STL:SKEW:LANE?
Description	This query returns the lane of the skew to be inserted.
Parameters	<Pt> = Port number
Response	<lane> = <BINARY NUMERIC RESPONSE DATA>
Example	`SON:STIM:TX1:STL:SKEW:LANE?` → `#B0100`
Note	This command can be used on 40G

11.4 Result

11.4.1 SONet:RX<Pt>:IFETch?

Syntax	SONet:RX<Pt>:IFETch? <parameter>
Description	This query fetches a SONET interval if available.
Parameters	<Pt> = Port number
	({<parameter>} + {,}∗) = <EXPRESSION PROGRAM DATA> The response format is listed for each parameter. **Alarms** LOS: Loss of signal. Response: <Seconds>,<Ratio> GAIS: Generic alarm indication signal. Response: <Seconds>,<Ratio> LOF: Loss of frame. Response: <Seconds>,<Ratio> OOF: Out of frame. Response: <Seconds>,<Ratio> AISL: Multiplex section alarm indication signal. Response: <Seconds>,<Ratio> RDIL: Multiplex section remote defect indicator. Response: <Seconds>,<Ratio> AISP: Administrative unit - alarm indication signal. Response: <Seconds>,<Ratio> LOPP: Administrative unit - loss of pointer. Response: <Seconds>,<Ratio> TIMP: High-order path trace identifier mismatch. Response: <Seconds>,<Ratio> PLMP: Payload label mismatch. Response: <Seconds>,<Ratio> UNEQP: High-order path unequipped. Response: <Seconds>,<Ratio> RDIP: High-order path remote defect indicator. Response: <Seconds>,<Ratio> AISV: Tributary unit alarm indication signal[1]. Response: <Seconds>,<Ratio> LOPV: Tributary unit loss of pointer[1]. Response: <Seconds>,<Ratio> LOMV: Tributary unit loss of multi frame[1]. Response: <Seconds>,<Ratio> TIMV: Low-order path trace identifier mismatch[1]. Response: <Seconds>,<Ratio> UNEQV: Low-order path unequipped[1]. Response: <Seconds>,<Ratio> RDIV: Low-order path remote defect indicator[1]. Response: <Seconds>,<Ratio> PLMV: Low-order path payload label mismatch[1]. Response: <Seconds>,<Ratio> LSS: Loss of signal synchronization[1]. Response: <Seconds>,<Ratio> **Errors** A1A2: Response: <Count>,<Ratio> B1: Response: <Count>,<Ratio> B2: Response: <Count>,<Ratio> REIL: Response: <Count>,<Ratio> B3: Response: <Count>,<Ratio> REIP: Response: <Count>,<Ratio> V5LPB3: [1]Response: <Count>,<Ratio> ERRPRBS: Pattern errors[1]. Response: <Count>,<Ratio> ERRPRBSBLK: Pattern block errors[1]. Response: <Count>,<Ratio> REIV: Response: <Count>,<Ratio> STSNDF: Response: <Count>,<Ratio> VTNDF: Response: <Count>,<Ratio> SAPS: Switch APS. Response: <Count>,<Ratio> **Rx frequency** FREQ: Frequency deviation. Response: <ppm> FREQDIF: Frequency difference (RX1-RX2). Response: <ppm> (Only for RX1, RX2 is always NaN (section 1.6.1)) **Mux quality** MFES: Forward ES. Response: <Count>,<Ratio%> MFSES: Forward SES. Response: <Count>,<Ratio%> MFUNAV: Forward UNAV. Response: <Count>,<Ratio%> MBES: Backward ES. Response: <Count>,<Ratio%> MBSES: Backward SES. Response: <Count>,<Ratio%> MBUNAV: Backward UNAV. Response: <Count>,<Ratio%> **STS3CSPE quality** STS3CSPEFES: Forward ES. Response: <Count>,<Ratio%> STS3CSPEFSES: Forward SES. Response: <Count>,<Ratio%>

continued on next page...

. . . continued from previous page

	STS3CSPEFUNAV: Forward UNAV. Response: <Count>,<Ratio%> STS3CSPEBES: Backward ES. Response: <Count>,<Ratio%> STS3CSPEBSES: Backward SES. Response: <Count>,<Ratio%> STS3CSPEBUNAV: Backward UNAV. Response: <Count>,<Ratio%> **STS1SPE quality**[1] STS1SPEFES: Forward ES. Response: <Count>,<Ratio%> STS1SPEFSES: Forward SES. Response: <Count>,<Ratio%> STS1SPEFUNAV: Forward UNAV. Response: <Count>,<Ratio%> STS1SPEBES: Backward ES. Response: <Count>,<Ratio%> STS1SPEBSES: Backward SES. Response: <Count>,<Ratio%> STS1SPEBUNAV: Backward UNAV. Response: <Count>,<Ratio%> **VT2 quality**[1] VT2FES: Forward ES. Response: <Count>,<Ratio%> VT2FSES: Forward SES. Response: <Count>,<Ratio%> VT2FUNAV: Forward UNAV. Response: <Count>,<Ratio%> VT2BES: Backward ES. Response: <Count>,<Ratio%> VT2BSES: Backward SES. Response: <Count>,<Ratio%> VT2BUNAV: Backward UNAV. Response: <Count>,<Ratio%> **VT15 quality**[1] VT15FES: Forward ES. Response: <Count>,<Ratio%> VT15FSES: Forward SES. Response: <Count>,<Ratio%> VT15FUNAV: Forward UNAV. Response: <Count>,<Ratio%> VT15BES: Backward ES. Response: <Count>,<Ratio%> VT15BSES: Backward SES. Response: <Count>,<Ratio%> VT15BUNAV: Backward UNAV. Response: <Count>,<Ratio%> **Bulk quality** ES: ES. Response: <Count>,<Ratio%> SES: SES. Response: <Count>,<Ratio%> UNAV: UNAV. Response: <Count>,<Ratio%> **STS pointer** STSPOINT: AU pointer. Response: <Count> STSNEG: Negative. Response: <Count> STSPOS: Positive. Response: <Count> **VT pointer**[1] VTPOINT: VT pointer. Response: <Count> VTNEG: Negative. Response: <Count> VTPOS: Positive. Response: <Count> **Justification**[1] JNEG: Negative justification. Response: <Count> JPOS: Positive justification. Response: <Count> **MUX performance**[1] PMPES: Performance MUX PO limit ES. Response: <Count>,<Ratio%> PMPSES: Performance MUX PO limit SES. Response: <Count>,<Ratio%> PMPBBE: Performance MUX PO limit BBE. Response: <Count>,<Ratio%> PMBS1ES: Performance MUX BIS limit S1ES. Response: <Count> PMBS2ES: Performance MUX BIS limit S2ES. Response: <Count> PMBS1SES: Performance MUX BIS limit S1SES. Response: <Count> PMBS2SES: Performance MUX BIS limit S2SES. Response: <Count> PMFSTAT: Performance MUX forward status. Response: <STRING RESPONSE DATA> PMFES: Performance MUX forward ES. Response: <Count>,<Ratio%> PMFSES: Performance MUX forward SES. Response: <Count>,<Ratio%> PMFUNAV: Performance MUX forward UNAV. Response: <Count>,<Ratio%> PMFBBE: Performance MUX forward BBE. Response: <Count>,<Ratio%> PMBSTAT: Performance MUX backward status. Response: <STRING RESPONSE DATA> PMBES: Performance MUX backward ES. Response: <Count>,<Ratio%>

continued on next page. . .

. . . continued from previous page

	PMBSES: Performance MUX backward SES. Response: <Count>,<Ratio%> PMBUNAV: Performance MUX backward UNAV. Response: <Count>,<Ratio%> PMBBBE: Performance MUX backward BBE. Response: <Count>,<Ratio%> **STS3CSPE performance**[1] PSTS3CSPEPES: Performance VC4 PO limit ES. Response: <Count>,<Ratio%> PSTS3CSPEPSES: Performance VC4 PO limit SES. Response: <Count>,<Ratio%> PSTS3CSPEPBBE: Performance VC4 PO limit BBE. Response: <Count>,<Ratio%> PSTS3CSPEBS1ES: Performance VC4 BIS limit S1ES. Response: <Count> PSTS3CSPEBS2ES: Performance VC4 BIS limit S2ES. Response: <Count> PSTS3CSPEBS1SES: Performance VC4 BIS limit S1SES. Response: <Count> PSTS3CSPEBS2SES: Performance VC4 BIS limit S2SES. Response: <Count> PSTS3CSPEFSTAT: Performance VC4 forward status. Response: <STRING RESPONSE DATA> PSTS3CSPEFES: Performance VC4 forward ES. Response: <Count>,<Ratio%> PSTS3CSPEFSES: Performance VC4 forward SES. Response: <Count>,<Ratio%> PSTS3CSPEFUNAV: Performance VC4 forward UNAV. Response: <Count>,<Ratio%> PSTS3CSPEFBBE: Performance VC4 forward BBE. Response: <Count>,<Ratio%> PSTS3CSPEBSTAT: Performance VC4 backward status. Response: <STRING RESPONSE DATA> PSTS3CSPEBES: Performance VC4 backward ES. Response: <Count>,<Ratio%> PSTS3CSPEBSES: Performance VC4 backward SES. Response: <Count>,<Ratio%> PSTS3CSPEBUNAV: Performance VC4 backward UNAV. Response: <Count>,<Ratio%> PSTS3CSPEBBBE : Performance VC4 backward BBE. Response: <Count>,<Ratio%> **VT2 performance**[1] PVT2PES: Performance VC12 PO limit ES. Response: <Count>,<Ratio%> PVT2PSES: Performance VC12 PO limit SES. Response: <Count>,<Ratio%> PVT2PBBE: Performance VC12 PO limit BBE. Response: <Count>,<Ratio%> PVT2BS1ES: Performance VC12 BIS limit S1ES. Response: <Count> PVT2BS2ES: Performance VC12 BIS limit S2ES. Response: <Count> PVT2BS1SES: Performance VC12 BIS limit S1SES. Response: <Count> PVT2BS2SES: Performance VC12 BIS limit S2SES. Response: <Count> PVT2FSTAT: Performance VC12 forward status. Response: <STRING RESPONSE DATA> PVT2FES: Performance VC12 forward ES. Response: <Count>,<Ratio%> PVT2FSES: Performance VC12 forward SES. Response: <Count>,<Ratio%> PVT2FUNAV: Performance VC12 forward UNAV. Response: <Count>,<Ratio%> PVT2FBBE: Performance VC12 forward BBE. Response: <Count>,<Ratio%> PVT2BSTAT: Performance VC12 backward status. Response: <STRING RESPONSE DATA> PVT2BES: Performance VC12 backward ES. Response: <Count>,<Ratio%> PVT2BSES: Performance VC12 backward SES. Response: <Count>,<Ratio%> PVT2BUNAV: Performance VC12 backward UNAV. Response: <Count>,<Ratio%> PVT2BBBE: Performance VC12 backward BBE. Response: <Count>,<Ratio%> **VT1.5 performance**[1] PVT15PES: Performance VC11 PO limit ES. Response: <Count>,<Ratio%> PVT15PSES: Performance VC11 PO limit SES. Response: <Count>,<Ratio%> PVT15PBBE: Performance VC11 PO limit BBE. Response: <Count>,<Ratio%> PVT15BS1ES: Performance VC11 BIS limit S1ES. Response: <Count> PVT15BS2ES: Performance VC11 BIS limit S2ES. Response: <Count> PVT15BS1SES: Performance VC11 BIS limit S1SES. Response: <Count> PVT15BS2SES: Performance VC11 BIS limit S2SES. Response: <Count> PVT15FSTAT: Performance VC11 forward status. Response: <STRING RESPONSE DATA> PVT15FES: Performance VC11 forward ES. Response: <Count>,<Ratio%> PVT15FSES: Performance VC11 forward SES. Response: <Count>,<Ratio%> PVT15FUNAV: Performance VC11 forward UNAV. Response: <Count>,<Ratio%> PVT15FBBE: Performance VC11 forward BBE. Response: <Count>,<Ratio%>

continued on next page. . .

. . . continued from previous page

	PVT15BSTAT: Performance VC11 backward status. Response: <STRING RESPONSE DATA> PVT15BES: Performance VC11 backward ES. Response: <Count>,<Ratio%> PVT15BSES: Performance VC11 backward SES. Response: <Count>,<Ratio%> PVT15BUNAV: Performance VC11 backward UNAV. Response: <Count>,<Ratio%> PVT15BBBE: Performance VC11 backward BBE. Response: <Count>,<Ratio%> **STS1SPE performance**[1] PSTS1SPEPES: Performance VC3 PO limit ES. Response: <Count>,<Ratio%> PSTS1SPEPSES: Performance VC3 PO limit SES. Response: <Count>,<Ratio%> PSTS1SPEPBBE: Performance VC3 PO limit BBE. Response: <Count>,<Ratio%> PSTS1SPEBS1ES: Performance VC3 BIS limit S1ES. Response: <Count> PSTS1SPEBS2ES: Performance VC3 BIS limit S2ES. Response: <Count> PSTS1SPEBS1SES: Performance VC3 BIS limit S1SES. Response: <Count> PSTS1SPEBS2SES: Performance VC3 BIS limit S2SES. Response: <Count> PSTS1SPEFSTAT: Performance VC3 forward status. Response: <STRING RESPONSE DATA> PSTS1SPEFES: Performance VC3 forward ES. Response: <Count>,<Ratio%> PSTS1SPEFSES: Performance VC3 forward SES. Response: <Count>,<Ratio%> PSTS1SPEFUNAV: Performance VC3 forward UNAV. Response: <Count>,<Ratio%> PSTS1SPEFBBE: Performance VC3 forward BBE. Response: <Count>,<Ratio%> PSTS1SPEBSTAT: Performance VC3 backward status. Response: <STRING RESPONSE DATA> PSTS1SPEBES: Performance VC3 backward ES. Response: <Count>,<Ratio%> PSTS1SPEBSES: Performance VC3 backward SES. Response: <Count>,<Ratio%> PSTS1SPEBUNAV: Performance VC3 backward UNAV. Response: <Count>,<Ratio%> PSTS1SPEBBBE: Performance VC3 backward BBE. Response: <Count>,<Ratio%> **Frequency** DEV: Deviation(ppm). Response: <Min>,<Max>,<Avg> **SDH TCM**[1] TCUNEQ: Tandem connection unequipped. response: <Count>,<Ratio%> TCLTC: Response: <Count>,<Ratio%> TCTIM: Tandem connection trace identifier mismatch. Response: <Count>,<Ratio%> TCAIS: Tandem connection alarm indication signal. Response: <Count>,<Ratio%> TCRDI: Tandem connection remote defect indicator. Response: <Count>,<Ratio%> TCODI: Tandem Connection Outgoing Defect Indicator. Response: <Count>,<Ratio%> TCIEC: Response: <Count>,<Ratio%> TCBIP2: Response: <Count>,<Ratio%> TCREI: Response: <Count>,<Ratio%> TCOEI: Response: <Count>,<Ratio%> TAPID: TCM APID. Response N1 or N2 TCM ID: <id> **STL** LOFSTL: STL LOF. Response: <Seconds> x 4 lanes LORSTL: STL LOR. Response: <Seconds> x 4 lanes OOFSTL: STL OOF. Response: <Frames> x 4 lanes OORSTL: STL OOR. Response: <Frames> x 4 lanes A1A2STL: A1A2-STL. Response: <Count> x 4 lanes OLA: OLA. Response: <Seconds>,<Ratio> RSKEW: Relative Skew. Response: <Nanoseconds> x 4 lanes MMAP: Marker Map. Response: <NR1> x 4 lanes
Response	{(<result>), }* = <EXPRESSION RESPONSE DATA> Format: Numeric List Each result is formatted according to the specification in the parameter field. Values that are not relevant or applicable for the current measurement return NaN (section 1.6.1).
Example	`SON:RX1:IFET? (LOS,LOF,OOF)` → `(3,0.00532),(4,0.00709),(5,0.00887)`
Notes	This command fetches the results from the interval selected using the `MEASurement:SETup:SELect` command (see section 18.2.2).

continued on next page. . .

. . . continued from previous page

	[1] Requires the current interface/application is installed and is currently active in the measurement. If requested result is not available, NaN (section 1.6.1) is returned. If there is one or more results, the last ”,” is always removed.

11.4.2 SONet:PMOVement:RX<Pt>:FETCh?

Syntax	SONet:PMOVement:RX<Pt>:FETCh? <pointer>,<min_time>,<max_time>
Description	This query fetches the pointer values.
Parameters	<pointer> = <CHARACTER PROGRAM DATA> STS: STS3 or STS1 pointer movements VT: VT pointer movements. VTG or VT2 or VT1.5 depending on SONET interface setup.
	<min_time> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=31536000* Beginning of the interval in seconds.
	<max_time> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=31536000* End of the interval in seconds.
Response	{<pointermovements>}* = <EXPRESSION RESPONSE DATA> The response is all pointer movements taking place in the requested time interval. Format: (pointerChange0,timestamp0),(pointerChange1,timestamp1),. . .
Example	`SON:PMOV:RX1:FETC? AU,0,3600` → `(1,10.000),(-1,15.002)`
Notes	The interval is closed, i.e., it includes both endpoints. The displayed resolution of the timestamps is 1/1000 second = 0.001s.

11.5 Status

11.5.1 SONet:STATus:RX<Pt>:AESummary[:EVENt]?

Syntax	SONet:STATus:RX<Pt>:AESummary[:EVENt]?
Description	This query returns the alarms and errors summary event register. The content of this event register is summarized in DB2 of the STATus:INTerface:PORT<Pt>:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Physical and high-order alarm summary DB2 (2) = Low-order path and tandem connection alarm summary DB3 (4) = Error section 1 summary DB4 (8) = Error section 2 summary DB5 (16) = STL alarm summary DB6 (32) = STL error summary DB7 - DB16 = NOT USED
Example	`SON:STAT:RX1:AES?` → `3`
Note	SDH can embed E1, E3 or E4 so if a PDH signal is embedded it is a good idea to send the following query STAT:INT:PORT<Pt>[:EVENt]? to see if there is alarms or error at the embedded signal.

11.5.2 SONet:STATus:RX<Pt>:AESummary:CONDition?

Syntax	SONet:STATus:RX<Pt>:AESummary:CONDition?
Description	This query returns alarms and errors summary condition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> Same as SONet:STATus:RX<Pt>:AESummary[:EVENt]?
Example	`SON:STAT:RX1:AES:COND?` → `3`
Note	

11.5.3 SONet:STATus:RX<Pt>:ALARm<section>[:EVENt]?

Syntax	SONet:STATus:RX<Pt>:ALARm<section>[:EVENt]?
Description	This query returns one of the alarms event register. These registers are summarized in DB1/DB2/DB5 of the SONet:STATus:RX<Pt>:AESummary:CONDition register.
Parameters	<Pt> = Port number
	<section> = Physical- and high-order path alarms(1), Low-order path and tandem connection alarms(2) or STL alarms(3)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> <section> = 1: DB1 (1) = LOS, Loss of signal DB2 (2) = LOF, Loss of frame DB3 (4) = OOF, Out of frame DB4 (8) = AISL, Multiplex section - alarm indication signal DB5 (16) = RDIL, Multiplex section - remote defect indicator DB6 (32) = AISP, Administrative unit - alarm indication signal DB7 (64) = LOPP, Administrative unit - loss of pointer DB8 (128) = TIMP, High-order path - trace identifier mismatch DB9 (256) = PLMP, High-order path - payload label mismatch DB10 (512) = UNEQP, High-order path - unequipped DB11 (1024) = RDIP, High-order path - remote defect indicator DB12 (2048) = AISV, Tributary unit - alarm indication signal DB13 (4096) = LOPV, Tributary unit - loss of pointer DB14 (8192) = LOMV, Tributary unit - loss Of multi frame DB15 (16384) = G-AIS, Generic alarm indication signal DB16 = NOT USED <section> = 2: DB1 (1) = TIMV, Low-order path - trace identifier mismatch DB2 (2) = UNEQV, Low-order path - unequipped DB3 (4) = RDIV, Low-order path - remote defect indicator DB4 (8) = LSS, Loss of signal synchronization DB5 = NOT USED DB6 (32) = PLMV, Low-order path - payload label mismatch DB7 (64) = TC-UNEQ, Tandem connection - unequipped DB8 (128) = TC-LTC, Tandem connection - loss of tandem connection DB9 (256) = TC-TIM, Tandem connection - trace identifier mismatch DB10 (512) = TC-AIS, Tandem connection - alarm indication signal DB11 (1024) = TC-RDI, Tandem connection - remote defect indicator DB12 (2048) = TC-ODI, Tandem connection - outgoing defect indicator DB13 - DB16 = NOT USED <section> = 3: DB1 (1) = STL LOF DB2 (2) = STL OOF DB3 (4) = STL LOR DB4 (8) = STL OOR DB5 (16) = OLA DB6 - DB16 = NOT USED
Example	`SON:STAT:RX1:ALAR1?` → `1`
Note	

11.5.4 SONet:STATus:RX<Pt>:ALARm<section>:CONDition?

Syntax	SONet:STATus:RX<Pt>:ALARm<section>:CONDition?
Description	This query returns one of the alarms condition registers.
Parameters	<Pt> = Port number
	<section> = Physical- and high-order path alarms(1), Low-order path and tandem connection alarms(2) or STL alarms(3)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> Same as SONet:STATus:RX<Pt>:ALARm<section>[:EVENt]?
Example	`SON:STAT:RX1:ALAR1:COND?` → `1`
Note	

11.5.5 SONet:STATus:RX<Pt>:ERRor<section>[:EVENt]?

Syntax	SONet:STATus:RX<Pt>:ERRor<section>[:EVENt]?
Description	This query returns the errors event register. The content of this register is summarized in DB3/DB4/DB6 of the SONet:STATus:RX<Pt>:AESummary:CONDition register.
Parameters	<Pt> = Port number
	<section> = (1-3)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> <section> = 1: DB1 (1) = A1A2 DB2 (2) = B1 DB3 (4) = B2 DB4 (8) = REIL DB5 (16) = B3 DB6 (32) = REIP DB7 (64) = V5/LP-B3 DB8 (128) = REIV DB9 (256) = PRBS DB10 (512) = VT-NDF DB11 (1024) = STS-NDF DB12 (2048) = APS DB13 - DB16 = NOT USED <section> = 2: DB1 (1) = TU-NEG DB2 (2) = TU-POS DB3 (4) = AU-NEG DB4 (8) = AU-POS DB5 not used DB6 (32) = TC-IEC DB7 (64) = TC-BIP-2 DB8 (128) = TC-REI DB9 (256) = TC-OEI DB10 - DB16 = NOT USED <section> = 3: DB1 (1) = A1A2-STL DB2 - DB16 = NOT USED
Example	`SON:STAT:RX1:ERR1?` → `3`
Note	

11.5.6 SONet:STATus:RX<Pt>:ERRor<section>:CONDition?

Syntax	SONet:STATus:RX<Pt>:ERRor<section>:CONDition?
Description	This query returns errors condition register.
Parameters	<Pt> = Port number
	<section> = (1-2)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> Same as SONet:STATus:RX<Pt>:ERRor<section>[:EVENt]?
Example	`SON:STAT:RX1:ERR1:COND?` → `3`
Note	

11.5.7 SONet:STATus:RX<Pt>:PSLevel?

Syntax	SONet:STATus:RX<Pt>:PSLevel?
Description	This query returns the physical signal level. Unit: dBm.
Parameter	<Pt> = Port number
Response	<signallevel> = <STRING RESPONSE DATA> "N/A": Module not present or not ready. - Electrical(STM1) - "<power> dBm": Min: "< -48 dBm", Max: "Exceeds Level" - Optical(Both module types) - "<power> dBm": Min: "< -27 dBm", Max: "Exceeds Level"
Example	`SON:STAT:RX1:PSL?` → `"-3 dBm"`
Note	

11.5.8 SONet:STATus:TX<Pt>:PSLevel?

Syntax	SONet:STATus:TX<Pt>:PSLevel?
Description	This query returns the physical signal level. Unit: dBm.
Parameter	<Pt> = Port number
Response	<signallevel> = <STRING RESPONSE DATA> "<power> dBm": Min: "< -27 dBm", Max: "Exceeds Level" "N/A": Module not present or not ready.
Example	`SON:STAT:TX1:PSL?` → `"-3 dBm"`
Note	Only available for Optical.

11.5.9 SONet:STATus:RX<Pt>:PDEViation?

Syntax	SONet:STATus:RX<Pt>:PDEViation? [<unit>]
Description	This query returns the physical deviation.
Parameters	<Pt> = Port number
	<unit> = <CHARACTER PROGRAM DATA> PPM = Parts per million BPS = Bits per second *DEFault = PPM*
Response	<deviation> = <NR2 NUMERIC RESPONSE DATA>
Example	`SON:STAT:RX1:PDEV? PPM` → `0`
Note	

11.5.10 SONet:STATus:RX<Pt>:PBRate?

Syntax	SONet:STATus:RX<Pt>:PBRate?
Description	This query returns the physical bit rate. Unit: bps.
Parameter	<Pt> = Port number
Response	<bitrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:STAT:RX1:PBR?` → `155520000`
Note	

11.5.11 SONet:STATus:RX<Pt>:DIFFerence?

Syntax	SONet:STATus:RX<Pt>:DIFFerence?
Description	This query returns physical bit rate difference between port A and port B (RX1 - RX2). Units: ppm, bps and bits.
Parameter	<Pt> = Port number
Response	<ppm> = <NR1 NUMERIC RESPONSE DATA>
	<bps> = <NR1 NUMERIC RESPONSE DATA>
	<acc> = <NR1 NUMERIC RESPONSE DATA> Accumulated difference in bits.
Example	`SON:STAT:RX1:DIFF?` → `-1,-4,-324`
Note	A valid response is only available if both RX1 and RX2 are on. Using either RX1 or RX2 will give the same results.

11.5.12 SONet:STATus:RX<Pt>:RACCumulated

Syntax	SONet:STATus:RX<Pt>:RACCumulated
Description	This command resets the accumulated difference.
Parameter	<Pt> = Port number
Response	None.
Example	`SON:STAT:RX1:RACC`
Note	Using either RX1 or RX2 will give the same results.

11.5.13 SONet:STATus:RX<Pt>:PPBRate?

Syntax	SONet:STATus:RX<Pt>:PPBRate?
Description	This query returns physical pattern bit rate. Unit: bps.
Parameter	<Pt> = Port number
Response	<bitrate> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:STAT:RX1:PPBR?` → `149760000`
Note	

11.5.14 SONet:STATus:RX<Pt>:CAPTure:TOH:TRACe?

Syntax	SONet:STATus:RX<Pt>:CAPTure:TOH:TRACe?
Description	This query returns the TOH trace (J0) for the latest captured frames.
Parameter	<Pt> = Port number
Response	<J0_trace> = <STRING RESPONSE DATA>
Example	`SON:STAT:RX1:CAPT:TOH:TRAC?` → `"Message_Test_J0"`
Note	If one of the alarms LOS or LOF is present, an empty string is returned.

11.5.15 SONet:STATus:RX<Pt>:CAPTure<Frame>:TOH?

Syntax	SONet:STATus:RX<Pt>:CAPTure<Frame>:TOH? <TOH-byte>
Description	This query returns the TOH bytes from the selected frame. 256 new frames are captured every second.
Parameters	<Pt> = Port number
	<Frame> = Frame number (1-64)
	<TOH-byte> = <CHARACTER PROGRAM DATA> A1: Returns 3 bytes. A2: Returns 3 bytes. J0: Returns 3 bytes. B1: Returns 3 bytes. E1: Returns 3 bytes. F1: Returns 3 bytes. D1: Returns 3 bytes. D2: Returns 3 bytes. D3: Returns 3 bytes. H1: Returns 3 bytes. H2: Returns 3 bytes. H3: Returns 3 bytes. B2: Returns 3 bytes. K1: Returns 3 bytes. K2: Returns 3 bytes. D4: Returns 3 bytes. D5: Returns 3 bytes. D6: Returns 3 bytes. D7: Returns 3 bytes. D8: Returns 3 bytes. D9: Returns 3 bytes. D10: Returns 3 bytes. D11: Returns 3 bytes. D12: Returns 3 bytes. S1: Returns 3 bytes. Z2: Returns 3 bytes. E2: Returns 3 bytes. M0: Returns 1 byte. [1] M1: Returns 1 byte.
Response	<byte1>[,<byte2>[,<byte3>]] = <HEXADECIMAL NUMERIC RESPONSE DATA> Refer to <TOH-byte> parameter description above to see how many bytes this command returns.

Examples	`SON:STAT:RX1:CAPT64:TOH? A1` → `#HF6,#HF6,#HF6`
	`SON:STAT:RX2:CAPT23:TOH? H1` → `#H69,#H93,#H93`
	`SON:STAT:RX1:CAPT1:TOH? M1` → `#H00`

Note	If one of the alarms LOS or LOF is present NaN (section 1.6.1) is returned. [1] Only valid for SONet:RX<Pt>:OCLevel = 192 or 768.

11.5.16 SONet:STATus:RX<Pt>:CAPTure:STS3:POH:TRACe?

Syntax	SONet:STATus:RX<Pt>:CAPTure:STS3:POH:TRACe?
Description	This query returns the STS3 path overhead trace (J1) for the latest captured frames. 64 new frames are captured every second.
Parameter	<Pt> = Port number
Response	<J1_trace> = <STRING RESPONSE DATA>
Example	`SON:STAT:RX1:CAPT:STS3:POH:TRAC?` → `"Message_Test_J1"`
Note	If one of the alarms LOS, LOF, UNEQP or LOPP is present, an empty string is returned.

11.5.17 SONet:STATus:RX<Pt>:CAPTure<Frame>:STS3:POH?

Syntax	SONet:STATus:RX<Pt>:CAPTure<Frame>:STS3:POH? <POH-byte>
Description	This query returns the STS3 path overhead bytes from the selected frame. 256 new frames are captured every second.
Parameters	<Pt> = Port number
	<Frame> = Frame number (1-64)
	<POH-byte> = <CHARACTER PROGRAM DATA> J1: J1 byte. B3: B3 byte. C2: C2 byte. G1: G1 byte. F2: F2 byte. H4: H4 byte. Z3: Z3 byte. Z4: Z4 byte. Z5: Z5 byte.
Response	<byte> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:STAT:RX1:CAPT1:STS3:POH? H4` → `#HFF`
Note	If one of the alarms LOS, LOF, UNEQP or LOPP is present NaN (section 1.6.1) is returned.

11.5.18 SONet:STATus:RX<Pt>:CAPTure:STS1:POH:TRACe?

Syntax	SONet:STATus:RX<Pt>:CAPTure:STS1:POH:TRACe?
Description	This command queries the STS1 path overhead trace (J1) for the latest captured frames. 64 new frames are captured every second.
Parameter	<Pt> = Port number
Response	<J1_trace> = <STRING RESPONSE DATA>
Example	`SON:STAT:RX1:CAPT:STS1:POH:TRAC?` → `"Message_Test_J1"`
Note	If one of the alarms LOS, LOF, UNEQP or UNEQV is present, an empty string is returned.

11.5.19 SONet:STATus:RX<Pt>:CAPTure<Frame>:STS1:POH?

Syntax	SONet:STATus:RX<Pt>:CAPTure<Frame>:STS1:POH? <POH-byte>
Description	This query returns the STS1 path overhead bytes from the selected frame. 64 new frames are captured every second.
Parameters	<Pt> = Port number
	<Frame> = Frame number (1-64)
	<POH-byte> = <CHARACTER PROGRAM DATA> J1: J1 byte. B3: B3 byte. C2: C2 byte. G1: G1 byte. F2: F2 byte. H4: H4 byte. Z3: Z3 byte. Z4: Z4 byte. Z5: Z5 byte.
Response	<byte> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:STAT:RX1:CAPT1:STS1:POH? C2` → `#HA3`
Note	If one of the alarms LOS, LOF, UNEQP or UNEQV is present NaN (section 1.6.1) is returned. Use the SONet:RX<Pt>:STS1:POH:CMASk command to selected which bytes to bee captured.

11.5.20 SONet:STATus:RX<Pt>:CAPTure:VT2:POH:TRACe?

Syntax	SONet:STATus:RX<Pt>:CAPTure:VT2:POH:TRACe?
Description	This query returns VT2 path overhead trace (J2) for the latest captured frames. 64 new frames are captured every second.
Parameter	<Pt> = Port number
Response	<J2_trace> = <STRING RESPONSE DATA>
Example	`SON:STAT:RX1:CAPT:VT2:POH:TRAC?` → `"Message_Test_J2"`
Note	If one of the alarms LOS, LOF, UNEQP or UNEQV is present, an empty string is returned.

11.5.21 SONet:STATus:RX<Pt>:CAPTure<Frame>:VT2:POH?

Syntax	SONet:STATus:RX<Pt>:CAPTure<Frame>:VT2:POH? <POH-byte>
Description	This query returns the VT2 path overhead bytes from the selected frame. 64 new frames are captured every second.
Parameters	<Pt> = Port number
	<Frame> = Frame number (1-64)
	<POH-byte> = <CHARACTER PROGRAM DATA> V5: V5-SL byte. J2: J2 byte. Z6: Z6 byte. Z7: Z7 byte.
Response	<byte> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:STAT:RX1:CAPT1:VT2:POH? V5` → `#H04`
Note	If one of the alarms LOS, LOF, UNEQP or UNEQV is present NaN (section 1.6.1) is returned.

11.5.22 SONet:STATus:RX<Pt>:CAPTure:VT15:POH:TRACe?

Syntax	SONet:STATus:RX<Pt>:CAPTure:VT15:POH:TRACe?
Description	This query returns VT1.5 path overhead trace (J2) for the latest captured frames. 64 new frames are captured every second.
Parameter	<Pt> = Port number
Response	<J2_trace> = <STRING RESPONSE DATA>
Example	`SON:STAT:RX1:CAPT:VT15:POH:TRAC?` → `"Message_Test_J2"`
Note	If one of the alarms LOS, LOF, UNEQP or UNEQV is present, an empty string is returned.

11.5.23 SONet:STATus:RX<Pt>:CAPTure<Frame>:VT15:POH?

Syntax	SONet:STATus:RX<Pt>:CAPTure<Frame>:VT15:POH? <POH-byte>
Description	This query returns the VT1.5 path overhead bytes from the selected frame. 64 new frames are captured every second.
Parameters	<Pt> = Port number
	<Frame> = Frame number (1-64)
	<POH-byte> = <CHARACTER PROGRAM DATA> V5: V5-SL byte. J2: J2 byte. Z6: Z6 byte. Z7: Z7 byte.
Response	<byte> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:STAT:RX1:CAPT1:VT15:POH? V5` → `#H04`
Note	If one of the alarms LOS, LOF, UNEQP or UNEQV is present NaN (section 1.6.1) is returned.

11.5.24 SONet:STATus:RX<Pt>:CAPTure:STS1:STS1:POH:TRACe?

Syntax	SONet:STATus:RX<Pt>:CAPTure:STS1:STS1:POH:TRACe?
Description	This query returns the STS1 path overhead trace (J1) for the latest captured frames. 64 new frames are captured every second.
Parameters	<Pt> = Port number
Response	<J1_trace> = <STRING RESPONSE DATA>
Example	`SON:STAT:RX1:CAPT:STS1:STS1:POH:TRAC?` → `"Message_Test_J1"`
Note	If one of the alarms LOS, LOF, UNEQP or LOPP is present, an empty string is returned.

11.5.25 SONet:STATus:RX<Pt>:CAPTure<Frame>:STS1:STS1:POH?

Syntax	SONet:STATus:RX<Pt>:CAPTure<Frame>:STS1:STS1:POH? <POH-byte>
Description	This query returns the STS1 path overhead bytes from the selected frame. 256 new frames are captured every second.
Parameters	<Pt> = Port number
	<Frame> = Frame number (1-64)
	<POH-byte> = <CHARACTER PROGRAM DATA> J1: J1 byte. B3: B3 byte. C2: C2 byte. G1: G1 byte. F2: F2 byte. H4: H4 byte. Z3: Z3 byte. Z4: Z4 byte. Z5: Z5 byte.
Response	<byte> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`SON:STAT:RX1:CAPT1:STS1:STS1:POH? H4` → `#HFF`
Note	If one of the alarms LOS, LOF, UNEQP or LOPP is present NaN (section 1.6.1) is returned.

11.6 APS

11.6.1 SONet:APS:STARt

Syntax	SONet:APS:STARt
Description	This command starts the APS (Automatic Protection Switching).
Parameter	None.
Response	None.
Example	`SON:APS:STAR`
Note	

11.6.2 SONet:APS:STOP

Syntax	SONet:APS:STOP
Description	This command stops the APS (Automatic Protection Switching) command.
Parameter	None.
Response	None.
Example	`SON:APS:STOP`
Note	

11.6.3 SONet:APS:RX<Pt>:PINTerpret?

Syntax	SONet:APS:RX<Pt>:PINTerpret?
Description	This query returns the protocol interpretation.
Parameter	<Pt> = Port number
Response	<interpretation> = <STRING RESPONSE DATA>

Example	`SON:APS:RX1:PINT?` →
	`"00:00:19 | Number 0 | Time:  0.000ms`
	`----------`
	`k1:  Signal degrade (protection)`
	` Destination Node (K1) 2`
	`k2:  Short`
	` Source Node (K2) 7"`

Note	

11.6.4 SONet:APS:RX<Pt>:NUMBer?

Syntax	SONet:APS:RX<Pt>:NUMBer?
Description	This query returns the number of times an APS Protocol event has occurred.
Parameter	<Pt> = Port number
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:APS:RX1:NUMB?` → `17`
Note	

11.6.5 SONet:APS:RX<Pt>:ATIMe?

Syntax	SONet:APS:RX<Pt>:ATIMe?
Description	This query returns the average time of the reference event occurrences. Unit: ms.
Parameter	<Pt> = Port number
Response	<avg> = <NR2 NUMERIC RESPONSE DATA>
Example	`SON:APS:RX1:ATIM?` → `4.000`
Note	The maximum measurable time is 10000 ms. The maximum measurable time will be responded if the result exceeds 10000 ms.

11.6.6 SONet:APS:RX<Pt>:MTIMe?

Syntax	SONet:APS:RX<Pt>:MTIMe?
Description	This query returns the maximum time of the reference event occurrences. Unit: ms.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
	<limit exceeded> = <NR1 NUMERIC RESPONSE DATA> Returns 1, if the reference maximum limit has been exceeded.
Example	`SON:APS:RX1:MTIM?` → `29.17,0`
Note	The maximum measurable time is 10000 ms. The maximum measurable time will be responded if the result exceeds 10000 ms.

11.6.7 SONet:APS:RX<Pt>:LTIMe?

Syntax	SONet:APS:RX<Pt>:LTIMe?
Description	This query returns the least (minimum) time of the reference event occurrences. Unit: ms.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`SON:APS:RX1:LTIM?` → `29.17`
Note	The maximum measurable time is 10000 ms. The maximum measurable time will be responded if the result exceeds 10000 ms.

11.6.8 SONet:APS:RX<Pt>:CTIMe?

Syntax	SONet:APS:RX<Pt>:CTIMe?
Description	This query returns the current time of the reference event occurrences. Unit: ms.
Parameter	<Pt> = Port number
Response	<current> = <NR2 NUMERIC RESPONSE DATA>
Example	`SON:APS:RX1:CTIM?` → `4.000`
Note	The maximum measurable time is 10000 ms. The maximum measurable time will be responded if the result exceeds 99999.999 ms.

11.6.9 SONet:APS:RX<Pt>:EVENt

Syntax	SONet:APS:RX<Pt>:EVENt <event>
Description	This command sets the Time Reference event.
Parameters	<Pt> = Port number
	<event> = <CHARACTER PROGRAM DATA> LOS = Loss of signal LOF = Loss of frame OOF = Out of frame AISL = MS alarm indication signal RDIL = MS remote defect indicator APSS = APS switch-over AISP = AU alarm indication signal LOPP = AU loss of pointer TIMP = HP trace identifier mismatch PLMP = HP payload label mismatch UNEQP = HP unequipped LOMV = TU loss of multiframe AISV = TU alarm indication signal LOPV = TU loss of pointer TIMV = LP trace identifier mismatch PLMV = LP payload label mismatch UNEQV = LP unequipped A1A2 = Frame alignment word error B1 = B1 checksum byte error B2 = B2 checksum byte error REIL = MS remote error indication B3 = B3 checksum byte error V5 = B3 checksum of the low-order path PERRor = Pattern error ANYERROR = Any errors *DEFault = LOS*
Response	None.
Example	`SON:APS:RX1:EVEN LOF`
Note	

Syntax	SONet:APS:RX<Pt>:EVENt?
Description	This query returns the time reference event.
Parameter	<Pt> = Port number
Response	<event> = <CHARACTER RESPONSE DATA>
Example	`SON:APS:RX1:EVEN?` → `LOF`
Note	

11.6.10 SONet:APS:RX<Pt>:MLIMit

Syntax	SONet:APS:RX<Pt>:MLIMit <max>
Description	This command sets the time reference maximum limit. Unit: ms.
Parameters	<Pt> = Port number
	<max> = <NUMERIC PROGRAM DATA> *MINimum = 0.000, MAXimum = 10000.000, DEFault = 50.000*
Response	None.
Example	`SON:APS:RX1:MLIM 50.000`
Note	

Syntax	SONet:APS:RX<Pt>:MLIMit?
Description	This query returns the time reference maximum limit. Unit: ms.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
Example	`SON:APS:RX1:MLIM?` → `50.000`
Note	

11.6.11 SONet:APS:RX<Pt>:PERiod

Syntax	SONet:APS:RX<Pt>:PERiod <period>
Description	This command sets the error free period.
Parameters	<Pt> = Port number
	<period> = <CHARACTER PROGRAM DATA> 1,10,20,30,40,50,60,70,80,90,100 Unit ms *DEFault = 1*
Response	None
Example	`SON:APS:RX1:PER 1`
Note	

Syntax	SONet:APS:RX<Pt>:PERiod?
Description	This query returns the error free period.
Parameter	<Pt> = Port number
Response	<period> = <CHARACTER PROGRAM DATA>
Example	`SON:APS:RX1:PER?` → `1`
Note	

11.6.12 SONet:APS:RX<Pt>:CONFig

Syntax	SONet:APS:RX<Pt>:CONFig <protection>,<path>
Description	This command sets the protection type and the path or architecture.
Parameters	<Pt> = Port number
	<protection> = <CHARACTER PROGRAM DATA> RING = Ring protection LINear = Linear *DEFault = RING*
	<path> = <CHARACTER PROGRAM DATA> Ring Path: SHRT = Short path LONG = Long path *DEFault = SHRT* Linear Architecture: 1P1 = 1+1 Architecture 1N = 1:n Architecture
Response	None.
Example	`SON:APS:RX1:CONF RING,SHRT`
Notes	

Syntax	SONet:APS:RX<Pt>:CONFig?
Description	This query returns the protection type and the path or architecture.
Parameter	<Pt> = Port number
Response	<protection> = <CHARACTER RESPONSE DATA>
	<path> = <CHARACTER PROGRAM DATA>
Example	`SON:APS:RX1:CONF?` → `RING,SHRT`
Note	

11.6.13 SONet:APS:RX<Pt>:RTYPe

Syntax	SONet:APS:RX<Pt>:RTYPe <type>
Description	This command sets the request type for the transmitter.
Parameter	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> Ring Request Types: for `SONet:APS:RX<Pt>:CONFig?` → `RING,....` RLOP = Lockout of protection (span) RFSS = Forced switch (span) RFSR = Forced switch (ring) RSFS = Signal fail (span) RSFR = Signal fail (ring) RSDP = Signal degrade (protection) RSDS = Signal degrade (span) RSDR = Signal degrade (ring) RMSS = Manual switch (span) RMSR = Manual switch (ring) RWTR = Wait to restore REXS = Exercise (span) REXR = Exercise (ring) RRRS = Reverse request (span) RRRR = Reverse request (ring) RNRQ = No request *DEFault = RLOP* Linear Request Types: for `SDH:APS:RX<Pt>:CONFig?` → `LIN,....` LLOP = Lockout of protection LFSW = Forced switch LSFH = Signal fail (high priority) LSFL = Signal fail (low priority) LSDH = Signal degrade (high priority) LSDL = Signal degrade (low priority) LMSW = Manual switch (ring) LWTR = Wait to restore LEXC = Exercise LRRQ = Reverse request LDNR = Do not revert LNRQ = No request *DEFault = LLOP*
Response	None.
Example	`SON:APS:RX1:RTYP RLOP`
Notes	Use the SONet:APS:RX<Pt>:APPLy command to apply the request to the transmitter.

Syntax	SONet:APS:RX<Pt>:RTYPe?
Description	This query returns the request type for the transmitter.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`SON:APS:RX1:RTYP?` → `RLOP`
Note	

11.6.14 SONet:APS:RX<Pt>:K1

Syntax	SONet:APS:RX<Pt>:K1 <value>
Description	This command sets the destination node/source channel (K1).
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=15, DEFault = 0*
Response	None.
Example	`SON:APS:RX1:K1 3`
Notes	Cannot be changed if the SONet:APS:CONFig command is set to LIN, 1P1. Changes to SONet:APS:CONFig will reset this value to 0.

Syntax	SONet:APS:RX<Pt>:K1?
Description	This query returns the destination node/source channel (K1).
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:APS:RX1:K1?` → `3`
Note	

11.6.15 SONet:APS:RX<Pt>:K2

Syntax	SONet:APS:RX<Pt>:K2 <value>
Description	This command sets the source node/bridged channel (K2).
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=15, DEFault = 0*
Response	None.
Example	`SON:APS:RX1:K2 3`
Notes	Cannot be changed if the SONet:APS:CONFig command is set to LIN, 1P1. Changes to SONet:APS:CONFig will reset this value to 0.

Syntax	SONet:APS:RX<Pt>:K2?
Description	This query returns the source node/bridged channel (K2).
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:APS:RX1:K2?` → `3`
Note	

11.6.16 SONet:APS:RX<Pt>:APPLy

Syntax	SONet:APS:RX<Pt>:APPLy
Description	This command applies the K1/K2 request setup to the transmitter.
Parameter	<Pt> = Port number
Response	None.
Example	`SON:APS:RX1:APPL`
Notes	

11.7 Tributary Scan

11.7.1 SONet:TSCan:STARt

Syntax	SONet:TSCan:STARt
Description	This command starts the tributary scan test.
Parameter	None.
Response	None.
Example	`SON:TSC:STAR`
Note	It is possible to run only one test or measurement at a time. The SONET interface must be active for at least one of the receivers.

11.7.2 SONet:TSCan:STOP

Syntax	SONet:TSCan:STOP
Description	This command stops the tributary scan test.
Parameter	None.
Response	None.
Example	`SON:TSC:STOP`
Note	

11.7.3 SONet:TSCan:RX<Pt>:NHOContainer?

Syntax	SONet:TSCan:RX<Pt>:NHOContainer?
Description	This query returns the number of High-order containers (STS3/STS1)
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:TSC:RX1:NHOC?` → `4`
Note	If '0' is returned, no High-order containers were found. To get more information, use the SONet:TSCan:RX<Pt>:MUX? query.

11.7.4 SONet:TSCan:RX<Pt>:GHOContainer?

Syntax	SONet:TSCan:RX<Pt>:GHOContainer?
Description	This query returns the state of High-order containers (STS3/STS1)
Parameter	<Pt> = Port number
Response	({<value>,}+) = <NR1 NUMERIC RESPONSE DATA> The values are presented in ascending order, meaning that STS3 #1 is the first on the list. 0 = No alarms or errors. 1 = Alarms or errors present.
Example	`SON:TSC:RX1:GHOC?` → `(0,0,1,0)`
Note	If there is one or more results, the last "," is always removed.

11.7.5 SONet:TSCan:RX<Pt>:SHOContainer

Syntax	SONet:TSCan:RX<Pt>:SHOContainer <STS>
Description	This command sets the High-order container (STS3/STS1) for scanning.
Parameters	<Pt> = Port number
	<STS> = High-order container number *MINimum=1, MAXimum=768*
Response	None.
Example	`SON:TSC:RX1:SHOC 2`
Note	

11.7.6 SONet:TSCan:RX<Pt>:SHOContainer?

Syntax	SONet:TSCan:RX<Pt>:SHOContainer?
Description	This query returns the High-order container (STS3/STS1) set for scanning.
Parameter	<Pt> = Port number
Response	<STS> = High-order container number
Example	`SON:TSC:RX1:SHOC?` → `2`
Note	

11.7.7 SONet:TSCan:RX<Pt>:DHOContainer?

Syntax	SONet:TSCan:RX<Pt>:DHOContainer? <STS3/STS1>
Description	This query returns the detailed alarm and error information from a High-order container (STS3/STS1).
Parameters	<Pt> = Port number
	<VC4> = High-order container number *MINimum=1, MAXimum=768*
Response	<selected>, = <STRING RESPONSE DATA>. {RXn}␣{High-order}, separated by one space character. RXn = RX1 or RX2 High-order = STS3#0 . . . STS3#64 or STS1#0 . . . STS1#192
	({<alarmserrors>,}∗) = <STRING RESPONSE DATA> List of alarms and errors.
Example	`SON:TSC:RX1:DHOC? 1` → `"RX1 STS3#1",("B1","A1A2")`
Note	If there is one or more responses, the last ”,” is always removed.

11.7.8 SONet:TSCan:RX<Pt>:NLOContainer?

Syntax	SONet:TSCan:RX<Pt>:NLOContainer?
Description	This query returns the number of Low-order containers (STS1/VT2/VT1.5).
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:TSC:RX1:NLOC?` → `3`
Note	

11.7.9 SONet:TSCan:RX<Pt>:GLOContainer?

Syntax	SONet:TSCan:RX<Pt>:GLOContainer?
Description	This query returns the state of Low-order containers (STS1/VT2/VT1.5).
Parameter	<Pt> = Port number
Response	({<value>,}∗) = <NR1 NUMERIC RESPONSE DATA> The values are presented in ascending order, meaning that STS3 #1 is the first in the list. 0 = No alarms or errors. 1 = Alarms or errors present.
Example	`SON:TSC:RX1:GLOC?` → `(0,1,1)`
Note	If there is one or more responses, the last ”,” is always removed.

11.7.10 SONet:TSCan:RX<Pt>:DLOContainer?

Syntax	SONet:TSCan:RX<Pt>:DLOContainer? <STS1/VT2/VT15>
Description	This query returns the detailed alarm and error information from a Low-order container (STS1/VT2/VT1.5).
Parameters	<Pt> = Port number
	<STS1/VT2/VT15> = Low-order container number *MINimum=1, MAXimum=84*
Response	<selected>, = <STRING RESPONSE DATA>. {RXn}␣{High-order}:{Low-order}, First separated by one space character and next with a colon char. RXn = RX1 or RX2 High-order = STS3#0 . . . STS3#64 or STS1#0 . . . STS1#192 Low-order = STS1#1 . . . STS1#3 or VT2#1 . . . VT2#63 or VT1.5#1 . . . VT1.5#84
	({<alarmserrors>, }∗) = <STRING RESPONSE DATA> List of alarms and errors.
Example	`SON:TSC:RX1:DLOC? 1` → `"RX1 STS3#1:STS1#1",("B1","A1A2")`
Notes	To select the High-order container, use the SONet:TSCan:RX<rx>:SHOContainer command. If there is one or more responses, the last ”,” is always removed.

11.7.11 SONet:TSCan:RX<Pt>:DMUX?

Syntax	SONet:TSCan:RX<Pt>:DMUX?
Description	This query returns the detailed alarm and error information from the MUX.
Parameter	<Pt> = Port number
Response	<selected> = <STRING RESPONSE DATA>. {Rx} MUX
	({<alarmserrors>, }∗) = <STRING RESPONSE DATA> List of alarms and errors.
Example	`SON:TSC:RX1:DMUX?` → `"Rx1 MUX",("LOS","LOF","OOF")`
Notes	This query is only available when no High-order containers are present. To determine if MUX is available, use the SONet:TSCan:RX<Pt>:NHOContainer? query. If there is one or more responses, the last ”,” is always removed.

11.8 RTD

This section document commands for the Round Trip Delay application.
Commands for general RTD settings are described in section 17.1 on page 953.

11.8.1 SONet:RTD:RX<Pt>:MLIMit

Syntax	SONet:RTD:RX<Pt>:MLIMit <max>
Description	This command sets the time reference maximum limit. Unit: us.
Parameters	<Pt> = Port number
	<max> = <NUMERIC PROGRAM DATA> *MINimum = 0.0, MAXimum = 10000000.0, DEFault = MAXimum*
Response	None.
Example	`SON:RTD:RX1:MLIM 0.0`
Note	

Syntax	SONet:RTD:RX<Pt>:MLIMit?
Description	This query returns the time reference maximum limit. Unit: us.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
Example	`SON:RTD:RX1:MLIM?` → `0.0`
Note	

11.8.2 SONet:RTD:RX<Pt>:NUMBer?

Syntax	SONet:RTD:RX<Pt>:NUMBer?
Description	This query returns the number of the RTD data.
Parameter	<Pt> = Port number
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`SON:RTD:RX1:NUMB?` → `2`
Note	

11.8.3 SONet:RTD:RX<Pt>:ATIMe?

Syntax	SONet:RTD:RX<Pt>:ATIMe?
Description	This query returns the average time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<avg> = <NR2 NUMERIC RESPONSE DATA>
Example	`SON:RTD:RX1:ATIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

11.8.4 SONet:RTD:RX<Pt>:MTIMe?

Syntax	SONet:RTD:RX<Pt>:MTIMe?
Description	This query returns the maximum time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
	<limit exceeded> = <NR1 NUMERIC RESPONSE DATA> Returns 1, if the reference maximum limit has been exceeded.
Example	`SON:RTD:RX1:MTIM?` → `1.0,0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

11.8.5 SONet:RTD:RX<Pt>:LTIMe?

Syntax	SONet:RTD:RX<Pt>:LTIMe?
Description	This query returns the least (minimum) time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`SON:RTD:RX1:LTIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

11.8.6 SONet:RTD:RX<Pt>:CTIMe?

Syntax	SONet:RTD:RX<Pt>:CTIMe?
Description	This query returns the current time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`SON:RTD:RX1:CTIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds 99999999.9 us.

Chapter 12

Ethernet

By default, the Ethernet settings cannot be changed during a measurement. Use `ETH:PORT<Pt>:SETT:ACIS` to change this behavior (see section 12.7.13).

12.1 Port Setup

12.1.1 ETHernet:PORT<Pt>:ITYPe

Syntax	ETHernet:PORT<Pt>:ITYPe <interface>
Description	This command sets the interface type.
Parameters	<Pt> = Port number
	<interface> = <CHARACTER PROGRAM DATA> ELECtrical: Electrical interface (10/100/1000 Mbps) SFP: SFP optical interface (100/1000 Mbps) SFPP: SFP+ optical interface (10 Gbps) SFP28: SFP28 optical interface (25 Gbps) QSFP: QSFP optical interface (40 Gbps) QSFP28: QSFP28 optical interface (100 Gbps) CXP: CXP optical interface (100 Gbps) CFP: CFP optical interface (40/100 Gbps) CFP2: CFP2 optical interface (100 Gbps) CFP4: CFP4 optical interface (100 Gbps) QSFP28ADpt: CFP2-QSFP28 Adaptor interface (100 Gbps) OPTical: Obsolete. For CMA 3000 backward compatibility only. Same as SFP. 10Gbps: Obsolete. For CMA 3000 backward compatibility only. Same as SFPP. *DEFault = ELECtrical*
Response	None.
Example	`ETH:PORT1:ITYP ELEC`
Notes	

Syntax	ETHernet:PORT<Pt>:ITYPe?
Description	This query returns the interface type.
Parameter	<Pt> = Port number
Response	<interface> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:ITYP?` → `ELEC`
Note	

12.1.2 ETHernet:PORT<Pt>:MODE

Syntax	ETHernet:PORT<Pt>:MODE <mode>
Description	The command sets the port mode
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: Port off ANEGotiate: Port ON in auto negotiation mode FORCed: Port ON in forced mode *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:MODE ANEG`
Note	ANEGotiate does not apply to the 10Gbps interface.

Syntax	ETHernet:PORT<Pt>:MODE?
Description	This query returns the port mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:MODE?` → `ANEG`
Note	

12.1.3 ETHernet:PORT<Pt>:ANEGotiate

Syntax	ETHernet:PORT<Pt>:ANEGotiate <speedduplex>
Description	This command sets the speed(s) and duplex to auto negotiate for electrical interface.
Parameters	<Pt> = Port number
	({<speedduplex>} + {, }∗) = <EXPRESSION PROGRAM DATA> 10MH: 10Mbps half duplex 10MF: 10Mbps full duplex 100MH: 100Mbps half duplex 100MF: 100Mbps full duplex 1GF: 1Gbps full duplex
Response	None.
Example	`ETH:PORT1:ANEG (10MH,100MF)`
Note	

Syntax	ETHernet:PORT<Pt>:ANEGotiate?
Description	This query returns the auto negotiation speed(s) and duplex for electrical interface.
Parameter	<Pt> = Port number
Response	{(<speedduplex>), }* = <EXPRESSION RESPONSE DATA> <speedduplex> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:ANEG?` → `(10MH,100MF)`
Note	

12.1.4 ETHernet:PORT<Pt>:FMODe

Syntax	ETHernet:PORT<Pt>:FMODe <mode>
Description	This command set the speed/duplex in forced mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> 10MH: 10Mbps half duplex 10MF: 10Mbps full duplex 100MH: 100Mbps half duplex 100MF: 100Mbps full duplex 1GF: 1Gbps full duplex 10GF: 10Gbps full duplex 25GF: 25Gbps full duplex 40GF: 40Gbps full duplex 100GF: 100bps full duplex *DEFault = 10MH*
Response	None.
Example	`ETH:PORT1:FMOD 100MF`
Note	

Syntax	ETHernet:PORT<Pt>:FMODe?
Description	This query returns forced mode speed/duplex.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:FMOD?` → `100MF`
Note	

12.1.5 ETHernet:PORT<Pt>:CMODe

Syntax	ETHernet:PORT<Pt>:CMODe <mode>
Description	This command sets the 1000Mbps clock mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> MASTer: Master PMASter: Prefer master SLAVe: Slave PSLave: Prefer slave *DEFault = PMASter*
Response	None.
Example	`ETH:PORT1:CMOD MAST`
Note	

Syntax	ETHernet:PORT<Pt>:CMODe?
Description	This query returns the 1000Mbps clock mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:CMOD?` → `SLAV`
Note	

12.1.6 ETHernet:PORT<Pt>:MDICrossover

Syntax	ETHernet:PORT<Pt>:MDICrossover <mode>
Description	This command sets the MDI Crossover mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> AUTO MDI MDIX *DEFault = AUTO*
Response	None.
Example	`ETH:PORT1:MDIC AUTO`
Note	

Syntax	ETHernet:PORT<Pt>:MDICrossover?
Description	This query returns the MDI Crossover mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:MDIC?` → `AUTO`
Note	

12.1.7 ETHernet:PORT<Pt>:TIMing:SOURce

Syntax	ETHernet:PORT<Pt>:TIMing:SOURce <source>
Description	This command set the transmitter timing source.
Parameter	<Pt> = Port number
	<source> = <CHARACTER PROGRAM DATA> INTernal: Internal clock. EXTernal: External input clock (2MHz/E1(SETS)/T1(BITS)). GPS: GPS signal. RX: Received clock. PTP: PTP(IEEE 1588v2) recovered clock. *DEFault = INTernal*
Response	None.
Response	None.
Example	`ETH:PORT1:TIM:SOUR INT`
Note	

Syntax	ETHernet:PORT<Pt>:TIMing:SOURce?
Description	This query returns the transmitter timing source.
Parameter	<Pt> = Port number
Response	<source> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:TIM:SOUR?` → `INT`
Note	

12.1.8 ETHernet:PORT<Pt>:PFRames

Syntax	ETHernet:PORT<Pt>:PFRames <respond>
Description	This command enable or disable respond to PAUSE frames.
Parameters	<Pt> = Port number
	<respond> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PFR ON`
Note	

Syntax	ETHernet:PORT<Pt>:PFRames?
Description	This query returns the state of respond to PAUSE frames.
Parameter	<Pt> = Port number
Response	<respond> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PFR?` → `1`
Note	

12.1.9 ETHernet:PORT<Pt>:FEC

Syntax	ETHernet:PORT<Pt>:FEC <mode>
Description	The command sets the FEC mode
Parameters	<Pt> = Port number
	<mode> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:FEC ON`
Note	This command can be used on MU110013A. This command can be used when interface type is CFP2 or QSFP28 Adpt..

Syntax	ETHernet:PORT<Pt>:FEC?
Description	This query returns the FEC mode.
Parameter	<Pt> = Port number
Response	<respond> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:FEC?` → `1`
Note	This command can be used on MU110013A. This command can be used when interface type is CFP2 or QSFP28 Adpt..

12.2 WAN

12.2.1 ETHernet:PORT<Pt>:WAN[:ENABle]

Syntax	ETHernet:PORT<Pt>:WAN[:ENABle] <enable>
Description	This command enables or disables the WAN layer of the 10Gbps interface.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Examples	`ETH:PORT1:WAN ON`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:WAN[:ENABle]?
Description	This query returns whether or not WAN is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:WAN? → 1`
Note	This command can be used on V2.00 or later.

12.2.2 ETHernet:PORT<Pt>:WAN:TERMinology

Syntax	ETHernet:PORT<Pt>:WAN:TERMinology <terminology>
Description	This command sets the WAN terminology.
Parameters	<Pt> = Port number
	<terminology> = <CHARACTER PROGRAM DATA> SONet: SONET terminology SDH: SDH terminology *DEFault = SONet*
Response	None.
Example	`ETH:PORT1:WAN:TERM SON`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:WAN:TERMinology?
Description	This query returns the WAN terminology.
Parameter	<Pt> = Port number
Response	<terminology> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:WAN:TERM? → SON`
Note	This command can be used on V2.00 or later.

12.2.3 ETHernet:PORT<Pt>:WAN:SOH:DEFault

Syntax	ETHernet:PORT<Pt>:WAN:SOH:DEFault
Description	This command sets all WAN section overhead bytes to their default value.
Parameter	<Pt> = Port number
Response	None.
Example	`ETH:PORT1:WAN:SOH:DEF`
Note	This command can be used on V2.00 or later.

12.2.4 ETHernet:PORT<Pt>:WAN:SOH:TRACe

Syntax	ETHernet:PORT<Pt>:WAN:SOH:TRACe <string>[,<idlechar>]
Description	This command sets the WAN section overhead trace (J0) to the specified string.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> This quoted string is used as section trace string.
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`ETH:PORT1:WAN:SOH:TRAC "Anritsu Network Master",#H20`
Note	If the entered trace string is more than 15 characters long, the string will be truncated. This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:WAN:SOH:TRACe?
Description	This query returns the WAN section overhead trace string and idle char.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>
	<idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:WAN:SOH:TRAC?` → `"Anritsu Network Master",#H20`
Note	This command can be used on V2.00 or later.

12.2.5 ETHernet:PORT<Pt>:WAN:SOH:TRACe:CRC

Syntax	ETHernet:PORT<Pt>:WAN:SOH:TRACe:CRC <mode>
Description	This command sets the SOH trace (J0) CRC mode (OFF/ON).
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: CRC OFF ON : CRC ON *DEFault = ON*
Response	None.
Example	`ETH:PORT1:WAN:SOH:TRAC:CRC ON`
Note	

Syntax	ETHernet:PORT<Pt>:WAN:SOH:TRACe:CRC?
Description	This query returns the mode of the SOH trace (J0) CRC.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:WAN:SOH:TRAC:CRC?` → `ON`
Note	

12.2.6 ETHernet:PORT<Pt>:WAN:SOH

Syntax	ETHernet:PORT<Pt>:WAN:SOH <SOH-byte>,<value1>[,<value2>[,<value3>]]
Description	This command sets the value of the specified bytes in the WAN section overhead.
Parameters	<Pt> = Port number
	<SOH-byte> = <CHARACTER PROGRAM DATA> A1: 3 bytes. A2: 3 bytes. J0: 3 bytes. Note: The defined trace string will be disabled. B1: 2 bytes. <value3> is ignored. E1: 3 bytes. F1: 3 bytes. D1: 3 bytes. D2: 3 bytes. D3: 3 bytes.

continued on next page...

. . . continued from previous page

	K1: 3 bytes. K2: 3 bytes. D4: 3 bytes. D5: 3 bytes. D6: 3 bytes. D7: 3 bytes. D8: 3 bytes. D9: 3 bytes. D10: 3 bytes. D11: 3 bytes. D12: 3 bytes. S1: 3 bytes. Z2: 3 bytes. E2: 3 bytes. M0: 1 byte. <value2> and <value3> are ignored. M1: 1 byte. <value2> and <value3> are ignored.
	<value1> = <NUMERIC PROGRAM DATA>
	<value2> = <NUMERIC PROGRAM DATA>
	<value3> = <NUMERIC PROGRAM DATA> *MINimum = #H00, MAXimum = #HFF*
Response	None.

Examples	`ETH:PORT1:WAN:SOH A1,#H00,#H00,#H00`
	`ETH:PORT1:WAN:SOH D7,#HFD,#H20,#H1A`
	`ETH:PORT1:WAN:SOH B1,#H04,#H05`
	`ETH:PORT1:WAN:SOH M1,#H00`

Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:WAN:SOH? <SOH-byte>
Description	This query returns the value of the specified bytes in the WAN section overhead.
Parameters	<Pt> = Port number
	<SOH-byte> = <CHARACTER PROGRAM DATA>
Response	<value1> = <HEXADECIMAL NUMERIC RESPONSE DATA> [,<value2> = <HEXADECIMAL NUMERIC RESPONSE DATA> [,<value3> = <HEXADECIMAL NUMERIC RESPONSE DATA>]]

Examples	`ETH:PORT1:WAN:SOH? A1` → `#H00,#H00,#H00`
	`ETH:PORT1:WAN:SOH? D7` → `#HFD,#H20,#H1A`
	`ETH:PORT1:WAN:SOH? B1` → `#H04,#H05`
	`ETH:PORT1:WAN:SOH? M1` → `#H00`

Note	This command can be used on V2.00 or later.

12.2.7 ETHernet:PORT<Pt>:WAN:POH:DEFault

Syntax	ETHernet:PORT<Pt>:WAN:POH:DEFault
Description	This command sets all WAN path overhead bytes to their default value.
Parameter	<Pt> = Port number
Response	None.
Example	`ETH:PORT1:WAN:POH:DEF`
Note	This command can be used on V2.00 or later.

12.2.8 ETHernet:PORT<Pt>:WAN:POH:TRACe

Syntax	ETHernet:PORT<Pt>:WAN:POH:TRACe <string>[,<idlechar>]
Description	This command sets the WAN path trace (J1) to the specified string and it's idle char.
Parameters	<Pt> = Port number
	<string> = <STRING PROGRAM DATA> This quoted string is used as section trace string.
	<idlechar> = <NUMERIC PROGRAM DATA> *DEFault = #H20*
Response	None.
Example	`ETH:PORT1:WAN:POH:TRAC "Anritsu Network Master",#H20`
Note	If the entered trace string is more than 15 characters long, the string will be truncated. This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:WAN:POH:TRACe?
Description	This query returns the WAN path trace string and idle char.
Parameter	<Pt> = Port number
Response	<string> = <STRING RESPONSE DATA>
	<idlechar> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:WAN:POH:TRAC?` → `"Anritsu Network Master",#H20`
Note	This command can be used on V2.00 or later.

12.2.9 ETHernet:PORT<Pt>:WAN:POH:TRACe:CRC

Syntax	ETHernet:PORT<Pt>:WAN:POH:TRACe:CRC <mode>
Description	This command sets the POH trace (J0) CRC mode (OFF/ON).
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: CRC OFF ON : CRC ON *DEFault = ON*
Response	None.
Example	`ETH:PORT1:WAN:POH:TRAC:CRC ON`
Note	

Syntax	ETHernet:PORT<Pt>:WAN:POH:TRACe:CRC?
Description	This query returns the mode of the POH trace (J0) CRC.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:WAN:POH:TRAC:CRC?` → `ON`
Note	

12.2.10 ETHernet:PORT<Pt>:WAN:POH

Syntax	ETHernet:PORT<Pt>:WAN:POH <POH-byte>,<value>
Description	This command sets the value of the specified byte in the WAN path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA> C2: C2 byte. G1: G1 byte. F2: Z1 / F2 byte. H4: Z2 / H4 byte. F3: Z3 / F3 byte. K3: Z4 / K3 byte. N1: Z5 / N1 byte.
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255*
Response	None.
Example	`ETH:PORT1:WAN:POH C2,0`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:WAN:POH? <POH-byte>
Description	This query returns the value of the specified byte in the WAN path overhead.
Parameters	<Pt> = Port number
	<POH-byte> = <CHARACTER PROGRAM DATA>
Response	<value> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Examples	`ETH:PORT1:WAN:POH? C2` → `#H0` `ETH:PORT1:WAN:POH? H4` → `#HFF`
Note	This command can be used on V2.00 or later.

12.3 Reflector

12.3.1 ETHernet:PORT<Pt>:REFLector:SWAP[:ENABle]

Syntax	ETHernet:PORT<Pt>:REFLector:SWAP[:ENABle] <enable>
Description	This command sets the Swap Parameter Enable.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT1:REFL:SWAP ON`
Note	

Syntax	ETHernet:PORT<Pt>:REFLector:SWAP[:ENABle]?
Description	This query returns the Swap Parameter Enable.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:REFL:SWAP? → 1`
Note	

12.3.2 ETHernet:PORT<Pt>:REFLector:SSMac

Syntax	ETHernet:PORT<Pt>:REFLector:SSMac <enable>
Description	This command enables/disables swapping of specific MAC addresses when the port is in reflector mode.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:REFL:SSM ON`
Note	

Syntax	ETHernet:PORT<Pt>:REFLector:SSMac?
Description	This query returns whether or not swapping of specific MAC addresses is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:REFL:SSM? → 1`
Note	

12.3.3 ETHernet:PORT<Pt>:REFLector:SMAC

Syntax	ETHernet:PORT<Pt>:REFLector:SMAC <address>
Description	This command sets the specific MAC address to swap when MAC address swapping is enabled.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> MAC address
Response	None.
Example	`ETH:PORT1:REFL:SMAC "00-50-C2-35-D2-EF"`
Note	

Syntax	ETHernet:PORT<Pt>:REFLector:SMAC?
Description	This query returns the specific MAC address to swap.
Parameter	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:REFL:SMAC? → "00-50-C2-35-D2-EF"`
Note	

12.3.4 ETHernet:PORT<Pt>:REFLector:IPSWap

Syntax	ETHernet:PORT<Pt>:REFLector:IPSWap <enable>
Description	This command enables/disables the IP address swap mode when the port is in reflector mode.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:REFL:IPSW ON`
Note	

Syntax	ETHernet:PORT<Pt>:REFLector:IPSWap?
Description	This query returns the IP address swap mode.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:REFL:IPSW? → 1`
Note	

12.3.5 ETHernet:PORT<Pt>:REFLector:PSWap

Syntax	ETHernet:PORT<Pt>:REFLector:PSWap <enable>
Description	This command enables/disables swapping of port numbers in UDP and TCP frames when the port is in reflector mode.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:REFL:PSW ON`
Note	

Syntax	ETHernet:PORT<Pt>:REFLector:PSWap?
Description	This query returns whether or not port swapping is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:REFL:PSW? → 1`
Note	

12.3.6 ETHernet:PORT<Pt>:REFLector:ATCP

Syntax	ETHernet:PORT<Pt>:REFLector:ATCP <enable>
Description	This command enables/disables ACK on TCP frames when the port is in reflector mode.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:REFL:ATCP ON`
Note	

Syntax	ETHernet:PORT<Pt>:REFLector:ATCP?
Description	This query returns whether or not ACK on TCP frames is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:REFL:ATCP? → 1`
Note	

12.4 PCS

12.4.1 ETHernet:PORT<Pt>:PCS:MMAPing:LANE

Syntax	ETHernet:PORT<Pt>:PCS:MMAPing:LANE <value>
Description	This command sets the PCS lane marker assignment.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List List consist of the value of the lane marker. 40G: 0 to 3 100G: 0 to 19
Response	None.
Example	`ETH:PORT1:PCS:MMAP:LANE (1,3,0,2)` `ETH:PORT1:PCS:MMAP:LANE (0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19)`
Note	This command can be used on 40/100G.

Syntax	ETHernet:PORT<Pt>:PCS:MMAPing:LANE?
Description	This query returns the PCS lane marker assignment.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION RESPONSE DATA> Format: Numeric List
Example	`ETH:PORT1:PCS:MMAP:LANE?` → `(0,1,2,3)` → `(0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19)`
Note	This command can be used on 40/100G.

12.5 Traffic

BERT, RFC-6349, Ping, and Traceroute applications use stream number one only. Setting other streams has no effect in these applications.

12.5.1 ETHernet:TRAFfic:GENerator:STARt

Syntax	ETHernet:TRAFfic:GENerator:STARt
Description	This command starts the traffic generator.
Parameter	None.
Response	None.
Example	`ETH:TRAF:GEN:STAR`
Note	This command applies to all ports.

12.5.2 ETHernet:PORT<Pt>:TRAFfic:GENerator:STARt

Syntax	ETHernet:PORT<Pt>:TRAFfic:GENerator:STARt
Description	This command starts the traffic generator.
Parameter	<Pt> = Port number
Response	None.
Example	`ETH:PORT1:TRAF:GEN:STAR`
Note	This command requires that the Port is enabled (`ETHernet:PORT<Pt>:MODE`).

12.5.3 ETHernet:TRAFfic:GENerator:STOP

Syntax	ETHernet:TRAFfic:GENerator:STOP
Description	This command stops the traffic generator.
Parameter	None.
Response	None.
Example	`ETH:TRAF:GEN:STOP`
Note	This command applies to all ports.

12.5.4 ETHernet:PORT<Pt>:TRAFfic:GENerator:STOP

Syntax	ETHernet:PORT<Pt>:TRAFfic:GENerator:STOP
Description	This command stops the traffic generator.
Parameter	<Pt> = Port number
Response	None.
Example	`ETH:PORT1:TRAF:GEN:STOP`
Note	

12.5.5 ETHernet:PORT<Pt>:TRAFfic:GENerator:STATus?

Syntax	ETHernet:PORT<Pt>:TRAFfic:GENerator:STATus?
Description	This query returns the current status of the traffic generator.
Parameter	<Pt> = Port number
Response	<stat> = <NR1 NUMERIC RESPONSE DATA> 0: Traffic generator inactive 1: Traffic generator active
Example	`ETH:PORT1:TRAF:GEN:STAT?` → `1`
Note	

12.5.6 ETHernet:PORT<Pt>:TRAFfic:DMODe

Syntax	ETHernet:PORT<Pt>:TRAFfic:DMODe <mode>
Description	This command sets the duration mode for the traffic generator.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> CONTinuous: Continuous FRAMes: Frames SEConds: Seconds *DEFault = CONTinuous*
Response	None.
Example	`ETH:PORT1:TRAF:DMOD CONT`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:DMODe?
Description	This query returns the duration mode for the traffic generator.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:TRAF:DMOD?` → `SEC`
Note	

12.5.7 ETHernet:PORT<Pt>:TRAFfic:DURation

Syntax	ETHernet:PORT<Pt>:TRAFfic:DURation <duration>
Description	This command sets the duration for the traffic generator.
Parameters	<Pt> = Port number
	<duration> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 2000000000, DEFault = 5*
Response	None.
Example	`ETH:PORT1:TRAF:DUR 1`
Note	Unit for this command can be either [frames] or [seconds] depending on `DMODe`.

Syntax	ETHernet:PORT<Pt>:TRAFfic:DURation?
Description	This query returns the duration for the traffic generator.
Parameter	<Pt> = Port number
Response	<duration> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:DUR? → 1`
Note	

12.5.8 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:TXMode

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:TXMode <mode>
Description	This command sets the transmitting mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<mode> = <CHARACTER PROGRAM DATA> NORMal: BURSt: Burst transmitting *DEFault = NORMal*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:TXM BURS`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:TXMode?
Description	This query returns the transmitting mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:TXM? → BURS`
Note	

12.5.9 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:PROFile

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:PROFile <profile>
Description	This command sets the stream Line Load Profile.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<profile> = <CHARACTER PROGRAM DATA> CONStant: Constant line load profile RAMP: Ramp line load profile *DEFault = CONStant*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:LL:PROF CONS`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:PROFile?
Description	This query returns the stream Line Load Profile.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<profile> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:LL:PROF?` → `RAMP`
Note	

12.5.10 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:PROFile:MRAMp [:MODe]

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:PROFile:MRAMp[:MODe] <enable>
Description	This command enables/disables milliseconds ramp.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:LL:PROF:MRAM ON`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:PROFile:MRAMp[:MODe]?
Description	This query returns if milliseconds ramp is enabled/disabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:LL:PROF:MRAM?` → `0`
Note	

12.5.11 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad[:CONStant]

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad[:CONStant] <load>
Description	This command sets the stream Line Load in constant mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<load> = <NUMERIC PROGRAM DATA> *MINimum=0.0000, MAXimum=100.0000* [1]*, DEFault=100.0000* *Allowed suffixes = PCT, MBPS, KBPS, IFG. Default = PCT*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:LL 10.0000`
Notes	The minimum possible line load is 0.0008PCT. The maximum combined line load for all 16 streams is 100PCT. [1] The maximum allowed Line Load varies depending on the stream frame size. MINimum, MAXimum and DEFault are all in PCT.

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad[:CONStant]? [<suffix>]
Description	This query returns the stream Line Load is constant mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second IFG: Inter Frame Gap *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:LL? → 10.0000`
Note	

12.5.12 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP[:MODE]

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP[:MODE] <mode>
Description	This command sets the ramp mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<mode> = <CHARACTER PROGRAM DATA> KEEPend: Maintain line load level at ramp end INVert: Invert ramp REPeat: Repeat ramp *DEFault = KEEPend*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:LL:RAMP KEEP`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP[:MODE]?
Description	This query returns the ramp mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:LL:RAMP? → INV`
Note	

12.5.13 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:STARt

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:STARt <load>
Description	This command sets the stream initial line load in ramp mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<load> = <NUMERIC PROGRAM DATA> *MINimum=0.0000, MAXimum=100.0000* [1]*, DEFault=0.0000* *Allowed suffixes = PCT, MBPS, KBPS, IFG. Default = PCT*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:LL:RAMP:STAR 10.0000`
Notes	The minimum possible line load is 0.0008PCT. The maximum combined line load for all 16 streams are 100PCT. [1] The maximum allowed Line Load varies depending on the stream frame size. MINimum, MAXimum and DEFault are all in PCT.

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:STARt? [<suffix>]
Description	This query returns the stream initial line load in ramp mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second IFG: Inter Frame Gap *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:LL:RAMP:STAR?` → `10.0000`
Note	

12.5.14 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:END

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:END <load>
Description	This command sets the stream end line load in ramp mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<load> = <NUMERIC PROGRAM DATA> *MINimum=0.0000, MAXimum=100.0000* [1]*, DEFault=0.0000* *Allowed suffixes = PCT, MBPS, KBPS, IFG. Default = PCT*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:LL:RAMP:END 10.0000`
Notes	The minimum possible line load is 0.0008PCT. The maximum combined line load for all 16 streams are 100PCT. [1] The maximum allowed Line Load varies depending on the stream frame size. MINimum, MAXimum and DEFault are all in PCT.

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:END? [<suffix>]
Description	This query returns the stream end line load in ramp mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second IFG: Inter Frame Gap *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:LL:RAMP:END?` → `10.0000`
Note	

12.5.15 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:STEP

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:STEP <load>
Description	This command sets the stream line load step size, in ramp mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<load> = <NUMERIC PROGRAM DATA> *MINimum=0.0000, MAXimum=100.0000* [1]*, DEFault=1.0000* *Allowed suffixes = PCT, MBPS, KBPS, IFG. Default = PCT*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:LL:RAMP:STEP 10.0000`
Notes	The minimum possible line load is 0.0008PCT. [1] The maximum allowed Line Load varies depending on the stream frame size. MINimum, MAXimum and DEFault are all in PCT.

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:STEP? [<suffix>]
Description	This query returns the stream line load step size, in ramp mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second IFG: Inter Frame Gap *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:LL:RAMP:STEP?` → `10.0000`
Note	

12.5.16 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:DURation

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:DURation <dur>
Description	This command sets the stream step line load duration in ramp mode. Unit: Seconds.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<dur> = <NUMERIC PROGRAM DATA> *MINimum = 3, MAXimum = 3600, DEFault = 10*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:LL:RAMP:DUR 10`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:RAMP:DURation?
Description	This query returns the stream step line load duration in ramp mode. Unit: Seconds.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<load> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:LL:RAMP:DUR?` → `10`
Note	

12.5.17 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:PROFile:MRAMp :DURation

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:PROFile:MRAMp:DURation <dur>
Description	This command sets the stream step line load duration in milliseconds ramp mode. Unit: Seconds.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<dur> = <NUMERIC PROGRAM DATA> *MINimum = 0.001, MAXimum = 3.000, DEFault = 0.001*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:LL:PROF:MRAM:DUR 1.000`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:LLoad:PROFile:MRAMp:DURation?
Description	This query returns the stream step line load duration in milliseconds ramp mode. Unit: Seconds.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<load> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:LL:PROF:MRAM:DUR?` → `1.000`
Note	

12.5.18 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:PROFile

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:PROFile <profile>
Description	This command sets the stream frame size profile.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<profile> = <CHARACTER PROGRAM DATA> CONStant: Constant frame size STEPped: Stepped frame size profile RANDom: Random frame size profile BINCrement: 1 byte increment *DEFault = CONStant*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:FSIZ:PROF CONS`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:PROFile?
Description	This query returns the stream frame size profile.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<profile> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:FSIZ:PROF?` → `RAND`
Note	

12.5.19 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe[:STARt]

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe[:STARt] <size>
Description	This command sets start frame size. Unit: Bytes
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:FSIZ 120`
Notes	This is used as the frame size for the Constant profile and start frame size for the Stepped and Random profiles. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe[:STARt]?
Description	This query returns the start frame size. Unit: Bytes
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:FSIZ?` → `100`
Note	

12.5.20 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:END

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:END <size>
Description	This command sets end frame size. Unit: Bytes
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:FSIZ:END 320`
Notes	This is used as the end frame size for the Stepped and Random profiles. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:END?
Description	This query returns the end frame size. Unit: Bytes
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:FSIZ:END?` → `320`
Note	

12.5.21 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:STEP

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:STEP <size>
Description	This command sets step frame size. Unit: Bytes
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:FSIZ:STEP 100`
Notes	This is used as the step frame size for the Stepped profile.

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:STEP?
Description	This query returns the step frame size. Unit: Bytes
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:FSIZ:STEP?` → `100`
Note	

12.5.22 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:DURation

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:DURation <duration>
Description	This command sets the step duration. Unit: Seconds.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<duration> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 3600, DEFault = 1*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:FSIZ:DUR 5`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:FSIZe:DURation?
Description	This query returns the step duration. Unit: Seconds.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<duration> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:FSIZ:DUR?` → `10`
Note	

12.5.23 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:PROFile

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:PROFile <profile>
Description	This command sets the stream profile.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<profile> = <CHARACTER PROGRAM DATA> DATA: Data VIDeo: Video VOICe: Voice *DEFault = DATA*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:PROF VID`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:PROFile?
Description	This query returns the stream profile.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<profile> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:PROF?` → `VID`
Note	

12.5.24 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:ENCoding:VIDeo

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:ENCoding:VIDeo <codec>
Description	This command sets the stream encoding video codec.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<codec> = <CHARACTER PROGRAM DATA> SDMPEG2: SDTV (MPEG2) HDMPEG2: HDTV (MPEG2) HDMPEG4: HDTV (MPEG4) *DEFault = SDMPEG2*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:ENC:VID HDMPEG2`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:ENCoding:VIDeo?
Description	This query returns the stream encoding video codec.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<codec> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:ENC:VID?` → `HDMPEG2`
Note	

12.5.25 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:ENCoding:VOICe

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:ENCoding:VOICe <codec>
Description	This command sets the stream encoding voice codec.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<codec> = <CHARACTER PROGRAM DATA> G711: VoIP G.711 G7231: VoIP G.723.1 G729: VoIP G.729 *DEFault = G711*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:ENC:VOIC G7231`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:ENCoding:VOICe?
Description	This query returns the stream encoding voice codec.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<codec> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:ENC:VOIC?` → `G7231`
Note	

12.5.26 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:NCHannels

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:NCHannels <channel>
Description	This command sets the number of channels.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<channel> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 100000, DEFault = 1*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:NCH 3`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:NCHannels?
Description	This query returns the number of channels.
Parameter	<Pt> = Port number
Response	<channel> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:NCH?` → `3`
Note	

12.5.27 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:MODe

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:MODe <mode>
Description	This command sets the burst mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<mode> = <CHARACTER PROGRAM DATA> OFF : OFF CONStant : constant RAMP : ramp *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:BURS:MOD RAMP`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:MODe?
Description	This query returns the burst mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:BURS:MOD?` → `RAMP`
Note	

12.5.28 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:NBURst:MODe

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:NBURst:MODe <enable>
Description	This command enables/disables number of burst mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:BURS:NBUR:MOD ON`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:NBURst:MODe?
Description	This query returns if number of burst mode is enabled/disabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:BURS:NBUR:MOD?` → `0`
Note	

12.5.29 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:NBURst:BNUMber

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:NBURst:BNUMber <number>
Description	This command sets the number of burst .
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<number> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 65535, DEFault = 1*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:BURS:NBUR:BNUM 100`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:NBURst:BNUMber?
Description	This query returns the number of burst.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:BURS:NBUR:BNUM?` → `100`
Note	

12.5.30 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:PDUTy [:DUTY]

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:PDUTy[:DUTY] <duty>, <period>
Description	This command sets the burst configuration. Duty is prior than period . Period Unit: Milliseconds
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<duty> = <NUMERIC PROGRAM DATA> *MINimum = 0.00000, MAXimum = 100.00000, DEFault = 0.00000*
	<period> = <NUMERIC PROGRAM DATA> *MINimum = 0.00001, MAXimum = 5000.0000, DEFault = 1.0000*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:BURS:CONS:PDUT 100.0000, 1.0000`
Note	

12.5.31 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:PDUTy :PERiod

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:PDUTy:PERiod <duty>, <period>
Description	This command sets the burst configuration. Period is prior than duty. Period Unit: Milliseconds
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<duty> = <NUMERIC PROGRAM DATA> *MINimum = 0.00000, MAXimum = 100.00000, DEFault = 0.00000*
	<period> = <NUMERIC PROGRAM DATA> *MINimum = 0.00001, MAXimum = 5000.0000, DEFault = 1.0000*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:BURS:CONS:PDUT:PER 100.0000, 1.0000`
Note	

12.5.32 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:DUTY?

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:DUTY?
Description	This query returns the duty percentage.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<duty> = <NUMERIC PROGRAM DATA>
Example	`ETH:PORT1:TRAF:STR1:BURS:CONS:DUTY?` → `100.00000`
Note	

12.5.33 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:PERiod?

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:PERiod?
Description	This query returns the period milliseconds.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<period> = <NUMERIC PROGRAM DATA>
Example	`ETH:PORT1:TRAF:STR1:BURS:CONS:PER?` → `1.00000`
Note	

12.5.34 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:BLENgth

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:BLENgth <length>[,<suffix>]
Description	This command sets the burst length of constant.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 8.0000, MAXimum = 919117647.0000, DEFault = 1.0000*
	<suffix> = <CHARACTER PROGRAM DATA> FRAMes : number of frames BYTes : bytes MSEConds : milli seconds *DEFault = FRAMes*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:BURS:CONS:BLEN 10`
Notes	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:BLENgth? [<suffix>]
Description	This query returns the burst length of constant.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> FRAMes: number of frames BYTes: bytes MSEConds: milli seconds *DEFault = FRAMes*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:BURS:CONS:BLEN?` → `10`
Note	

12.5.35 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:BGAP

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:BGAP <length>[,<suffix>]
Description	This command sets the inter burst gap of constant.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 128 bytes, MAXimum = 62500000000 bytes, DEFault = 1024 bytes*
	<suffix> = <CHARACTER PROGRAM DATA> BYTes : bytes MSEConds : milli seconds *DEFault = BYTes*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:BURS:CONS:BGAP 10`
Notes	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:CONStant:BGAP? [<suffix>]
Description	This query returns the inter burst gap of constant.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> BYTes: bytes MSEConds: milli seconds *DEFault = BYTes*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:BURS:CONS:BGAP?` → `10`
Note	

12.5.36 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:RMODe

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:RMODe <mode>
Description	This command sets the burst ramp mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<mode> = <CHARACTER PROGRAM DATA> KEEPend : keep end REPeat : repeat *DEFault = KEEPend*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:BURS:RAMP:RMOD KEEP`
Note	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:RMODe?
Description	This query returns the burst ramp mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:BURS:RAMP:RMOD?` → `KEEP`
Note	

12.5.37 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:BLENgth :STARt

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:BLENgth:STARt <length>[,<suffix>]
Description	This command sets the burst ramp start length of ramp.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 8.0000, MAXimum = 919117647.0000, DEFault = 1.0000*
	<suffix> = <CHARACTER PROGRAM DATA> FRAMes : number of frames BYTes : bytes MSEConds : milli seconds *DEFault = FRAMes*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:BURS:RAMP:BLEN:STAR 10`
Notes	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:BLENgth:STARt? [<suffix>]
Description	This query returns the burst ramp end length of ramp.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> FRAMes: number of frames BYTes: bytes MSEConds: milli seconds *DEFault = FRAMes*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:BURS:RAMP:BLEN:STAR?` → `10`
Note	

12.5.38 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:BLENgth :END

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:BLENgth:END <length>[,<suffix>]
Description	This command sets the burst ramp end length of ramp.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 8.0000, MAXimum = 919117647.0000, DEFault = 1.0000*
	<suffix> = <CHARACTER PROGRAM DATA> FRAMes : number of frames BYTes : bytes MSEConds : seconds *DEFault = FRAMes*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:BURS:RAMP:BLEN:END 10`
Notes	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:BLENgth:END? [<suffix>]
Description	This query returns the burst ramp start length of ramp.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> FRAMes: number of frames BYTes: bytes MSEConds: milli seconds *DEFault = FRAMes*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:BURS:RAMP:BLEN:END? → 10`
Note	

12.5.39 ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:BGAP

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:BGAP <length>[,<suffix>]
Description	This command sets the inter burst gap of ramp.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 128 bytes, MAXimum = 62500000000 bytes, DEFault = 1024 bytes*
	<suffix> = <CHARACTER PROGRAM DATA> BYTes : bytes MSEConds : milli seconds *DEFault = BYTes*
Response	None.
Example	`ETH:PORT1:TRAF:STR1:BURS:RAMP:BGAP 10`
Notes	

Syntax	ETHernet:PORT<Pt>:TRAFfic:STReam<St>:BURSt:RAMP:BGAP? [<suffix>]
Description	This query returns the inter burst gap of ramp.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> BYTes: bytes MSEConds: milli seconds *DEFault = BYTes*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:TRAF:STR1:BURS:RAMP:BGAP? → 10`
Note	

12.6 Frame Content

12.6.1 ETHernet:PORT<Pt>:STReam<St>:FRAMed

Syntax	ETHernet:PORT<Pt>:STReam<St>:FRAMed <enable>
Description	This command enables/disables framed Ethernet.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT1:STR1:FRAM OFF`
Note	BER is the only allowed protocol when framed Ethernet is OFF (unframed).

Syntax	ETHernet:PORT<Pt>:STReam<St>:FRAMed?
Description	This query returns if framed Ethernet is enabled/disabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:FRAM?` → `0`
Note	

12.6.2 ETHernet:PORT<Pt>:STReam<St>:MAC:SOURce

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:SOURce <address>
Description	This command sets the MAC source address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> MAC address
Response	None.
Example	`ETH:PORT1:STR1:MAC:SOUR "00-50-C2-35-D2-EF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:SOURce?
Description	This query returns the MAC source address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:MAC:SOUR?` → `"00-50-C2-35-D2-EF"`
Note	

12.6.3 ETHernet:PORT<Pt>:STReam<St>:MAC:DESTination

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:DESTination <address>
Description	This command sets the MAC destination address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> MAC address
Response	None.
Example	`ETH:PORT1:STR1:MAC:DEST "00-50-C2-35-D2-EF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:DESTination?
Description	This query returns the MAC destination address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:MAC:DEST?` → `"00-50-C2-35-D2-EF"`
Note	

12.6.4 ETHernet:PORT<Pt>:STReam<St>:MAC:ARP

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:ARP <enable>
Description	This command enables/disables use of ARP to resolve destination MAC address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:MAC:ARP ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:ARP?
Description	This query returns whether or not ARP is used.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:MAC:ARP?` → `1`
Note	

12.6.5 ETHernet:PORT<Pt>:STReam<St>:MAC:ARP:LKUP

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:ARP:LKUP
Description	This command execute ARP. this command valid only when IpV4 is selected on layer3.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	None.
Example	`ETH:PORT1:STR1:MAC:ARP:LKUP`
Note	

12.6.6 ETHernet:PORT<Pt>:STReam<St>:MAC:ARP:RESult?

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:ARP:RESult?
Description	This query returns the ARP condition. this command valid only when IpV4 is selected on layer3.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<status> = <CHARACTER RESPONSE DATA> SUCCESS: ARP lookup succeeded. TIMEOUT: ARP lookup timed out.
Example	`ETH:PORT1:STR1:MAC:ARP:RES?` → `SUCCESS`
Note	

12.6.7 ETHernet:PORT<Pt>:STReam<St>:MAC:NDP

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:NDP <enable>
Description	This command enables/disables use of NDP to resolve destination MAC address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:MAC:NDP ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:NDP?
Description	This query returns whether or not NDP is used.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:MAC:NDP? → 1`
Note	

12.6.8 ETHernet:PORT<Pt>:STReam<St>:MAC:NDP:LKUP

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:NDP:LKUP
Description	This command execute NDP. this command valid only when IPv6 is selected on layer3.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	None.
Example	`ETH:PORT1:STR1:MAC:NDP:LKUP`
Note	

12.6.9 ETHernet:PORT<Pt>:STReam<St>:MAC:NDP:RESult?

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:NDP:RESult?
Description	This query returns the NDP condition. this command valid only when IPv6 is selected on layer3.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<status> = <CHARACTER RESPONSE DATA> SUCCESS: ARP lookup succeeded. TIMEOUT: ARP lookup timed out.
Example	`ETH:PORT1:STR1:MAC:NDP:RES? → SUCCESS`
Note	

12.6.10 ETHernet:PORT<Pt>:STReam<St>:MAC:BROadcast

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:BROadcast <percentage>
Description	This command sets the percentage of stream packets that will be transmitted as broadcast packets.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<percentage> = <NUMERIC PROGRAM DATA> *MINimum=0.0, MAXimum=100.0, DEFault=0.0*
Response	None.
Example	`ETH:PORT1:STR1:MAC:BRO 10.0`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:BROadcast?
Description	This query returns the stream broadcast percentage.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<percentage> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:MAC:BRO?` → `10.5`
Note	

12.6.11 ETHernet:PORT<Pt>:STReam<St>:MAC:DEFault

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:DEFault <enable>
Description	This command enables/disables use of default source MAC address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:MAC:DEF ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:DEFault?
Description	This query returns if the default source MAC address is used.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:MAC:DEF?` → `1`
Note	

12.6.12 ETHernet:PORT<Pt>:STReam<St>:MAC:ETYPe

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:ETYPe <type>
Description	This command sets MAC level Ethertype.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<type> = <NUMERIC PROGRAM DATA> Acceptable values: #H8100 #H88A8 #H9100 #H9200
Response	None.
Example	`ETH:PORT1:STR1:MAC:ETYP #H8100`
Note	This command can only be used when MPLS is disabled and VLAN is enabled. When IPv4 and IPv6 is disabled this level is set by :MAC:L3EType command (see section 12.6.13).

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:ETYPe?
Description	This query returns MAC level Ethertype.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<type> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:MAC:ETYP?` → `#H8100`
Note	This command can only be used when MPLS is disabled and VLAN is enabled.

12.6.13 ETHernet:PORT<Pt>:STReam<St>:MAC:L3EType

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:L3EType <type>
Description	This command sets the Ethertype for layer three when no protocol is defined for this layer.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<type> = <NUMERIC PROGRAM DATA> *MINimum=1501, MAXimum=65535, DEFault=1501*
Response	None.
Example	`ETH:PORT1:STR1:MAC:L3ET #H0800`
Note	This command can only be used when IPv4 or IPv6 is disabled. When VLAN is enabled this level is set by :MAC:ETYPe command (see section 12.6.12).

Syntax	ETHernet:PORT<Pt>:STReam<St>:MAC:L3EType?
Description	This query returns the Ethertype for layer three when no protocol is defined for this layer.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<type> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:MAC:L3ET?` → `#0800`
Note	This command can only be used when IPv4 or IPv6 is disabled.

12.6.14 ETHernet:PORT<Pt>:STReam<St>:MPLS[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS[:ENABle] <enable>
Description	This command enables/disables MPLS.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:MPLS ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS[:ENABle]?
Description	This query returns the state of MPLS.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:MPLS?` → `1`
Note	

12.6.15 ETHernet:PORT<Pt>:STReam<St>:MPLS:LCount

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:LCount <levels>
Description	This command sets the number of active MPLS levels.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<levels> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8, DEFault=1*
Response	None.
Example	`ETH:PORT1:STR1:MPLS:LC 2`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:LCount?
Description	This query returns the number of active MPLS levels.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<levels> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:MPLS:LC?` → `2`
Note	

12.6.16 ETHernet:PORT<Pt>:STReam<St>:MPLS:LEVel<Lv>:LABel

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:LEVel<Lv>:LABel <label>
Description	This command sets the MPLS label.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = MPLS level (1-N[1])
	<label> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1048575, DEFault=0*
Response	None.
Example	`ETH:PORT1:STR1:MPLS:LEV1:LAB 1048575`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active MPLS Levels (see section 12.32.81). Level 1 corresponds to the level at the top of the label stack and N is the level at the bottom of the stack (bottom of stack flag set).

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:LEVel<Lv>:LABel?
Description	This query returns the MPLS label.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = MPLS level (1-N)
Response	<label> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:MPLS:LEV1:LAB?` → `1048575`
Note	

12.6.17 ETHernet:PORT<Pt>:STReam<St>:MPLS:LEVel<Lv>:EBITs

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:LEVel<Lv>:EBITs <value>
Description	This command sets the MPLS experimental bits.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = MPLS level (1-N[1])
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=7, DEFault=0*
Response	None.
Example	`ETH:PORT1:STR1:MPLS:LEV1:EBIT 5`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active MPLS Levels (see section 12.32.81). Level 1 corresponds to the level at the top of the label stack and N is the level at the bottom of the stack (bottom of stack flag set).

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:LEVel<Lv>:EBITs?
Description	This query returns the MPLS experimental bits.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = MPLS level (1-N)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:MPLS:LEV1:EBIT?` → `5`
Note	

12.6.18 ETHernet:PORT<Pt>:STReam<St>:MPLS:LEVel<Lv>:TTL

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:LEVel<Lv>:TTL <value>
Description	This command sets the MPLS time to live.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = MPLS level (1-N[1])
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=32*
Response	None.
Example	`ETH:PORT1:STR1:MPLS:LEV1:TTL 32`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active MPLS Levels (see section 12.32.81). Level 1 corresponds to the level at the top of the label stack and N is the level at the bottom of the stack (bottom of stack flag set).

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:LEVel<Lv>:TTL?
Description	This query returns the MPLS time to live.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = MPLS level (1-N)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:MPLS:LEV1:TTL?` → `32`
Note	

12.6.19 ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet[:ENABle] <enable>
Description	This command enables/disables Ethernet over MPLS
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:MPLS:ETH ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet[:ENABle]?
Description	This query returns the state of Ethernet over MPLS.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:MPLS:ETH?` → `1`
Note	

12.6.20 ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:AINCrement

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:AINCrement <enable>
Description	This command enables/disables EoMPLS auto increment of sequence numbers.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:MPLS:ETH:AINC ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:AINCrement?
Description	This query returns the state of EoMPLS auto increment of sequence numbers.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:MPLS:ETH:AINC?` → `1`
Note	

12.6.21 ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:SMAC

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:SMAC <address>
Description	This command sets EoMPLS source MAC address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> MAC address
Response	None.
Example	`ETH:PORT1:STR1:MPLS:ETH:SMAC "66-60-C2-35-D3-EF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:SMAC?
Description	This query returns the EoMPLS source MAC address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:MPLS:ETH:SMAC?` → `"66-60-C2-35-D3-EF"`
Note	

12.6.22 ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:DMAC

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:DMAC <address>
Description	This command sets the EoMPLS destination MAC address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> MAC address
Response	None.
Example	`ETH:PORT1:STR1:MPLS:ETH:DMAC "66-60-C2-35-D3-EF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:DMAC?
Description	This query returns the EoMPLS destination MAC address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:MPLS:ETH:DMAC?` → `"66-60-C2-35-D3-EF"`
Note	

12.6.23 ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:ETYPe

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:ETYPe <type>
Description	This command sets the EoMPLS Ethertype.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<type> = <NUMERIC PROGRAM DATA> Acceptable values: #H8100 #H88A8 #H9100 #H9200
Response	None.
Example	`ETH:PORT1:STR1:MPLS:ETH:ETYP #H8100`
Note	This command can only be used when MPLS and VLAN are enabled.

Syntax	ETHernet:PORT<Pt>:STReam<St>:MPLS:ETHernet:ETYPe?
Description	This query returns the EoMPLS Ethertype.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<type> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:MPLS:ETH:ETYP?` → `#H8100`
Note	

12.6.24 ETHernet:PORT<Pt>:STReam<St>:MIM[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM[:ENABle] <enable>
Description	This command enables/disables MiM.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:MIM ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM[:ENABle]?
Description	This query returns whether or not MiM is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:MIM?` → `1`
Note	

12.6.25 ETHernet:PORT<Pt>:STReam<St>:MIM:BTAG:DEI

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:BTAG:DEI <enable>
Description	This command sets the B-TAG DEI bit.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:MIM:BTAG:DEI ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:BTAG:DEI?
Description	This query returns the B-TAG DEI bit.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:MIM:BTAG:DEI?` → `1`
Note	

12.6.26 ETHernet:PORT<Pt>:STReam<St>:MIM:BTAG:PRiority

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:BTAG:PRiority <priority>
Description	This command sets the B-TAG Priority (PCP).
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<priority> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:MIM:BTAG:PR 7`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:BTAG:PRiority?
Description	This query returns the B-TAG Priority (PCP).
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<priority> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:MIM:BTAG:PR?` → `7`
Note	

12.6.27 ETHernet:PORT<Pt>:STReam<St>:MIM:BTAG:VID

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:BTAG:VID <vid>
Description	This command sets the B-TAG Backbone VLAN ID.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<vid> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 4095, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:MIM:BTAG:VID 1024`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:BTAG:VID?
Description	This query returns the B-TAG Backbone VLAN ID.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<vid> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:MIM:BTAG:VID?` → `1024`
Note	

12.6.28 ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:UCA

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:UCA <enable>
Description	This command sets the I-TAG UCA bit.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:MIM:ITAG:UCA ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:UCA?
Description	This query returns the I-TAG UCA bit.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:MIM:ITAG:UCA? → 1`
Note	

12.6.29 ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:DEI

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:DEI <enable>
Description	This command sets the I-TAG DEI bit.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:MIM:ITAG:DEI ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:DEI?
Description	This query returns the I-TAG DEI bit.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:MIM:ITAG:DEI? → 1`
Note	

12.6.30 ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:PRiority

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:PRiority <priority>
Description	This command sets the I-TAG Priority (PCP).
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<priority> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:MIM:ITAG:PR 7`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:PRiority?
Description	This query returns the I-TAG Priority (PCP).
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<priority> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:MIM:ITAG:PR?` → `7`
Note	

12.6.31 ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:SID

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:SID <sid>
Description	This command sets the I-TAG SID.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<sid> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 16777215, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:MIM:ITAG:SID 1`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:ITAG:SID?
Description	This query returns the I-TAG SID.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<sid> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:MIM:ITAG:SID?` → `1`
Note	

12.6.32 ETHernet:PORT<Pt>:STReam<St>:MIM:ETHernet:SMAC

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:ETHernet:SMAC <address>
Description	This command sets the encapsulated customer source MAC address.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> MAC address
Response	None.
Example	`ETH:PORT1:STR1:MIM:ETH:SMAC "00-50-C2-35-D2-EF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:ETHernet:SMAC?
Description	This query returns the encapsulated customer source MAC address.
Parameter	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:MIM:ETH:SMAC?` → `"00-50-C2-35-D2-EF"`
Note	

12.6.33 ETHernet:PORT<Pt>:STReam<St>:MIM:ETHernet:DMAC

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:ETHernet:DMAC <address>
Description	This command sets the encapsulated customer destination MAC address.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> MAC address
Response	None.
Example	`ETH:PORT1:STR1:MIM:ETH:DMAC "00-50-C2-35-D2-EF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:ETHernet:DMAC?
Description	This query returns the encapsulated customer destination MAC address.
Parameter	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:MIM:ETH:DMAC?` → `"00-50-C2-35-D2-EF"`
Note	

12.6.34 ETHernet:PORT<Pt>:STReam<St>:MIM:ETHernet:ETYPe

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:ETHernet:ETYPe <type>
Description	This command sets the encapsulated customer Ethernet Type.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<type> = <NUMERIC PROGRAM DATA> Acceptable values: #H8100 #H88A8 #H9100 #H9200
Response	None.
Example	`ETH:PORT1:STR1:MIM:ETH:ETYP #H8100`
Note	This command can only be used when MiM and VLAN is enabled. When IPv4 and IPv6 is disabled this level is set by :MAC:L3EType command (see section 12.6.13).

Syntax	ETHernet:PORT<Pt>:STReam<St>:MIM:ETHernet:ETYPe?
Description	This query returns the encapsulated customer Ethernet Type.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<type> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:MIM:ETH:ETYP?` → `#H8100`
Note	

12.6.35 ETHernet:PORT<Pt>:STReam<St>:VLAN[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:VLAN[:ENABle] <enable>
Description	This command enables/disables VLAN.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:VLAN ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:VLAN[:ENABle]?
Description	This query returns whether or not VLAN is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<boolean> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:VLAN?` → `1`
Note	

12.6.36 ETHernet:PORT<Pt>:STReam<St>:VLAN:LCount

Syntax	ETHernet:PORT<Pt>:STReam<St>:VLAN:LCount <levels>
Description	This command sets the number of active VLAN levels.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<levels> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8, DEFault=1*
Response	None.
Example	`ETH:PORT1:STR1:VLAN:LC 2`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:VLAN:LCount?
Description	This query returns the number of active VLAN levels.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<levels> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:VLAN:LC?` → `2`
Note	

12.6.37 ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:ID

Syntax	ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:ID <number>
Description	This command sets the VLAN ID.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = VLAN level (1-N[1])
	<number> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=4095, DEFault=0*
Response	None.
Example	`ETH:PORT1:STR1:VLAN:LEV1:ID 1024`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels (see section 12.32.97). Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame).

Syntax	ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:ID?
Description	This query returns the VLAN ID.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = VLAN level (1-N)
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:VLAN:LEV1:ID?` → `1024`
Note	

12.6.38 ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:CFI

Syntax	ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:CFI <enable>
Description	This command enables/disables the VLAN canonical format indicator.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = VLAN level (1-N[1])
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:VLAN:LEV1:CFI ON`

continued on next page...

. . . continued from previous page

Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels (see section 12.32.97). Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame). [2] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition. [3] This command is as same as `ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:DEI`.

Syntax	ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:CFI?
Description	This query returns the state of the VLAN canonical format indicator.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = VLAN level (1-N)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:VLAN:LEV1:CFI?` → `1`
Note	[1] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition. [2] This command is as same as `ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:DEI?`.

12.6.39 ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:DEI

Syntax	ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:DEI <enable>
Description	This command enables/disables the VLAN canonical format indicator.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = VLAN level (1-N[1])
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:VLAN:LEV1:DEI ON`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels (see section 12.32.97). Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame). [2] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition. [3] This command is as same as `ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:CFI`.

Syntax	ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:DEI?
Description	This query returns the state of the VLAN canonical format indicator.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = VLAN level (1-N)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:VLAN:LEV1:DEI?` → `1`
Note	[1] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition. [2] This command is as same as `ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:CFI?`.

12.6.40 ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:PRiority

Syntax	ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:PRiority <priority>
Description	This command sets the VLAN priority.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = VLAN level (1-N[1])
	<priority> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=7, DEFault=0*
Response	None.
Example	`ETH:PORT1:STR1:VLAN:LEV1:PR 7`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels (see section 12.32.97). Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame).

Syntax	ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:PRiority?
Description	This query returns the VLAN priority.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = VLAN level (1-N)
Response	<priority> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:VLAN:LEV1:PR?` → `7`
Note	

12.6.41 ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:ETYPe

Syntax	ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:ETYPe <type>
Description	This command sets the VLAN Ethertype.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = VLAN level (1-N)[1]
	<type> = <NUMERIC PROGRAM DATA> Acceptable values: #H8100 #H88A8 #H9100 #H9200 *DEFault=#H8100*
Response	None.
Example	`ETH:PORT1:STR1:VLAN:LEV1:ETYP #H8100`
Notes	[1] It is only possible to use this command for levels (1-M), where M is the number of active VLAN Levels minus one (see section 12.32.97). Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame). It is not possible to use this command for VLAN level N as Ethertype because: - When IPv4 or IPv6 is enabled this level is automatically set according to the selected higher-level protocol. - When IPv4 and IPv6 is disabled this level is set by :MAC:L3EType command (see section 12.6.13). MAC level Ethertype is set by :MAC:ETYPe command (see section 12.6.12).

Syntax	ETHernet:PORT<Pt>:STReam<St>:VLAN:LEVel<Lv>:ETYPe?
Description	This query returns the VLAN Ethertype.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<Lv> = VLAN level (1-N[1])
Response	<type> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:VLAN:LEV1:ETYP?` → `#H8100`
Note	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels (see section 12.32.97).

12.6.42 ETHernet:PORT<Pt>:STReam<St>:LLC[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:LLC[:ENABle] <enable>
Description	This command enables/disables LLC1 (logical link control type 1).
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:LLC ON`
Note	Disabling LLC will automatically also disable SNAP.

Syntax	ETHernet:PORT<Pt>:STReam<St>:LLC[:ENABle]?
Description	This query returns whether or not LLC1 is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:LLC?` → `1`
Note	

12.6.43 ETHernet:PORT<Pt>:STReam<St>:SNAP[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:SNAP[:ENABle] <enable>
Description	This command enables/disables SNAP.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:SNAP ON`
Note	Enabling SNAP will automatically also enable LLC.

Syntax	ETHernet:PORT<Pt>:STReam<St>:SNAP[:ENABle]?
Description	This query returns whether or not SNAP is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable > = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:SNAP?` → `1`
Note	

12.6.44 ETHernet:PORT<Pt>:STReam<St>:L2Custom[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:L2Custom[:ENABle] <enable>
Description	This command enables/disables L2Custom.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:L2C ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:L2Custom[:ENABle]?
Description	This query returns whether or not L2Custom is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<boolean> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:L2C?` → `1`
Note	

12.6.45 ETHernet:PORT<Pt>:STReam<St>:IPV4[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4[:ENABle] <enable>
Description	This command enables/disables IPv4.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT1:STR1:IPV4 ON`
Note	Enabling of this parameter will disable IPv6

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4[:ENABle]?
Description	This query returns whether or not IPv4 is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4?` → `1`
Note	

12.6.46 ETHernet:PORT<Pt>:STReam<St>:IPV4:SOURce

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:SOURce <address>
Description	This command sets the IPv4 source address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> IPv4 address
Response	None.
Example	`ETH:PORT1:STR1:IPV4:SOUR "172.29.2.36"`
Note	Only the character '.' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:SOURce?
Description	This query returns the IPv4 source address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:SOUR?` → `"172.29.2.36"`
Note	

12.6.47 ETHernet:PORT<Pt>:STReam<St>:IPV4:DESTination

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:DESTination <address>
Description	This command sets the IPv4 destination address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> IPv4 address
Response	None.
Example	`ETH:PORT1:STR1:IPV4:DEST "172.29.2.36"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:DESTination?
Description	This query returns the IPv4 destination address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:DEST?` → `"172.29.2.36"`
Note	

12.6.48 ETHernet:PORT<Pt>:STReam<St>:IPV4:GATeway[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:GATeway[:ENABle] <enable>
Description	This command enables/disables use of the IPv4 gateway.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:IPV4:GAT ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:GATeway[:ENABle]?
Description	This query returns whether or not the IPv4 gateway is used.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:GAT?` → `1`
Note	

12.6.49 ETHernet:PORT<Pt>:STReam<St>:IPV4:GATeway:ADDRess

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:GATeway:ADDRess <address>
Description	This command sets the IPv4 default gateway.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> IPv4 address
Response	None.
Example	`ETH:PORT1:STR1:IPV4:GAT:ADDR "172.29.2.36"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:GATeway:ADDRess?
Description	This query returns the IPv4 default gateway.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:GAT:ADDR?` → `"172.29.2.36"`
Note	

12.6.50 ETHernet:PORT<Pt>:STReam<St>:IPV4:NETMask

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:NETMask <mask>
Description	This command sets the IPv4 netmask.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<mask> = <STRING PROGRAM DATA> IPv4 netmask
Response	None.
Example	`ETH:PORT1:STR1:IPV4:NETM "255.255.255.0"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:NETMask?
Description	This query returns the IPv4 netmask.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<mask> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:NETM?` → `"255.255.255.0"`
Note	

12.6.51 ETHernet:PORT<Pt>:STReam<St>:IPV4:HOST

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:HOST <host>
Description	This command sets the IPv4 destination host name which is used in case DNS is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<host> = <STRING PROGRAM DATA> Host name (max. 255 characters)
Response	None.
Example	`ETH:PORT1:STR1:IPV4:HOST "www.anritsu.com"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:HOST?
Description	This query return the IPv4 destination host name.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<host> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:HOST?` → `"www.anritsu.com"`
Note	

12.6.52 ETHernet:PORT<Pt>:STReam<St>:IPV4:TTL

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:TTL <value>
Description	This command sets the IPv4 time to live.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=32*
Response	None.
Example	`ETH:PORT1:STR1:IPV4:TTL 32`
Note	This command sets the same storage parameter within the instrument as the IPv6 command `ETHernet:PORT<Pt>:STReam<St>:IPV6:HLIMit` (see section 12.6.74).

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:TTL?
Description	This query returns the IPv4 time to live.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:TTL?` → `32`
Note	

12.6.53 ETHernet:PORT<Pt>:STReam<St>:IPV4:TOS

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:TOS <value>
Description	This command sets the IPv4 DSCP/TOS byte.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=0*
Response	None.
Examples	`ETH:PORT1:STR1:IPV4:TOS #HFF` `ETH:PORT1:STR1:IPV4:TOS 128`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:TOS?
Description	This query returns the IPv4 DSCP/TOS byte.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:TOS?` → `255`
Note	

12.6.54 ETHernet:PORT<Pt>:STReam<St>:IPV4:IDENtifier

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:IDENtifier <value>
Description	This command sets the IPv4 identifier field.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=65535, DEFault=43981*
Response	None.
Examples	`ETH:PORT1:STR1:IPV4:IDEN #HABCD` `ETH:PORT1:STR1:IPV4:IDEN 12345`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:IDENtifier?
Description	This query returns the IPv4 identifier field.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:IDEN?` → `43981`
Note	

12.6.55 ETHernet:PORT<Pt>:STReam<St>:IPV4:AINCrement

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:AINCrement <enable>
Description	This command enables/disables auto increment of the IPv4 identifier field.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:IPV4:AINC OFF`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:AINCrement?
Description	This query returns whether or not auto increment of the IPv4 identifier field is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:AINC? → 0`
Note	

12.6.56 ETHernet:PORT<Pt>:STReam<St>:IPV4:MF

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:MF <boolean>
Description	This command sets/clears the IPv4 more fragments flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<boolean> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:IPV4:MF 1`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:MF?
Description	This query returns the IPv4 more fragments flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<boolean> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:MF? → 1`
Note	

12.6.57 ETHernet:PORT<Pt>:STReam<St>:IPV4:DF

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:DF <boolean>
Description	This command sets/clears the IPv4 don't fragment flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<boolean> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT1:STR1:IPV4:DF 1`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:DF?
Description	This query returns the IPv4 don't fragment flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<boolean> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:DF?` → `1`
Note	

12.6.58 ETHernet:PORT<Pt>:STReam<St>:IPV4:RES

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:RES <boolean>
Description	This command sets/clears the IPv4 reserved flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<boolean> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:IPV4:RES 0`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:RES?
Description	This query returns the IPv4 reserved flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<boolean> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:RES?` → `0`
Note	

12.6.59 ETHernet:PORT<Pt>:STReam<St>:IPV4:PROTocol

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:PROTocol <value>
Description	This command sets the IPv4 protocol byte.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=253*
Response	None.
Examples	`ETH:PORT1:STR1:IPV4:PROT #HFD` `ETH:PORT1:STR1:IPV4:PROT 17`
Notes	This command sets the same storage parameter within the instrument as the IPv6 command `ETHernet:PORT<Pt>:STReam<St>:IPV6:NHeader` (see section 12.6.73). In case UDP or TCP is enabled, the value set by this command is not used. Instead values 17 (UDP) or 6 (TCP) are automatically inserted in the IPv4 header.

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:PROTocol?
Description	This query returns the IPv4 protocol byte.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:PROT?` → `6`
Note	

12.6.60 ETHernet:PORT<Pt>:STReam<St>:IPV4:TLENgth?

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:TLENgth?
Description	This query returns the Total Length field in the IP header.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:TLEN?` → `62`
Note	

12.6.61 ETHernet:PORT<Pt>:STReam<St>:IPV4:CHECksum?

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:CHECksum?
Description	This query returns the checksum field in the IP header.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<checksum> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:CHEC?` → `#HA2E3`
Note	

12.6.62 ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP[:ENABle] <enable>
Description	This command enables/disables use of the IPv4 IGMP.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:IPV4:IGMP ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP[:ENABle]?
Description	This query returns whether or not the IPv4 IGMP is used.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:IGMP?` → `1`
Note	

12.6.63 ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:VERSion

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:VERSion <version>
Description	This command sets the IGMP version.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<version> = <CHARACTER PROGRAM DATA> V1: IGMPv1 V2: IGMPv2 V3: IGMPv3 *DEFault = V2*
Response	None.
Example	`ETH:PORT1:STR1:IPV4:IGMP:VERS V1`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:VERSion?
Description	This query returns the IGMP version.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<version> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:IGMP:VERS?` → `V2`
Note	

12.6.64 ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:GROup[:ADDRess]

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:GROup[:ADDRess] <address>
Description	This command sets the IPv4 IGMP group address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> IPv4 address
Response	None.
Example	`ETH:PORT1:STR1:IPV4:IGMP:GRO "239.1.1.5"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:GROup[:ADDRess]?
Description	This query returns the IPv4 IGMP group address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:IGMP:GRO?` → `"239.1.1.5"`
Note	

12.6.65 ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:SOURce[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:SOURce[:ENABle] <enable>
Description	This command enables/disables use of the IPv4 IGMPv3 source address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:IPV4:IGMP:SOUR ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:SOURce[:ENABle]?
Description	This query returns whether or not the IPv4 IGMPv3 source is used.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:IGMP:SOUR?` → `1`
Note	

12.6.66 ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:SOURce:ADDRess

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:SOURce:ADDRess <address>
Description	This command sets the IPv4 IGMPv3 source address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> IPv4 address
Response	None.
Example	`ETH:PORT1:STR1:IPV4:IGMP:SOUR:ADDR "192.168.1.5"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV4:IGMP:SOURce:ADDRess?
Description	This query returns the IPv4 IGMPv3 source address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV4:IGMP:SOUR:ADDR? → "192.168.1.5"`
Note	

12.6.67 ETHernet:PORT<Pt>:STReam<St>:DNS[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:DNS[:ENABle] <enable>
Description	This command enables/disables DNS.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:DNS ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:DNS[:ENABle]?
Description	This query returns whether or not DNS is used.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:DNS? → 1`
Note	

12.6.68 ETHernet:PORT<Pt>:STReam<St>:DNS:PRIMary

Syntax	ETHernet:PORT<Pt>:STReam<St>:DNS:PRIMary <address>
Description	This command sets primary DNS sever address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> IPv4 address
Response	None.
Example	`ETH:PORT1:STR1:DNS:PRIM "172.29.2.36"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:DNS:PRIMary?
Description	This query returns the primary DNS server address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:DNS:PRIM?` → `"172.29.2.36"`
Note	

12.6.69 ETHernet:PORT<Pt>:STReam<St>:DNS:SECondary

Syntax	ETHernet:PORT<Pt>:STReam<St>:DNS:SECondary <address>
Description	This command sets secondary DNS sever address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> IPv4 address
Response	None.
Example	`ETH:PORT1:STR1:DNS:SEC "172.29.2.37"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:DNS:SECondary?
Description	This query returns the secondary DNS server address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:DNS:SEC?` → `"172.29.2.37"`
Note	

12.6.70 ETHernet:PORT<Pt>:STReam<St>:IPV6[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6[:ENABle] <enable>
Description	This command enables/disables IPv6.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:IPV6 ON`
Note	Enabling of this parameter will disable IPv4

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6[:ENABle]?
Description	This query returns whether or not IPv6 is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6?` → `1`
Note	

12.6.71 ETHernet:PORT<Pt>:STReam<St>:IPV6:TCLass

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:TCLass <value>
Description	This command sets the IPv6 traffic class byte.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=0*
Response	None.
Examples	`ETH:PORT1:STR1:IPV6:TCL #HFF` `ETH:PORT1:STR1:IPV6:TCL 128`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:TCLass?
Description	This query returns the IPv6 traffic class byte.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:TCL?` → `255`
Note	

12.6.72 ETHernet:PORT<Pt>:STReam<St>:IPV6:FLABel

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:FLABel <value>
Description	This command sets the IPv6 flow label field.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1048575, DEFault=0*
Response	None.
Examples	`ETH:PORT1:STR1:IPV6:FLAB #HFFFFF` `ETH:PORT1:STR1:IPV6:FLAB 1024`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:FLABel?
Description	This query returns the IPv6 flow label field.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:FLAB?` → `1024`
Note	

12.6.73 ETHernet:PORT<Pt>:STReam<St>:IPV6:NHeader

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:NHeader <value>
Description	This command sets the IPv6 next header field.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=253*
Response	None.
Examples	`ETH:PORT1:STR1:IPV6:NH #HFF` `ETH:PORT1:STR1:IPV6:NH 128`
Notes	This command sets the same storage parameter within the instrument as the IPv4 command `ETHernet:PORT<Pt>:STReam<St>:IPV4:PROTocol` (see section 12.6.59). In case UDP or TCP is enabled, the value set by this command is not used. Instead values 17 (UDP) or 6 (TCP) are automatically inserted in the IPv6 header.

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:NHeader?
Description	This query returns the IPv6 next header field.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:NH?` → `233`
Note	

12.6.74 ETHernet:PORT<Pt>:STReam<St>:IPV6:HLIMit

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:HLIMit <value>
Description	This command sets the IPv6 hop limit field.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=32*
Response	None.
Example	`ETH:PORT1:STR1:IPV6:HLIM 32`
Note	This command sets the same storage parameter within the instrument as the IPv4 command `ETHernet:PORT<Pt>:STReam<St>:IPV4:TTL` (see section 12.6.52).

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:HLIMit?
Description	This query returns the IPv6 hop limit field.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:HLIM?` → `32`
Note	

12.6.75 ETHernet:PORT<Pt>:STReam<St>:IPV6:SOURce

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:SOURce <address>
Description	This command sets the IPv6 source address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> IPv6 address
Response	None.
Example	`ETH:PORT1:STR1:IPV6:SOUR "1234:5678:9ABC:DEF0:1234:5678:9ABC:DEF0"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:SOURce?
Description	This query returns the IPv6 source address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:SOUR?` → `"1234:5678:9ABC:DEF0:1234:5678:9ABC:DEF0"`
Note	

12.6.76 ETHernet:PORT<Pt>:STReam<St>:IPV6:DESTination

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:DESTination <address>
Description	This command sets the IPv6 destination address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> IPv6 address
Response	None.
Example	`ETH:PORT1:STR1:IPV6:DEST "1234:5678:9ABC:DEF0:1234:5678:9ABC:DEF0"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:DESTination?
Description	This query returns the IPv6 destination address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:DEST?` → `"1234:5678:9ABC:DEF0:1234:5678:9ABC:DEF0"`
Note	

12.6.77 ETHernet:PORT<Pt>:STReam<St>:IPV6:GATeway[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:GATeway[:ENABle] <enable>
Description	This command enables/disables use of the IPv6 gateway.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:IPV6:GAT ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:GATeway[:ENABle]?
Description	This query returns whether or not the IPv6 gateway is used.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:GAT?` → `1`
Note	

12.6.78 ETHernet:PORT<Pt>:STReam<St>:IPV6:GATeway:ADDRess

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:GATeway:ADDRess <address>
Description	This command sets the IPv6 default gateway.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> IPv6 address
Response	None.
Example	`ETH:PORT1:STR1:IPV6:GAT:ADDR "1234:5678:9ABC:DEF0:1234:5678:9ABC:DEF0"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:GATeway:ADDRess?
Description	This query returns the IPv6 default gateway.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:GAT:ADDR?` → `"1234:5678:9ABC:DEF0:1234:5678:9ABC:DEF0"`
Note	

12.6.79 ETHernet:PORT<Pt>:STReam<St>:IPV6:PREFix

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:PREFix <mask>
Description	This command sets the IPv6 prefix mask.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<mask> = <STRING PROGRAM DATA> IPv6 prefix mask
Response	None.
Example	`ETH:PORT1:STR1:IPV6:PREF "FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:0000"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:PREFix?
Description	This query returns the IPv6 prefix mask.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<mask> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:PREF?` → `"FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:0000"`
Note	

12.6.80 ETHernet:PORT<Pt>:STReam<St>:IPV6:ADDRconfig

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:ADDRconfig <mode>
Description	This command sets the address configuration mode for IPv6.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<mode> = <CHARACTER PROGRAM DATA> MANual: Manual SLESs: Stateless *DEFault = SLESs*
Response	None.
Example	`ETH:PORT1:STR1:IPV6:ADDR SLES`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:ADDRconfig?
Description	This query returns the address configuration mode for IPv6.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:ADDR?` → `SLES`
Note	

12.6.81 ETHernet:PORT<Pt>:STReam<St>:IPV6:ADDRconfig:RESult?

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:ADDRconfig:RESult?
Description	This query returns the result of address config for IPv6.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<result> = <CHARACTER RESPONSE DATA> SUCCESS: Address configuration succeeded. TIMEOUT: Address configuration timed out.
Example	`ETH:PORT1:STR1:IPV6:ADDR:RES?` → `TIMEOUT`
Note	

12.6.82 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:IID

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:IID <id>
Description	This command sets the interface ID for IPv6.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<id> = <STRING PROGRAM DATA> The interface ID string must consist of 16 hexadecimal digits.
Response	None.
Example	`ETH:PORT1:STR1:IPV6:SLES:IID "00-00-00-00-00-00-00-00"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:IID?
Description	This query returns the address configuration mode for IPv6.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<mode> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:SLES:IID?` → `"00-00-00-00-00-00-00-00"`
Note	

12.6.83 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:IID:AUTO

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:IID:AUTO <enable>
Description	This command enables/disables Interface ID.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:IPV6:SLES:IID:AUTO ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:IID:AUTO?
Description	This query returns whether or not Interface ID is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:SLES:IID:AUTO?` → `1`
Note	

12.6.84 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:LINKlocal?

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:LINKlocal?
Description	This query returns the link local address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:SLES:LINK?` → `"1234:5678:9ABC:DEF0:1234:5678:9ABC:DEF0"`
Note	

12.6.85 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:RAFLag?

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:RAFLag?
Description	This query returns the stateless RA flags.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<flags> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:SLES:RAFL?` → `"0x00"`
Note	

12.6.86 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:SRCMac?

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:SRCMac?
Description	This query returns the stateless source MAC address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:SLES:SRCM?` → `"00-00-00-00-00-00"`
Note	

12.6.87 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:PREFix?

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:PREFix?
Description	This query returns the stateless prefix.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<prefix> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:SLES:PREF?` → `"1234:5678:9ABC:DEF0:1234:5678:9ABC:DEF0"`
Note	

12.6.88 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:PRFLag?

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:PRFLag?
Description	This query returns the stateless prefix flags.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<flags> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:SLES:PRFL?` → `"0x00"`
Note	

12.6.89 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:LTIMe?

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:LTIMe?
Description	This query returns the stateless life time.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<time> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:SLES:LTIM?` → `"Fri Jan 8 14:24:44 2010"`
Note	

12.6.90 ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:RENew

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:SLESs:RENew
Description	This command execute Stateless address configration. This command valid only when IpV6 is selected on layer3.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	None.
Example	`ETH:PORT1:STR1:IPV6:SLES:REN`
Note	

12.6.91 ETHernet:PORT<Pt>:STReam<St>:IPV6:PLENgth?

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:PLENgth?
Description	This query returns the Payload Length field in the IP header.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:PLEN?` → `62`
Note	

12.6.92 ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD[:ENABle] <enable>
Description	This command enables/disables use of the IPv6 MLD.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:IPV6:MLD ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD[:ENABle]?
Description	This query returns whether or not the IPv6 MLD is used.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:MLD?` → `1`
Note	

12.6.93 ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:VERSion

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:VERSion <version>
Description	This command sets the MLD version.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<version> = <CHARACTER PROGRAM DATA> V1: MLDv1 V2: MLDv2 *DEFault = V1*
Response	None.
Example	`ETH:PORT1:STR1:IPV6:MLD:VERS V1`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:VERSion?
Description	This query returns the MLD version.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<version> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:MLD:VERS?` → `V1`
Note	

12.6.94 ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:GROup[:ADDRess]

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:GROup[:ADDRess] <address>
Description	This command sets the IPv6 MLD group address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> IPv6 address
Response	None.
Example	`ETH:PORT1:STR1:IPV6:MLD:GRO "FF05:0000:0000:0000:0000:0000:0000:0333"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:GROup[:ADDRess]?
Description	This query returns the IPv6 MLD group address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:MLD:GRO?` → `"FF05:0000:0000:0000:0000:0000:0000:0333"`
Note	

12.6.95 ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:SOURce[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:SOURce[:ENABle] <enable>
Description	This command enables/disables use of the IPv6 MLDv2 source address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:IPV6:MLD:SOUR ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:SOURce[:ENABle]?
Description	This query returns whether or not the IPv6 MLDv2 source is used.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:MLD:SOUR?` → `1`
Note	

12.6.96 ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:SOURce:ADDRess

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:SOURce:ADDRess <address>
Description	This command sets the IPv6 MLDv2 source address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<address> = <STRING PROGRAM DATA> IPv6 address
Response	None.
Example	`ETH:PORT1:STR1:IPV6:MLD:SOUR:ADDR "2001:0001:0001:0001:0000:0000:0000:0005"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:IPV6:MLD:SOURce:ADDRess?
Description	This query returns the IPv6 MLDv2 source address.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:IPV6:MLD:SOUR:ADDR?` → `"2001:0001:0001:0001:0000:0000:0000:0005"`
Note	

12.6.97 ETHernet:PORT<Pt>:STReam<St>:L3Custom[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:L3Custom[:ENABle] <enable>
Description	This command enables/disables Layer3 CUSTOM.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:L3C ON`
Note	Enabling of this parameter will disable IPv4 and IPv6

Syntax	ETHernet:PORT<Pt>:STReam<St>:L3Custom[:ENABle]?
Description	This query returns whether or not Layer3 CUSTOM is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:L3C?` → `1`
Note	

12.6.98 ETHernet:PORT<Pt>:STReam<St>:DHCP[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:DHCP[:ENABle] <enable>
Description	This command enables/disables DHCP.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:DHCP ON`
Note	When DHCP is enabled a new DHCP discover message is transmitted. Use the query `ETH:PORT1:STR1:DHCP:LET?` (see section 12.6.102) to check if lease is obtained.

Syntax	ETHernet:PORT<Pt>:STReam<St>:DHCP[:ENABle]?
Description	This query returns whether or not DHCP is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:DHCP?` → `1`
Note	

12.6.99 ETHernet:PORT<Pt>:STReam<St>:DHCP:RENew

Syntax	ETHernet:PORT<Pt>:STReam<St>:DHCP:RENew <enable>
Description	This command enables/disables renewal of DHCP lease when link is reestablished.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:DHCP:REN ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:DHCP:RENew?
Description	This query returns whether or not DHCP lease is renewed when link reestablished.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:DHCP:REN?` → `1`
Note	

12.6.100 ETHernet:PORT<Pt>:STReam<St>:DHCP:DNS

Syntax	ETHernet:PORT<Pt>:STReam<St>:DHCP:DNS <enable>
Description	This command enables/disables whether DNS server info should be obtained through DHCP.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:DHCP:DNS ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:DHCP:DNS?
Description	This query returns whether or not DNS server info is obtained through DHCP.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:DHCP:DNS?` → `1`
Note	

12.6.101 ETHernet:PORT<Pt>:STReam<St>:DHCP:GATeway

Syntax	ETHernet:PORT<Pt>:STReam<St>:DHCP:GATeway <enable>
Description	This command enables/disables whether gateway setup should be obtained through DHCP.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:DHCP:GAT ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:DHCP:GATeway?
Description	This query returns whether or not gateway setup is obtained through DHCP.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:DHCP:GAT?` → `1`
Note	

12.6.102 ETHernet:PORT<Pt>:STReam<St>:DHCP:LETime?

Syntax	ETHernet:PORT<Pt>:STReam<St>:DHCP:LETime?
Description	This query returns the current DHCP lease expire time.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<time> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:DHCP:LET?` → `"Fri Jan 8 14:24:44 2010"`
Note	"N/A" is returned if DHCP is disabled or a lease has not been obtained .

12.6.103 ETHernet:PORT<Pt>:STReam<St>:UDP[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:UDP[:ENABle] <enable>
Description	This command enables/disables UDP.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:UDP ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:UDP[:ENABle]?
Description	This query returns whether or not UDP is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:UDP?` → `1`
Note	

12.6.104 ETHernet:PORT<Pt>:STReam<St>:UDP:SPORt

Syntax	ETHernet:PORT<Pt>:STReam<St>:UDP:SPORt <value>
Description	This command sets the UDP source port.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 65535, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:UDP:SPOR 22`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:UDP:SPORt?
Description	This query returns the UDP source port
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:UDP:SPOR?` → `22`
Note	

12.6.105 ETHernet:PORT<Pt>:STReam<St>:UDP:DPORt

Syntax	ETHernet:PORT<Pt>:STReam<St>:UDP:DPORt <value>
Description	This command sets the UDP destination port.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 65535, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:UDP:DPOR 22`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:UDP:DPORt?
Description	This query returns the UDP destination port.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:UDP:DPOR?` → `22`
Note	

12.6.106 ETHernet:PORT<Pt>:STReam<St>:UDP:LENGth?

Syntax	ETHernet:PORT<Pt>:STReam<St>:UDP:LENGth?
Description	This query returns the Length field in the UDP header.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:UDP:LENG?` → `62`
Note	

12.6.107 ETHernet:PORT<Pt>:STReam<St>:UDP:CNULl

Syntax	ETHernet:PORT<Pt>:STReam<St>:UDP:CNULl <enable>
Description	This command enables/disables forcing of the UDP header checksum to null.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:UDP:CNUL ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:UDP:CNULl?
Description	This query returns whether or not forcing of UDP checksum to null is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:UDP:CNUL?` → `1`
Note	

12.6.108 ETHernet:PORT<Pt>:STReam<St>:UDP:CHECksum?

Syntax	ETHernet:PORT<Pt>:STReam<St>:UDP:CHECksum?
Description	This query returns the checksum field in the UDP header.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<checksum> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:UDP:CHEC?` → `#HA2E3`
Note	

12.6.109 ETHernet:PORT<Pt>:STReam<St>:TCP[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP[:ENABle] <enable>
Description	This command enables/disables TCP.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:TCP ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP[:ENABle]?
Description	This query returns whether or not TCP is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP?` → `1`
Note	

12.6.110 ETHernet:PORT<Pt>:STReam<St>:TCP:ACONnect

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:ACONnect <enable>
Description	This command enables/disables automatic TCP connect.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:TCP:ACON ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:ACONnect?
Description	This query returns whether or not automatic TCP connect is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:ACON?` → `1`
Note	

12.6.111 ETHernet:PORT<Pt>:STReam<St>:TCP:LMODe

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:LMODe <enable>
Description	This command enables/disables TCP listen mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:TCP:LMOD ON`
Note	Requires ACONnect to be enabled.

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:LMODe?
Description	This query returns whether or not TCP listen mode is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:LMOD?` → `1`
Note	

12.6.112 ETHernet:PORT<Pt>:STReam<St>:TCP:SPORt

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:SPORt <value>
Description	This command sets the TCP source port.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 65535, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:TCP:SPOR 22`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:SPORt?
Description	This query returns the TCP source port
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:SPOR?` → `22`
Note	

12.6.113 ETHernet:PORT<Pt>:STReam<St>:TCP:DPORt

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:DPORt <value>
Description	This command sets the TCP destination port.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 65535, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:TCP:DPOR 22`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:DPORt?
Description	This query returns the TCP destination port.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:DPOR?` → `22`
Note	

12.6.114 ETHernet:PORT<Pt>:STReam<St>:TCP:SEQuence

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:SEQuence <value>
Description	This command sets the TCP sequence number.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=4294967295, DEFault=0*
Response	None.
Example	`ETH:PORT1:STR1:TCP:SEQ 123456`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:SEQuence?
Description	This query returns the TCP sequence number.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:SEQ?` → `123456`
Note	

12.6.115 ETHernet:PORT<Pt>:STReam<St>:TCP:AINCrement

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:AINCrement <enable>
Description	This command enables/disables auto increment of the TCP sequence number.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:TCP:AINC ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:AINCrement?
Description	This query returns whether or not auto increment of the TCP sequence number is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:AINC?` → `1`
Note	

12.6.116 ETHernet:PORT<Pt>:STReam<St>:TCP:ACKNowledge

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:ACKNowledge <value>
Description	This command sets the TCP acknowledgment number.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 4294967295, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:TCP:ACKN 123456`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:ACKNowledge?
Description	This query returns the TCP acknowledgment number.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:ACKN?` → `123456`
Note	

12.6.117 ETHernet:PORT<Pt>:STReam<St>:TCP:REServed

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:REServed <value>
Description	This command sets the TCP reserved value.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 63, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:TCP:RES 10`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:REServed?
Description	This query returns the TCP reserved value.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:RES?` → `10`
Note	

12.6.118 ETHernet:PORT<Pt>:STReam<St>:TCP:FCWR

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FCWR <enable>
Description	This command enables/disables the TCP CWR flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:TCP:FCWR ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FCWR?
Description	This query returns the TCP CWR flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:FCWR?` → `1`
Note	

12.6.119 ETHernet:PORT<Pt>:STReam<St>:TCP:FECE

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FECE <enable>
Description	This command enables/disables the TCP ECE flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:TCP:FECE ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FECE?
Description	This query returns the TCP ECE flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:FECE?` → `1`
Note	

12.6.120 ETHernet:PORT<Pt>:STReam<St>:TCP:FURG

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FURG <enable>
Description	This command enables/disables the TCP URG flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:TCP:FURG ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FURG?
Description	This query returns the TCP URG flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:FURG?` → `1`
Note	

12.6.121 ETHernet:PORT<Pt>:STReam<St>:TCP:FACK

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FACK <enable>
Description	This command enables/disables the TCP ACK flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:TCP:FACK ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FACK?
Description	This query returns the TCP ACK flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:FACK?` → `1`
Note	

12.6.122 ETHernet:PORT<Pt>:STReam<St>:TCP:FPSH

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FPSH <enable>
Description	This command enables/disables the TCP PSH flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:TCP:FPSH ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FPSH?
Description	This query returns the TCP PSH flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:FPSH?` → `1`
Note	

12.6.123 ETHernet:PORT<Pt>:STReam<St>:TCP:FRST

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FRST <enable>
Description	This command enables/disables the TCP RST flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:TCP:FRST ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FRST?
Description	This query returns the TCP RST flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:FRST?` → `1`
Note	

12.6.124 ETHernet:PORT<Pt>:STReam<St>:TCP:FSYN

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FSYN <enable>
Description	This command enables/disables the TCP SYN flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:TCP:FSYN ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FSYN?
Description	This query returns the TCP SYN flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:FSYN?` → `1`
Note	

12.6.125 ETHernet:PORT<Pt>:STReam<St>:TCP:FFIN

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FFIN <enable>
Description	This command enables/disables the TCP FIN flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:TCP:FFIN ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:FFIN?
Description	This query returns the TCP FIN flag.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:FFIN? → 1`
Note	

12.6.126 ETHernet:PORT<Pt>:STReam<St>:TCP:WINDow

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:WINDow <value>
Description	This command sets the TCP window size.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 65535, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:TCP:WIND 1000`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:WINDow?
Description	This query returns the TCP window size.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:WIND? → 1000`
Note	

12.6.127 ETHernet:PORT<Pt>:STReam<St>:TCP:UPOinter

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:UPOinter <value>
Description	This command sets the TCP urgent pointer.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 65535, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:TCP:UPO 1000`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:UPOinter?
Description	This query returns the TCP urgent pointer.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:UPO?` → `1000`
Note	

12.6.128 ETHernet:PORT<Pt>:STReam<St>:TCP:CHECksum?

Syntax	ETHernet:PORT<Pt>:STReam<St>:TCP:CHECksum?
Description	This query returns the checksum field in the TCP header.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<length> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:TCP:CHEC?` → `#HA2E3`
Note	

12.6.129 ETHernet:PORT<Pt>:STReam1:BER:UTGenerator

Syntax	ETHernet:PORT<Pt>:STReam1:BER:UTGenerator <enable>
Description	This command enables/disables automatic start of the transmitter when running a BER measurement.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:BER:UTG ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam1:BER:UTGenerator?
Description	This query return if the transmitter automatically starts when running a BER measurement.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:BER:UTG?` → `1`
Note	

12.6.130 ETHernet:PORT<Pt>:STReam1:BER:SEQuence

Syntax	ETHernet:PORT<Pt>:STReam1:BER:SEQuence <enable>
Description	This command enables/disables sequence checking when using unframed Ethernet.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT1:STR1:BER:SEQ ON`
Note	This is only valid when `FRAMed` is `OFF`.

Syntax	ETHernet:PORT<Pt>:STReam1:BER:SEQuence?
Description	This query returns if sequence checking is enabled/disabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:BER:SEQ?` → `1`
Note	

12.6.131 ETHernet:PORT<Pt>:STReam1:BER:SDMeasure[:ENABled]

Syntax	ETHernet:PORT<Pt>:STReam1:BER:SDMeasure[:ENABled] <enable>
Description	This command enables/disables Service Disruption Measurement.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:BER:SDM ON`
Note	This is only valid when `FRAMed` is `ON`.

Syntax	ETHernet:PORT<Pt>:STReam1:BER:SDMeasure[:ENABled]?
Description	This query returns if Service Disruption Measurement is enabled/disabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:BER:SDM? → 1`
Note	

12.6.132 ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:DTYPe

Syntax	ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:DTYPe <type>
Description	This command sets the Disruption Type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> PACK: Packet LOS: Loss of Signal
Response	None.
Example	`ETH:PORT1:STR1:BER:SDM:DTYP PACK`
Note	

Syntax	ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:DTYPe?
Description	This query returns the Disruption Type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STR1:BER:SDM:DTYP? → PACK`
Note	

12.6.133 ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:MLIMit

Syntax	ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:MLIMit <value>
Description	This command sets the Max Limit. Unit: Microseconds.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 5000000, DEFault = 50000*
Response	None.
Example	`ETH:PORT1:STR1:BER:SDM:MLIM 10`
Note	

Syntax	ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:MLIMit?
Description	This query returns the Max Limit. Unit: Microseconds.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:BER:SDM:MLIM? → 10`
Note	

12.6.134 ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:MDISruption

Syntax	ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:MDISruption <value>
Description	This command sets the Min Disruption Frames.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 5000, DEFault = 10*
Response	None.
Example	`ETH:PORT1:STR1:BER:SDM:MDIS 5`
Note	

Syntax	ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:MDISruption?
Description	This query return the Min Disruption Frames.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:BER:SDM:MDIS?` → `5`
Note	

12.6.135 ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:EFPeriod

Syntax	ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:EFPeriod <value>
Description	This command sets the LOS Error Free Period. Unit: Milliseconds
Parameters	<Pt> = Port number
	<value> = <NR1 NUMERIC RESPONSE DATA> *MINimum = 0.1, MAXimum = 1000.0, DEFault = 10.0*
Response	None.
Example	`ETH:PORT1:STR1:BER:SDM:EFP 5.0`
Note	

Syntax	ETHernet:PORT<Pt>:STReam1:BER:SDMeasure:EFPeriod?
Description	This query return the LOS Error Free Period. Unit: Milliseconds
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:BER:SDM:EFP?` → `5.0`
Note	

12.6.136 ETHernet:PORT<Pt>:STReam1:BER:STHResholds[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam1:BER:STHResholds[:ENABle] <enable>
Description	This command enables/disables BER Sequence error thresholds.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Examples	`ETH:PORT1:STR1:BER:STHR ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam1:BER:STHResholds[:ENABle]?
Description	This query returns whether or not BER Sequence error thresholds are enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Examples	`ETH:PORT1:STR1:BER:STHR?` → `1`
Note	

12.6.137 ETHernet:PORT<Pt>:STReam1:BER:STHResholds:VALue

Syntax	ETHernet:PORT<Pt>:STReam1:BER:STHResholds:VALue <value>
Description	This command sets the Sequence errors threshold value.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 4294967295, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:BER:STHR:VAL 100`
Note	

Syntax	ETHernet:PORT<Pt>:STReam1:BER:STHResholds:VALue?
Description	This query returns the Sequence errors threshold value.
Parameters	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:BER:STHR:VAL? → 10`
Note	

12.6.138 ETHernet:PORT<Pt>:STReam<St>:PAYLoad

Syntax	ETHernet:PORT<Pt>:STReam<St>:PAYLoad <pattern>
Description	This command sets stream payload pattern
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<pattern> = <CHARACTER PROGRAM DATA> FOX: 'Fox' pattern 5555: 'All-5s' pattern PRBS9: PRBS9 sequence PRBS11: PRBS11 sequence PRBS15: PRBS15 sequence PRBS20: PRBS20 sequence PRBS23: PRBS23 sequence PRBS29: PRBS29 sequence PRBS31: PRBS31 sequence HFTest: HF test pattern CRPAT: Compliant random Pattern JTPAT: Jitter tolerance pattern SPAT: Supply noise test sequence USER32BIT: 32 bit user defined pattern. USER16BIT: Obsolete. For CMA 3000 backward compatibility only. Same as USER32BIT. *DEFault = PRBS23*
Response	None.
Example	`ETH:PORT1:STR1:PAYL FOX`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:PAYLoad?
Description	This query returns the stream payload pattern
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<pattern> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STR1:PAYL? → SPAT`
Note	

12.6.139 ETHernet:PORT<Pt>:STReam<St>:UP16

Syntax	ETHernet:PORT<Pt>:STReam<St>:UP16 <pattern>
Description	This command sets the 16 bit user pattern.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<pattern> = <STRING PROGRAM DATA> Valid characters: '0' and '1' The string must consist of 1 to 16 characters.
Response	None.
Example	`ETH:PORT1:STR1:UP16 "101101"`
Note	This command is for backward compatibility only, and a query command is not available. Actually the new 32 bit user pattern is set by this command.

12.6.140 ETHernet:PORT<Pt>:STReam<St>:UP32

Syntax	ETHernet:PORT<Pt>:STReam<St>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PAYLoad` is `USER32BIT`.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters (one bit resolution).
Response	None.
Example	`ETH:PORT1:STR1:UP32 "01101"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<pattern> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:UP32?` → `"01101"`
Note	

12.6.141 ETHernet:PORT<Pt>:STReam:PCMA

Syntax	ETHernet:PORT<Pt>:STReam:PCMA <enable>
Description	This command set PRBS pattern for CMA 3000 compatibility.
Parameter	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR:PCMA ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam:PCMA?
Description	This query returns whether PRBS pattern for CMA 3000 compatibility.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR:PCMA?` → `1`
Note	

12.6.142 ETHernet:PORT<Pt>:STReam:CROSsprbs

Syntax	ETHernet:PORT<Pt>:STReam:CROSsprbs <enable>
Description	This command set Enable or Disable cross pattern by Frame by Frame.
Parameter	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR:CROS ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam:CROSsprbs?
Description	This query returns Enable or Disable cross pattern by Frame by Frame.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR:CROS?` → `1`
Note	

12.6.143 ETHernet:PORT<Pt>:STReam<St>:LATency

Syntax	ETHernet:PORT<Pt>:STReam<St>:LATency <enable>
Description	This command enables/disables stream latency measurement.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:LAT ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:LATency?
Description	This query returns whether or not stream latency measurement is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:LAT?` → `1`
Note	

12.6.144 ETHernet:PORT<Pt>:STReam<St>:JITTer

Syntax	ETHernet:PORT<Pt>:STReam<St>:JITTer <enable>
Description	This command enables/disables stream jitter measurement.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:JITT ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:JITTer?
Description	This query returns whether or not stream jitter measurement is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:JITT?` → `1`
Note	

12.6.145 ETHernet:PORT<Pt>:STReam<St>:FLOSs

Syntax	ETHernet:PORT<Pt>:STReam<St>:FLOSs <enable>
Description	This command enables/disables multistream frame loss measurement.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:STR1:FLOS ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:FLOSs?
Description	This query returns whether or not multistream frame loss measurement is enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR1:FLOS?` → `1`
Note	

12.6.146 ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds[:ENABle] <enable>
Description	This command enables/disables Frameloss thresholds.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Examples	`ETH:PORT1:STR1:FLOS:THR ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds[:ENABle]?
Description	This query returns whether or not Frameloss thresholds are enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Examples	`ETH:PORT1:STR1:FLOS:THR?` → `1`
Note	

12.6.147 ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds:MODE

Syntax	ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds:MODE <mode>
Description	This command sets the Frameloss thresholds mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<mode> = <NUMERIC PROGRAM DATA> COUNT: Count RATE: Rate *DEFault=COUNT*
Response	None.
Example	`ETH:PORT1:STR1:FLOS:THR:MODE COUNT`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds:MODE? <mode>
Description	This query returns the Frameloss thresholds mode.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:FLOS:THR:MODE?` → `COUNT`
Note	

12.6.148 ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds:COUNt

Syntax	ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds:COUNt <value>
Description	This command sets the Frameloss thresholds count value.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 4294967295, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:FLOS:THR:COUN 100`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds:COUNt?
Description	This query returns the Frameloss thresholds count value.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:FLOS:THR:COUN?` → `100`
Note	

12.6.149 ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds:RATio

Syntax	ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds:RATio <ratio>
Description	This command sets the Frameloss thresholds ratio% value.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<ratio> = <NUMERIC PROGRAM DATA> *MINimum = 0.00000, MAXimum = 100.00000, DEFault = 0.00000*
Response	None.
Example	`ETH:PORT1:STR1:FLOS:THR:RAT 10.00000`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:FLOSs:THResholds:RATio?
Description	This query returns the Frameloss thresholds ratio% value.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:FLOS:THR:RAT?` → `10.00000`
Note	

12.6.150 ETHernet:PORT<Pt>:STReam<St>:LATency:THResholds[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:LATency:THResholds[:ENABle] <enable>
Description	This command enables/disables Latency thresholds.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Examples	`ETH:PORT1:STR1:LAT:THR ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:LATency:THResholds[:ENABle]?
Description	This query returns whether or not Latency thresholds are enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Examples	`ETH:PORT1:STR1:LAT:THR? → 1`
Note	

12.6.151 ETHernet:PORT<Pt>:STReam<St>:LATency:THResholds:VALue

Syntax	ETHernet:PORT<Pt>:STReam<St>:LATency:THResholds:VALue <value>
Description	This command sets the Latency thresholds value.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 429496729.500, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:LAT:THR:VAL 100.000`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:LATency:THResholds:VALue?
Description	This query returns the level for the Latency thresholds value.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:LAT:THR:VAL? → 10.000`
Note	

12.6.152 ETHernet:PORT<Pt>:STReam<St>:JITTer:THResholds[:ENABle]

Syntax	ETHernet:PORT<Pt>:STReam<St>:JITTer:THResholds[:ENABle] <enable>
Description	This command enables/disables Jitter thresholds.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Examples	`ETH:PORT1:STR1:JITT:THR ON`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:JITTer:THResholds[:ENABle]?
Description	This query returns whether or not Jitter thresholds are enabled.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Examples	`ETH:PORT1:STR1:JITT:THR?` → `1`
Note	

12.6.153 ETHernet:PORT<Pt>:STReam<St>:JITTer:THResholds:VALue

Syntax	ETHernet:PORT<Pt>:STReam<St>:JITTer:THResholds:VALue <value>
Description	This command sets the Jitter thresholds value.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 429496729.500, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:JITT:THR:VAL 100.00`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:JITTer:THResholds:VALue?
Description	This query returns the Jitter thresholds value.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:JITT:THR:VAL?` → `10.000`
Note	

12.6.154 ETHernet:PORT<Pt>:STReam:TSTamp:CCTimestamp

Syntax	ETHernet:PORT<Pt>:STReam:TSTamp:CCTimestamp <version>
Description	This command sets the Latency/Jitter time stamp compatible configuration.
Parameters	<Pt> = Port number
	<version> = <CHARACTER PROGRAM DATA> V0906: From V9.06 or later [1] V0100: V9.05 or older *DEFault = V0906*
Response	None.
Example	`ETH:PORT1:STR:TST:CCT V0906`
Note	[1] Only available on MU100010A or MU100011A.

Syntax	ETHernet:PORT<Pt>:STReam:TSTamp:CCTimestamp?
Description	This query returns the Latency/Jitter time stamp compatible configuration.
Parameters	<Pt> = Port number
Response	<version> = <CHARACTER PROGRAM DATA>
Example	`ETH:PORT1:STR:TST:CCT?` → `V0906`
Note	

12.6.155 ETHernet:PORT<Pt>:STReam:TSTamp:TSRC

Syntax	ETHernet:PORT<Pt>:STReam:TSTamp:TSRC <source>
Description	This command sets the time stamp source.
Parameters	<Pt> = Port number
	<source> = <CHARACTER PROGRAM DATA> INTernal: Internal GPS: GPS R1PPS: 1PPS(Ext. ref. 10MHz) [1] RGPS: GPS(Ext. ref. 10MHz) [1] *DEFault = INTernal*
Response	None.
Example	`ETH:PORT1:STR:TST:TSRC GPS`
Note	[1] Only available on MU100011A.

Syntax	ETHernet:PORT<Pt>:STReam:TSTamp:TSRC?
Description	This query returns the time stamp source.
Parameters	<Pt> = Port number
Response	<version> = <CHARACTER PROGRAM DATA>
Example	`ETH:PORT1:STR:TST:TSRC?` → `GPS`
Note	

12.6.156 ETHernet:PORT<Pt>:STReam:TSTamp:GCABle

Syntax	ETHernet:PORT<Pt>:STReam:TSTamp:GCABle <delay>
Description	This command sets the delay time of GPS antenna cable length. Unit: nano Seconds.
Parameters	<Pt> = Port number
	<delay> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1000000000, DEFault=0*
Response	None.
Example	`ETH:PORT1:STR:TST:GCAB 35`
Note	

Syntax	ETHernet:PORT<Pt>:STReam:TSTamp:GCABle?
Description	This query returns the delay time of GPS antenna cable length.
Parameter	<Pt> = Port number
Response	<delay> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR:TST:GCAB?` → `35`
Note	

12.6.157 ETHernet:PORT<Pt>:STReam:TSTamp:SYNC?

Syntax	ETHernet:PORT<Pt>:STReam:TSTamp:SYNC?
Description	This query returns if time stamp source is synchronize with target source.
Parameter	<Pt> = Port number
Response	<sync> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:STR:TST:SYNC?` → `1`
Note	

12.6.158 ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:FIELd

Syntax	ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:FIELd <protocol>
Description	This command sets the variable field protocol.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<No> = Variable field number (1-2)
	<protocol> = <CHARACTER PROGRAM DATA> OFF: OFF SMAC: Source MAC Address DMAC: Destination MAC Address VLANx: VLAN ID where x is the VLAN level (1-8) BTAG: B-Tag VID ITAG: I-Tag SID MPLSx: MPLS where x is the MPLS level (1-8) MIMDMAC: MiM Destination MAC Address EMDM: EoMPLS destination MAC Address MIMSMAC: MiM Source MAC Address EMSM: EoMPLS source MAC Address SIV4: Source IPv4 DIV4: Destination IPv4 SIV6: Source IPv6 DIV6: Destination IPv6 CUST: Custom Header PCID: PC_ID (2Byte) SEQID: SEQ_ID (2Byte) PCID4B: PC_ID (4Byte) SEQID4B: SEQ_ID (4Byte) RTCID: RTC_ID RSTID: RST_ID FLOWID: FLOW_ID ODRINFO: Ordering Information MSGHEAD: Message header SUBHEAD: Sub Header *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STR1:VAR1:FIEL SMAC`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:FIELd?
Description	This query returns the variable field protocol.
Parameter	<Pt> = Port number
	<St> = Stream number (1-16)
	<No> = Variable field number (1-2)
Response	<protocol> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STR1:VAR1:FIEL?` → `SMAC`
Note	

12.6.159 ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:TYPE

Syntax	ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:TYPE <type>
Description	This command sets the variable field type.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<No> = Variable field number (1-2)
	<type> = <CHARACTER PROGRAM DATA> INCREMENT: Increment DECREMENT: Decrement RANDOM: Random *DEFault = INCREMENT*
Response	None.
Example	`ETH:PORT1:STR1:VAR1:TYPE INCREMENT`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:TYPE?
Description	This query returns the variable field type.
Parameter	<Pt> = Port number
	<St> = Stream number (1-16)
	<No> = Variable field number (1-2)
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STR1:VAR1:TYPE?` → `INCREMENT`
Note	

12.6.160 ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:OFFSet

Syntax	ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:OFFSet <offset>
Description	This command sets the variable field offset.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<No> = Variable field number (1-2)
	<offset> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 127, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:VAR1:OFFS 1`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:OFFSet?
Description	This query returns the variable field offset.
Parameter	<Pt> = Port number
	<St> = Stream number (1-16)
	<No> = Variable field number (1-2)
Response	<offset> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:VAR1:OFFS?` → `1`
Note	

12.6.161 ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:LENGth

Syntax	ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:LENGth <length>
Description	This command sets the variable field length.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<No> = Variable field number (1-2)
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 32, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:VAR1:LENG 1`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:LENGth?
Description	This query returns the variable field length.
Parameter	<Pt> = Port number
	<St> = Stream number (1-16)
	<No> = Variable field Number (1-2)
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:VAR1:LENG?` → `1`
Note	

12.6.162 ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:STARt

Syntax	ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:STARt <start>
Description	This command sets the variable field start value.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<No> = Variable field number (1-2)
	<start> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 4294967295, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:VAR1:STAR 1`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:STARt?
Description	This query returns the variable field start value.
Parameter	<Pt> = Port number
	<St> = Stream number (1-16)
	<No> = Variable field number (1-2)
Response	<start> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:VAR1:STAR?` → `1`
Note	

12.6.163 ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:END

Syntax	ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:END <end>
Description	This command sets the variable field end value.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<No> = Variable field number (1-2)
	<end> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 4294967295, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:VAR1:END 10`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:END?
Description	This query returns the variable field end value.
Parameter	<Pt> = Port number
	<St> = Stream number (1-16)
	<No> = Variable field number (1-2)
Response	<end> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:VAR1:END?` → `10`
Note	

12.6.164 ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:STEP

Syntax	ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:STEP <step>
Description	This command sets the variable field step value.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<No> = Variable field number (1-2)
	<step> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 2147483648, DEFault = 1*
Response	None.
Example	`ETH:PORT1:STR1:VAR1:STEP 1`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:VARiable<No>:STEP?
Description	This query returns the variable field step value.
Parameter	<Pt> = Port number
	<St> = Stream number (1-16)
	<No> = Variable field number (1-2)
Response	<step> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:VAR1:STEP?` → `1`
Note	

12.6.165 ETHernet:PORT<Pt>:STReam<St>:CUSTom:LENGth

Syntax	ETHernet:PORT<Pt>:STReam<St>:CUSTom:LENGth <length>
Description	This command sets the custom header length.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 2, MAXimum = 256*
Response	None.
Example	`ETH:PORT1:STR1:CUST:LENG 2`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:CUSTom:LENGth?
Description	This query returns the custom header length.
Parameter	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:CUST:LENG?` → `2`
Note	

12.6.166 ETHernet:PORT<Pt>:STReam<St>:CUSTom:DATa

Syntax	ETHernet:PORT<Pt>:STReam<St>:CUSTom:DATa <data>
Description	This command sets the custom header data.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:CUST:DAT "33FF0011"`
Note	

Syntax	ETHernet:PORT<Pt>:STReam<St>:CUSTom:DATa?
Description	This query returns the custom header data.
Parameter	<Pt> = Port number
	<St> = Stream number (1-16)
Response	<data> = <STRING PROGRAM DATA>
Example	`ETH:PORT1:STR1:CUST:DAT?` → `"33FF0011"`
Note	

12.7 Settings

12.7.1 ETHernet:PORT<Pt>:SETTings:AARP

Syntax	ETHernet:PORT<Pt>:SETTings:AARP <enable>
Description	This command enables/disables answering incoming ARP requests.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT1:SETT:AARP ON`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:AARP?
Description	This query returns the state of answering incoming ARP requests.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:AARP? → 1`
Note	

12.7.2 ETHernet:PORT<Pt>:SETTings:ANDP

Syntax	ETHernet:PORT<Pt>:SETTings:ANDP <enable>
Description	This command enables/disables answering incoming NDP requests.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT1:SETT:ANDP ON`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:ANDP?
Description	This query returns enables/disables answering incoming NDP requests.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:ANDP? → 1`
Note	

12.7.3 ETHernet:PORT<Pt>:SETTings:APINg

Syntax	ETHernet:PORT<Pt>:SETTings:APINg <enable>
Description	This command enables/disables answering incoming PING requests.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT1:SETT:APIN ON`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:APINg?
Description	This query returns the state of answering incoming PING requests.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:APIN? → 1`
Note	

12.7.4 ETHernet:PORT<Pt>:SETTings:RLFSignaling

Syntax	ETHernet:PORT<Pt>:SETTings:RLFSignaling <enable>
Description	This command enables/disables respond to link fault signaling.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT1:SETT:RLFS ON`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:RLFSignaling?
Description	This query returns the state of respond to link fault signaling.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:RLFS? → 1`
Note	

12.7.5 ETHernet:PORT<Pt>:SETTings:ACMA

Syntax	ETHernet:PORT<Pt>:SETTings:ACMA <enable>
Description	This command enables/disables answering incoming Network Master configuration frames.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SETT:ACMA ON`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:ACMA?
Description	This query returns the state of answering incoming Network Master configuration frames.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:ACMA? → 1`
Note	

12.7.6 ETHernet:PORT<Pt>:SETTings:PLENenght

Syntax	ETHernet:PORT<Pt>:SETTings:PLENenght <length>
Description	This command sets the expected preamble length at 1Gbps and lower speeds.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 3, MAXimum = 15, DEFault = 8*
Response	None.
Example	`ETH:PORT1:SETT:PLEN 8`
Notes	

Syntax	ETHernet:PORT<Pt>:SETTings:PLENenght?
Description	This query returns the expected preamble length at 1Gbps and lower speeds.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:SETT:PLEN? → 8`
Note	

12.7.7 ETHernet:PORT<Pt>:SETTings:IPViolations

Syntax	ETHernet:PORT<Pt>:SETTings:IPViolations <enable>
Description	This command sets ignore preamble violations.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SETT:IPV ON`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:IPViolations?
Description	This query returns ignore preamble violations.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:IPV?` → `1`
Note	

12.7.8 ETHernet:PORT<Pt>:SETTings:ILThreshold

Syntax	ETHernet:PORT<Pt>:SETTings:ILThreshold <length>
Description	This command sets the IFG lower threshold.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 8, MAXimum = 27, DEFault = 12*
Response	None.
Example	`ETH:PORT1:SETT:ILT 12`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:ILThreshold?
Description	This query returns the IFG lower threshold.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:SETT:ILT?` → `12`
Note	

12.7.9 ETHernet:PORT<Pt>:SETTings:FIViolations

Syntax	ETHernet:PORT<Pt>:SETTings:FIViolations <enable>
Description	This command enables/disables filter IFG violations caused by master/slave clock synchronization.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT1:SETT:FIV ON`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:FIViolations?
Description	This query returns the state of filter IFG violations caused by master/slave clock synchronization.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:FIV?` → `1`
Note	

12.7.10 ETHernet:PORT<Pt>:SETTings:JFSize

Syntax	ETHernet:PORT<Pt>:SETTings:JFSize <length>
Description	This command sets the jumbo frame size upper limit.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 1519, MAXimum = 16000, DEFault = 9018*
Response	None.
Example	`ETH:PORT1:SETT:JFS 9018`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:JFSize?
Description	This query returns the jumbo frame size upper limit.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:SETT:JFS? → 9018`
Note	

12.7.11 ETHernet:PORT<Pt>:SETTings:ERFault

Syntax	ETHernet:PORT<Pt>:SETTings:ERFault <enable>
Description	This command sets enable remote fault detection in Autonegotiation.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT1:SETT:ERF ON`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:ERFault?
Description	This query returns enable remote fault detection in Autonegotiation.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:ERF? → 1`
Note	

12.7.12 ETHernet:PORT<Pt>:SETTings:FRESults

Syntax	ETHernet:PORT<Pt>:SETTings:FRESults <format>
Description	This command sets the format of the results.
Parameters	<Pt> = Port number
	<format> = <CHARACTER PROGRAM DATA> SI: SI prefix notation ENG: Engineering exponent notation SCI: Scientific exponent notation UNF: Unformatted *DEFault = SI*
Response	None.
Example	`ETH:PORT1:SETT:FRES ENG`
Note	This only affects the GUI and reports. This setting applies to all ports.

Syntax	ETHernet:PORT<Pt>:SETTings:FRESults?
Description	This query returns the format of the results.
Parameter	<Pt> = Port number
Response	<format> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:SETT:FRES? → ENG`
Note	This setting applies to all ports.

12.7.13 ETHernet:PORT<Pt>:SETTings:ACISetup

Syntax	ETHernet:PORT<Pt>:SETTings:ACISetup <enable>
Description	This command enables/disables allow changes to interface setup while measuring.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SETT:ACIS ON`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:ACISetup?
Description	This query returns the state of allow changes to interface setup while measuring.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:ACIS? → 1`
Note	

12.7.14 ETHernet:PORT<Pt>:SETTings:ASTGenerator

Syntax	ETHernet:PORT<Pt>:SETTings:ASTGenerator <enable>
Description	This command enables/disables automatically start the traffic generator when measurement is started.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SETT:ASTG ON`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:ASTGenerator?
Description	This query returns the state of automatically start the traffic generator when measurement is started.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:ASTG? → 1`
Note	

12.7.15 ETHernet:PORT<Pt>:SETTings:BER:OBAMeasuring

Syntax	ETHernet:PORT<Pt>:SETTings:BER:OBAMeasuring <enable>
Description	This command enables/disables only show BER alarms when measuring.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SETT:BER:OBAM ON`
Note	This setting applies to all ports.

Syntax	ETHernet:PORT<Pt>:SETTings:BER:OBAMeasuring?
Description	This query returns if BER alarms should only be shown when measuring.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:BER:OBAM? → 1`
Note	This setting applies to all ports.

12.7.16 ETHernet:PORT<Pt>:SETTings:BER:IAFFilter

Syntax	ETHernet:PORT<Pt>:SETTings:BER:IAFFilter <enable>
Description	This command enables/disables include addresses in frame filter on receiver.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT1:SETT:BER:IAFF OFF`
Note	This setting applies to all ports.

Syntax	ETHernet:PORT<Pt>:SETTings:BER:IAFFilter?
Description	This query returns if include addresses in frame filter on receiver is enabled/disabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:BER:IAFF?` → `0`
Note	This setting applies to all ports.

12.7.17 ETHernet:PORT<Pt>:SETTings:BER:CLFrames

Syntax	ETHernet:PORT<Pt>:SETTings:BER:CLFrames <enable>
Description	This command enables/disables count lost frames as pattern errors.
Parameter	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SETT:BER:CLF ON`
Note	This setting applies to all ports.

Syntax	ETHernet:PORT<Pt>:SETTings:BER:CLFrames?
Description	This query return if count lost frames as pattern errors is enabled/disabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:BER:CLF?` → `1`
Note	This setting applies to all ports.

12.7.18 ETHernet:PORT<Pt>:SETTings:BER:HIDeframeloss

Syntax	ETHernet:PORT<Pt>:SETTings:BER:HIDeframeloss <enable>
Description	This command show/hide Frame loss secs. count.
Parameter	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SETT:BER:HID ON`
Note	This setting applies to all ports.

Syntax	ETHernet:PORT<Pt>:SETTings:BER:HIDeframeloss?
Description	This query return if Frame loss secs. count is show/hide.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:BER:HID?` → `1`
Note	This setting applies to all ports.

12.7.19 ETHernet:PORT<Pt>:SETTings:CINBand[:ENABle]

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand[:ENABle] <enable>
Description	This command enables/disables in-band control.
Parameter	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SETT:CINB ON`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand[:ENABle]?
Description	This query returns whether or not in-band control is used.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:CINB? → 1`
Note	

12.7.20 ETHernet:PORT<Pt>:SETTings:CINBand:PASSword[:ENABle]

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:PASSword[:ENABle] <enable>
Description	This command enables/disables password of in-band control.
Parameter	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SETT:CINB:PASS ON`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:PASSword[:ENABle]?
Description	This query returns whether or not password of in-band control is used.
Parameter	<Pt> = Port number
Response	None.
Example	`ETH:PORT1:SETT:CINB:PASS? → 1`
Note	

12.7.21 ETHernet:PORT<Pt>:SETTings:CINBand:PASSword:VALue

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:PASSword:VALue <value>
Description	This command sets password value of in-band control.
Parameter	<Pt> = Port number <value> = <STRING RESPONSE DATA>
Response	None.
Example	`ETH:PORT1:SETT:CINB:PASS:VAL "Anritsu"`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:PASSword:VALue?
Description	This query returns the password value of in-band control.
Parameter	<Pt> = Port number
Response	<value> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:SETT:CINB:PASS:VAL? → "Anritsu"`
Note	

12.7.22 ETHernet:PORT<Pt>:SETTings:CINBand:MAC:SOURce

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:MAC:SOURce <address>
Description	This command sets the MAC source address of in-band control.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> MAC address
Response	None.
Example	`ETH:PORT1:SETT:CINB:MAC:SOUR "00-50-C2-35-D2-EF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:MAC:SOURce?
Description	This query returns the MAC source address of in-band control.
Parameters	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:SETT:CINB:MAC:SOUR?` → `"00-50-C2-35-D2-EF"`
Note	

12.7.23 ETHernet:PORT<Pt>:SETTings:CINBand:MAC:DEFault

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:MAC:DEFault <enable>
Description	This command enables/disables use of the default source MAC address of in-band control.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SETT:CINB:MAC:DEF ON`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:MAC:DEFault?
Description	This query returns whether or not the default source MAC address of in-band control is used.
Parameters	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:CINB:MAC:DEF?` → `1`
Note	

12.7.24 ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:SOURce

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:SOURce <address>
Description	This command sets the IPv4 source address of in-band control.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> IPv4 address
Response	None.
Example	`ETH:PORT1:SETT:CINB:IPV4:SOUR "172.29.2.36"`
Note	Only the character '.' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:SOURce?
Description	This query returns the IPv4 source address of in-band control.
Parameters	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:SETT:CINB:IPV4:SOUR?` → `"172.29.2.36"`
Note	

12.7.25 ETHernet:PORT<Pt>:SETTings:CINBand:DHCP[:ENABle]

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:DHCP[:ENABle] <enable>
Description	This command enables/disables use of DHCP of in-band control.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SETT:CINB:DHCP ON`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:DHCP[:ENABle]?
Description	This query returns whether or not DHCP of in-band control is used.
Parameters	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:CINB:DHCP? → 1`
Note	

12.7.26 ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:NETMask

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:NETMask <mask>
Description	This command sets the IPv4 netmask of in-band control.
Parameters	<Pt> = Port number
	<mask> = <STRING PROGRAM DATA> IPv4 netmask
Response	None.
Example	`ETH:PORT1:SETT:CINB:IPV4:NETM "255.255.255.0"`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:NETMask?
Description	This query returns the IPv4 netmask of in-band control.
Parameters	<Pt> = Port number
Response	<mask> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:SETT:CINB:IPV4:NETM? → "255.255.255.0"`
Note	

12.7.27 ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:GATeway[:ENABle]

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:GATeway[:ENABle] <enable>
Description	This command enables/disables use of the IPv4 gateway of in-band control.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SETT:CINB:IPV4:GAT ON`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:GATeway[:ENABle]?
Description	This query returns whether or not the IPv4 gateway of in-band control is used.
Parameters	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SETT:CINB:IPV4:GAT? → 1`
Note	

12.7.28 ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:GATeway:ADDRess

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:GATeway:ADDRess <address>
Description	This command sets the IPv4 default gateway of in-band control.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> IPv4 address
Response	None.
Example	`ETH:PORT1:SETT:CINB:IPV4:GAT:ADDR "172.29.2.36"`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:IPV4:GATeway:ADDRess?
Description	This query returns the IPv4 default gateway of in-band control.
Parameters	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:SETT:CINB:IPV4:GAT:ADDR?` → `"172.29.2.36"`
Note	

12.7.29 ETHernet:PORT<Pt>:SETTings:CINBand:VLAN:LCount

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:VLAN:LCount <levels>
Description	This command sets the number of active VLAN levels of in-band control.
Parameters	<Pt> = Port number
	<levels> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=2, DEFault=0*
Response	None.
Example	`ETH:PORT1:SETT:CINB:VLAN:LC 2`
Note	

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:VLAN:LCount?
Description	This query returns the number of active VLAN levels of in-band control.
Parameters	<Pt> = Port number
Response	<levels> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:SETT:CINB:VLAN:LC?` → `2`
Note	

12.7.30 ETHernet:PORT<Pt>:SETTings:CINBand:VLAN:LEVel<Lv>:ID

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:VLAN:LEVel<Lv>:ID <number>
Description	This command sets the VLAN ID of in-band control.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N[1])
	<number> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=4095, DEFault=0*
Response	None.
Example	`ETH:PORT1:SETT:CINB:VLAN:LEV1:ID 1024`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels (see section 12.7.29). Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame).

Syntax	ETHernet:PORT<Pt>:SETTings:CINBand:VLAN:LEVel<Lv>:ID?
Description	This query returns the VLAN ID of in-band control.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N)
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:SETT:CINB:VLAN:LEV1:ID?` → `1024`
Note	

12.8 Filter

Generally eight separate filters are available for each port of the instrument. In the following section these filters are identified using the `:FILTer<Ft>` program mnemonic, where `<Ft>` = Filter number (1-8).

12.8.1 ETHernet:PORT<Pt>:FILTer[:ENABle]

Syntax	ETHernet:PORT<Pt>:FILTer[:ENABle] <enable>
Description	This command enables/disables filtering.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Examples	`ETH:PORT1:FILT ON`
Note	This is the default node for `ETHernet:PORT<Pt>:FILTer`

Syntax	ETHernet:PORT<Pt>:FILTer[:ENABle]?
Description	This query returns whether or not filtering is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Examples	`ETH:PORT1:FILT?` → `1`
Note	This is the default node for `ETHernet:PORT<Pt>:FILTer?`

12.8.2 ETHernet:PORT<Pt>:FILTer:RESet

Syntax	ETHernet:PORT<Pt>:FILTer:RESet
Description	This command resets the filter settings to the default values.
Parameter	<Pt> = Port number
Response	None.
Example	`ETH:PORT1:FILT:RES`
Note	Note this does not change `ETHernet:PORT<Pt>:FILTer[:ENABle]`

12.8.3 ETHernet:PORT<Pt>:FILTer:ENCapsulation

Syntax	ETHernet:PORT<Pt>:FILTer:ENCapsulation <ether-type>, <snap>, <llc>
Description	This command sets the allowed encapsulation types.
Parameters	<Pt> = Port number
	<ether-type> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
	<snap> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
	<llc> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:FILT:ENC ON, OFF, ON`
Notes	Setting all off will disable the encapsulation filter. For the encapsulation filter to be active, at least one of the eight available general filters must be enabled. The general filters are enabled using the `ETHernet:PORT<Pt>:FILTer:FILTer<Ft>[:ENABle]` command (see 12.8.5).

Syntax	ETHernet:PORT<Pt>:FILTer:ENCapsulation?
Description	This query returns the allowed encapsulation types.
Parameter	<Pt> = Port number
Response	<ether-type> = <BOOLEAN RESPONSE DATA>
	<snap> = <BOOLEAN RESPONSE DATA>
	<llc> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:FILT:ENC?` → `1, 0, 1`
Note	

12.8.4 ETHernet:PORT<Pt>:FILTer:PROTocols

Syntax	ETHernet:PORT<Pt>:FILTer:PROTocols <type>,<filter>[,<mask>]
Description	This commands enables/disables protocol type filters and masks.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> DMAC: Destination MAC Address SMAC: Source MAC Address VLANx: VLAN ID where x is the VLAN level (1-8) BTAG: MiM B-Tag ITAG: MiM I-Tag MIMDMAC: MiM Destination MAC Address MIMSMAC: MiM Source MAC Address MPLSx: MPLS where x is the MPLS level (1-8) EMDM: EoMPLS destination MAC Address EMSM: EoMPLS source MAC Address SIV4: Source IPv4 DIV4: Destination IPv4 POFF: Pattern/Offset DPOR: Destination TCP/UDP port SIV6: Source IPv6 HIV6: Hop destination IPv6 FIV6: Final destination IPv6 SPOR: Source TCP/UDP port
	<filter> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
	<mask> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Examples	`ETH:PORT1:FILT:PROT MPLS4, ON` `ETH:PORT1:FILT:PROT MPLS4, ON, ON`
Note	<mask> can only be enabled if <filter> is enabled.

Syntax	ETHernet:PORT<Pt>:FILTer:PROTocols? <type>
Description	This query returns if the protocol type filters and masks are enabled/disabled.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA>
Response	<filter> = <BOOLEAN RESPONSE DATA>
	<mask> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:FILT:PROT? MPLS4` → `1, 1`
Note	

12.8.5 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>[:ENABle]

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>[:ENABle] <enable>
Description	This command enables/disables the specific filter.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8) (see 12.8)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1 ON`
Note	This is the default node for `ETHernet:PORT<Pt>:FILTer:FILTer<Ft>`

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>[:ENABle]?
Description	This query returns whether or not the specific filter is enabled.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Examples	`ETH:PORT1:FILT:FILT1?` → `1`
Note	This is the default node for `ETHernet:PORT<Pt>:FILTer:FILTer<Ft>?`

12.8.6 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SMAC

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SMAC <filter>[,<mask>]
Description	This command sets the MAC source filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <STRING PROGRAM DATA>
	<mask> = <STRING PROGRAM DATA>
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:SMAC "00-50-C2-35-D2-EF"` `ETH:PORT1:FILT:FILT1:SMAC "00-50-C2-35-D2-EF", "FF-FF-FF-FF-FF-FF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SMAC?
Description	This query returns the MAC source filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <STRING RESPONSE DATA>
	<mask> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:SMAC?` → `"00-50-C2-35-D2-EF", "FF-FF-FF-FF-FF-FF"`
Note	

12.8.7 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:DMAC

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:DMAC <filter>[,<mask>]
Description	This command sets the MAC destination filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <STRING PROGRAM DATA>
	<mask> = <STRING PROGRAM DATA>
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:DMAC "00-50-C2-35-D2-EF"` `ETH:PORT1:FILT:FILT1:DMAC "00-50-C2-35-D2-EF", "FF-FF-FF-FF-FF-FF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:DMAC?
Description	This query returns the MAC destination filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <STRING RESPONSE DATA>
	<mask> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:DMAC?` → `"00-50-C2-35-D2-EF", "FF-FF-FF-FF-FF-FF"`
Note	

12.8.8 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:EMSMac

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:EMSMac <filter>[,<mask>]
Description	This command sets the EoMPLS MAC source filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <STRING PROGRAM DATA>
	<mask> = <STRING PROGRAM DATA>
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:EMSM "00-50-C2-35-D2-EF"` `ETH:PORT1:FILT:FILT1:EMSM "00-50-C2-35-D2-EF", "FF-FF-FF-FF-FF-FF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:EMSMac?
Description	This query returns the EoMPLS MAC source filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <STRING RESPONSE DATA>
	<mask> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:EMSM?` → `"00-50-C2-35-D2-EF", "FF-FF-FF-FF-FF-FF"`
Note	

12.8.9 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:EMDMac

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:EMDMac <filter>[,<mask>]
Description	This command sets the EoMPLS MAC destination filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <STRING PROGRAM DATA>
	<mask> = <STRING PROGRAM DATA>
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:EMDM "00-50-C2-35-D2-EF"` `ETH:PORT1:FILT:FILT1:EMDM "00-50-C2-35-D2-EF", "FF-FF-FF-FF-FF-FF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:EMDMac?
Description	This query returns the EoMPLS MAC destination filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <STRING RESPONSE DATA>
	<mask> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:EMDM?` → `"00-50-C2-35-D2-EF", "FF-FF-FF-FF-FF-FF"`
Note	

12.8.10 ETHernet:PORT<Pt>:FILTer:POFFset

Syntax	ETHernet:PORT<Pt>:FILTer:POFFset <offset>
Description	This commands sets the pattern offset for the pattern filter.
Parameters	<Pt> = Port number
	<offset> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 15956, DEFault = 2*
Response	None.
Example	`ETH:PORT1:FILT:POFF 3`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:POFFset?
Description	This query returns the pattern offset for the pattern filter.
Parameter	<Pt> = Port number
Response	<offset> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:POFF?` → `3`
Note	

12.8.11 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:PATTern

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:PATTern <filter>[,<mask>]
Description	This command sets the Pattern filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum =* $2^{32} - 1$*, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum =* $2^{32} - 1$*, DEFault = 0*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:PATT 170` `ETH:PORT1:FILT:FILT1:PATT #HAA, #HFF`
Note	If the pattern is shorter than 32 bit it will be padded with '0' until a length of 32 bit is reached. (e.i., "101" becomes "00000000000000000000000000000101")

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:PATTern?
Description	This query returns the Pattern filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:PATT?` → `170, 256`
Note	

12.8.12 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SPORt

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SPORt <filter>[,<mask>]
Description	This command sets the TCP/UDP source port filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 65535, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 65535, DEFault = 65535*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:SPOR 81` `ETH:PORT1:FILT:FILT1:SPOR 1, 1`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SPORt?
Description	This query returns the TCP/UDP source port filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:SPOR?` → `1,1`
Note	

12.8.13 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:DPORt

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:DPORt <filter>[,<mask>]
Description	This command sets the TCP/UDP destination port filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <NUMERIC PROGRAM DATA>
	MINimum = 0, MAXimum = 65535, DEFault = 0
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 65535, DEFault = 0*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:DPOR 81` `ETH:PORT1:FILT:FILT1:DPOR 1,1`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:DPORt?
Description	This query returns the TCP/UDP destination port filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:DPOR?` → `1,1`
Note	

12.8.14 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SIV4

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SIV4 <filter>[,<mask>]
Description	This command sets the IPv4 source filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <STRING PROGRAM DATA>
	<mask> = <STRING PROGRAM DATA>
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:SIV4 "192.168.0.1"` `ETH:PORT1:FILT:FILT1:SIV4 "192.168.0.1", "255.255.0.0"`
Note	Only the character '.' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SIV4?
Description	This query returns the IPv4 source filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <STRING RESPONSE DATA>
	<mask> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:SIV4?` → `"192.168.0.1", "255.255.0.0"`
Note	

12.8.15 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:DIV4

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:DIV4 <filter>[,<mask>]
Description	This command sets the IPv4 destination filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <STRING PROGRAM DATA>
	<mask> = <STRING PROGRAM DATA>
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:DIV4 "192.168.0.1"` `ETH:PORT1:FILT:FILT1:DIV4 "192.168.0.1", "255.255.0.0"`
Note	Only the character '.' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:DIV4?
Description	This query returns the IPv4 destination filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <STRING RESPONSE DATA>
	<mask> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:DIV4?` → `"192.168.0.1", "255.255.0.0"`
Note	

12.8.16 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SIV6

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SIV6 <filter>[,<mask>]
Description	This command sets the IPv6 source filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <STRING PROGRAM DATA>
	<mask> = <STRING PROGRAM DATA>
Response	None.

Examples	`ETH:PORT1:FILT:FILT1:SIV6 "2002:0000:0000:0000:0000:0000:0000:001F"`
	`ETH:PORT1:FILT:FILT1:SIV6 "2002:0000:0000:0000:0000:0000:0000:0000",`
	`"FFFF:0000:0000:0000:0000:0000:0000:0000"`

Note	Only the character ':' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:SIV6?
Description	This query returns the IPv6 source filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <STRING RESPONSE DATA>
	<mask> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:SIV6?` → `"2002:0000:0000:0000:0000:0000:0000:0000",` `"FFFF:0000:0000:0000:0000:0000:0000:0000"`
Note	

12.8.17 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:HIV6

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:HIV6 <filter>[,<mask>]
Description	This command sets the IPv6 hop destination filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <STRING PROGRAM DATA>
	<mask> = <STRING PROGRAM DATA>
Response	None.

Examples	`ETH:PORT1:FILT:FILT1:HIV6 "2002:0000:0000:0000:0000:0000:0000:001F"`
	`ETH:PORT1:FILT:FILT1:HIV6 "2002:0000:0000:0000:0000:0000:0000:0000",`
	`"FFFF:0000:0000:0000:0000:0000:0000:0000"`

Note	Only the character ':' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:HIV6?
Description	This query returns the IPv6 hop destination filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <STRING RESPONSE DATA>
	<mask> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:HIV6?` → `"2002:0000:0000:0000:0000:0000:0000:0000",` `"FFFF:0000:0000:0000:0000:0000:0000:0000"`
Note	

12.8.18 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:FIV6

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:FIV6 <filter>[,<mask>]
Description	This command sets the IPv6 final destination filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <STRING PROGRAM DATA>
	<mask> = <STRING PROGRAM DATA>
Response	None.

Examples	`ETH:PORT1:FILT:FILT1:FIV6 "2002:0000:0000:0000:0000:0000:0000:001F"`
	`ETH:PORT1:FILT:FILT1:FIV6 "2002:0000:0000:0000:0000:0000:0000:0000",`
	`"FFFF:0000:0000:0000:0000:0000:0000:0000"`

Note	Only the character ':' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:FIV6?
Description	This query returns the IPv6 final destination filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <STRING RESPONSE DATA>
	<mask> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:FIV6?` → `"2002:0000:0000:0000:0000:0000:0000:0000",` `"FFFF:0000:0000:0000:0000:0000:0000:0000"`
Note	

12.8.19 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:LABel

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:LABel <filter>[,<mask>]
Description	This command sets the MPLS level filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1048575, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1048575, DEFault = 1048575*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:MPLS1:LAB 10` `ETH:PORT1:FILT:FILT1:MPLS1:LAB 10, 10`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:LABel?
Description	This query returns the MPLS level filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:MPLS1:LAB?` → `10,10`
Note	

12.8.20 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:EBITs

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:EBITs <filter>[,<mask>]
Description	This command sets the MPLS experimental bits filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 7*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:MPLS1:EBIT 3` `ETH:PORT1:FILT:FILT1:MPLS1:EBIT 3, 1`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:EBITs?
Description	This query returns the MPLS experimental bits filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:MPLS1:EBIT?` → `3,1`
Note	

12.8.21 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:BOLStack

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:BOLStack <filter>[,<mask>]
Description	This command sets the MPLS bottom of label stack filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1, DEFault = 1*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:MPLS1:BOLS 1` `ETH:PORT1:FILT:FILT1:MPLS1:BOLS 1, 1`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:BOLStack?
Description	This query returns the MPLS bottom of label stack filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:MPLS1:BOLS?` → `1,1`
Note	

12.8.22 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:TTL

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:TTL <filter>[,<mask>]
Description	This command sets the MPLS TTL filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255, DEFault = 255*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:MPLS1:TTL 3` `ETH:PORT1:FILT:FILT1:MPLS1:TTL 3, 1`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MPLS<Lv>:TTL?
Description	This query returns the MPLS TTL filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:MPLS1:TTL?` → `3,1`
Note	

12.8.23 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:ID

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:ID <filter>[,<mask>]
Description	This command sets the VLAN ID filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 4095, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 4095, DEFault = 4095*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:VLAN1:ID 3` `ETH:PORT1:FILT:FILT1:VLAN1:ID 3, 1`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:ID?
Description	This query returns the VLAN ID filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:VLAN1:ID?` → `3,1`
Note	

12.8.24 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:CFI

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:CFI <filter>[,<mask>]
Description	This command sets the VLAN CFI filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1, DEFault = 1*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:VLAN1:CFI 1` `ETH:PORT1:FILT:FILT1:VLAN1:CFI 1, 1`
Note	[1] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition. [2] This command is as same as `ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:DEI`.

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:CFI?
Description	This query returns the VLAN CFI filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:VLAN1:CFI?` → `3,1`
Note	[1] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition. [2] This command is as same as `ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:DEI?`.

12.8.25 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:DEI

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:DEI <filter>[,<mask>]
Description	This command sets the VLAN DEI filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1, DEFault = 1*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:VLAN1:DEI 1` `ETH:PORT1:FILT:FILT1:VLAN1:DEI 1, 1`
Note	[1] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition. [2] This command is as same as `ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:CFI`.

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:DEI?
Description	This query returns the VLAN DEI filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:VLAN1:DEI? → 3,1`
Note	[1] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition. [2] This command is as same as `ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:CFI?`.

12.8.26 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:PRiority

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:PRiority <filter>[,<mask>]
Description	This command sets the VLAN priority filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 7*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:VLAN1:PR 3` `ETH:PORT1:FILT:FILT1:VLAN1:PR 3, 1`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:VLAN<Lv>:PRiority?
Description	This query returns the VLAN priority filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<Lv> = Level number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:VLAN1:PR? → 3,1`
Note	

12.8.27 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMSMac

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMSMac <filter>[,<mask>]
Description	This command sets the MiM source MAC address filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <STRING PROGRAM DATA>
	<mask> = <STRING PROGRAM DATA>
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:MMSM "00-50-C2-35-D2-EF"` `ETH:PORT1:FILT:FILT1:MMSM "00-50-C2-35-D2-EF", "FF-FF-FF-FF-FF-FF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMSMac?
Description	This query returns the MiM source MAC address filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <STRING RESPONSE DATA>
	<mask> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:MMSM?` → `"00-50-C2-35-D2-EF", "FF-FF-FF-FF-FF-FF"`
Note	

12.8.28 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMDMac

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMDMac <filter>[,<mask>]
Description	This command sets the MIM destination MAC filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <STRING PROGRAM DATA>
	<mask> = <STRING PROGRAM DATA>
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:MMDM "00-50-C2-35-D2-EF"` `ETH:PORT1:FILT:FILT1:MMDM "00-50-C2-35-D2-EF", "FF-FF-FF-FF-FF-FF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMDMac?
Description	This query returns the MIM destination MAC filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <STRING RESPONSE DATA>
	<mask> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:MMDM?` → `"00-50-C2-35-D2-EF", "FF-FF-FF-FF-FF-FF"`
Note	

12.8.29 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMBTAG:ID

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMBTAG:ID <filter>[,<mask>]
Description	This command sets the MiM B-Tag VLAN ID filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 4095, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 4095, DEFault = 4095*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:MMBTAG:ID 3` `ETH:PORT1:FILT:FILT1:MMBTAG:ID 3, 1`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMBTAG:ID?
Description	This query returns the MiM B-Tag VLAN ID filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:MMBTAG:ID?` → `3,1`
Note	

12.8.30 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMBTAG:PRiority

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMBTAG:PRiority <filter>[,<mask>]
Description	This command sets the MiM B-Tag Priority (PCP) filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 7*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:MMBTAG:PR 3` `ETH:PORT1:FILT:FILT1:MMBTAG:PR 3, 1`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMBTAG:PRiority?
Description	This query returns the MiM B-Tag Priority (PCP) filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:MMBTAG:PR?` → `3,1`
Note	

12.8.31 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMBTAG:DEI

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMBTAG:DEI <filter>[,<mask>]
Description	This command sets the MiM B-Tag DEI filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1, DEFault = 1*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:MMBTAG:DEI 1` `ETH:PORT1:FILT:FILT1:MMBTAG:DEI 1, 1`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMBTAG:DEI?
Description	This query returns the MiM B-Tag DEI filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:MMBTAG:DEI?` → `1,1`
Note	

12.8.32 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:ID

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:ID <filter>[,<mask>]
Description	This command sets the MiM I-Tag Service ID filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 16777215, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 16777215, DEFault = 16777215*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:MMITAG:ID 3` `ETH:PORT1:FILT:FILT1:MMITAG:ID 3, 1`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:ID?
Description	This query returns the MiM I-Tag Service ID filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:MMITAG:ID?` → `3,1`
Note	

12.8.33 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:PRiority

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:PRiority <filter>[,<mask>]
Description	This command sets the MiM I-Tag Priority (PCP) filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 7*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:MMITAG:PR 3` `ETH:PORT1:FILT:FILT1:MMITAG:PR 3, 1`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:PRiority?
Description	This query returns the MiM I-Tag Priority (PCP) filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:MMITAG:PR?` → `3,1`
Note	

12.8.34 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:DEI

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:DEI <filter>[,<mask>]
Description	This command sets the MiM I-Tag DEI filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1, DEFault = 1*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:MMITAG:DEI 1` `ETH:PORT1:FILT:FILT1:MMITAG:DEI 1, 1`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:DEI?
Description	This query returns the MiM I-Tag DEI filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:MMITAG:DEI?` → `1,1`
Note	

12.8.35 ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:UCA

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:UCA <filter>[,<mask>]
Description	This command sets the MiM I-Tag UCA filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
	<filter> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1, DEFault = 0*
	<mask> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1, DEFault = 1*
Response	None.
Examples	`ETH:PORT1:FILT:FILT1:MMITAG:UCA 1` `ETH:PORT1:FILT:FILT1:MMITAG:UCA 1, 1`
Note	

Syntax	ETHernet:PORT<Pt>:FILTer:FILTer<Ft>:MMITAG:UCA?
Description	This query returns the MiM I-Tag UCA filter and mask.
Parameters	<Pt> = Port number
	<Ft> = Filter number (1-8)
Response	<filter> = <NR1 NUMERIC RESPONSE DATA>
	<mask> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:FILT:FILT1:MMITAG:UCA?` → `1,1`
Note	

12.9 Thresholds

12.9.1 ETHernet:PORT<Pt>:THResholds[:ENABle]

Syntax	ETHernet:PORT<Pt>:THResholds[:ENABle] <enable>
Description	This command enables/disables thresholds.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Examples	`ETH:PORT1:THR ON`
Note	This is the default node for `ETHernet:PORT<Pt>:THResholds`

Syntax	ETHernet:PORT<Pt>:THResholds[:ENABle]?
Description	This query returns whether or not thresholds are enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Examples	`ETH:PORT1:THR? → 1`
Note	This is the default node for `ETHernet:PORT<Pt>:THResholds?`

12.9.2 ETHernet:PORT<Pt>:THResholds:SELect

Syntax	ETHernet:PORT<Pt>:THResholds:SELect <type>, <enable>
Description	This command enables/disables the specific threshold type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> ALL: All thresholds UTIL: Utilization THR: Throughput ERR: Errored frames Measurement thresholds: COL: Collision rate UNI: Unicast frames MULTI: Multicast frames BROAD: Broadcast frames PAUSE: Pause frames FRAG: Fragmented frames UNDER: Undersized frames OVER: Oversized frames FCS: FCS errored frames IFG: IFG violations PRE: Preamble violations DIFF: Difference Tx-Rx DIFFO: Same as DIFF OVRFCSERR: Oversized & FCS Errored frames IPCHKSUM: IP checksum errored frames *DEFault = ALL*
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:THR:SEL UTIL, ON`
Note	Measurement thresholds are only active when a measurement is running.

Syntax	ETHernet:PORT<Pt>:THResholds:SELect? <type>
Description	This query returns whether or not the specific threshold type is enabled.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA>
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:THR:SEL? UTIL` → `1`
Note	

12.9.3 ETHernet:PORT<Pt>:THResholds:VALue

Syntax	ETHernet:PORT<Pt>:THResholds:VALue <type>, <compare>, <value>
Description	This command sets the level for the specific threshold type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> UTIL: Utilization[1] THR: Throughput[2] ERR: Errored frames Measurement thresholds: COL: Collision rate UNI: Unicast frames MULTI: Multicast frames BROAD: Broadcast frames PAUSE: Pause frames FRAG: Fragmented frames UNDER: Undersized frames OVER: Oversized frames FCS: FCS errored frames IFG: IFG violations PRE: Preamble violations DIFF: Difference Tx-Rx[3] DIFFO: Same as DIFF OVRFCSERR: Oversized & FCS Errored frames IPCHKSUM: IP checksum errored frames
	<compare> = <CHARACTER PROGRAM DATA> GT: Greater than LT: Less than GTEQ: Greater than or equal to LTEQ: Less than or equal to *DEFault = LT*
	<value> = <NUMERIC PROGRAM DATA> Absolute values: Decimals are rounded. Maximum is 4000000000 Percentage values: Allows one decimal, Maximum is 100.0 *DEFault = 0, MINimum = 0* *Allowed Suffixes = PCT*
Response	None.
Example	`ETH:PORT1:THR:VAL UTIL, GT, 10.5PCT`
Notes	Measurement thresholds are only active when a measurement is running. [1] Utilization only supports percentage values. [2] Throughput only supports absolute values, the unit is MBPS, and has one decimal. Maximum = 10000.0 [3] Difference only supports absolute count values.

Syntax	ETHernet:PORT<Pt>:THResholds:VALue? <type>
Description	This query returns the level for the specific threshold type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA>
Response	<compare> = <CHARACTER RESPONSE DATA>
	<value> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:THR:VAL? UTIL` → `GT, 10.5PCT`
Note	

12.10 SyncE

12.10.1 ETHernet:SYNCe:PTHRough:FCAPture

Syntax	ETHernet:SYNCe:PTHRough:FCAPture <enable>
Description	This command enables/disables SyncE frame capture when the ports are in pass-through mode.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:SYNC:PTHR:FCAP OFF`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:SYNCe:PTHRough:FCAPture?
Description	This query returns whether or not SyncE frame capture is enabled when the ports are in pass-through mode.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:SYNC:PTHR:FCAP?` → `0`
Note	This command can be used on V2.00 or later.

12.10.2 ETHernet:PORT<Pt>:SYNCe:PTHRough:QLEVel[:ENABle]

Syntax	ETHernet:PORT<Pt>:SYNCe:PTHRough:QLEVel[:ENABle] <enable>
Description	This command enables/disables modification of quality level in SSM messages.
Parameters	<Pt> = Port number Port number 1 denotes messages passing from port 1 to port 2 Port number 2 denotes messages passing from port 2 to port 1
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SYNC:PTHR:QLEV OFF`
Note	This is the default node for `ETHernet:PORT<Pt>:SYNCe:PTHRough:QLEVel` This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:SYNCe:PTHRough:QLEVel[:ENABle]?
Description	This query returns whether or not modification of quality level in SSM messages is enabled.
Parameter	<Pt> = Port number Port number 1 denotes messages passing from port 1 to port 2 Port number 2 denotes messages passing from port 2 to port 1
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SYNC:PTHR:QLEV?` → `0`
Note	This command can be used on V2.00 or later.

12.10.3 ETHernet:PORT<Pt>:SYNCe:PTHRough:QLEVel:USER

Syntax	ETHernet:PORT<Pt>:SYNCe:PTHRough:QLEVel:USER <level>
Description	This command sets the quality level to be inserted in SSM messages.
Parameters	<Pt> = Port number
	<level> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=15, DEFault=0*
Response	None.
Example	`ETH:PORT1:SYNC:PTHR:QLEV:USER 0`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:SYNCe:PTHRough:QLEVel:USER?
Description	This query returns the quality level to be inserted in SSM messages.
Parameter	<Pt> = Port number
Response	<level> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:SYNC:PTHR:QLEV:USER?` → `0`
Note	This command can be used on V2.00 or later.

12.10.4 ETHernet:PORT<Pt>:SYNCe[:ENABle]

Syntax	ETHernet:PORT<Pt>:SYNCe[:ENABle] <enable>
Description	This command enables/disables SyncE for the port.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SYNC ON`
Note	This is the default node for `ETHernet:PORT<Pt>:SYNCe`

Syntax	ETHernet:PORT<Pt>:SYNCe[:ENABle]?
Description	This query returns whether or not SyncE is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SYNC?` → `1`
Note	

12.10.5 ETHernet:PORT<Pt>:SYNCe:MODE

Syntax	ETHernet:PORT<Pt>:SYNCe:MODE <mode>
Description	This command set the SyncE operation mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> MONitor : Non-synchronous/Monitor-only mode SYNChronous : Active synchronous mode *DEFault = MONitor*
Response	None.
Example	`ETH:PORT1:SYNC:MODE MON`
Note	

Syntax	ETHernet:PORT<Pt>:SYNCe:MODE?
Description	This query returns the SyncE operation mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:SYNC:MODE?` → `SYNC`
Note	

12.10.6 ETHernet:PORT<Pt>:SYNCe:FCAPture

Syntax	ETHernet:PORT<Pt>:SYNCe:FCAPture <enable>
Description	This command enables/disables SyncE frame capture.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SYNC:FCAP OFF`
Note	

Syntax	ETHernet:PORT<Pt>:SYNCe:FCAPture?
Description	This query returns whether or not SyncE frame capture is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SYNC:FCAP?` → `1`
Note	

12.10.7 ETHernet:PORT<Pt>:SYNCe:QLEVel

Syntax	ETHernet:PORT<Pt>:SYNCe:QLEVel <qualityLevel>
Description	This command sets the quality level of the transmitter clock.
Parameters	<Pt> = Port number
	<qualityLevel> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=15, DEFault=0*
Response	None.
Example	`ETH:PORT1:SYNC:QLEV 0`
Note	

Syntax	ETHernet:PORT<Pt>:SYNCe:QLEVel?
Description	This query returns configured clock quality level.
Parameter	<Pt> = Port number
Response	<qualityLevel> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:SYNC:QLEV?` → `0`
Note	

12.10.8 ETHernet:PORT<Pt>:SYNCe:MAC

Syntax	ETHernet:PORT<Pt>:SYNCe:MAC <address>
Description	This command sets the MAC source address for transmitted ESMC messages.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> MAC address
Response	None.
Example	`ETH:PORT1:SYNC:MAC "00-50-C2-35-D2-EF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:SYNCe:MAC?
Description	This query returns the MAC source address for transmitted ESMC messages.
Parameter	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:SYNC:MAC?` → `"00-50-C2-35-D2-EF"`
Note	

12.10.9 ETHernet:PORT<Pt>:SYNCe:MAC:AUTO

Syntax	ETHernet:PORT<Pt>:SYNCe:MAC:AUTO <automatic>
Description	This command enables/disables automatic generation of source MAC address.
Parameters	<Pt> = Port number
	<automatic> = <CHARACTER PROGRAM DATA> ON: Lock the MAC parameter to the value of default MAC source address. OFF: Unlock the MAC parameter value. *DEFault = ON*
Response	None.
Example	`ETH:PORT1:SYNC:MAC:AUTO ON`
Note	

Syntax	ETHernet:PORT<Pt>:SYNCe:MAC:AUTO?
Description	This query returns whether or not automatic generation of source MAC address is enabled.
Parameter	<Pt> = Port number
Response	<automatic> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:SYNC:MAC:AUTO?` → `ON`
Note	

12.10.10 ETHernet:PORT<Pt>:SYNCe:EFLag

Syntax	ETHernet:PORT<Pt>:SYNCe:EFLag <behavior>
Description	This command set the SyncE Event flag behavior.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> DYNamic : Event flag bit is dynamically set ST0 : Event flag bit is set to 0 ST1 : Event flag bit is set to 1 *DEFault = DYNamic*
Response	None.
Example	`ETH:PORT1:SYNC:EFL DYN`
Note	

Syntax	ETHernet:PORT<Pt>:SYNCe:EFLag?
Description	This query returns the SyncE Event flag behavior.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:SYNC:EFL?` → `DYN`
Note	

12.11 Precision Time Protocol - IEEE 1588v2

12.11.1 ETHernet:PORT<Pt>:PTP[:ENABle]

Syntax	ETHernet:PORT<Pt>:PTP[:ENABle] <enable>
Description	This command enables/disables the Precision Time Protocol.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP ON`
Note	This is the default node for `ETHernet:PORT<Pt>:PTP`

Syntax	ETHernet:PORT<Pt>:PTP[:ENABle]?
Description	This query returns whether or not the Precision Time Protocol is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP? → 1`
Note	

12.11.2 ETHernet:PORT<Pt>:PTP:SONLy

Syntax	ETHernet:PORT<Pt>:PTP:SONLy <enable>
Description	This command enables/disables slave only clock mode.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT1:PTP:SONL ON`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:SONLy?
Description	This query returns whether or not slave only clock mode is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:SONL? → 1`
Note	

12.11.3 ETHernet:PORT<Pt>:PTP:CAST

Syntax	ETHernet:PORT<Pt>:PTP:CAST <cast>
Description	This command sets the multicast/unicast mode.
Parameters	<Pt> = Port number
	<cast> = <CHARACTER PROGRAM DATA> MULTicast: Multicast mode UNICast: Unicast mode *DEFault = UNICast*
Response	None.
Example	`ETH:PORT1:PTP:CAST DEF`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:CAST?
Description	This query returns the multicast/unicast mode.
Parameter	<Pt> = Port number
Response	<cast> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:CAST? → MULT`
Note	

12.11.4 ETHernet:PORT<Pt>:PTP:UNEGotiate

Syntax	ETHernet:PORT<Pt>:PTP:UNEGotiate <enable>
Description	This command enables/disables unicast negotiate mode.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT1:PTP:UNEG ON`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:UNEGotiate?
Description	This query returns whether or not unicast negotiate mode is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:UNEG? → 1`
Note	

12.11.5 ETHernet:PORT<Pt>:PTP:PROFile

Syntax	ETHernet:PORT<Pt>:PTP:PROFile <profile>
Description	This command sets the PTP profile.
Parameters	<Pt> = Port number
	<profile> = <CHARACTER PROGRAM DATA> UDEFined: User Defined G82651: G.8265.1 G82751: G.8275.1 G82752: G.8275.2 SMPTe: SMPTE 2059-2 *DEFault = UDEFined*
Response	None.
Example	`ETH:PORT1:PTP:PROF G82651`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:PROFile?
Description	This query returns the PTP profile.
Parameter	<Pt> = Port number
Response	<profile> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:PROF? → G82651`
Note	

12.11.6 ETHernet:PORT<Pt>:PTP:DOMain

Syntax	ETHernet:PORT<Pt>:PTP:DOMain <domain>
Description	This command sets the PTP domain Id.
Parameters	<Pt> = Port number
	<domain> = <CHARACTER PROGRAM DATA> DEFault: Default domain for multicast ALT1: Alternate 1 for multicast ALT2: Alternate 2 for multicast ALT3: Alternate 3 for multicast UDEFault: Default domain for unicast 5..255 : Alternative domains
Response	None.
Example	`ETH:PORT1:PTP:DOM DEF`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:DOMain?
Description	This query returns the PTP domain Id.
Parameter	<Pt> = Port number
Response	<domain> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:DOM?` → `DEF`
Note	

12.11.7 ETHernet:PORT<Pt>:PTP:TPRotocol

Syntax	ETHernet:PORT<Pt>:PTP:TPRotocol <protocol>
Description	This command sets the transport protocol to use for PTP communication.
Parameters	<Pt> = Port number
	<protocol> = <CHARACTER PROGRAM DATA> IPV4: UDP IPv4 IPV6: UDP IPv6 IEEE: IEEE 802.3 *DEFault = IPV4*
Response	None.
Example	`ETH:PORT1:PTP:TPR IPV6`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:TPRotocol?
Description	This query returns the transport protocol to use for PTP communication.
Parameter	<Pt> = Port number
Response	<protocol> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:TPR?` → `IPV6`
Note	

12.11.8 ETHernet:PORT<Pt>:PTP:MAC[:SOURce]

Syntax	ETHernet:PORT<Pt>:PTP:MAC[:SOURce] <address>
Description	This command sets the PTP MAC source address.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> MAC address
Response	None.
Example	`ETH:PORT1:PTP:MAC "00-50-C2-35-D2-EF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:PTP:MAC[:SOURce]?
Description	This query returns the PTP MAC source address.
Parameter	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:PTP:MAC?` → `"00-50-C2-35-D2-EF"`
Note	

12.11.9 ETHernet:PORT<Pt>:PTP:MAC:AUTO

Syntax	ETHernet:PORT<Pt>:PTP:MAC:AUTO <automatic>
Description	This command enables/disables automatic generation of PTP source MAC address.
Parameters	<Pt> = Port number
	<automatic> = <CHARACTER PROGRAM DATA> ON: Lock the MAC parameter to the value of default MAC source address. OFF: Unlock the MAC parameter value. *DEFault = ON*
Response	None.
Example	`ETH:PORT1:PTP:MAC:AUTO ON`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:MAC:AUTO?
Description	This query returns whether or not automatic generation of PTP source MAC address is enabled.
Parameter	<Pt> = Port number
Response	<automatic> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:MAC:AUTO?` → `ON`
Note	

12.11.10 ETHernet:PORT<Pt>:PTP:MAC:DESTination

Syntax	ETHernet:PORT<Pt>:PTP:MAC:DESTination <address>
Description	This command sets the PTP MAC destination address.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> MAC address
Response	None.
Example	`ETH:PORT1:PTP:MAC:DEST "00-50-C2-35-D2-EF"`
Note	Only the character '-' is accepted as separator. This value is used when `ETHernet:PORT<Pt>:PTP:SONLy` is enabled and `ETHernet:PORT<Pt>:PTP:CAST` is set to `UNICast`.

Syntax	ETHernet:PORT<Pt>:PTP:MAC:DESTination?
Description	This query returns the PTP MAC destination address.
Parameter	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:PTP:MAC:DEST?` → `"00-50-C2-35-D2-EF"`
Note	

12.11.11 ETHernet:PORT<Pt>:PTP:MAC:SMULticast

Syntax	ETHernet:PORT<Pt>:PTP:MAC:SMULticast <address type>
Description	This command sets multicast destination MAC addresses.
Parameters	<Pt> = Port number
	<address type> = <CHARACTER PROGRAM DATA> AUTO: Use default multicast addresses.Peer delay multicast address is set according to IEEE1588v2 standard one. NONForwardable: Use only non-forwardable multicast address 01-80-C2-00-00-0E.This parameter should be used for G.8275.1. FORWardable: Use only forwardable multicast address 01-1B-19-00-00-00.This parameter should be used for G.8275.1. *DEFault = AUTO*
Response	None.
Example	`ETH:PORT1:PTP:MAC:SMUL NONF`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:MAC:SMULticast?
Description	This query returns multicast destination MAC addresses
Parameter	<Pt> = Port number
Response	<address type> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:MAC:SMUL?` → `FORW`
Note	

12.11.12 ETHernet:PORT<Pt>:PTP:MAC:ARP

Syntax	ETHernet:PORT<Pt>:PTP:MAC:ARP
Description	this command execute ARP for IEEE1588v2.
Parameters	<Pt> = Port number
Response	None.
Example	`ETH:PORT1:PTP:MAC:ARP`
Note	

12.11.13 ETHernet:PORT<Pt>:PTP:MAC:ARP:RESult?

Syntax	ETHernet:PORT<Pt>:PTP:MAC:ARP:RESult?
Description	This query returns that ARP lookup succeeded or arp lookup timed out.
Parameters	<Pt> = Port number
Response	<result> = <CHARACTER RESPONSE DATA> SUCCESS: ARP lookup succeeded. TIMEOUT: ARP lookup timed out.
Example	`ETH:PORT1:PTP:MAC:ARP:RES?` → `SUCCESS`
Note	

12.11.14 ETHernet:PORT<Pt>:PTP:MAC:NDP

Syntax	ETHernet:PORT<Pt>:PTP:MAC:NDP
Description	This command execute NDP for PTP
Parameters	<Pt> = Port number
Response	None.
Example	`ETH:PORT1:PTP:MAC:NDP`
Note	

12.11.15 ETHernet:PORT<Pt>:PTP:MAC:NDP:RESult?

Syntax	ETHernet:PORT<Pt>:PTP:MAC:NDP:RESult?
Description	This query returns that NDP lookup succeeded or NDP lookup timed out.
Parameters	<Pt> = Port number
Response	<result> = <CHARACTER RESPONSE DATA> SUCCESS: NDP lookup succeeded. TIMEOUT: NDP lookup timed out.
Example	`ETH:PORT1:PTP:MAC:NDP:RES?` → `SUCCESS`
Note	

12.11.16 ETHernet:PORT<Pt>:PTP:PTYPe

Syntax	ETHernet:PORT<Pt>:PTP:PTYPe <payload type>
Description	This command sets payload type for IPv6 header.
Parameters	<Pt> = Port number
	<payload type> = <CHARACTER PROGRAM DATA> AUTO: IPv6 Payload length is set by auto. NULL: IPv6 Payload length is set 0. *DEFault = AUTO*
Response	None.
Example	`ETH:PORT1:PTP:PTYP NULL`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:PTYPe?
Description	This query returns payload type for IPv6 header.
Parameter	<Pt> = Port number
Response	<address type> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:PTYP?` → `NULL`
Note	

12.11.17 ETHernet:PORT<Pt>:PTP:EXT

Syntax	ETHernet:PORT<Pt>:PTP:EXT <enable>
Description	This command enables/disables the extended IPv6 header.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:EXT ON`
Notes	

Syntax	ETHernet:PORT<Pt>:PTP:EXT?
Description	This query returns the extended IPv6 header is enables or disables .
Parameters	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:EXT?` → `1`
Note	

12.11.18 ETHernet:PORT<Pt>:PTP:EXT:TYPE

Syntax	ETHernet:PORT<Pt>:PTP:EXT:TYPE <type>
Description	This command sets the type of extension header for PTP. This command is only valid when IPv6 is selected on layer3.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> HOP: Hop-by-Hop Options ROUTing: Routing(Type 0) DESTination: Destination Options
Response	None.
Example	`ETH:PORT1:PTP:EXT:TYPE HOP`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:EXT:TYPE?
Description	This query returns the type of extension header for PTP. This command is only valid when IPv6 is selected on layer3.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:EXT:TYPE?` → `HOP`
Note	

12.11.19 ETHernet:PORT<Pt>:PTP:EXT:LENGth

Syntax	ETHernet:PORT<Pt>:PTP:EXT:LENGth <length>
Description	This command sets the length of extension header for PTP. This command is only valid when IPv6 is selected on layer3.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum=6, MAXimum=254, DEFault=6*
Response	None.
Example	`ETH:PORT1:PTP:EXT:LENG 6`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:EXT:LENGth?
Description	This query returns the length of extension header for PTP. This command is only valid when IPv6 is selected on layer3.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:EXT:LENG?` → `0`
Note	

12.11.20 ETHernet:PORT<Pt>:PTP:EXT:DATA

Syntax	ETHernet:PORT<Pt>:PTP:EXT:DATA <data>
Description	This command sets the data of extension header for PTP. This command is only valid when IPv6 is selected on layer3.
Parameters	<Pt> = Port number
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:PTP:EXT:DATA "0123456789AB"`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:EXT:DATA?
Description	This query returns the data of extension header for PTP. This command is only valid when IPv6 is selected on layer3.
Parameter	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:PTP:EXT:DATA?` → `"0123456789AB"`
Note	

12.11.21 ETHernet:PORT<Pt>:PTP:IP:IPV4:SOURce

Syntax	ETHernet:PORT<Pt>:PTP:IP:IPV4:SOURce <address>
Description	This command sets the PTP IPv4 source address.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> IPv4 address
Response	None.
Example	`ETH:PORT1:PTP:IP:IPV4:SOUR "172.29.2.36"`
Note	Only the character '.' is accepted as separator. This value is used when `ETHernet:PORT<Pt>:PTP:TPRotocol` is set to `IPV4`

Syntax	ETHernet:PORT<Pt>:PTP:IP:IPV4:SOURce?
Description	This query returns the PTP IPv4 source address.
Parameter	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:PTP:IP:IPV4:SOUR?` → `"172.29.2.36"`
Note	

12.11.22 ETHernet:PORT<Pt>:PTP:IP:IPV4:DESTination

Syntax	ETHernet:PORT<Pt>:PTP:IP:IPV4:DESTination <address>
Description	This command sets the PTP IPv4 destination address. Used in unicast slave mode.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> IPv4 address
Response	None.
Example	`ETH:PORT1:PTP:IP:IPV4:DEST "172.29.2.36"`
Note	Only the character '.' is accepted as separator. This value is used when `ETHernet:PORT<Pt>:PTP:SONLy` is enabled and `ETHernet:PORT<Pt>:PTP:TPRotocol` is set to `IPV4` and `ETHernet:PORT<Pt>:PTP:CAST` is set to `UNICast`.

Syntax	ETHernet:PORT<Pt>:PTP:IP:IPV4:DESTination?
Description	This query returns the PTP IPv4 destination address.
Parameter	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:PTP:IP:IPV4:DEST?` → `"172.29.2.36"`
Note	

12.11.23 ETHernet:PORT<Pt>:PTP:IPV4:IGMP[:ENABle]

Syntax	ETHernet:PORT<Pt>:PTP:IPV4:IGMP[:ENABle] <enable>
Description	This command enables/disables use of the PTP IGMP.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:IPV4:IGMP ON`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:IPV4:IGMP[:ENABle]?
Description	This query returns whether or not the PTP IGMP is used.
Parameters	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:IPV4:IGMP?` → `1`
Note	

12.11.24 ETHernet:PORT<Pt>:PTP:IPV4:IGMP:VERSion

Syntax	ETHernet:PORT<Pt>:PTP:IPV4:IGMP:VERSion <version>
Description	This command sets the PTP IGMP version.
Parameters	<Pt> = Port number
	<version> = <CHARACTER PROGRAM DATA> V1: IGMPv1 V2: IGMPv2 V3: IGMPv3 *DEFault = V2*
Response	None.
Example	`ETH:PORT1:PTP:IPV4:IGMP:VERS V1`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:IPV4:IGMP:VERSion?
Description	This query returns the PTP IGMP version.
Parameters	<Pt> = Port number
Response	<version> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:IPV4:IGMP:VERS?` → `V2`
Note	

12.11.25 ETHernet:PORT<Pt>:PTP:IP:IPV6:SOURce

Syntax	ETHernet:PORT<Pt>:PTP:IP:IPV6:SOURce <address>
Description	This command sets the PTP IPv6 source address.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> IPv6 address
Response	None.
Example	`ETH:PORT1:PTP:IP:IPV6:SOUR "1234:5678:9ABC:DEF0:1234:5678:9ABC:DEF0"`
Note	This value is used when `ETHernet:PORT<Pt>:PTP:TPRotocol` is set to `IPV6`

Syntax	ETHernet:PORT<Pt>:PTP:IP:IPV6:SOURce?
Description	This query returns the PTP IPv6 source address.
Parameter	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:PTP:IP:IPV6:SOUR?` → `"1234:5678:9ABC:DEF0:1234:5678:9ABC:DEF0"`
Note	

12.11.26 ETHernet:PORT<Pt>:PTP:IP:IPV6:DESTination

Syntax	ETHernet:PORT<Pt>:PTP:IP:IPV6:DESTination <address>
Description	This command sets the PTP IPv6 destination address. Used in unicast slave mode.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> IPv6 address
Response	None.
Example	`ETH:PORT1:PTP:IP:IPV6:DEST "1234:5678:9ABC:DEF0:1234:5678:9ABC:DEF0"`
Note	This value is used when `ETHernet:PORT<Pt>:PTP:SONLy` is enabled and `ETHernet:PORT<Pt>:PTP:TPRotocol` is set to `IPV6` and `ETHernet:PORT<Pt>:PTP:CAST` is set to `UNICast`.

Syntax	ETHernet:PORT<Pt>:PTP:IP:IPV6:DESTination?
Description	This query returns the PTP IPv6 destination address.
Parameter	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:PTP:IP:IPV6:DEST?` → `"1234:5678:9ABC:DEF0:1234:5678:9ABC:DEF0"`
Note	

12.11.27 ETHernet:PORT<Pt>:PTP:IPV6:ADDRconfig

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:ADDRconfig <mode>
Description	This command sets the PTP address configuration mode for IPv6.
Parameters	<Pt> = Port number
	<St> = Stream number (1-16)
	<mode> = <CHARACTER PROGRAM DATA> MANual: Manual SLESs: Stateless *DEFault = SLESs*
Response	None.
Example	`ETH:PORT1:PTP:IPV6:ADDR SLES`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:ADDRconfig?
Description	This query returns the PTP address configuration mode for IPv6.
Parameters	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:IPV6:ADDR?` → `SLES`
Note	

12.11.28 ETHernet:PORT<Pt>:PTP:IPV6:ADDRconfig:RESult?

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:ADDRconfig:RESult?
Description	This query returns the PTP result of address config for IPv6.
Parameters	<Pt> = Port number
Response	<result> = <CHARACTER RESPONSE DATA> SUCCESS: Address configuration succeeded. TIMEOUT: Address configuration timed out.
Example	`ETH:PORT1:PTP:IPV6:ADDR:RES?` → `TIMEOUT`
Note	

12.11.29 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:IID

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:SLESs:IID <mode>
Description	This command sets the PTP interface ID for IPv6.
Parameters	<Pt> = Port number
	<id> = <STRING PROGRAM DATA> The interface ID string must consist of 16 hexadecimal digits.
Response	None.
Example	`ETH:PORT1:PTP:IPV6:SLES:IID "00-00-00-00-00-00-00-00"`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:SLESs:IID?
Description	This query returns the PTP address configuration mode for IPv6.
Parameters	<Pt> = Port number
Response	<mode> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:PTP:IPV6:SLES:IID?` → `"00-00-00-00-00-00-00-00"`
Note	

12.11.30 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:IID:AUTO

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:SLESs:IID:AUTO <enable>
Description	This command enables/disables Interface ID for PTP.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:IPV6:SLES:IID:AUTO ON`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:SLESs:IID:AUTO?
Description	This query returns whether or not Interface ID for PTP is enabled.
Parameters	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:IPV6:SLES:IID:AUTO?` → `1`
Note	

12.11.31 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:LINKlocal?

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:SLESs:LINKlocal?
Description	This query returns the PTP link local address.
Parameters	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:PTP:IPV6:SLES:LINK?` → `"1234:5678:9ABC:DEF0:1234:5678:9ABC:DEF0"`
Note	

12.11.32 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:RAFLag?

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:SLESs:RAFLag?
Description	This query returns the PTP stateless RA flags.
Parameters	<Pt> = Port number
Response	<flags> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:PTP:IPV6:SLES:RAFL?` → `"0x00"`
Note	

12.11.33 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:SRCMac?

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:SLESs:SRCMac?
Description	This query returns the PTP stateless source MAC address.
Parameters	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:PTP:IPV6:SLES:SRCM?` → `"00-00-00-00-00-00"`
Note	

12.11.34 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:PREFix?

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:SLESs:PREFix?
Description	This query returns the PTP stateless prefix.
Parameters	<Pt> = Port number
Response	<prefix> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:PTP:IPV6:SLES:PREF?` → `"1234:5678:9ABC:DEF0"`
Note	

12.11.35 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:PRFLag?

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:SLESs:PRFLag?
Description	This query returns the PTP stateless prefix flags.
Parameters	<Pt> = Port number
Response	<flags> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:PTP:IPV6:SLES:PRFL?` → `"0x00"`
Note	

12.11.36 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:LTIMe?

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:SLESs:LTIMe?
Description	This query returns the PTP stateless life time.
Parameters	<Pt> = Port number
Response	<time> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:PTP:IPV6:SLES:LTIM?` → `"Fri Jan 8 14:24:44 2010"`
Note	

12.11.37 ETHernet:PORT<Pt>:PTP:IPV6:SLESs:RENew

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:SLESs:RENew
Description	This command execute PTP stateless address configration. This command valid only when IpV6 is selected on layer3.
Parameters	<Pt> = Port number
Response	None.
Example	`ETH:PORT1:PTP:IPV6:SLES:REN`
Note	

12.11.38 ETHernet:PORT<Pt>:PTP:IPV6:MLD[:ENABle]

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:MLD[:ENABle] <enable>
Description	This command enables/disables use of the PTP MLD.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:IPV6:MLD ON`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:MLD[:ENABle]?
Description	This query returns whether or not the PTP MLD is used.
Parameters	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:IPV6:MLD?` → `1`
Note	

12.11.39 ETHernet:PORT<Pt>:PTP:IPV6:MLD:VERSion

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:MLD:VERSion <version>
Description	This command sets the PTP MLD version.
Parameters	<Pt> = Port number
	<version> = <CHARACTER PROGRAM DATA> V1: MLDv1 V2: MLDv2 *DEFault = V1*
Response	None.
Example	`ETH:PORT1:PTP:IPV6:MLD:VERS V1`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:IPV6:MLD:VERSion?
Description	This query returns the PTP MLD version.
Parameters	<Pt> = Port number
Response	<version> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:IPV6:MLD:VERS?` → `V2`
Note	

12.11.40 ETHernet:PORT<Pt>:PTP:IP:DSCP:EVENt

Syntax	ETHernet:PORT<Pt>:PTP:IP:DSCP:EVENt <value>
Description	This command sets the DSCP value to use for PTP event messages.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=63, DEFault=0*
Response	None.
Example	`ETH:PORT1:PTP:IP:DSCP:EVEN 0`
Note	This value is used when `ETHernet:PORT<Pt>:PTP:TPRotocol` is set to `IPV4` or `IPV6`.

Syntax	ETHernet:PORT<Pt>:PTP:IP:DSCP:EVENt?
Description	This query returns the DSCP value to use for PTP event messages.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:IP:DSCP:EVEN?` → `0`
Note	

12.11.41 ETHernet:PORT<Pt>:PTP:IP:DSCP:NORMal

Syntax	ETHernet:PORT<Pt>:PTP:IP:DSCP:NORMal <value>
Description	This command sets the DSCP value to use for normal PTP messages (not PTP events).
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=63, DEFault=0*
Response	None.
Example	`ETH:PORT1:PTP:IP:DSCP:NORM 0`
Note	This value is used when `ETHernet:PORT<Pt>:PTP:TPRotocol` is set to `IPV4` or `IPV6`.

Syntax	ETHernet:PORT<Pt>:PTP:IP:DSCP:NORMal?
Description	This query returns the DSCP value to use for normal PTP messages (not PTP events).
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:IP:DSCP:NORM?` → `0`
Note	

12.11.42 ETHernet:PORT<Pt>:PTP:TUNNeling

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling <tunneling>
Description	This command sets the tunneling mode to use for PTP communication.
Parameters	<Pt> = Port number
	<tunneling> = <CHARACTER PROGRAM DATA> NONE: No tunneling VLAN: VLAN MPLS: MPLS *DEFault = NONE*
Response	None.
Example	`ETH:PORT1:PTP:TUNN NONE`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling?
Description	This query returns the tunneling protocol to use for PTP communication.
Parameter	<Pt> = Port number
Response	<tunneling> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:TUNN?` → `NONE`
Note	

12.11.43 ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LCount

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LCount <levels>
Description	This command sets the number of active VLAN levels.
Parameters	<Pt> = Port number
	<levels> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8, DEFault=1*
Response	None.
Example	`ETH:PORT1:PTP:TUNN:VLAN:LC 2`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LCount?
Description	This query returns the number of active VLAN levels.
Parameter	<Pt> = Port number
Response	<levels> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:TUNN:VLAN:LC?` → `2`
Note	

12.11.44 ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:ID

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:ID <number>
Description	This command sets the VLAN ID.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N[1])
	<number> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=4095, DEFault=0*
Response	None.
Example	`ETH:PORT1:PTP:TUNN:VLAN:LEV1:ID 1024`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels (see section 12.32.97). Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame).

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:ID?
Description	This query returns the VLAN ID.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N)
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:TUNN:VLAN:LEV1:ID?` → `1024`
Note	

12.11.45 ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:CFI

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:CFI <enable>
Description	This command enables/disables the VLAN canonical format indicator.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N[1])
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:TUNN:VLAN:LEV1:CFI 1`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels (see section 12.32.97). Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame). [2] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition. [3] This command is as same as `ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:DEI`.

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:CFI?
Description	This query returns the state of the VLAN canonical format indicator.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:TUNN:VLAN:LEV1:CFI?` → `1`
Note	[1] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition. [2] This command is as same as `ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:DEI?`.

12.11.46 ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:DEI

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:DEI <enable>
Description	This command enables/disables the VLAN canonical format indicator.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N[1])
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:TUNN:VLAN:LEV1:DEI 1`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels (see section 12.32.97). Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame). [2] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition. [3] This command is as same as `ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:CFI`.

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:DEI?
Description	This query returns the state of the VLAN canonical format indicator.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:TUNN:VLAN:LEV1:DEI?` → `1`
Note	[1] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition. [2] This command is as same as `ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:CFI?`.

12.11.47 ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:PRiority

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:PRiority <priority>
Description	This command sets the VLAN priority.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N[1])
	<priority> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=7, DEFault=0*
Response	None.
Example	`ETH:PORT1:PTP:TUNN:VLAN:LEV1:PR 7`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels (see section 12.32.97). Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame).

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:PRiority?
Description	This query returns the VLAN priority.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N)
Response	<priority> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:TUNN:VLAN:LEV1:PR?` → `7`
Note	

12.11.48 ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:ETYPe

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:ETYPe <type>
Description	This command sets the VLAN Ethertype.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N)[1]
	<type> = <NUMERIC PROGRAM DATA> Acceptable values: #H8100 #H88A8 #H9100 #H9200 *DEFault=#H8100*
Response	None.
Example	`ETH:PORT1:PTP:TUNN:VLAN:LEV1:ETYP #H8100`
Notes	[1] It is only possible to use this command for levels (1-M), where M is the number of active VLAN Levels minus one(see section 12.32.97). Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame). It is not possible to use this command for VLAN level N as Ethertype because this level is automatically set according to the selected higher-level protocol.

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:VLAN:LEVel<Lv>:ETYPe?
Description	This query returns the VLAN Ethertype.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N[1])
Response	<type> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:TUNN:VLAN:LEV1:ETYP?` → `#H86DD`
Note	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels (see section 12.32.97).

12.11.49 ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LCount

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LCount <levels>
Description	This command sets the number of active MPLS levels.
Parameters	<Pt> = Port number
	<levels> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8, DEFault=1*
Response	None.
Example	`ETH:PORT1:PTP:TUNN:MPLS:LC 2`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LCount?
Description	This query returns the number of active MPLS levels.
Parameter	<Pt> = Port number
Response	<levels> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:TUNN:MPLS:LC?` → `2`
Note	

12.11.50 ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LEVel<Lv>:LABel

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LEVel<Lv>:LABel <label>
Description	This command sets the MPLS label.
Parameters	<Pt> = Port number
	<Lv> = MPLS level (1-N[1])
	<label> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1048575, DEFault=0*
Response	None.
Example	`ETH:PORT1:PTP:TUNN:MPLS:LEV1:LAB 1048575`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active MPLS Levels (see section 12.32.81). Level 1 corresponds to the level at the top of the label stack and N is the level at the bottom of the stack (bottom of stack flag set).

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LEVel<Lv>:LABel?
Description	This query returns the MPLS label.
Parameters	<Pt> = Port number
	<Lv> = MPLS level (1-N)
Response	<label> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:TUNN:MPLS:LEV1:LAB?` → `1048575`
Note	

12.11.51 ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LEVel<Lv>:EBITs

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LEVel<Lv>:EBITs <value>
Description	This command sets the MPLS experimental bits.
Parameters	<Pt> = Port number
	<Lv> = MPLS level (1-N[1])
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=7, DEFault=0*
Response	None.
Example	`ETH:PORT1:PTP:TUNN:MPLS:LEV1:EBIT 5`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active MPLS Levels (see section 12.32.81). Level 1 corresponds to the level at the top of the label stack and N is the level at the bottom of the stack (bottom of stack flag set).

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LEVel<Lv>:EBITs?
Description	This query returns the MPLS experimental bits.
Parameters	<Pt> = Port number
	<Lv> = MPLS level (1-N)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:TUNN:MPLS:LEV1:EBIT?` → `5`
Note	

12.11.52 ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LEVel<Lv>:TTL

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LEVel<Lv>:TTL <value>
Description	This command sets the MPLS time to live.
Parameters	<Pt> = Port number
	<Lv> = MPLS level (1-N[1])
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=32*
Response	None.
Example	`ETH:PORT1:PTP:TUNN:MPLS:LEV1:TTL 32`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active MPLS Levels (see section 12.32.81). Level 1 corresponds to the level at the top of the label stack and N is the level at the bottom of the stack (bottom of stack flag set).

Syntax	ETHernet:PORT<Pt>:PTP:TUNNeling:MPLS:LEVel<Lv>:TTL?
Description	This query returns the MPLS time to live.
Parameters	<Pt> = Port number
	<Lv> = MPLS level (1-N)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:TUNN:MPLS:LEV1:TTL?` → `32`
Note	

12.11.53 ETHernet:PORT<Pt>:PTP:FCAPture

Syntax	ETHernet:PORT<Pt>:PTP:FCAPture <enable>
Description	This command enables/disables PTP frame capture.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:FCAP OFF`
Note	Only frames for the configured domain ID will be captured.

Syntax	ETHernet:PORT<Pt>:PTP:FCAPture?
Description	This query returns whether or not PTP frame capture is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:FCAP?` → `1`
Note	

12.11.54 ETHernet:PORT<Pt>:PTP:FCAPture:DIRecotry

Syntax	ETHernet:PORT<Pt>:PTP:FCAPture:DIRecotry <directory>
Description	This command sets the PTP frame capture directory.
Parameters	<Pt> = Port number
	<directory> = <CHARACTER PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:PTP:FCAP:DIR "pcapdir"`
Note	Only frames for the configured domain ID will be captured.

Syntax	ETHernet:PORT<Pt>:PTP:FCAPture:DIRecotry?
Description	This query returns the PTP frame capture directory.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:FCAP:DIR?` → `"Internal/pcapdir"`
Note	There must be a connected application server for this command to be recognized as a legal command.
	The application server must be in the idle state.
	Directory must be selected to the `Internal/` directory or a sub-directory hereof.
	When a USB storage device is mounted, files can stored via the Usb/ directory.
	If the USB storage is not accessible, directory will be "Internal/" even if "Usb/" is selected.

12.11.55 ETHernet:PORT<Pt>:PTP:ELOG

Syntax	ETHernet:PORT<Pt>:PTP:ELOG <enable>
Description	This command enables/disables Extended logging in the IEEE 1588 clock log. All received and transmitted PTP messages will be logged.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT1:PTP:ELOG OFF`
Note	Only frames for the configured domain ID will be captured.

Syntax	ETHernet:PORT<Pt>:PTP:ELOG?
Description	This query returns whether or not Extended logging in the IEEE 1588 clock log is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:ELOG?` → `1`
Note	

12.11.56 ETHernet:PORT<Pt>:PTP:SMODe

Syntax	ETHernet:PORT<Pt>:PTP:SMODe <mode>
Description	This command sets the PTP step mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> ONE: One step mode TWO: Two step mode *DEFault = ONE*
Response	None.
Example	`ETH:PORT1:PTP:SMOD ON`
Note	<mode> = `ONE` forces `ETHernet:PORT<Pt>:PTP:DMEChanism` to `RRESponse`

Syntax	ETHernet:PORT<Pt>:PTP:SMODe?
Description	This query returns the PTP step mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:SMOD?` → `TW`
Note	

12.11.57 ETHernet:PORT<Pt>:PTP:DMEChanism

Syntax	ETHernet:PORT<Pt>:PTP:DMEChanism <mechanism>
Description	This command sets the PTP delay mechanism.
Parameters	<Pt> = Port number
	<mechanism> = <CHARACTER PROGRAM DATA> RRESponse: Request-Response delay mechanism PEER: Peer delay mechanism *DEFault = RRESponse*
Response	None.
Example	`ETH:PORT1:PTP:DMEC RRES`
Note	<mechanism> = `PEER` forces `ETHernet:PORT<Pt>:PTP:SMODe` to `TWO`

Syntax	ETHernet:PORT<Pt>:PTP:DMEChanism?
Description	This query returns the PTP delay mechanism.
Parameter	<Pt> = Port number
Response	<mechanism> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:DMEC?` → `RRES`
Note	

12.11.58 ETHernet:PORT<Pt>:PTP:CLOCk:SOURce

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:SOURce <source>
Description	This command sets the PTP clock source.
Parameters	<Pt> = Port number
	<source> = <CHARACTER PROGRAM DATA> INTernal: Use the internal instrument clock. GPS: Use the attach GPS device for time reference. *DEFault = INTernal*
Response	None.
Example	`ETH:PORT1:PTP:CLOC:SOUR INT`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:SOURce?
Description	This query returns the PTP clock source.
Parameter	<Pt> = Port number
Response	<source> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:CLOC:SOUR?` → `INT`
Note	

12.11.59 ETHernet:PORT<Pt>:PTP:CLOCk:IDENtity

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:IDENtity <id>
Description	This command sets the PTP clock identity.
Parameters	<Pt> = Port number
	<id> = <STRING PROGRAM DATA> The identity string must consist of 16 hexadecimal digits.
Response	None.
Example	`ETH:PORT1:PTP:CLOC:IDEN "00-00-00-00-00-00-00-00"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:IDENtity?
Description	This query returns the PTP clock identity.
Parameter	<Pt> = Port number
Response	<id> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:PTP:CLOC:IDEN?` → `"00-00-00-00-00-00-00-00"`
Note	

12.11.60 ETHernet:PORT<Pt>:PTP:CLOCk:IDENtity:AUTO

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:IDENtity:AUTO <automatic>
Description	This command enables/disables automatic generation of an identity from the PTP source MAC address.
Parameters	<Pt> = Port number
	<automatic> = <CHARACTER PROGRAM DATA> ON: Lock the identity parameter to the value of the PTP MAC source address. OFF: Unlock the identity parameter value. *DEFault = ON*
Response	None.
Example	`ETH:PORT1:PTP:CLOC:IDEN:AUTO ON`
Note	Auto generation is performed as described in IEEE 1588v2.

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:IDENtity:AUTO?
Description	This query returns whether or not automatic generation of an identity from the PTP source MAC address is enabled.
Parameter	<Pt> = Port number
Response	<automatic> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:CLOC:IDEN:AUTO? → ON`
Note	

12.11.61 ETHernet:PORT<Pt>:PTP:CLOCk:P1

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:P1 <priority>
Description	This command sets the PTP clock "priority 1" priority .
Parameters	<Pt> = Port number
	<priority> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=255*
Response	None.
Example	`ETH:PORT1:PTP:CLOC:P1 255`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:P1?
Description	This query returns the PTP clock "priority 1" priority.
Parameter	<Pt> = Port number
Response	<priority> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:CLOC:P1? → 255`
Note	

12.11.62 ETHernet:PORT<Pt>:PTP:CLOCk:P2

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:P2 <priority>
Description	This command sets the PTP clock "priority 2" priority .
Parameters	<Pt> = Port number
	<priority> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=255*
Response	None.
Example	`ETH:PORT1:PTP:CLOC:P2 255`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:P2?
Description	This query returns the PTP clock "priority 2" priority.
Parameter	<Pt> = Port number
Response	<priority> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:CLOC:P2? → 255`
Note	

12.11.63 ETHernet:PORT<Pt>:PTP:CLOCk:CLASs

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:CLASs <class>
Description	This command sets the PTP clock class.
Parameters	<Pt> = Port number
	<class> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=255*
Response	None.
Example	`ETH:PORT1:PTP:CLOC:CLAS 255`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:CLASs?
Description	This query returns the PTP clock class.
Parameter	<Pt> = Port number
Response	<class> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:CLOC:CLAS?` → `255`
Note	

12.11.64 ETHernet:PORT<Pt>:PTP:CLOCk:ACCuracy

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:ACCuracy <accuracy>
Description	This command sets the PTP clock accuracy.
Parameters	<Pt> = Port number
	<accuracy> = <CHARACTER PROGRAM DATA> NS25: 25 nano seconds NS100: 100 nano seconds NS250: 250 nano seconds US1: 1 micro second US2: 2.5 micro seconds US10: 10 micro seconds US25: 25 micro seconds US100: 100 micro seconds US250: 250 micro seconds MS1: 1 milli second MS2: 2.5 milli seconds MS10: 10 milli seconds MS25: 25 milli seconds MS100: 100 milli seconds MS250: 250 milli seconds S1: 1 second S10: 10 seconds PS10: Above 10 seconds UNKNown: Unknown accuracy USER: Use user defined value EPRTC: T-GM connected to a ePRTC in locked-mode[1] CPRTC: T-GM connected to a PRTC in locked-mode[1] NPRTC: T-GM not connected to a PRTC in locked-mode, or T-BC[1] *DEFault = UNKNown*
Response	None.
Example	`ETH:PORT1:PTP:CLOC:ACC UNKN`
Note	[1] Only available when the Profile is G.8275.1 or G.8275.2.

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:ACCuracy?
Description	This query returns the PTP clock accuracy.
Parameter	<Pt> = Port number
Response	<accuracy> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:CLOC:ACC?` → `UNKN`
Note	

12.11.65 ETHernet:PORT<Pt>:PTP:CLOCk:ACCuracy:USER

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:ACCuracy:USER <value>
Description	This command sets the user defined PTP clock accuracy.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=254*
Response	None.
Example	`ETH:PORT1:PTP:CLOC:ACC:USER 0`
Note	This value is used when `ETHernet:PORT<Pt>:PTP:CLOCk:ACCuracy` is set to `USER`.

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:ACCuracy:USER?
Description	This query returns the user defined PTP clock accuracy.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:CLOC:ACC:USER?` → `254`
Note	

12.11.66 ETHernet:PORT<Pt>:PTP:CLOCk:TSOurce

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:TSOurce <source>
Description	This command sets the PTP clock time source value.
Parameters	<Pt> = Port number
	<source> = <CHARACTER PROGRAM DATA> ATOMic: Atomicclock GPS: GPS clock TRADio: Terrestrial Radio PTP: PTP clock NTP: NTP clock HSET: Hand set OTHer: Other IOSCillator: Internal Oscillator SARB: SMPTE (ARB) SLOCal: SMPTE (Local) USER: Use user defined value *DEFault = IOSCillator*
Response	None.
Example	`ETH:PORT1:PTP:CLOC:TSO IOSC`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:TSOurce?
Description	This query returns the PTP clock time source value.
Parameter	<Pt> = Port number
Response	<source> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:CLOC:TSO?` → `IOSC`
Note	

12.11.67 ETHernet:PORT<Pt>:PTP:CLOCk:TSOurce:USER

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:TSOurce:USER <value>
Description	This command sets the user defined PTP clock time source value.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=160*
Response	None.
Example	`ETH:PORT1:PTP:CLOC:TSO:USER 255`
Note	This value is used when `ETHernet:PORT<Pt>:PTP:CLOCk:TSOurce` is set to `USER`.

Syntax	ETHernet:PORT<Pt>:PTP:CLOCk:TSOurce:USER?
Description	This query returns the user defined PTP clock time source value.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:CLOC:TSO:USER? → 255`
Note	

12.11.68 ETHernet:PORT<Pt>:PTP:TIMing:AINTerval

Syntax	ETHernet:PORT<Pt>:PTP:TIMing:AINTerval <interval>
Description	This command sets the PTP timing announce interval.
Parameters	<Pt> = Port number
	<interval> = <CHARACTER PROGRAM DATA> FS8: 1/8 second FS4: 1/4 second FS2: 1/2 second S05: 1/2 second (for backward compatibility only) S1: 1 second S2: 2 seconds S4: 4 seconds S8: 8 seconds S16: 16 seconds S32: 32 seconds *DEFault = S2*
Response	None.
Example	`ETH:PORT1:PTP:TIM:AINT S2`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:TIMing:AINTerval?
Description	This query returns the PTP timing announce interval.
Parameter	<Pt> = Port number
Response	<interval> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:TIM:AINT? → S1`
Note	

12.11.69 ETHernet:PORT<Pt>:PTP:TIMing:ATIMeout

Syntax	ETHernet:PORT<Pt>:PTP:TIMing:ATIMeout <intervals>
Description	This command sets the PTP timing announce timeout value. Unit: number of announce intervals.
Parameters	<Pt> = Port number
	<intervals> = <NUMERIC PROGRAM DATA> *MINimum=2, MAXimum=255, DEFault=3*
Response	None.
Example	`ETH:PORT1:PTP:TIM:ATIM 3`
Note	The actual timeout period will be `ATIMeout` multiplied by `AINTerval`

Syntax	ETHernet:PORT<Pt>:PTP:TIMing:ATIMeout?
Description	This query returns the PTP timing announce timeout value. Unit: number of announce intervals.
Parameter	<Pt> = Port number
Response	<intervals> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:TIM:ATIM?` → `3`
Note	

12.11.70 ETHernet:PORT<Pt>:PTP:TIMing:UOFFset

Syntax	ETHernet:PORT<Pt>:PTP:TIMing:UOFFset <offset>
Description	This command sets the nominal offset between UTC time and PTP/TAI time. The value is used for conversion and comparison of UTC and PTP/TAI time.
Parameters	<Pt> = Port number
	<offset> = <NUMERIC PROGRAM DATA> *MINimum=-32768, MAXimum=32767, DEFault=35*
Response	None.
Example	`ETH:PORT1:PTP:TIM:UOFF 35`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:TIMing:UOFFset?
Description	This query returns the nominal offset between UTC time and PTP/TAI time.
Parameter	<Pt> = Port number
Response	<offset> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:TIM:UOFF?` → `34`
Note	

12.11.71 ETHernet:PORT<Pt>:PTP:TIMing:SINTerval

Syntax	ETHernet:PORT<Pt>:PTP:TIMing:SINTerval <interval>
Description	This command sets the PTP timing sync interval.
Parameters	<Pt> = Port number
	<interval> = <CHARACTER PROGRAM DATA> FS128: 1/128 second FS64: 1/64 second FS32: 1/32 second FS16: 1/16 second FS8: 1/8 second FS4: 1/4 second FS2: 1/2 second S05: 1/2 second (for backward compatibility only) S1: 1 second S2: 2 seconds S4: 4 seconds S8: 8 seconds S16: 16 seconds S32: 32 seconds *DEFault = S1*
Response	None.
Example	`ETH:PORT1:PTP:TIM:SINT S1`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:TIMing:SINTerval?
Description	This query returns the PTP timing sync interval.
Parameter	<Pt> = Port number
Response	<interval> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:TIM:SINT?` → `S1`
Note	

12.11.72 ETHernet:PORT<Pt>:PTP:TIMing:DRINterval

Syntax	ETHernet:PORT<Pt>:PTP:TIMing:DRINterval <interval>
Description	This command sets the PTP timing delay request interval.
Parameters	<Pt> = Port number
	<interval> = <CHARACTER PROGRAM DATA> FS128: 1/128 second FS64: 1/64 second FS32: 1/32 second FS16: 1/16 second FS8: 1/8 second FS4: 1/4 second FS2: 1/2 second S05: 1/2 second (for backward compatibility only) S1: 1 second S2: 2 seconds S4: 4 seconds S8: 8 seconds S16: 16 seconds S32: 32 seconds DISabled: Use one-way-operation. (Disable delay requests/response) *DEFault = S1*
Response	None.
Example	`ETH:PORT1:PTP:TIM:DRIN S1`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:TIMing:DRINterval?
Description	This query returns the PTP timing delay request interval.
Parameter	<Pt> = Port number
Response	<interval> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:TIM:DRIN?` → `S1`
Note	

12.11.73 ETHernet:PORT<Pt>:PTP:TIMing:UDURation

Syntax	ETHernet:PORT<Pt>:PTP:TIMing:UDURation <timeout>
Description	This command sets the PTP timing unicast duration. Unit: Seconds.
Parameters	<Pt> = Port number
	<timeout> = <NUMERIC PROGRAM DATA> *MINimum = 60, MAXimum = 1000, DEFault = 300*
Response	None.
Example	`ETH:PORT1:PTP:TIM:UDUR 300`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:TIMing:UDURation?
Description	This query returns the PTP timing unicast duration.
Parameter	<Pt> = Port number
Response	<duration> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:TIM:UDUR?` → `300`
Note	

12.11.74 ETHernet:PORT<Pt>:PTP:TIMing:GCABle

Syntax	ETHernet:PORT<Pt>:PTP:TIMing:GCABle <delay>
Description	This command sets the delay time of GPS antenna cable length. Unit: nano Seconds.
Parameters	<Pt> = Port number
	<delay> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1000000000, DEFault=0*
Response	None.
Example	`ETH:PORT1:PTP:TIM:GCAB 35`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:TIMing:GCABle?
Description	This query returns the delay time of GPS antenna cable length.
Parameter	<Pt> = Port number
Response	<delay> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:TIM:GCAB? → 35`
Note	

12.11.75 ETHernet:PORT<Pt>:PTP:FFIeld?

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld?
Description	This query returns the flagField.
Parameter	<Pt> = Port number
Response	<value> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:PTP:FFI? → #H0000`
Note	

12.11.76 ETHernet:PORT<Pt>:PTP:FFIeld:AMFLag

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:AMFLag <enable>
Description	This command sets the alternateMasterFlag of flagField.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:FFI:AMFL OFF`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:AMFLag?
Description	This query returns the alternateMasterFlag of flagField.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:FFI:AMFL? → 0`
Note	

12.11.77 ETHernet:PORT<Pt>:PTP:FFIeld:L61

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:L61 <enable>
Description	This command sets the leap61 of flagField.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:FFI:L61 OFF`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:L61?
Description	This query returns the leap61 of flagField.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:FFI:L61?` → `0`
Note	

12.11.78 ETHernet:PORT<Pt>:PTP:FFIeld:L59

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:L59 <enable>
Description	This command sets the leap59 of flagField.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:FFI:L59 OFF`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:L59?
Description	This query returns the leap59 of flagField.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:FFI:L59?` → `0`
Note	

12.11.79 ETHernet:PORT<Pt>:PTP:FFIeld:CUOValid

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:CUOValid <enable>
Description	This command sets the currentUtcOffsetValid of flagField.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:FFI:CUOV OFF`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:CUOValid?
Description	This query returns the currentUtcOffsetValid of flagField.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:FFI:CUOV?` → `0`
Note	

12.11.80 ETHernet:PORT<Pt>:PTP:FFIeld:PTIMescale

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:PTIMescale <enable>
Description	This command sets the ptpTimescale of flagField.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:FFI:PTIM OFF`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:PTIMescale?
Description	This query returns the ptpTimescale of flagField.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:FFI:PTIM?` → `0`
Note	

12.11.81 ETHernet:PORT<Pt>:PTP:FFIeld:TTRaceable

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:TTRaceable <enable>
Description	This command sets the timeTraceable of flagField.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:FFI:TTR OFF`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:TTRaceable?
Description	This query returns the timeTraceable of flagField.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:FFI:TTR?` → `0`
Note	

12.11.82 ETHernet:PORT<Pt>:PTP:FFIeld:FTRaceable

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:FTRaceable <enable>
Description	This command sets the frequencyTraceable of flagField.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:FFI:FTR OFF`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:FTRaceable?
Description	This query returns the frequencyTraceable of flagField.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:FFI:FTR?` → `0`
Note	

12.11.83 ETHernet:PORT<Pt>:PTP:FFIeld:SUNCertain

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:SUNCertain <enable>
Description	This command sets the synchronizationUncertain of flagField.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:FFI:SUNC OFF`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:FFIeld:SUNCertain?
Description	This query returns the synchronizationUncertain of flagField.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:FFI:SUNC?` → `0`
Note	

12.11.84 ETHernet:PORT<Pt>:PTP:POPTions:IRTLv

Syntax	ETHernet:PORT<Pt>:PTP:POPTions:IRTLv <enable>
Description	This command sets the PTP profile options INTERFACE RATE TLV.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:PTP:POPT:IRTL OFF`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:POPTions:IRTLv?
Description	This query returns the PTP profile options INTERFACE RATE TLV.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:PTP:POPT:IRTL?` → `0`
Note	

12.11.85 ETHernet:PORT<Pt>:PTP:POPTions:IRTLv:SPEed

Syntax	ETHernet:PORT<Pt>:PTP:POPTions:IRTLv:SPEed <speed>
Description	This command sets the PTP profile options INTERFACE RATE TLV Speed.
Parameters	<Pt> = Port number
	<speed> = <CHARACTER PROGRAM DATA> 10M: 10Mbsp 100M: 100Mbps 1G: 1Gbps 10G: 10Gbps 25G: 25Gbps 40G: 40Gbps 100G: 100Gbps *DEFault = 10M*
Response	None.
Example	`ETH:PORT1:PTP:POPT:IRTL:SPE 10G`
Note	

Syntax	ETHernet:PORT<Pt>:PTP:POPTions:IRTLv:SPEed?
Description	This query returns the PTP profile options INTERFACE RATE TLV Speed.
Parameter	<Pt> = Port number
Response	<speed> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:PTP:POPT:IRTL:SPE?` → `10G`
Note	

12.11.86 ETHernet:PORT<Pt>:PTP:LOG?

Syntax	ETHernet:PORT<Pt>:PTP:LOG?
Description	This query returns a lists of logged PTP events.
Parameter	<Pt> = Port number
Response	<log> = <STRING RESPONSE DATA>

Example	`ETH:PORT1:PTP:LOG?` →
	"
	`10:25:26.729 IEEE1588v2 ENABLED`
	`10:25:26.729 Wall Clock:  10:26:00.000099450`
	`10:25:26.740 Master:  0050FFFEC235D2E:0`
	`10:25:26.740 Master:  cls:255 acc:254 var:4d2c`
	`10:25:26.740 Master:  prio1:255 prio2:255`
	`10:25:26.740 Clock State:  LISTENING`
	"

Note	If the list is empty an execution error will be reported.

12.12 Follow

12.12.1 ETHernet:FOLLow:TRAFfic

Syntax	ETHernet:FOLLow:TRAFfic <enable>
Description	This command sets traffic setup in Test-"Generator" for PORT2 to follow PORT1.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:FOLL:TRAF ON`
Note	This command is compatible with CMA3000.

Syntax	ETHernet:FOLLow:TRAFfic?
Description	This query returns whether or not traffic setup in Test-"Generator" for PORT2 follows PORT1.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:FOLL:TRAF?` → `1`
Note	This command is compatible with CMA3000.

12.12.2 ETHernet:FOLLow:STReam<St>

Syntax	ETHernet:FOLLow:STReam<St> <enable>
Description	This command sets setup in "Stream - Profile" and "Stream - Measurement" for PORT2 stream <St> to follow PORT1 stream <St>.
Parameters	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:FOLL:STR2 ON`
Note	This command is compatible with CMA3000.

Syntax	ETHernet:FOLLow:STReam<St>?
Description	This query returns whether or not setup in "Stream - Profile" and "Stream - Measurement" for PORT2 stream <St> follows PORT1 stream <St>.
Parameter	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:FOLL:STR1?` → `0`
Note	This command is compatible with CMA3000.

12.12.3 ETHernet:FOLLow:GATeway<St>

Syntax	ETHernet:FOLLow:GATeway<St> <enable>
Description	This command sets gateway setup for PORT2 stream <St> to follow PORT1 stream <St>.
Parameters	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:FOLL:GAT1 OFF`
Note	This will only have an effect if ETHernet:FOLLow:STReam<St> is enabled. This command is compatible with CMA3000.

Syntax	ETHernet:FOLLow:GATeway<St>?
Description	This query returns whether or not gateway setup for PORT2 stream <St> follows PORT1 stream <St>.
Parameter	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:FOLL:GAT1?` → `0`
Note	This command is compatible with CMA3000.

12.12.4 ETHernet:FOLLow:SETTings

Syntax	ETHernet:FOLLow:SETTings <enable>
Description	This command sets settings setup (Incoming Frames, Receiver Setup, In-band Setup, and Miscellaneous) in SETUP screen for PORT2 to follow PORT1.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:FOLL:SETT ON`
Note	This command is compatible with CMA3000.

Syntax	ETHernet:FOLLow:SETTings?
Description	This query returns whether or not settings setup (Incoming Frames, Receiver Setup, In-band Setup, and Miscellaneous) in SETUP screen for PORT2 follows PORT1.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:FOLL:SETT?` → `1`
Note	This command is compatible with CMA3000.

12.12.5 ETHernet:FOLLow:FILTer

Syntax	ETHernet:FOLLow:FILTer <enable>
Description	This command sets filter setup for PORT2 to follow PORT1.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:FOLL:FILT OFF`
Note	This command is compatible with CMA3000.

Syntax	ETHernet:FOLLow:FILTer?
Description	This query returns whether or not filter setup for PORT2 follows PORT1.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:FOLL:FILT?` → `0`
Note	This command is compatible with CMA3000.

12.12.6 ETHernet:FOLLow:THResholds

Syntax	ETHernet:FOLLow:THResholds <enable>
Description	This command sets threshold setup for PORT2 to follow PORT1.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:FOLL:THR ON`
Note	This command is compatible with CMA3000.

Syntax	ETHernet:FOLLow:THResholds?
Description	This query returns whether or not threshold setup for PORT2 follows PORT1.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:FOLL:THR?` → `1`
Note	This command is compatible with CMA3000.

12.12.7 ETHernet:FOLLow:SYNCe

Syntax	ETHernet:FOLLow:SYNCe <enable>
Description	This command sets SyncE setup for PORT2 to follow PORT1.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:FOLL:SYNC ON`
Note	This command is compatible with CMA3000.

Syntax	ETHernet:FOLLow:SYNCe?
Description	This query returns whether or not SyncE setup for PORT2 follows PORT1.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:FOLL:SYNC?` → `1`
Note	This command is compatible with CMA3000.

12.12.8 ETHernet:FOLLow:PTP

Syntax	ETHernet:FOLLow:PTP <enable>
Description	This command sets Precision Time Protocol setup for PORT2 to follow PORT1.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:FOLL:PTP ON`
Note	This command is compatible with CMA3000.

Syntax	ETHernet:FOLLow:PTP?
Description	This query returns whether or not Precision Time Protocol setup for PORT2 follows PORT1.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:FOLL:PTP?` → `1`
Note	This command is compatible with CMA3000.

12.12.9 ETHernet:PORT<Pt>:FOLLow:TRAFfic

Syntax	ETHernet:PORT<Pt>:FOLLow:TRAFfic <enable>
Description	This command sets traffic setup in Test-"Generator" for port <Pt> to follow PORT1.
Parameter	<Pt> = Port number (2-4)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT2:FOLL:TRAF ON`
Note	

Syntax	ETHernet:PORT<Pt>:FOLLow:TRAFfic?
Description	This query returns whether or not traffic setup in Test-"Generator" for port <Pt> follows PORT1.
Parameter	<Pt> = Port number (2-4)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT2:FOLL:TRAF?` → `1`
Note	

12.12.10 ETHernet:PORT<Pt>:FOLLow:STReam<St>

Syntax	ETHernet:PORT<Pt>:FOLLow:STReam<St> <enable>
Description	This command sets setup in "Stream - Profile" and "Stream - Measurement" for port <Pt> stream <St> to follow PORT1 stream <St>.
Parameter	<Pt> = Port number (2-4)
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT2:FOLL:STR1 ON`
Note	

Syntax	ETHernet:PORT<Pt>:FOLLow:STReam<St>?
Description	This query returns whether or not stream setup in "Stream - Profile" and "Stream - Measurement" for port <Pt> stream <St> follows PORT1 stream <St>.
Parameter	<Pt> = Port number (2-4)
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT2:FOLL:STR1?` → `1`
Note	

12.12.11 ETHernet:PORT<Pt>:FOLLow:GATeway<St>

Syntax	ETHernet:PORT<Pt>:FOLLow:GATeway<St> <enable>
Description	This command sets gateway setup for port <Pt> stream <St> to follow PORT1 stream <St>.
Parameter	<Pt> = Port number (2-4)
	<St> = Stream number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT2:FOLL:GAT1 OFF`
Note	This will only have an effect if ETH:PORT<Pt>:FOLLow:STReam<St> is enabled.

Syntax	ETHernet:PORT<Pt>:FOLLow:GATeway<St>?
Description	This query returns whether or not gateway setup for port <Pt> stream <St> follows PORT1 stream <St>.
Parameter	<Pt> = Port number (2-4)
	<St> = Stream number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT2:FOLL:GAT1?` → `0`
Note	

12.12.12 ETHernet:PORT<Pt>:FOLLow:SETTings

Syntax	ETHernet:PORT<Pt>:FOLLow:SETTings <enable>
Description	This command sets settings setup (Incoming Frames, Receiver Setup, In-band Setup, and Miscellaneous) for port <Pt> to follow PORT1.
Parameter	<Pt> = Port number (2-4)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT2:FOLL:SETT ON`
Note	

Syntax	ETHernet:PORT<Pt>:FOLLow:SETTings?
Description	This query returns whether or not settings setup (Incoming Frames, Receiver Setup, In-band Setup, and Miscellaneous) for port <Pt> follows PORT1.
Parameter	<Pt> = Port number (2-4)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT2:FOLL:SETT?` → `1`
Note	

12.12.13 ETHernet:PORT<Pt>:FOLLow:FILTer

Syntax	ETHernet:PORT<Pt>:FOLLow:FILTer <enable>
Description	This command sets filter setup for port <Pt> to follow PORT1.
Parameter	<Pt> = Port number (2-4)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT2:FOLL:FILT OFF`
Note	

Syntax	ETHernet:PORT<Pt>:FOLLow:FILTer?
Description	This query returns whether or not filter setup for port <Pt> follows PORT1.
Parameter	<Pt> = Port number (2-4)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT2:FOLL:FILT?` → `0`
Note	

12.12.14 ETHernet:PORT<Pt>:FOLLow:THResholds

Syntax	ETHernet:PORT<Pt>:FOLLow:THResholds <enable>
Description	This command sets threshold setup for port <Pt> to follow PORT1.
Parameter	<Pt> = Port number (2-4)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT2:FOLL:THR ON`
Note	

Syntax	ETHernet:PORT<Pt>:FOLLow:THResholds?
Description	This query returns whether or not threshold setup for port <Pt> follows PORT1.
Parameter	<Pt> = Port number (2-4)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT2:FOLL:THR?` → `1`
Note	

12.12.15 ETHernet:PORT<Pt>:FOLLow:SYNCe

Syntax	ETHernet:PORT<Pt>:FOLLow:SYNCe <enable>
Description	This command sets SyncE setup for port <Pt> to follow PORT1.
Parameter	<Pt> = Port number (2-4)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT2:FOLL:SYNC ON`
Note	

Syntax	ETHernet:PORT<Pt>:FOLLow:SYNCe?
Description	This query returns whether or not SyncE setup for port <Pt> follows PORT1.
Parameter	<Pt> = Port number (2-4)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT2:FOLL:SYNC? → 1`
Note	

12.12.16 ETHernet:PORT<Pt>:FOLLow:PTP

Syntax	ETHernet:PORT<Pt>:FOLLow:PTP <enable>
Description	This command sets Precision Time Protocol (IEEE 1588v2) setup for port <Pt> to follow PORT1.
Parameter	<Pt> = Port number (2-4)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT2:FOLL:PTP ON`
Note	

Syntax	ETHernet:PORT<Pt>:FOLLow:PTP?
Description	This query returns whether or not Precision Time Protocol (IEEE 1588v2) setup for port <Pt> follows PORT1.
Parameter	<Pt> = Port number (2-4)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT2:FOLL:PTP? → 1`
Note	

12.12.17 ETHernet:PORT<Pt>:FOLLow:WAN

Syntax	ETHernet:PORT<Pt>:FOLLow:WAN <enable>
Description	This command sets WAN setup for port <Pt> to follow PORT1.
Parameter	<Pt> = Port number (2-4)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:PORT2:FOLL:WAN ON`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:FOLLow:WAN?
Description	This query returns whether or not WAN setup for port <Pt> follows PORT1.
Parameter	<Pt> = Port number (2-4)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT2:FOLL:WAN? → 1`
Note	This command can be used on V2.00 or later.

12.13 Stimuli

12.13.1 ETHernet:PORT<Pt>:STIMuli:ALARm

Syntax	ETHernet:PORT<Pt>:STIMuli:ALARm <alarm>
Description	This command sets the stimuli alarm state.
Parameters	<Pt> = Port number
	<alarm> = <CHARACTER PROGRAM DATA> NONE: No alarm LINK: No link REM: Remote fault LF: Local fault[1] *DEFault = NONE*
Response	None.
Example	`ETH:PORT1:STIM:ALAR LINK`
Note	[1] Requires 10G as the active interface.

Syntax	ETHernet:PORT<Pt>:STIMuli:ALARm?
Description	This query returns the stimuli alarm state.
Parameter	<Pt> = Port number
Response	<alarm> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STIM:ALAR?` → `LINK`
Note	

12.13.2 ETHernet:PORT<Pt>:STIMuli:ERRor

Syntax	ETHernet:PORT<Pt>:STIMuli:ERRor <error>
Description	This command sets the stimuli error/violation type.
Parameters	<Pt> = Port number
	<error> = <CHARACTER PROGRAM DATA> NONE: No error IFG: IFG[1] FCS: FCS PRE: Preamble PAUS: PAUSE frames IPCS: Wrong IP checksum FRAG: Fragmented IP L4CS: Wrong layer 4 checksum PRBS: PRBS bit error BSEQ: BER sequence error SYMB: Error symbol / Invalid 10G block[2] FSYMB: FEC symbol error *DEFault = NONE*
Response	None.
Example	`ETH:PORT1:STIM:ERR L4CS`
Note	[1] Not available when 10Gbps is enabled. [2] When 10G is enabled, invalid 10G blocks are inserted instead of error symbols.

Syntax	ETHernet:PORT<Pt>:STIMuli:ERRor?
Description	This query returns the stimuli error/violation type.
Parameter	<Pt> = Port number
Response	<error> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STIM:ERR?` → `L4CS`
Note	

12.13.3 ETHernet:PORT<Pt>:STIMuli:EINSertion

Syntax	ETHernet:PORT<Pt>:STIMuli:EINSertion <mode>
Description	This command sets the stimuli error insertion mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: Stimuli disabled MANual B1: Burst every 1 second B10: Burst every 10 seconds BE2: Burst · 1E-02 BE3: Burst · 1E-03 BE4: Burst · 1E-04 BE5: Burst · 1E-05 BE6: Burst · 1E-06 BE7: Burst · 1E-07 *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STIM:EINS MAN`
Note	If insertion is set to MANual, errors are inserted with SYST:STIM:INS. See section 2.3.23. BE2-BE7 can only be used with PRBS bit error.

Syntax	ETHernet:PORT<Pt>:STIMuli:EINSertion?
Description	This query returns the stimuli error insertion mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STIM:EINS?` → `MAN`
Note	

12.13.4 ETHernet:PORT<Pt>:STIMuli:EBLength

Syntax	ETHernet:PORT<Pt>:STIMuli:EBLength <length>
Description	This command sets the error burst length.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 1*[1], *MAXimum = 65535*[1], *DEFault = 1*[1]
Response	None.
Example	`ETH:PORT1:STIM:EBL 128`
Note	[1] The following error types have other MINimum, MAXimum and DEFault values. IFG: *MINimum = 2, MAXimum = 100, DEFault = 2* FRAG: *MINimum = 2, MAXimum = 5, DEFault = 2* PRBS: *MINimum = 1, MAXimum = 255, DEFault = 1* BSEQ: *MINimum = 2, MAXimum = 2, DEFault = 2*

Syntax	ETHernet:PORT<Pt>:STIMuli:EBLength?
Description	This query returns the error burst length.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STIM:EBL?` → `128`
Note	

12.13.5 ETHernet:PORT<Pt>:STIMuli:FEC:ERRor:LANE

Syntax	ETHernet:PORT<Pt>:STIMuli:FEC:ERRor:LANE <content>
Description	This command sets the lane number.
Parameters	<Pt> = Port number
	<content> = <NUMERIC PROGRAM DATA> *MINimum=#B0000, MAXimum=#B1111, DEFault=#B0000*
Response	None.
Example	`ETH:PORT1:STIM:FEC:ERR:LANE #B1001` This command add error into lane 0 and 3.
Note	This command can be used on 100G(on MU110013A CFP2 or QSFP28 Adpt.)

Syntax	ETHernet:PORT<Pt>:STIMuli:FEC:ERRor:LANE?
Description	This query returns the lane number.
Parameter	<Pt> = Port number
Response	<content> = <BINARY NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STIM:FEC:ERR:LANE?` → `#B0000`
Note	This command can be used on 100G(on MU110013A CFP2 or QSFP28 Adpt.)

12.13.6 ETHernet:PORT<Pt>:STIMuli:WAN:ALARm

Syntax	ETHernet:PORT<Pt>:STIMuli:WAN:ALARm <alarm>
Description	This command sets the WAN stimuli alarm state.
Parameters	<Pt> = Port number
	<alarm> = <CHARACTER PROGRAM DATA> NONE: No alarm LOS: Loss of signal LOF: Loss of frame OOF: Out of frame STIM: Section/Mux section trace identifier mismatch LAIS: Line/Mux section - alarm indication signal LRDI: Line/Mux section - remote defect indicator PAIS: Path/Administrative unit - alarm indication signal PLOP: Path/Administrative unit - loss of pointer PTIM: Path/High-order path - trace identifier mismatch PPLM: Path/High-order path - payload label mismatch PUNEQ: Path/High-order path - unequipped PRDI: Path enhanced/High-order path - remote defect indicator LCD: Loss of code-group delineation *DEFault = NONE*
Response	None.
Example	`ETH:PORT1:STIM:WAN:ALAR LOS`
Note	[1] Only available when the WAN terminology is SONET. This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:STIMuli:WAN:ALARm?
Description	This query returns the WAN stimuli alarm state.
Parameter	<Pt> = Port number
Response	<alarm> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STIM:WAN:ALAR?` → `LOS`
Note	This command can be used on V2.00 or later.

12.13.7 ETHernet:PORT<Pt>:STIMuli:WAN:ERRor

Syntax	ETHernet:PORT<Pt>:STIMuli:WAN:ERRor <error>
Description	This command sets the WAN stimuli error/violation type.
Parameters	<Pt> = Port number
	<error> = <CHARACTER PROGRAM DATA> NONE: No error A1A2: Frame alignment word B1: B1 checksum byte B2: B2 checksum byte LREI: Line/Multiplex section - remote error indication B3: B3 checksum byte PREI: Path/High-order path - remote error indication *DEFault = NONE*
Response	None.
Example	`ETH:PORT1:STIM:WAN:ERR B1`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:STIMuli:WAN:ERRor?
Description	This query returns the WAN stimuli error/violation type.
Parameter	<Pt> = Port number
Response	<error> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STIM:WAN:ERR?` → `B1`
Note	This command can be used on V2.00 or later.

12.13.8 ETHernet:PORT<Pt>:STIMuli:WAN:EINSertion

Syntax	ETHernet:PORT<Pt>:STIMuli:WAN:EINSertion <mode>
Description	This command sets the WAN stimuli error insertion mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: Stimuli disabled MANual *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STIM:WAN:EINS MAN`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:STIMuli:WAN:EINSertion?
Description	This query returns the WAN stimuli error insertion mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STIM:WAN:EINS?` → `MAN`
Note	This command can be used on V2.00 or later.

12.13.9 ETHernet:PORT<Pt>:STIMuli:WAN:EBLength

Syntax	ETHernet:PORT<Pt>:STIMuli:WAN:EBLength <length>
Description	This command sets the WAN stimuli error burst length.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 8000, DEFault = 1*
Response	None.
Example	`ETH:PORT1:STIM:WAN:EBL 128`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:STIMuli:WAN:EBLength?
Description	This query returns the WAN stimuli error burst length.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STIM:WAN:EBL?` → `128`
Note	This command can be used on V2.00 or later.

12.13.10 ETHernet:PORT<Pt>:STIMuli:DEViation

Syntax	ETHernet:PORT<Pt>:STIMuli:DEViation <deviation>
Description	This command sets the amount of deviation to apply to the transmitted Ethernet signal. Unit: ppm.
Parameters	<Pt> = Port number
	<deviation> = <NUMERIC PROGRAM DATA> MT1000A: *MINimum=-100, MAXimum=100, DEFault=0* MT1100A: *MINimum=-200.0, MAXimum=200.0, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STIM:DEV 0`
Note	

Syntax	ETHernet:PORT<Pt>:STIMuli:DEViation?
Description	This query returns the amount of deviation to apply to the transmitted Ethernet signal.
Parameter	<Pt> = Port number
Response	MT1000A: <deviation> = <NR1 NUMERIC RESPONSE DATA> MT1100A: <deviation> = <NR2 NUMERIC RESPONSE DATA>
Example	MT1000A: `ETH:PORT1:STIM:DEV?` → `0` MT1100A: `ETH:PORT1:STIM:DEV?` → `0.0`
Note	

12.13.11 ETHernet:PORT<Pt>:STIMuli:PCS:ALARm

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:ALARm <alarm>
Description	This command sets the stimuli alarm state.
Parameters	<Pt> = Port number
	<alarm> = <CHARACTER PROGRAM DATA> NONE: No alarm HIBER: High BER *DEFault = NONE*
Response	None.
Example	`ETH:PORT1:STIM:PCS:ALAR HIBER`
Note	[1] Requires 10G as the active interface. This command can be used on 10/40/100G.

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:ALARm?
Description	This query returns the stimuli alarm state.
Parameter	<Pt> = Port number
Response	<alarm> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STIM:PCS:ALAR?` → `HIBER`
Note	This command can be used on 10/40/100G.

12.13.12 ETHernet:PORT<Pt>:STIMuli:PCS:ERRor

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:ERRor <error>
Description	This command sets the stimuli error/violation type.
Parameters	<Pt> = Port number
	<error> = <CHARACTER PROGRAM DATA> NONE: No error INVBT00: INVBT2D: INVBT33: INVBT66: INVSH00: INVSH11: INVMARKER: BIP: *DEFault = NONE*
Response	None.
Example	`ETH:PORT1:STIM:PCS:ERR NONE`
Note	This command can be used on 10/40/100G. INVMARKER and BIP cannot be used on 10G.

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:ERRor?
Description	This query returns the stimuli error/violation type.
Parameter	<Pt> = Port number
Response	<error> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STIM:PCS:ERR?` → `NONE`
Note	This command can be used on 10/40/100G.

12.13.13 ETHernet:PORT<Pt>:STIMuli:PCS:ERRor:LANE

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:ERRor:LANE <content>
Description	This command sets the lane number.
Parameters	<Pt> = Port number
	<content> = <NUMERIC PROGRAM DATA> *MINimum=#B00000000000000000000, MAXimum=#B11111111111111111111, DEFault=#B10000000000000000000*
Response	None.
Example	`ETH:PORT1:STIM:PCS:ERR:LANE #B1001000001` This command add error into lane 0, 3 and 9.
Note	This command can be used on 40/100G.

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:ERRor:LANE?
Description	This query returns the lane number.
Parameter	<Pt> = Port number
Response	<content> = <BINARY NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STIM:PCS:ERR:LANE?` → `#B10010000010000000000`
Note	This command can be used on 40/100G.

12.13.14 ETHernet:PORT<Pt>:STIMuli:PCS:EINSertion

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:EINSertion <error>
Description	This command sets the PCS error insertion mode.
Parameters	<Pt> = Port number
	<error> = <CHARACTER PROGRAM DATA> OFF: Stimuli disabled MANual BURSt RATE ALTernate ALL *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:STIM:PCS:EINS MAN`
Note	This command can be used on 10/40/100G.

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:EINSertion?
Description	This query returns the PCS stimuli insertion mode.
Parameter	<Pt> = Port number
Response	<error> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STIM:PCS:EINS?` → `MAN`
Note	This command can be used on 10/40/100G.

12.13.15 ETHernet:PORT<Pt>:STIMuli:PCS:EBLength

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:EBLength <length>
Description	This command sets the PCS stimuli error burst length.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 256, DEFault = 1*
Response	None.
Example	`ETH:PORT1:STIM:PCS:EBL 128`
Note	This command can be used on 10/40/100G.

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:EBLength?
Description	This query returns the PCS stimuli error burst length.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STIM:PCS:EBL?` → `128`
Note	This command can be used on 10/40/100G.

12.13.16 ETHernet:PORT<Pt>:STIMuli:PCS:ERATe

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:ERATe <rate>
Description	This command sets the PCS stimuli error rate.
Parameters	<Pt> = Port number
	<rate> = <NUMERIC PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STIM:PCS:ERAT 1E-8`
Note	This command can be used on 10/40/100G.

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:ERATe?
Description	This query returns the PCS stimuli error rate.
Parameter	<Pt> = Port number
Response	<rate> = <NR3 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STIM:PCS:ERAT?` → `1E-8`
Note	This command can be used on 10/40/100G.

12.13.17 ETHernet:PORT<Pt>:STIMuli:PCS:EELength

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:EELength <length>
Description	This command sets the PCS stimuli alternate error length.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 256, DEFault = 1*
Response	None.
Example	`ETH:PORT1:STIM:PCS:EEL 128`
Note	This command can be used on 10/40/100G.

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:EELength?
Description	This query returns the PCS stimuli alternate error length.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STIM:PCS:EEL?` → `128`
Note	This command can be used on 10/40/100G.

12.13.18 ETHernet:PORT<Pt>:STIMuli:PCS:ENLength

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:ENLength <length>
Description	This command sets the PCS stimuli alternate normal length.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 3125000000, DEFault = 1*
Response	None.
Example	`ETH:PORT1:STIM:PCS:ENL 128`
Note	This command can be used on 10/40/100G.

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:ENLength?
Description	This query returns the PCS stimuli alternate normal length.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STIM:PCS:ENL?` → `128`
Note	This command can be used on 10/40/100G.

12.13.19 ETHernet:PORT<Pt>:STIMuli:PCS:SKEW:BIT

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:SKEW:BIT <bit>
Description	This command sets the PCS stimuli skew bit.
Parameters	<Pt> = Port number
	<bit> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 8448, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STIM:PCS:SKEW:BIT 128`
Note	This command can be used on 40/100G.

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:SKEW:BIT?
Description	This query returns the PCS stimuli skew bit.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STIM:PCS:SKEW:BIT?` → `128`
Note	This command can be used on 40/100G.

12.13.20 ETHernet:PORT<Pt>:STIMuli:PCS:SKEW:TYPE

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:SKEW:TYPE <error>
Description	This command sets the PCS stimuli skew type.
Parameters	<Pt> = Port number
	<error> = <CHARACTER PROGRAM DATA> TXLANE: PCS stimuli skew type Tx lane PHYLANE: PCS stimuli skew type physical lane *DEFault = TXLANE*
Response	None.
Example	`ETH:PORT1:STIM:PCS:SKEW:TYPE PHYLANE`
Note	This command can be used on 40/100G.

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:SKEW:TYPE?
Description	This query returns the PCS stimuli skew type.
Parameter	<Pt> = Port number
Response	<error> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STIM:PCS:SKEW:TYPE?` → `PHYLANE`
Note	This command can be used on 40/100G.

12.13.21 ETHernet:PORT<Pt>:STIMuli:PCS:SKEW:LANE

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:SKEW:LANE <content>
Description	This command sets the PCS stimuli skew lane number.
Parameters	<Pt> = Port number
	<content> = <NUMERIC PROGRAM DATA> *MINimum=#B00000000000000000000, MAXimum=#B11111111111111111111, DEFault=#B10000000000000000000*
Response	None.
Example	`ETH:PORT1:STIM:PCS:SKEW:LANE #B1001000001` This command add error into lane 0, 3 and 9.
Note	This command can be used on 40/100G.

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:SKEW:LANE?
Description	This query returns the PCS stimuli skew lane number.
Parameter	<Pt> = Port number
Response	<content> = <BINARY NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STIM:PCS:SKEW:LANE?` → `#B10010000010000000000`
Note	This command can be used on 40/100G.

12.13.22 ETHernet:PORT<Pt>:STIMuli:PCS:SKEW:NS?

Syntax	ETHernet:PORT<Pt>:STIMuli:PCS:SKEW:NS?
Description	This query returns the PCS stimuli skew(ns).
Parameter	<Pt> = Port number
Response	<content> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STIM:PCS:SKEW:NS?` → `819.2`
Note	This command can be used on 40/100G.

12.14 Result

12.14.1 ETHernet:PORT<Pt>:IFETch?

Syntax	ETHernet:PORT<Pt>:IFETch? <parameter>
Description	This query fetches an Ethernet interval if available.
Parameters	<Pt> = Port number
	({<parameter>} + {,}∗) = <EXPRESSION PROGRAM DATA> The response format is listed for each parameter. **Performance** UTIL: Utilization. Response: <Min%>,<Max%>,<Avg%> THR: Throughput (bps). Response: <Min>,<Max>,<Avg> FRAT: Frame rate (fps). Response: <Min>,<Max>,<Avg> DEV: Deviation(ppm). Response: <Min>,<Max>,<Avg> **Frame Stat** TFR: Total frames. Response: <Count>,<Ratio> TGFR: Total good frames. Response: <Count>,<Ratio> UFR: Unicast frames. Response: <Count>,<Ratio> MFR: Multicast frames. Response: <Count>,<Ratio> BFR: Broadcast frames. Response: <Count>,<Ratio> PFR: Pause frames. Response: <Count>,<Ratio> **MPLS Frame Stat** MPFR: MPLS frames. Response: <Count>,<Ratio> MLM: MPLS level max. Response: <Count> MLMIN: MPLS level min. Response: <Count> EMFR: EoMPLS frames. Response: <Count>,<Ratio> LMPLS<Lv>: Last received MPLS Label, Priority and TTL. <Lv> = MPLS level. Response: <label>,<priority>,<TTL> **VLAN Frame Stat** VFR: VLAN frames. Response: <Count>,<Ratio> VLM: VLAN level max. Response: <Count> VLMIN: VLAN level min. Response: <Count> LVLAN<Lv>: Last received VLAN ID and priority. <Lv> = VLAN level. Response: <id>,<priority> **MiM Frame Stat** MIMFR: MiM frames. Response: <Count>,<Ratio> LBTAG: Last received B-TAG VLAN ID and priority Response: <id>,<priority> LITAG: Last received I-TAG service ID and priority Response: <id>,<priority> **Frame Alarms & Errors** TEFR: Total errored frames. Response: <Count>,<Ratio> FFR: Fragmented frames. Response: <Count>,<Ratio> UNFR: Undersized frames. Response: <Count>,<Ratio> OVFR: Oversized frames. Response: <Count>,<Ratio> FEFR: FCS errored frames. Response: <Count>,<Ratio> ESFR: Error symbol frames. Response: <Count>,<Ratio> OVRFCSERR: Oversized & FCS errored frames. Response: <Count>,<Ratio> IPCHKSUM: IP checksum errored frames. Response: <Count>,<Ratio> I10GB: Invalid 10G Blocks[2]. Response: <Count>, <Ratio> COLL: Collisions. Response: <Count>,<Ratio> PV: Preamble violations. Response: <Count>,<Ratio> IV: IFG violations. Response: <Count>,<Ratio> LF: 10/40/100G Local Faults. Response: <Count>

continued on next page...

. . . continued from previous page

	10GLF: Obsolete. For CMA 3000 backward compatibility only. Same as LF. RF: 10/40/100G Remote Faults. Response: <Count> 10GRF: Obsolete. For CMA 3000 backward compatibility only. Same as RF. **Frame difference** TFRD: Total frames difference (Tx-Rx<ref-port>)[3]. Response: <Count> **Burst Stat** BUFR: Bursted frames. Response: <Count>,<Ratio> NOB: Number of bursts. Response: <Count> AVGB: Average burst size. Response: <Count>,<Bytes> MAXB: Maximum burst size. Response: <Count>,<Bytes> MINB: Minimum burst size. Response: <Count>,<Bytes> **Size Distribution (Rx)** R46: 46-63[4]. Response: <Count>,<Ratio> R64: 64-127. Response: <Count>,<Ratio> R128: 128-255. Response: <Count>,<Ratio> R256: 256-511. Response: <Count>,<Ratio> R512: 512-1023. Response: <Count>,<Ratio> R1024: 1024-1518. Response: <Count>,<Ratio> RJUM: Jumbo frames. Response: <Count>,<Ratio> FSIZ: Frame size (bytes). Response: <Min>,<Max>,<Avg> **Transmit Stat(Tx)** TTFR: Total frames. Response: <Count> TTGB: Total good bytes (bytes). Response: <Count> TUFR: Unicast frames. Response: <Count> TMFR: Multicast frames. Response: <Count> TBFR: Broadcast frames. Response: <Count> TFE: FCS errors. Response: <Count> TTE: Total errors. Response: <Count> T46: 46-63[4]. Response: <Count> T64: 64-127. Response: <Count> T128: 128-255. Response: <Count> T256: 256-511. Response: <Count> T512: 512-1023. Response: <Count> T1024: 1024-1518. Response: <Count> TJUM: Jumbo frames. Response: <Count> **Transmit Stat(Rx)** RTGB: Total good bytes (bytes). Response: <Count> Each parameter of the following items is described on **Frame Stat**. - Total frames. → TFR - Unicast frames. → UFR - Multicast frames. → MFR - Broadcast frames. → BFR Each parameter of the following items is described on **Frame Alarms & Errors**. - Total errored. → TEFR - FCS errored. → FEFR Each parameter of the following items is described on **Size Distribution(Rx)**. - 46-63. → R46 - 64-127. → R64 - 128-255. → R128 - 256-511. → R256 - 512-1023. → R512

continued on next page. . .

. . . continued from previous page

- 1024-1518. → R1024
- Jumbo frames. → RJUM

Multistream Transmit Stat
STFx: Stream x frames. Response: <CountTx>,<CountRx>
STBx: Stream x bytes (bytes). Response: <CountTx>,<CountRx>
x = Stream number (1-16)

Multistream Throughput Stat
SFPSx: Stream x frames per second (fps). Response: <CountTx>,<CountRx>
SBPSx: Stream x bits per seconds (bps). Response: <CountTx>,<CountRx>
x = Stream number (1-16)

Multistream Frame Loss
SFLx: Stream x frame loss. Response: <Count>,<Ratio>
x = Stream number (1-16)

Multistream Latency/Jitter
SLx: Latency (μs). Response: <Min>,<Max>,<Avg>
SJx: Jitter (μs). Response: <Min>,<Max>,<Avg>
x = Stream number (1-16)

BER - Alarms & Errors
LOS: Loss of signal. Response: <Count>,<Ratio>
NLS: No link Seconds. Response: <Count>,<Ratio>
RFS: Remote fault seconds. Response: <Count>,<Ratio>
BPBC: Pattern bit count. Response: <Count>
BPE: Pattern errors. Response: <Count>,<Ratio>
PSA: PRBS Sync Alarm.[6] Response: <Count>
BSE: Sequence errors. Response: <Count>
BSSL: Sequence sync. Lost. Response: <Count>
BFL: Frame loss. Response: <Count>
BFLS: Frame loss seconds. Response: <Count>
SDMD: Maximum disruption. Response: <Microseconds>
SDAD: Average disruption. Response: <Microseconds>,<Count>
MES: M.2100 ES. Response: <Seconds>,<Ratio>
MSES: M.2100 SES. Response: <Seconds>,<Ratio>
MALS: M.2100 ALS. Response: <Seconds>,<Ratio>
MUAT: M.2100 UAT. Response: <Seconds>,<Ratio>
MAVT: M.2100 AVT. Response: <Seconds>,<Ratio>
MEFS: M.2100 EFS. Response: <Seconds>,<Ratio>
TDL: Throughput Data Layer (bps). Response: <Min>,<Max>,<Avg>
TNL: Throughput Network Layer (bps). Response: <Min>,<Max>,<Avg>
TLL: Throughput Link Layer (bps). Response: <Min>,<Max>,<Avg>
TPPL: Throughput Physical Layer without Preamble (bps).
Response: <Min>,<Max>,<Avg>
TPL: Throughput Physical Layer (bps). Response: <Min>,<Max>,<Avg>
TUL: Throughput Utilization Layer (bps). Response: <Min>,<Max>,<Avg>

WAN - Alarms
WLOS: Loss of signal. Response: <Seconds>,<Ratio>
WLOF: Loss of frame. Response: <Seconds>,<Ratio>
WSEF: WSEF/OOF -Severely errored frame/Out of frame. Response: <Seconds>,<Ratio>
WSTIM: Section/Mux section - Trace identifier mismatch. Response: <Seconds>,<Ratio>
WLAIS: Line/Mux section - alarm indication signal. Response: <Seconds>,<Ratio>
WLRDI: Line/Mux section - remote defect indicator. Response: <Seconds>,<Ratio>
WPAIS: Path/Administrative unit - alarm indication signal.

continued on next page. . .

. . . continued from previous page

	Response: <Seconds>,<Ratio> WPLOP: Path/Administrative unit - loss of pointer. Response: <Seconds>,<Ratio> WPTIM: Path/High-order path - trace Identifier mismatch. Response: <Seconds>,<Ratio> WPPLM: Path/High-order path - payload label mismatch. Response: <Seconds>,<Ratio> WPUNEQ: Path/High-order path - unequipped. Response: <Seconds>,<Ratio> WPRDI: Path/High-order path - remote defect indication. Response: <Seconds>,<Ratio> WPLCD: Loss of code-group delineation. Response: <Seconds>,<Ratio> **WAN - Errors** WA1A2: Response: <Count>,<Ratio> WB1: Response: <Count>,<Ratio> WB2: Response: <Count>,<Ratio> WLREI: Line/Mux section - remote error indication. Response: <Count>,<Ratio> WB3: Response: <Count>,<Ratio> WPREI: Path/High-order path - remote error indication. Response: <Count>,<Ratio> **WAN - Mux quality** WMFES: Mux forward ES. Response: <Count>,<Ratio> WMFSES: Mux forward SES. Response: <Count>,<Ratio> WMFUNAV: Mux forward UNAV. Response: <Count>,<Ratio> WMBES: Mux backward ES. Response: <Count>,<Ratio> WMBSES: Mux backward SES. Response: <Count>,<Ratio> WMBUNAV: Mux backward UNAV Response: <Count>,<Ratio> **WAN - SPE quality** WSFES: SPE forward ES. Response: <Count>,<Ratio> WSFSES: SPE forward SES. Response: <Count>,<Ratio> WSFUNAV: SPE forward UNAV. Response: <Count>,<Ratio> WSBES: SPE backward ES. Response: <Count>,<Ratio> WSBSES: SPE backward SES. Response: <Count>,<Ratio> WSBUNAV: SPE backward UNAV Response: <Count>,<Ratio> **SyncE - Synchronous Ethernet** QLD: Quality level distribution. Response: A percentage value for each of the 16 QLs <Count>,... QLR: Quality level reports. Response: <Count>,<Rate> QLU: Quality level unavailable time (SSF). Response: <Seconds>,<Ratio> QLT: Quality level transmission count. Response: <Count>,<Ratio> **IEEE1588v2 PTP - Precision Time Protocol** PAC: Announce count. Response: <CountTx>,<CountRx> PSC: Sync count. Response: <CountTx>,<CountRx> PFUC: Follow up count. Response: <CountTx>,<CountRx> PDRC: Delay request count. Response: <CountTx>,<CountRx> PDRESC: Delay response count. Response: <CountTx>,<CountRx> PPDRC: Peer delay request count. Response: <CountTx>,<CountRx> PPDRESC: Peer delay response count. Response: <CountTx>,<CountRx> PPDFC: Peer delay follow-up count. Response: <CountTx>,<CountRx> SIGN: Signalling messages count. Response: <CountTx>,<CountRx> MGM: Management messages count. Response: <CountTx>,<CountRx> PO: Offset. Unit: Nano seconds. Response: <Min>,<Max>,<Avg> PAO: Absolute offset. Unit: Nano seconds. Response: <Min>,<Max>,<Avg> POD: Offset deviation. Unit: Nano seconds. Response: <Min>,<Max>,<Avg> POV: Offset variance. Unit: (seconds)^2. Response: <Min>,<Max>,<Avg> PMPD: Mean path delay. Unit: Nano seconds. Response: <Min>,<Max>,<Avg>

continued on next page. . .

. . . continued from previous page

	PPMPD: Peer mean path delay. Unit: Nano seconds. Response: <Min>,<Max>,<Avg> PPDV: Path delay variation. Unit: Nano seconds. Response: <Min>,<Max>,<Avg> PSIRCC: Signalling INTERFACE RATE TLV changed count. Response: <Count> PICST: IEEE Clock State Transitions. Response: <Count> PISTE: IEEE State Transition Events. Response: <Count> PIF: IEEE Faults. Response: <Count> PICGC: IEEE Changes in Grandmaster Clock. Response: <Count> PGW: GPS vs Wall clock. Unit: Nano seconds. Response: <Min>,<Max>,<Avg> **OAM 802.ah** OINF: Information count. Response: <CountTx>,<CountRx> OEVENT: Event count. Response: <CountTx>,<CountRx> OVREQ: Variable Request count. Response: <CountTx>,<CountRx> OVRES: Variable response count. Response: <CountTx>,<CountRx> OLBCK: Loopback count. Response: <CountTx>,<CountRx> ODUP: Duplicate count. Response: <CountTx>,<CountRx> OUNS: Unsupported count. Response: <CountTx>,<CountRx> OORG: Organization count. Response: <CountTx>,<CountRx> **OAM 802.ag and Y.1731** OCCM: CCM count. Response: <CountTx>,<CountRx> OLBM: LBM count. Response: <CountTx>,<CountRx> OLBR: LBR count. Response: <CountTx>,<CountRx> OLTM: LTM count. Response: <CountTx>,<CountRx> OLTR: LTR count. Response: <CountTx>,<CountRx> OOTH: Other count. Response: <CountTx>,<CountRx> OTOT: Total count. Response: <CountTx>,<CountRx> **Sync Test** BRATE: Bitrate[5]. Unit: bps. Response: <Count> BRATED: Bitrate difference[5]. Unit: ppb. Response: <Count> PHASE: Phase error result[5]. Unit: Nano seconds. Response: <Seconds> PHASEAVG: Average phase error result[5]. Unit: Nano seconds. Response: <Seconds> PHASEMIN: Minimum phase error result[5]. Unit: Nano seconds. Response: <Seconds> PHASEMAX: Maximum phase error result[5]. Unit: Nano seconds. Response: <Seconds> FILTERED: Filtered TE result[5]. Unit: Nano seconds. Response: <Seconds> FILTEREDAVG: Average filtered TE result[5]. Unit: Nano seconds. Response: <Seconds> FILTEREDMIN: Minimum filtered TE result[5]. Unit: Nano seconds. Response: <Seconds> FILTEREDMAX: Maximum filtered TE result[5]. Unit: Nano seconds. Response: <Seconds> PPS: 1PPS deviation result[5]. Unit: Nano seconds. Response: <Seconds> PPSAVG: Average 1PPS deviation average result[5]. Unit: Nano seconds. Response: <Seconds> PPSMIN: Minimum 1PPS deviation result[5]. Unit: Nano seconds. Response: <Seconds> PPSMAX: Maximum 1PPS deviation result[5]. Unit: Nano seconds. Response: <Seconds> SYNCAVG: Average sync message transmission time[5]. Unit: Nano seconds. Response: <Seconds> SYNCMIN: Minimum sync message transmission time[5]. Unit: Nano seconds. Response: <Seconds> SYNCMAX: Maximum sync message transmission time[5]. Unit: Nano seconds. Response: <Seconds> FOLLOWAVG: Average followup message transmission time[5]. Unit: Nano seconds. Response: <Seconds> FOLLOWMIN: Minimum followup message transmission time[5]. Unit: Nano seconds. Response: <Seconds> FOLLOWMAX: Maximum followup message transmission time[5]. Unit: Nano seconds. Response: <Seconds>

continued on next page. . .

. . . continued from previous page

	DELAYAVG: Average delay request Message transmission time[5]. Unit: Nano seconds. Response: <Seconds> DELAYMIN: Minimum delay request Message transmission time[5]. Unit: Nano seconds. Response: <Seconds> DELAYMAX: Maximum delay request Message transmission time[5]. Unit: Nano seconds. Response: <Seconds> CTE: cTE result[5]. Unit: Nano seconds. Response: <Seconds> DTE: dTE result[5]. Unit: Nano seconds. Response: <Seconds> MAXTE: Maximum TE result[5]. Unit: Nano seconds. Response: <Seconds> CTE1: cTE1 result[5]. Unit: Nano seconds. Response: <Seconds> CTE4: cTE4 result[5]. Unit: Nano seconds. Response: <Seconds> MAXTE1: Max—TE1— result[5]. Unit: Nano seconds. Response: <Seconds> MAXTE4: Max—TE4— result[5]. Unit: Nano seconds. Response: <Seconds> TERR: Terr result[5]. Unit: Nano seconds. Response: <Seconds> TERRABSMAX: Absolute maximum Terr result[5]. Unit: Nano seconds. Response: <Seconds> TERRMIN: Minimum Terr result[5]. Unit: Nano seconds. Response: <Seconds> TERRMAX: Maximum Terr result[5]. Unit: Nano seconds. Response: <Seconds> **PCS Alarms & Errors** ISH: Invalid Sync Header. Response: <Lane>[7],<Total> IAM: Invalid Alignment Marker. Response: <Lane>[7],<Total> BIP: BIP Error. (bit) Response: <Lane>[7],<Total> IBLOCK: Invalid Block. Response: <Count> HBER: High BER. Response: <Count>,<Ratio> **PCS Status** SHLOCK: Sync header Lock.(LOBL) 0: No Alarm, 1: Alarm. <Lane>[7] AMLOCK: Align Marker Lock.(LOAML) 0: No Alarm, 1: Alarm. <Lane>[7] MMAP: Marker Map. <Lane>[7] RSKEW: Relative Skew. Unit: Nano seconds. <Lane>[7] ALIGN: Alignment Status.(LOA) <Lane>[7] **FEC Errors** FECCORCW: FEC Corrected Codewords. Response: <Count> FECUCORCW: FEC Uncorrected Codewords. Response: <Count> FECSYMERR: FEC Symbol errors. Response: <Count> FECSYMERRRATE: FEC Symbol errors rate. Response: <Count> FECLOFA: Loss of Alignment. Response: <Count> FECMMAP: FEC Lane Marker. <Lane>[7] **Conditions** LOSTEXTREF: External reference 10MHz clock. Response: <Count>,<Ratio> LOSTREF1PPS: External 1PPS . Response: <Count>,<Ratio> LOSTGPS: GPS. Response: <Count>,<Ratio>
Response	{(<result>), }* = <EXPRESSION RESPONSE DATA> Format: Numeric List Each result is formatted according to the specification in the parameter field. Values that are not relevant or applicable for the current measurement return NaN (section 1.6.1).
Examples	`ETH:PORT1:IFET? (TGFR)` → `(6400,0.853)` `ETH:PORT1:IFET? (UFR,BFR)` → `(251923,0.900),(27992,0.100)`
Notes	This command fetches the results from the interval selected using the `MEASurement:SETup:SELect` command (see section 18.2.2). [1] Results are only available when the WAN terminology is SONET. [2] Results are only available at 10G. [3] <ref-port> is set with the command `ETH:PORT1:TFRD:PSEL`

continued on next page. . .

. . . continued from previous page

	[4] Result are only available at MPLS/IP over OTN. [5] Result are only available on the Sync Test application. [6] Result are only available when cross pattern by Frame by Frame is available. [7] <Lane> show the results of each lane. Example: 10GBE → (0), 40GBE → (0,0,0,0) If the requested result is not available, NaN (section 1.6.1) is returned. If there is one or more results, the last ”,” is always removed.

12.14.2 ETHernet:PORT<Pt>:TFETch?

Syntax	ETHernet:PORT<Pt>:TFETch? <parameter>
Description	This query fetches thresholds for a Ethernet interval if available.
Parameters	<Pt> = Port number
	({<parameter>} + {,}∗) = <EXPRESSION PROGRAM DATA> **Performance** UTIL: Utilization. THR: Throughput. **Frame Stat** UFR: Unicast frames. MFR: Multicast frames. BFR: Broadcast frames. PFR: Pause frames. FFR: Fragmented frames. UNFR: Undersized frames. OVFR: Oversized frames. FEFR: FCS errored frames. OVRFCSERR: Oversized & FCS errored frames. IPCHKSUM: IP checksum errored frames. COLL: Collisions. PV: Preamble violations. IV: IFG violations. TEFR: Total errored frames. **Transmit Stat** TFRD: Total frames difference (Tx-Rx<ref-port>)[1]. **BER - Alarms & Errors** BPE: Pattern errors. BSE: Sequence errors. SDMD: Maximum disruption. **Multistream Frame Loss** SFLx: Stream x frame loss. x = Stream number (1-16) **Multistream Latency/Jitter** SLx: Latency (μs). SJx: Jitter (μs). x = Stream number (1-16) **Sync Test** PHASE: 1PPS phase error[2]. FILTERED: Filtered TE[2]. PPS: 1PPS deviation[2]. SYNC: Sync message transmission time[2]. FOLLOW: Followup message transmission time[2]. DELAY: Delay request message transmission time[2].
Response	{(<result>), }* = <EXPRESSION RESPONSE DATA> Format: Numeric List <result> = <NR1 NUMERIC RESPONSE DATA> 0: Pass 1: Fail
Example	`ETH:PORT1:TFET? (TEFR,UTIL,TFRD)` → `(1),(1),(1,0)`

continued on next page. . .

. . . continued from previous page

Note	This query fetches from the interval selected using the `MEASurement:SETup:SELect` command (see section 18.2.2). [1] <ref-port>) is set with the command `ETH:PORT1:TFRD:PSEL` [2] Result are only available on the Sync Test application.

12.14.3 ETHernet:PORT<Pt>:TFRDiffrence:PSELection

Syntax	ETHernet:PORT<Pt>:TFRDiffrence:PSELection <port>
Description	This command sets the Rx port number for total frames difference.
Parameters	<Pt> = Port number
	<port> = <CHARACTER PROGRAM DATA> PORTx: Rx port number x
Response	None.
Example	`ETH:PORT1:TFRD:PSEL PORT1`
Note	

Syntax	ETHernet:PORT<Pt>:TFRDiffrence:PSELection?
Description	This query returns the Rx port number for total frames difference.
Parameter	<Pt> = Port number
Response	<port> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:TFRD:PSEL?` → `PORT1`
Note	

12.15 Status

12.15.1 ETHernet:STATus:PORT<Pt>:AESummary[:EVENt]?

Syntax	ETHernet:STATus:PORT<Pt>:AESummary[:EVENt]?
Description	This query returns the Ethernet alarms and errors summary event register. The content of this event register is summarized in DB6 of the STATus:INTerface:PORT<Pt>:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 (2) = Error summary DB3 (4) = WAN physical and Line/Mux alarm summary [1] DB4 (8) = WAN path/High-order alarm summary [1] DB5 (16) = WAN error summary [1] DB6 (32) = PCS alarm summary [2] DB7 (64) = PCS error summary [2] DB8 (128) = Sync Test alarm summary [3] DB9 (256) = FEC error summary [4] DB10 - DB16 = NOT USED
Example	`ETH:STAT:PORT1:AES?` → `1`
Note	[1] WAN errors/alarms are only valid if WAN is enabled. [2] PCS errors/alarms are only available at 40/100G. [3] Sync Test alarms are only valid if the Sync Test application. [4] FEC errors are only available at 100G(on MU110013A CFP2 or QSFP28 Adpt.).

12.15.2 ETHernet:STATus:PORT<Pt>:AESummary:CONDition?

Syntax	ETHernet:STATus:PORT<Pt>:AESummary:CONDition?
Description	This query returns the Ethernet alarms and errors summary condition register. DB1/DB3/DB4/DB6/DB8 is summary of ETHernet:STATus:PORT<Pt>:ALARm<section>[:EVENt]? DB2/DB5/DB7/DB9 is summary of ETHernet:STATus:PORT<Pt>:ERRor<section>[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> Same as ETHernet:STATus:PORT<Pt>:AESummary[:EVENt]?
Example	`ETH:STAT:PORT1:AES:COND?` → `1`
Note	

12.15.3 ETHernet:STATus:PORT<Pt>:ALARm<section>[:EVENt]?

Syntax	ETHernet:STATus:PORT<Pt>:ALARm<section>[:EVENt]?
Description	This query returns the alarms event register. The content of this register is summarized in DB1/DB3/DB4/DB6/DB8 of the ETHernet:STATus:PORT<Pt>:AESummary:CONDition register.
Parameters	<Pt> = Port number
	<section> = Ethernet(1), WAN Physical and Line/Mux(2), WAN Path/High-order(3), PCS(4), or Sync Test(5)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> <section> = 1: DB1 (1) = No link DB2 (2) = Remote fault DB3 (4) = Threshold(s) exceeded DB4 (8) = BER alarms [1] DB5 (16) = Loss of signal DB6 - DB8 = NOT USED

continued on next page...

. . . continued from previous page

	DB9 (256) = SyncE Timeout alarm DB10 (512) = PTP Synchronization Timeout alarm DB11 (1024) = Timing Source alarm DB12 - DB16 = NOT USED <section> = 2:[2] DB1 (1) = Loss of signal DB2 (2) = Loss of frame DB3 (4) = Severely errored frame/Out of frame DB4 (8) = Section/Mux section - trace identifier mismatch DB5 (16) = Line/Mux section - alarm indication signal DB6 (32) = Line/Mux section - remote defect indicator DB7 - DB16 = NOT USED <section> = 3:[2] DB1 (1) = Path/Administrative unit - alarm indication signal DB2 (2) = Path/Administrative unit - loss of pointer DB3 (4) = Path/High-order path - trace Identifier mismatch DB4 (8) = Path/High-order path - payload label mismatch DB5 (16) = Path/High-order path - unequipped DB6 (32) = Path/High-order path - remote defect indication DB7 (64) = Path - enhanced remote defect indication payload [3] DB8 (128) = Path - enhanced remote defect indication server [3] DB9 (256) = Path - enhanced remote defect indication connectivity [3] DB10 (512) = Loss of code-group delineation DB11 (1024) = Loss of signal synchronization DB12 - DB16 = NOT USED <section> = 4: DB1 (1) = High BER DB2 (2) = Alignment Status [4] DB3 - DB16 = NOT USED <section> = 5:[5] DB1 (1) = Ext.ref. clock DB2 (2) = Ext. ref. 1PPS DB3 (4) = 1PPS DB4 - DB16 = NOT USED
Example	`ETH:STAT:PORT1:ALAR1?` → `1`
Notes	[1] BER Alarms are only valid if BER is enabled. [2] WAN Alarms are only valid if WAN is enabled. [3] Only available when the WAN terminology is SONET. [4] Only available at 40/100G. [5] Only available on Sync Test application.

12.15.4 ETHernet:STATus:PORT<Pt>:ALARm<section>:CONDition?

Syntax	ETHernet:STATus:PORT<Pt>:ALARm<section>:CONDition?
Description	This query returns the alarms condition register.
Parameters	<Pt> = Port number
	<section> = Ethernet(1) , WAN Physical and Line/Mux(2), WAN Path/High-order(3), PCS(4), or Sync Test(5)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> Same as ETHernet:STATus:PORT<Pt>:ALARm<section>[:EVENt]?
Example	`ETH:STAT:PORT1:ALAR1:COND?` → `1`
Note	

12.15.5 ETHernet:STATus:PORT<Pt>:ERRor<section>[:EVENt]?

Syntax	ETHernet:STATus:PORT<Pt>:ERRor<section>[:EVENt]?
Description	This query returns the errors event register. The content of this register is summarized in DB2/DB5/DB7/DB9 of the ETHernet:STATus:PORT<Pt>:AESummary:CONDition register.
Parameters	<Pt> = Port number
	<section> = Ethernet(1), WAN(2), PCS(3) or FEC(4)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> <section> = 1: DB1 (1) = Errored frames DB2 (2) = Multistream frame loss [1] DB3 (4) = Bit errors DB4 - DB16 = NOT USED <section> = 2:[2] DB1 (1) = A1A2 DB2 (2) = B1 DB3 (4) = B2 DB4 (8) = Line/Mux section REI DB5 (16) = B3 DB6 (32) = Path/High-order path REI DB7 (64) = STS-192C/AU4-64C positive DB8 (128) = STS-192C/AU4-64C negative DB9 - DB16 = NOT USED <section> = 3:[3] DB1 (1) = Invalid Sync header DB2 (2) = Invalid Align Marker DB3 (4) = Invalid Block DB4 (8) = BIP Error DB5 - DB16 = NOT USED <section> = 4:[4] DB1 (1) = LOFA DB2 (2) = FEC Corrected Codewords DB3 (4) = FEC Uncorrected Codewords DB4 (8) = FEC Symbol Errors DB5 (16) = LOFAML DB6 - DB16 = NOT USED
Example	`ETH:STAT:PORT1:ERR1?` → `1`
Notes	[1] Multistream frame loss is only valid when multistream frame loss is enabled and a measurement is running. [2] WAN errors are only valid if WAN is enabled. [3] Only available at 40/100G. [3] Only available at 100G(on MU110013A CFP2 or QSFP28 Adpt.).

12.15.6 ETHernet:STATus:PORT<Pt>:ERRor<section>:CONDition?

Syntax	ETHernet:STATus:PORT<Pt>:ERRor<section>:CONDition?
Description	This query returns the errors condition register.
Parameters	<Pt> = Port number
	<section> = Ethernet(1), WAN(2), PCS(3), FEC(4)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> Same as ETHernet:STATus:PORT<Pt>:ERRor<section>[:EVENt]?
Example	`ETH:STAT:PORT1:ERR1:COND?` → `1`
Note	

12.15.7 ETHernet:STATus:PORT<Pt>:LINK?

Syntax	ETHernet:STATus:PORT<Pt>:LINK?
Description	This query returns if there is link.
Parameter	<Pt> = Port number
Response	<link> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:LINK?` → `1`
Note	

12.15.8 ETHernet:STATus:PORT<Pt>:INTerface?

Syntax	ETHernet:STATus:PORT<Pt>:INTerface?
Description	This query returns the current type of interface.
Parameter	<Pt> = Port number
Response	<interface> = <STRING RESPONSE DATA> "ELECTRICAL" "OPTICAL" "N/A": No link
Example	`ETH:STAT:PORT1:INT?` → `"ELECTRICAL"`
Note	

12.15.9 ETHernet:STATus:PORT<Pt>:FRAMes?

Syntax	ETHernet:STATus:PORT<Pt>:FRAMes?
Description	This query returns if frames are present.
Parameter	<Pt> = Port number
Response	<frames> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:FRAM?` → `1`
Note	

12.15.10 ETHernet:STATus:PORT<Pt>:MPLS?

Syntax	ETHernet:STATus:PORT<Pt>:MPLS?
Description	This query returns if any frames with MPLS and Ethernet over MPLS are present.
Parameter	<Pt> = Port number
Response	<mpls> = <NR1 NUMERIC RESPONSE DATA>
	<eompls> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:MPLS?` → `1,1`
Note	

12.15.11 ETHernet:STATus:PORT<Pt>:VLAN?

Syntax	ETHernet:STATus:PORT<Pt>:VLAN?
Description	This query returns if any frames with VLAN are present.
Parameter	<Pt> = Port number
Response	<vlan> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:VLAN?` → `1`
Note	

12.15.12 ETHernet:STATus:PORT<Pt>:SPeed?

Syntax	ETHernet:STATus:PORT<Pt>:SPeed?
Description	This query returns the current speed. Unit: Mbps.
Parameter	<Pt> = Port number
Response	<speed> = <NR1 NUMERIC RESPONSE DATA> 0 means no link.
Example	`ETH:STAT:PORT1:SP?` → `100`
Note	

12.15.13 ETHernet:STATus:PORT<Pt>:DUPLex?

Syntax	ETHernet:STATus:PORT<Pt>:DUPLex?
Description	This query returns the current duplex type.
Parameter	<Pt> = Port number
Response	<duplex> = <STRING RESPONSE DATA> "HDX": half duplex "FDX": full duplex "N/A": No link
Example	`ETH:STAT:PORT1:DUPL?` → `"HDX"`
Note	

12.15.14 ETHernet:STATus:PORT<Pt>:MDI?

Syntax	ETHernet:STATus:PORT<Pt>:MDI?
Description	This query returns the current MDI type.
Parameter	<Pt> = Port number
Response	<mdi> = <STRING RESPONSE DATA> "MDI" "MDIX" "N/A": No link
Example	`ETH:STAT:PORT1:MDI?` → `"MDI"`
Note	

12.15.15 ETHernet:STATus:PORT<Pt>:L10G?

Syntax	ETHernet:STATus:PORT<Pt>:L10G?
Description	Obsolete. For CMA 3000 backward compatibility only. Same as ETHernet:STATus:PORT<Pt>:LFS?
Parameter	<Pt> = Port number
Response	<lf> = <BOOLEAN RESPONSE DATA>
	<rf> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:L10G?` → `1, 0`
Note	

12.15.16 ETHernet:STATus:PORT<Pt>:LFS?

Syntax	ETHernet:STATus:PORT<Pt>:LFS?
Description	This query returns if any Local faults or Remote faults are present.
Parameter	<Pt> = Port number
Response	<lf> = <BOOLEAN RESPONSE DATA>
	<rf> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:LFS?` → `1, 0`
Note	

12.15.17 ETHernet:STATus:PORT<Pt>:ANComplete?

Syntax	ETHernet:STATus:PORT<Pt>:ANComplete?
Description	This query returns if auto negotiation is completed.
Parameter	<Pt> = Port number
Response	<ancomp> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:ANC?` → `1`
Note	

12.15.18 ETHernet:STATus:PORT<Pt>:PCAPable?

Syntax	ETHernet:STATus:PORT<Pt>:PCAPable?
Description	This query returns the Link Partner Ability - Pause Capable
Parameter	<Pt> = Port number
Response	<pcable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:PCAP?` → `1`
Note	ETHernet:STATus:PORT<Pt>:ANComplete? must be 1 before this query is valid.

12.15.19 ETHernet:STATus:PORT<Pt>:APRequest?

Syntax	ETHernet:STATus:PORT<Pt>:APRequest?
Description	This query returns the Link Partner Abilities - Asymmetric Pause Request
Parameter	<Pt> = Port number
Response	<apreq> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:APR?` → `1`
Note	ETHernet:STATus:PORT<Pt>:ANComplete? must be 1 before this query is valid.

12.15.20 ETHernet:STATus:PORT<Pt>:RFAult?

Syntax	ETHernet:STATus:PORT<Pt>:RFAult?
Description	This query returns the Link Partner Abilities - Remote Fault
Parameter	<Pt> = Port number
Response	<rfault> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:RFA?` → `1`
Note	ETHernet:STATus:PORT<Pt>:ANComplete? must be 1 before this query is valid.

12.15.21 ETHernet:STATus:PORT<Pt>:LCLock?

Syntax	ETHernet:STATus:PORT<Pt>:LCLock?
Description	This query returns the local clock status at electrical gigabit connection.
Parameter	<Pt> = Port number
Response	<interface> = <STRING RESPONSE DATA> "Slave" "Master" "Fault" "N/A": No link or not electrical gigabit connection.
Example	`ETH:STAT:PORT1:LCL?` → `"Master"`
Notes	ETHernet:STATus:PORT<Pt>:ANComplete? must be 1 before this query is valid.

12.15.22 ETHernet:STATus:PORT<Pt>:SADuplex?

Syntax	ETHernet:STATus:PORT<Pt>:SADuplex?
Description	This query returns the Link partner ability - speed and duplex
Parameter	<Pt> = Port number
Response	{(<sad>),}* = <EXPRESSION RESPONSE DATA> 10MH: 10Mbps half duplex 10MF: 10Mbps full duplex 100MH: 100Mbps half duplex 100MF: 100Mbps full duplex 1GH: 1Gbps half duplex 1GF: 1Gbps full duplex
Example	`ETH:STAT:PORT1:SAD?` → `(10MH,10MF,100MH,100MF)`
Note	ETHernet:STATus:PORT<Pt>:ANComplete? must be 1 before this query is valid.

12.15.23 ETHernet:STATus:PORT<Pt>:UTILization?

Syntax	ETHernet:STATus:PORT<Pt>:UTILization?
Description	This query returns the current utilization. Unit: %.
Parameter	<Pt> = Port number
Response	<util> = <NR2 NUMERIC RESPONSE DATA>
	<threshold> = <BOOLEAN RESPONSE DATA> 1: Threshold exceeded 0: Threshold not exceeded / disabled
Example	`ETH:STAT:PORT1:UTIL?` → `97.1, 0`
Note	

12.15.24 ETHernet:STATus:PORT<Pt>:THRoughput?

Syntax	ETHernet:STATus:PORT<Pt>:THRoughput?
Description	This query returns the current throughput. Unit: bps.
Parameter	<Pt> = Port number
Response	<thr> = <NR2 NUMERIC RESPONSE DATA>
	<threshold> = <BOOLEAN RESPONSE DATA> 1: Threshold exceeded 0: Threshold not exceeded / disabled
Example	`ETH:STAT:PORT1:THR?` → `97100000, 0`
Note	

12.15.25 ETHernet:STATus:PORT<Pt>:EFRames?

Syntax	ETHernet:STATus:PORT<Pt>:EFRames?
Description	This query returns the current amount of errored frames. Unit: %.
Parameter	<Pt> = Port number
Response	<errored> = <NR2 NUMERIC RESPONSE DATA>
	<threshold> = <BOOLEAN RESPONSE DATA> 1: Threshold exceeded 0: Threshold not exceeded / disabled
Example	`ETH:STAT:PORT1:EFR?` → `0.5, 0`
Note	

12.15.26 ETHernet:STATus:PORT<Pt>:PCS:SHLock?

Syntax	ETHernet:STATus:PORT<Pt>:PCS:SHLock?
Description	This query returns whether or not 40/100G PCS sync header is locked.
Parameter	<Pt> = Port number
Response	<shlock> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:PCS:SHL?` → `1`
Note	This command can be used on V2.00 or later.

12.15.27 ETHernet:STATus:PORT<Pt>:PCS:AMLock?

Syntax	ETHernet:STATus:PORT<Pt>:PCS:AMLock?
Description	This query returns whether or not 40/100G PCS alignment marker is locked.
Parameter	<Pt> = Port number
Response	<shlock> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:PCS:AML?` → `1`
Note	This command can be used on V2.00 or later.

12.15.28 ETHernet:STATus:PORT<Pt>:TIMing?

Syntax	ETHernet:STATus:PORT<Pt>:TIMing?
Description	This query returns the timing status.
Parameter	<Pt> = Port number
Response	<bitRate> = <NR1 NUMERIC RESPONSE DATA> Unit: bps.
	<bitRateDeviation> = <NR1 NUMERIC RESPONSE DATA> Unit: ppb.
Example	`ETH:STAT:PORT1:TIM?` → `10000001000,100`
Note	

12.15.29 ETHernet:STATus:PORT<Pt>:TIMing:SOURce?

Syntax	ETHernet:STATus:PORT<Pt>:TIMing:SOURce?
Description	This query returns the status of the timing source.
Parameter	<Pt> = Port number
Response	<timingSrcProblem> = <BOOLEAN RESPONSE DATA> `1` is returned if the timing signal is absent.
Example	`ETH:STAT:PORT1:TIM:SOUR?` → `0`
Note	

12.15.30 ETHernet:STATus:PORT<Pt>:SYNCe?

Syntax	ETHernet:STATus:PORT<Pt>:SYNCe?
Description	This query returns the status of the SyncE.
Parameter	<Pt> = Port number
Response	<qualityLevel> = <NR1 NUMERIC RESPONSE DATA>
	<qlTimeout> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:SYNC?` → `15,0`
Note	

12.15.31 ETHernet:STATus:PORT<Pt>:PTP:LCLock?

Syntax	ETHernet:STATus:PORT<Pt>:PTP:LCLock?
Description	This query returns the status of the PTP clock.
Parameter	<Pt> = Port number
Response	<clockState> = <STRING RESPONSE DATA> "INITIALIZING", "FAULTY", "DISABLED", "LISTENING", "PRE_MASTER", "MASTER", "PASSIVE", "UNCALIBRATED" or "SLAVE".
	<offset> = <NR2 NUMERIC RESPONSE DATA> Unit: Seconds.
	<meanPathDelay> = <NR2 NUMERIC RESPONSE DATA> Unit: Seconds.
	<delayAsymmetry> = <NR2 NUMERIC RESPONSE DATA> Unit: Seconds.
Example	`ETH:STAT:PORT1:PTP:LCL?` → `"Listening",0.001002003,0.123456789,0.000000100`
Note	

12.15.32 ETHernet:STATus:PORT<Pt>:PTP:WCLock?

Syntax	ETHernet:STATus:PORT<Pt>:PTP:WCLock?
Description	This query returns the status of the PTP wall clock.
Parameter	<Pt> = Port number (1-2)
Response	<dateTime> = <STRING RESPONSE DATA> Format: "YYYY-MM-DDThh:mm:ss"
	<offset> = <NR1 NUMERIC RESPONSE DATA> Offset from UTC time given by an attached GPS receiver. Unit: Nano seconds.
Example	`ETH:STAT:PORT1:PTP:WCL?` → `"2011-09-16T15:16:17",2345`
Note	

12.15.33 ETHernet:STATus:PORT<Pt>:PTP:PCLock?

Syntax	ETHernet:STATus:PORT<Pt>:PTP:PCLock?
Description	This query returns the properties of the PTP parent clock.
Parameter	<Pt> = Port number
Response	<parentId> = <STRING RESPONSE DATA>
	<parentPort> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:STAT:PORT1:PTP:PCL?` → `"00-00-00-00-00-00-00-00",0`
Note	

12.15.34 ETHernet:STATus:PORT<Pt>:PTP:GCLock?

Syntax	ETHernet:STATus:PORT<Pt>:PTP:GCLock?
Description	This query returns the properties of the grandmaster clock.
Parameter	<Pt> = Port number
Response	<clockId> = <STRING RESPONSE DATA>
	<class> = <NR1 NUMERIC RESPONSE DATA>
	<accuracy> = <NR1 NUMERIC RESPONSE DATA>
	<logVarianceOffsetAnnounced> = <NR3 NUMERIC RESPONSE DATA>
	<logVarianceOffsetObserved> = <NR3 NUMERIC RESPONSE DATA>
	<priority1> = <NR1 NUMERIC RESPONSE DATA>
	<priority2> = <NR1 NUMERIC RESPONSE DATA>
	<logVarianceOffsetAnnouncedRaw> = <NR1 NUMERIC RESPONSE DATA>
	<stepsRemoved> = <NR1 NUMERIC RESPONSE DATA>
	<timeSource> = <NR1 NUMERIC RESPONSE DATA>
	<utcOffset> = <NR1 NUMERIC RESPONSE DATA>
	<flagField> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:STAT:PORT1:PTP:GCL?` → `"00-00-00-00-00-00-00-00",255,254,1.23E-16,2.34E-16,255,255,22564,0,160,37,#H0000`
Note	

12.15.35 ETHernet:STATus:PORT<Pt>:PTP:FMASters?

Syntax	ETHernet:STATus:PORT<Pt>:PTP:FMASters?
Description	This query returns a list of maximum five foreign masters and their properties.
Parameter	<Pt> = Port number
Response	{(<master>), }* = <EXPRESSION RESPONSE DATA> <master> is split into three separate results (<clockId>,<port>,<announceCount>):
	<clockId> = <STRING RESPONSE DATA>
	<port> = <NR1 NUMERIC RESPONSE DATA>
	<announceCount> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:STAT:PORT1:PTP:FMAS?` → `("00-00-00-00-00-00-00-11",111,99), ("00-00-00-00-00-00-00-22",222,88)`
Note	If the list is empty an execution error will be reported.

12.15.36 ETHernet:STATus:PORT<Pt>:WAN:CAPTure:SOH:TRACe?

Syntax	ETHernet:STATus:PORT<Pt>:WAN:CAPTure:SOH:TRACe?
Description	This query returns the section overhead trace (J0) for the latest captured frames.
Parameter	<Pt> = Port number
Response	<J0_trace> = <STRING RESPONSE DATA>
Example	`ETH:STAT:PORT1:WAN:CAPT:SOH:TRAC?` → `"Message_Test_J0"`
Notes	If one of the alarms LOS or LOF is present, an empty string is returned. New frames are captured every second. This command can be used on V2.00 or later.

12.15.37 ETHernet:STATus:PORT<Pt>:WAN:CAPTure<Frame>:SOH?

Syntax	ETHernet:STATus:PORT<Pt>:WAN:CAPTure<Frame>:SOH? <SOH-byte>
Description	This query returns the section overhead bytes from the selected frame.
Parameters	<Pt> = Port number
	<Frame> = Frame number (1-64)
	<SOH-byte> = <CHARACTER PROGRAM DATA> A1: Returns 3 bytes. A2: Returns 3 bytes. J0: Returns 3 bytes. B1: Returns 3 bytes. E1: Returns 3 bytes. F1: Returns 3 bytes. D1: Returns 3 bytes. D2: Returns 3 bytes. D3: Returns 3 bytes. H1: Returns 3 bytes. H2: Returns 3 bytes. H3: Returns 3 bytes. B2: Returns 3 bytes. K1: Returns 3 bytes. K2: Returns 3 bytes. D4: Returns 3 bytes. D5: Returns 3 bytes. D6: Returns 3 bytes. D7: Returns 3 bytes. D8: Returns 3 bytes. D9: Returns 3 bytes. D10: Returns 3 bytes. D11: Returns 3 bytes. D12: Returns 3 bytes. S1: Returns 3 bytes.

continued on next page...

. . . continued from previous page

	Z2: Returns 3 bytes. E2: Returns 3 bytes. M0: Returns 1 byte. M1: Returns 1 byte.
Response	<byte1>[,<byte2>[,<byte3>]] = <HEXADECIMAL NUMERIC RESPONSE DATA> Refer to <SOH-byte> parameter description above to see how many bytes this command returns.

Examples	`ETH:STAT:PORT1:WAN:CAPT64:SOH? A1` → `#HF6,#HF6,#HF6`
	`ETH:STAT:PORT1:WAN:CAPT23:SOH? H1` → `#H69,#H93,#H93`
	`ETH:STAT:PORT1:WAN:CAPT1:SOH? M1` → `#H00`

Notes	If one of the alarms LOS or LOF is present NaN (section 1.6.1) is returned. 64 new frames are captured every second. This command can be used on V2.00 or later.

12.15.38 ETHernet:STATus:PORT<Pt>:WAN:CAPTure:POH:TRACe?

Syntax	ETHernet:STATus:PORT<Pt>:WAN:CAPTure:POH:TRACe?
Description	This query returns the VC4 path overhead trace (J1) for the latest captured frames.
Parameter	<Pt> = Port number
Response	<J1_trace> = <STRING RESPONSE DATA>
Example	`ETH:STAT:PORT1:WAN:CAPT:POH:TRAC?` → `"Message_Test_J1"`
Notes	If one of the alarms LOS, LOF, UNEQ or LOP is present, an empty string is returned. New frames are captured every second. This command can be used on V2.00 or later.

12.15.39 ETHernet:STATus:PORT<Pt>:WAN:CAPTure<Frame>:POH?

Syntax	ETHernet:STATus:PORT<Pt>:WAN:CAPTure<Frame>:POH? <POH-byte>
Description	This query returns the VC4 path overhead bytes from the selected frame.
Parameters	<Pt> = Port number
	<Frame> = Frame number (1-64)
	<POH-byte> = <CHARACTER PROGRAM DATA> J1: J1 byte. B3: B3 byte. C2: C2 byte. G1: G1 byte. F2: F2 byte. H4: H4 byte. F3: F3 byte. K3: K3 byte. N1: N1 byte.
Response	<byte> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:STAT:PORT1:WAN:CAPT1:POH? H4` → `#HFF`
Notes	If one of the alarms LOS, LOF, UNEQ or LOP is present NaN (section 1.6.1) is returned. 64 new frames are captured every second. This command can be used on V2.00 or later.

12.15.40 ETHernet:STATus:PORT<Pt>:TSRC:R10M?

Syntax	ETHernet:STATus:PORT<Pt>:TSRC:R10M?
Description	This query returns if external reference 10MHz is available for timestamp source.
Parameter	<Pt> = Port number
Response	<timestamp source availability> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:TSRC:R10M?` → `1`
Note	This command is available if both of the following conditions are met: - ETHernet:PORT<Pt>:STReam<St>:LATency, ETHernet:PORT<Pt>:STReam<St>:JITTer, or ETHernet:PORT<Pt>:STReam1:BER:SDMeasure[:ENABled] are enabled. - ETHernet:PORT<Pt>:STReam:TSTamp:TSRC is set to R1PPS or RGPS.

12.15.41 ETHernet:STATus:PORT<Pt>:TSRC:RPPS?

Syntax	ETHernet:STATus:PORT<Pt>:TSRC:RPPS?
Description	This query returns if external reference 1PPS is available for timestamp source.
Parameter	<Pt> = Port number
Response	<timestamp source availability> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:TSRC:RPPS?` → `1`
Note	This command is available if both of the following conditions are met: - ETHernet:PORT<Pt>:STReam<St>:LATency, ETHernet:PORT<Pt>:STReam<St>:JITTer, or ETHernet:PORT<Pt>:STReam1:BER:SDMeasure[:ENABled] are enabled. - ETHernet:PORT<Pt>:STReam:TSTamp:TSRC is set to R1PPS or RGPS.

12.15.42 ETHernet:STATus:PORT<Pt>:TSRC:GPS?

Syntax	ETHernet:STATus:PORT<Pt>:TSRC:GPS?
Description	This query returns if GPS is available for timestamp source.
Parameter	<Pt> = Port number
Response	<timestamp source availability> = <BOOLEAN RESPONSE DATA>
Example	`ETH:STAT:PORT1:TSRC:GPS?` → `1`
Note	This command is available if both of the following conditions are met: - ETHernet:PORT<Pt>:STReam<St>:LATency, ETHernet:PORT<Pt>:STReam<St>:JITTer, or ETHernet:PORT<Pt>:STReam1:BER:SDMeasure[:ENABled] are enabled. - ETHernet:PORT<Pt>:STReam:TSTamp:TSRC is set to GPS or RGPS.

12.16 Ping

12.16.1 ETHernet:PING:STARt

Syntax	ETHernet:PING:STARt
Description	This command starts the Ping test.
Parameter	None.
Response	None.
Example	`ETH:PING:STAR`
Note	

12.16.2 ETHernet:PING:STOP

Syntax	ETHernet:PING:STOP
Description	This command stops the Ping test.
Parameter	None.
Response	None.
Example	`ETH:PING:STOP`
Note	

12.16.3 ETHernet:PING:SETup:PORT<Pt>:DMODe

Syntax	ETHernet:PING:SETup:PORT<Pt>:DMODe <mode>
Description	This command sets the test mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> CONTinuous SEConds REQuests *DEFault = REQuests*
Response	None.
Example	`ETH:PING:SET:PORT1:DMOD SEC`
Note	This setting applies to all ports.

Syntax	ETHernet:PING:SETup:PORT<Pt>:DMODe?
Description	This query returns the test mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PING:SET:PORT1:DMOD?` → `SEC`
Note	This setting applies to all ports.

12.16.4 ETHernet:PING:SETup:PORT<Pt>:SDURation

Syntax	ETHernet:PING:SETup:PORT<Pt>:SDURation <seconds>
Description	This command sets the test duration. Unit: Seconds.
Parameters	<Pt> = Port number
	<seconds> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 2000000, DEFault = 30*
Response	None.
Example	`ETH:PING:SET:PORT1:SDUR 5`
Note	`DMODe` must be set to `SEConds` for this command to have effect. This setting applies to all ports.

Syntax	ETHernet:PING:SETup:PORT<Pt>:SDURation?
Description	This query returns the test duration. Unit: Seconds.
Parameter	<Pt> = Port number
Response	<seconds> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PING:SET:PORT1:SDUR?` → `5`
Note	This setting applies to all ports.

12.16.5 ETHernet:PING:SETup:PORT<Pt>:RDURation

Syntax	ETHernet:PING:SETup:PORT<Pt>:RDURation <requests>
Description	This command sets the test duration. Unit: Requests.
Parameters	<Pt> = Port number
	<requests> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 65000, DEFault = 10*
Response	None.
Example	`ETH:PING:SET:PORT1:RDUR 5`
Note	`DMODe` must be set to `REQuests` for this command to have effect. This setting applies to all ports.

Syntax	ETHernet:PING:SETup:PORT<Pt>:RDURation?
Description	This query returns the test duration. Unit: Requests.
Parameter	<Pt> = Port number
Response	<requests> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PING:SET:PORT1:RDUR?` → `5`
Note	This setting applies to all ports.

12.16.6 ETHernet:PING:SETup:PORT<Pt>:INTerval

Syntax	ETHernet:PING:SETup:PORT<Pt>:INTerval <interval>
Description	This command sets the requests interval. Unit: Seconds.
Parameters	<Pt> = Port number
	<interval> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 60, DEFault = 4*
Response	None.
Example	`ETH:PING:SET:PORT1:INT 5`
Note	0 means that request are send as fast as possible. This setting applies to all ports.

Syntax	ETHernet:PING:SETup:PORT<Pt>:INTerval?
Description	This query returns the requests interval. Unit: Seconds.
Parameter	<Pt> = Port number
Response	<seconds> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PING:SET:PORT1:INT?` → `5`
Note	This setting applies to all ports.

12.16.7 ETHernet:PING:SETup:PORT<Pt>:FSIZe

Syntax	ETHernet:PING:SETup:PORT<Pt>:FSIZe <size>
Description	This command sets the frame size. Unit: Bytes.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 70*
Response	None.
Example	`ETH:PING:SET:PORT1:FSIZ 100`
Notes	[1] The minimum allowed frame size varies depending on the stream frame setup. This setting applies to all ports.

Syntax	ETHernet:PING:SETup:PORT<Pt>:FSIZe?
Description	This query returns the frame size. Unit: Bytes.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PING:SET:PORT1:FSIZ?` → `100`
Note	This setting applies to all ports.

12.16.8 ETHernet:PING:SETup:PORT<Pt>:TOUT

Syntax	ETHernet:PING:SETup:PORT<Pt>:TOUT <timeout>
Description	This command sets the time out value. Unit: Milliseconds.
Parameters	<Pt> = Port number
	<timeout> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 15000, DEFault = 500*
Response	None.
Example	`ETH:PING:SET:PORT1:TOUT 100`
Note	This setting applies to all ports.

Syntax	ETHernet:PING:SETup:PORT<Pt>:TOUT?
Description	This query returns the time out value. Unit: Milliseconds.
Parameter	<Pt> = Port number
Response	<timeout> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PING:SET:PORT1:TOUT?` → `100`
Note	This setting applies to all ports.

12.16.9 ETHernet:PING:RESults:SUMMary?

Syntax	ETHernet:PING:RESults:SUMMary?
Description	This query returns the result summary.
Parameter	None.
Response	<sent> = <NR1 NUMERIC RESPONSE DATA> Number of sent requests.
	<received> = <NR1 NUMERIC RESPONSE DATA> Number of received responses.
	<lost> = <NR1 NUMERIC RESPONSE DATA> Number of request time outs.
Example	`ETH:PING:RES:SUMM?` → `10,7,3`
Note	

12.16.10 ETHernet:PING:RESults:RTT?

Syntax	ETHernet:PING:RESults:RTT?
Description	Returns round trip times. Unit: Milliseconds.
Parameter	None.
Response	<min> = <NR2 NUMERIC RESPONSE DATA> Minimum round trip time.
	<max> = <NR2 NUMERIC RESPONSE DATA> Maximum round trip time.
	<avg> = <NR2 NUMERIC RESPONSE DATA> Average round trip time.
Example	`ETH:PING:RES:RTT?` → `0.075,2.913,0.484`
Note	

12.16.11 ETHernet:PING:RESults:NREQuests?

Syntax	ETHernet:PING:RESults:NREQuests?
Description	Returns the number of sent requests.
Parameter	None.
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PING:RES:NREQ?` → `10`
Note	

12.16.12 ETHernet:PING:RESults:SREQuest?

Syntax	ETHernet:PING:RESults:SREQuest? <index>
Description	Returns the result of a single request.
Parameter	{(<index>), }* = <EXPRESSION PROGRAM DATA> Expression format: Numeric List *MINimum = 1*
Response	{(<result>), }* = <EXPRESSION RESPONSE DATA> <result> is split into 2 separate results (<rtt>,<seq-nr>):
	<rtt> = <NR2 NUMERIC RESPONSE DATA> Round trip time. Unit: Milliseconds. NaN (section 1.6.1) means time out.
	<seq-nr> = <NR1 NUMERIC RESPONSE DATA> Request sequence number.
Example	`ETH:PING:RES:SREQ? (1,3:5)` → `(2.913,1),(1.236,3),(3.528,4),(0.879,5)`
Notes	Use the `NREQuests` command above to get the maximum allowed index. Results are returned in the order they are entered in the <index> parameter.

12.17 Traceroute

12.17.1 ETHernet:TRACeroute:STARt

Syntax	ETHernet:TRACeroute:STARt
Description	This command starts the Traceroute test.
Parameter	None.
Response	None.
Example	`ETH:TRAC:STAR`
Note	

12.17.2 ETHernet:TRACeroute:STOP

Syntax	ETHernet:TRACeroute:STOP
Description	This command stops the Traceroute test.
Parameter	None.
Response	None.
Example	`ETH:TRAC:STOP`
Note	

12.17.3 ETHernet:TRACeroute:SETup:NATTempts

Syntax	ETHernet:TRACeroute:SETup:NATTempts <attempts>
Description	This command sets the number of attempts.
Parameter	<attempts> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 1000000, DEFault = 3*
Response	None.
Example	`ETH:TRAC:SET:NATT 5`
Note	

Syntax	ETHernet:TRACeroute:SETup:NATTempts?
Description	This query returns the number of attempts.
Parameter	None.
Response	<attempts> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TRAC:SET:NATT?` → `5`
Note	

12.17.4 ETHernet:TRACeroute:SETup:MNHops

Syntax	ETHernet:TRACeroute:SETup:MNHops <hops>
Description	This command sets the maximum number of hops.
Parameter	<hops> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 255, DEFault = 30*
Response	None.
Example	`ETH:TRAC:SET:MNH 5`
Note	

Syntax	ETHernet:TRACeroute:SETup:MNHops?
Description	This query returns the maximum number of hops.
Parameter	None.
Response	<hops> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TRAC:SET:MNH?` → `5`
Note	

12.17.5 ETHernet:TRACeroute:SETup:TOUT

Syntax	ETHernet:TRACeroute:SETup:TOUT <timeout>
Description	This command sets the time out value. Unit: Milliseconds.
Parameter	<timeout> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 15000, DEFault = 500*
Response	None.
Example	`ETH:TRAC:SET:TOUT 100`
Note	

Syntax	ETHernet:TRACeroute:SETup:TOUT?
Description	This query returns the time out value. Unit: Milliseconds.
Parameter	None.
Response	<timeout> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TRAC:SET:TOUT?` → `100`
Note	

12.17.6 ETHernet:TRACeroute:SETup:NTPHosts

Syntax	ETHernet:TRACeroute:SETup:NTPHosts <pings>
Description	This command sets the number of times to ping a host.
Parameter	<pings> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1000000, DEFault = 3*
Response	None.
Example	`ETH:TRAC:SET:NTPH 5`
Note	

Syntax	ETHernet:TRACeroute:SETup:NTPHosts?
Description	This query returns the number of times to ping a host.
Parameter	None.
Response	<pings> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TRAC:SET:NTPH?` → `5`
Note	

12.17.7 ETHernet:TRACeroute:RESults:NHOPs?

Syntax	ETHernet:TRACeroute:RESults:NHOPs?
Description	Returns the number of found hops.
Parameter	None.
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TRAC:RES:NHOP?` → `10`
Note	

12.17.8 ETHernet:TRACeroute:RESults:HOP?

Syntax	ETHernet:TRACeroute:RESults:HOP? <index>
Description	Returns the result from a single hop.
Parameter	{(<index>), }* = <EXPRESSION PROGRAM DATA> Expression format: Numeric List *MINimum = 1*
Response	{(<result>), }* = <EXPRESSION RESPONSE DATA> <result> is split into 5 separate results (<host>,<minrtt>,<maxrtt>,<avgrtt>,<timeouts>): <host> = <STRING RESPONSE DATA> Host IP address. "(d)" at the end of the string means that this is the destination.

continued on next page...

. . . continued from previous page

	”<timeout>” means that an IP address was unattainable. <minrtt> = <NR2 NUMERIC RESPONSE DATA> Minimum Round trip time. Unit: Milliseconds. NaN (section 1.6.1) means no ping replies. <maxrtt> = <NR2 NUMERIC RESPONSE DATA> Maximum Round trip time. Unit: Milliseconds. NaN (section 1.6.1) means no ping replies. <avgrtt> = <NR2 NUMERIC RESPONSE DATA> Average Round trip time. Unit: Milliseconds. NaN (section 1.6.1) means no ping replies. <timeouts> = <NR1 NUMERIC RESPONSE DATA> Number of ping timeouts. NaN (section 1.6.1) means no ping requests sent.
Example	`ETH:TRAC:RES:HOP? (1:2)` → `("192.168.1.1",0.7888,0.8278,0.8078,0),` `("192.168.2.1(d)",0.5263,0.6689,0.591667,0)`
Notes	Use `NHOPs` to get the maximum allowed index. Results are return in the order they are entered in the <index> parameter.

12.18 RFC2544

12.18.1 ETHernet:RFC:STARt

Syntax	ETHernet:RFC:STARt
Description	This command starts the RFC2544 test.
Parameter	None.
Response	None.
Example	`ETH:RFC:STAR`
Note	

12.18.2 ETHernet:RFC:STOP

Syntax	ETHernet:RFC:STOP
Description	This command stops the RFC2544 test.
Parameter	None.
Response	None.
Example	`ETH:RFC:STOP`
Note	

12.18.3 ETHernet:RFC:SETup:GENeral:MODE

Syntax	ETHernet:RFC:SETup:GENeral:MODE <mode>
Description	This command sets the RFC2544 test mode.
Parameter	<mode> = <CHARACTER PROGRAM DATA> SRouter: Switch/Router RLatency: Router latency SENetwork: Single ended network E2End: End to end network *DEFault = SR*
Response	None.
Examples	`ETH:RFC:SET:GEN:MODE SR` `ETH:RFC:SET:GEN:MODE E2E`
Note	

Syntax	ETHernet:RFC:SETup:GENeral:MODE?
Description	This query returns the RFC2544 test mode.
Parameter	None.
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:GEN:MODE?` → `SEN`
Note	

12.18.4 ETHernet:RFC:SETup:GENeral:TSELection

Syntax	ETHernet:RFC:SETup:GENeral:TSELection <test>
Description	This command sets the RFC2544 tests to be executed.
Parameter	({<test>} ∗ {,}∗) = <EXPRESSION PROGRAM DATA> THR: Throughput FLOS: Frame loss TAFL: Throughput and Frame loss LAT: Latency BURS: Burst
Response	None.
Example	`ETH:RFC:SET:GEN:TSEL (FLOS,LAT,BURS)`
Note	THR and/or FLOS can not be selected with TAFL. Setting no parameters will clear the test selection.

Syntax	ETHernet:RFC:SETup:GENeral:TSELection?
Description	This query returns the RFC2544 test that will be executed.
Parameter	None.
Response	{(<test>),}* = <EXPRESSION RESPONSE DATA>
Example	`ETH:RFC:SET:GEN:TSEL?` → `(FLOS,LAT,BUR)`
Note	

12.18.5 ETHernet:RFC:SETup:GENeral:E2E:OWAY

Syntax	ETHernet:RFC:SETup:GENeral:E2E:OWAY <enable>
Description	This command sets One Way testing for End to End network test.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:RFC:SET:GEN:E2E:OWAY ON`
Note	

Syntax	ETHernet:RFC:SETup:GENeral:E2E:OWAY?
Description	This query returns the state of End to End One Way testing.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:GEN:E2E:OWAY?` → `1`
Note	

12.18.6 ETHernet:RFC:SETup:GENeral:E2E:ADDRess

Syntax	ETHernet:RFC:SETup:GENeral:E2E:ADDRess <enable>
Description	This command enables/disables use of master source addresses for destination on slave side for End to End Network test.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:RFC:SET:GEN:E2E:ADDR ON`
Note	

Syntax	ETHernet:RFC:SETup:GENeral:E2E:ADDRess?
Description	This query returns whether or not master source addresses are used for destination on slave side.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:GEN:E2E:ADDR?` → `1`
Note	

12.18.7 ETHernet:RFC:SETup:GENeral:E2E:TDIRection

Syntax	ETHernet:RFC:SETup:GENeral:E2E:TDIRection <dir>
Description	This command sets the End to End One Way transmission direction.
Parameter	<dir> = <CHARACTER PROGRAM DATA> MASTer: Master to slave SLAVe: Slave to master *DEFault = SLAVe*
Response	None.
Example	`ETH:RFC:SET:GEN:E2E:TDIR MAST`
Note	

Syntax	ETHernet:RFC:SETup:GENeral:E2E:TDIRection?
Description	This query returns the End to End One Way transmission direction.
Parameter	None.
Response	<dir> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:GEN:E2E:TDIR?` → `MAST`
Note	

12.18.8 ETHernet:RFC:SETup:GENeral:E2E:SSTore

Syntax	ETHernet:RFC:SETup:GENeral:E2E:SSTore <enable>
Description	This commands enables/disables storing of results on the slave side for End to End Network test.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:RFC:SET:GEN:E2E:SST OFF`
Note	

Syntax	ETHernet:RFC:SETup:GENeral:E2E:SSTore?
Description	This query returns whether or not results are stored on slave side in End to End Network test.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:GEN:E2E:SST?` → `0`
Note	

12.18.9 ETHernet:RFC:SETup:GENeral:TLFrames

Syntax	ETHernet:RFC:SETup:GENeral:TLFrames <enable>
Description	This command enables/disables transmission of learning frames prior to test.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:RFC:SET:GEN:TLF OFF`
Note	

Syntax	ETHernet:RFC:SETup:GENeral:TLFrames?
Description	This query returns whether or not learning frames are used.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:GEN:TLF?` → `0`
Note	

12.18.10 ETHernet:RFC:SETup:GENeral:IAFFilter

Syntax	ETHernet:RFC:SETup:GENeral:IAFFilter <enable>
Description	This command selects if addresses should be included in frame filter on receiver.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:RFC:SET:GEN:IAFF OFF`
Note	

Syntax	ETHernet:RFC:SETup:GENeral:IAFFilter?
Description	This query returns whether or not addresses are included in frame filter on receiver.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:GEN:IAFF?` → `0`
Note	

12.18.11 ETHernet:RFC:SETup:GENeral:JITTer

Syntax	ETHernet:RFC:SETup:GENeral:JITTer <enable>
Description	This command enables/disables jitter measurement with latency test.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:RFC:SET:GEN:JITT ON`
Note	

Syntax	ETHernet:RFC:SETup:GENeral:JITTer?
Description	This query returns whether or not jitter measurement is enabled with latency test.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:GEN:JITT?` → `1`
Note	

12.18.12 ETHernet:RFC:SETup:GENeral:ACCumulate

Syntax	ETHernet:RFC:SETup:GENeral:ACCumulate <enable>
Description	This command sets if repeated steps should be accumulated.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:RFC:SET:GEN:ACC OFF`
Note	

Syntax	ETHernet:RFC:SETup:GENeral:ACCumulate?
Description	This query returns whether or not repeated steps are accumulated.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:GEN:ACC?` → `0`
Note	

12.18.13 ETHernet:RFC:SETup:GENeral:TCLayer

Syntax	ETHernet:RFC:SETup:GENeral:TCLayer <layer>
Description	This command selects the throughput calculation layer.
Parameter	<layer> = <CHARACTER PROGRAM DATA> UTIL: Utilization PHYP: Physical with preamble PHYS: Physical without preamble LINK: Link NETWork: Network DATA: Data *DEFault = PHYS*
Response	None.
Example	`ETH:RFC:SET:GEN:TCL LINK`
Note	

Syntax	ETHernet:RFC:SETup:GENeral:TCLayer?
Description	This query returns the throughput calculation layer.
Parameter	None.
Response	<layer> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:GEN:TCL?` → `LINK`
Note	

12.18.14 ETHernet:RFC:SETup:GENeral:TTYPe

Syntax	ETHernet:RFC:SETup:GENeral:TTYPe <layer>
Description	This command sets the throughput calculation type.
Parameter	<type> = <CHARACTER PROGRAM DATA> AVG: Average MAX: Maximum *DEFault = MAX*
Response	None.
Example	`ETH:RFC:SET:GEN:TTYP AVG`
Note	

Syntax	ETHernet:RFC:SETup:GENeral:TTYPe?
Description	This query returns the throughput calculation type.
Parameter	None.
Response	<layer> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:GEN:TTYP?` → `AVG`
Note	

12.18.15 ETHernet:RFC:SETup:GENeral:VERCompati

Syntax	ETHernet:RFC:SETup:GENeral:VERCompati <layer>
Description	This command sets the version compatibility for configuration.
Parameter	<type> = <CHARACTER PROGRAM DATA> V0906: V9.06 or older V0907: V9.07 or later *DEFault = V0907*
Response	None.
Example	`ETH:RFC:SET:GEN:VERC V0906`
Note	

Syntax	ETHernet:RFC:SETup:GENeral:VERCompati?
Description	This query returns the version compatibility value.
Parameter	None.
Response	<layer> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:GEN:VERC?` → `V0906`
Note	

12.19 RFC2544 - Throughput

12.19.1 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:MODE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:MODE <mode>
Description	This command sets the RFC2544 throughput frame size mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> USER: User defined STEPped: Stepped CONStant: Constant FLEXible: Flexible *DEFault = USER*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:MODE CONS`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:MODE?
Description	This query returns the RFC2544 throughput frame size mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:MODE?` → `CONS`
Note	

12.19.2 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:USER

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:USER {<size>}*
Description	This command sets the RFC2544 throughput user defined frame size(s).
Parameters	<Pt> = Port number
	({<size>} ∗ {, }∗) = <EXPRESSION PROGRAM DATA> 64: 64 bytes 128: 128 bytes 256: 256 bytes 512: 512 bytes 768: 768 bytes 1024: 1024 bytes 1280: 1280 bytes 1518: 1518 bytes JUMB: Jumbo frame size use `:JUMBo` to define.
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:USER (64,256,JUMB)`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:USER?
Description	This query returns the RFC2544 throughput user defined frame size(s).
Parameter	<Pt> = Port number
Response	{(<size>), }* = <EXPRESSION RESPONSE DATA> <size> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:USER?` → `(64,256,JUMB)`
Note	

12.19.3 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:JUMBo

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:JUMBo <size>
Description	This command sets the RFC2544 throughput jumbo frame size.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1519, MAXimum = 16000, DEFault = 1582*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:JUMB 6000`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:JUMBo?
Description	This query returns the RFC2544 throughput jumbo frame size.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:JUMB?` → `6000`
Note	

12.19.4 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:BEGin

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:BEGin <size>
Description	This command sets the RFC2544 throughput begin frame size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:BEG 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:BEGin?
Description	This query returns the RFC2544 throughput begin frame size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:BEG?` → `128`
Note	

12.19.5 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:END

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:END <size>
Description	This command sets the RFC2544 throughput end frame size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 256*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:END 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:END?
Description	This query returns the RFC2544 throughput end frame size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:END?` → `128`
Note	

12.19.6 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:STEP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:STEP <size>
Description	This command sets the RFC2544 throughput step frame size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:STEP 128`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:STEP?
Description	This query returns the RFC2544 throughput step frame size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:STEP?` → `128`
Note	

12.19.7 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:CONStant

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:CONStant <size>
Description	This command sets the RFC2544 throughput frame size for constant mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 256*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:CONS 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:CONStant?
Description	This query returns the RFC2544 throughput frame size for constant mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:CONS?` → `128`
Note	

12.19.8 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:FLEXible:COUNt

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:FLEXible:COUNt <count>
Description	This command sets the RFC2544 flexible frame count.
Parameters	<Pt> = Port number
	<count> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=10, DEFault=1*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:FLEX:COUN 3`
Note	

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:FLEXible:COUNt?
Description	This query returns the RFC2544 flexible frame count.
Parameter	<Pt> = Port number
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:FLEX:COUN?` → `3`
Note	

12.19.9 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:FLEXible<Count>:SIZE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:FLEXible<Count>:SIZE <size>
Description	This command sets the RFC2544 flexible frame size.
Parameters	<Pt> = Port number
	<Count> = Flexible frame number (1-10)
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:FLEX1:SIZE 384`
Notes	

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:FSIZe:FLEXible<Count>:SIZE?
Description	This query returns the RFC2544 flexible frame size.
Parameter	<Pt> = Port number
	<Count> = Flexible frame number (1-10)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:FSIZ:FLEX1:SIZE?` → `384`
Note	

12.19.10 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:STOP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:STOP <enable>
Description	This command enables or disables stop on no frame loss at maximum utilization.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:LL:STOP ON`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:STOP?
Description	This query return the state of stop on no frame loss at maximum utilization.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:LL:STOP?` → `1`
Note	

12.19.11 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:MINimum

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:MINimum <load>
Description	This command sets the minimum line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0008, MAXimum = 100.0000, DEFault = 1.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:LL:MIN 10`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. MINimum, MAXimum and DEFault are all in PCT. `:MINimum` must be smaller or equal to `:MAXimum`.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:MINimum? [<suffix>]
Description	This query returns the minimum line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:LL:MIN? → 10`
Note	

12.19.12 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:MAXimum

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:MAXimum <load>
Description	This command sets the maximum line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0008, MAXimum = 100.0000, DEFault = 100.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:LL:MAX 10`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. MINimum, MAXimum and DEFault are all in PCT. `:MAXimum` must be larger or equal to `:MINimum`.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:MAXimum? [<suffix>]
Description	This query returns the maximum line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:LL:MAX? → 10`
Note	

12.19.13 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:STEP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:STEP <load>
Description	This command sets the step line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0001, MAXimum = 100.0000, DEFault = 10.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:LL:STEP 10`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. MINimum, MAXimum and DEFault are all in PCT.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:STEP? [<suffix>]
Description	This query returns the step line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:LL:STEP? → 10`
Note	

12.19.14 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch[:ENABle]

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch[:ENABle] <enable>
Description	This command enables or disables the line load Auto Search.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Examples	`ETH:RFC:SET:PORT1:THR:LL:ASE ON`
Note	

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch[:ENABle]?
Description	This query returns whether or not the line load Auto Search is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:LL:ASE? → 1`
Note	

12.19.15 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch:MODE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch:MODE <mode>
Description	This command sets the auto search mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> SMARt BINary *DEFault = SMARt*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:LL:ASE:MODE BIN`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch:MODE?
Description	This query returns the auto search mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:LL:ASE:MODE?` → `BIN`
Note	

12.19.16 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch :RESolution

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch:RESolution <res>
Description	This command sets the auto search resolution.
Parameters	<Pt> = Port number
	<res> = <CHARACTER PROGRAM DATA> 01: 0.1% 1: 1% 10: 10% *DEFault = 01*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:LL:ASE:RES 10`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch:RESolution?
Description	This query returns the auto search resolution.
Parameter	<Pt> = Port number
Response	<res> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:LL:ASE:RES?` → `10`
Note	

12.19.17 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:DURation:STEP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:DURation:STEP <step>
Description	This command sets the step duration. Unit: seconds.
Parameters	<Pt> = Port number
	<step> = <NUMERIC PROGRAM DATA> *MINimum = 3, MAXimum = 1000000000, DEFault = 10*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:DUR:STEP 5`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:DURation:STEP?
Description	This query returns the step duration. Unit: Seconds. Unit: seconds.
Parameter	<Pt> = Port number
Response	<step> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:DUR:STEP?` → `5`
Note	

12.19.18 ETHernet:RFC:SETup:PORT<Pt>:THRoughput:DURation:REPeats

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:DURation:REPeats <rep>
Description	This command sets the number of repeats.
Parameters	<Pt> = Port number
	<rep> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1000, DEFault = 0*
Response	None.
Example	`ETH:RFC:SET:PORT1:THR:DUR:REP 5`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:THRoughput:DURation:REPeats?
Description	This query returns the number of repeats.
Parameter	<Pt> = Port number
Response	<rep> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:THR:DUR:REP?` → `5`
Note	

12.20 RFC2544 - Frame Loss

12.20.1 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:MODE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:MODE <mode>
Description	This command sets the RFC2544 frame loss frame size mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> USER: User defined STEPped: Stepped CONStant: Constant FLEXible: Flexible *DEFault = USER*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:MODE CONS`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:MODE?
Description	This query returns the RFC2544 frame loss frame size mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:MODE?` → `CONS`
Note	

12.20.2 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:USER

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:USER {<size>}*
Description	This command sets the RFC2544 frame loss user defined frame size(s) for user defined mode.
Parameters	<Pt> = Port number
	({<size>} ∗ {, }∗) = <EXPRESSION PROGRAM DATA> 64: 64 bytes 128: 128 bytes 256: 256 bytes 512: 512 bytes 768: 768 bytes 1024: 1024 bytes 1280: 1280 bytes 1518: 1518 bytes JUMB: Jumbo frame size use `:JUMBo` to define.
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:USER (64,256,JUMB)`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:USER?
Description	This query returns the RFC2544 frame loss user defined frame size(s) for user defined mode.
Parameter	<Pt> = Port number
Response	{(<size>), }* = <EXPRESSION RESPONSE DATA> <size> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:USER?` → `(64,256,JUMB)`
Note	

12.20.3 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:JUMBo

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:JUMBo <size>
Description	This command sets the RFC2544 frame loss jumbo frame size for user defined mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1519, MAXimum = 16000, DEFault = 1582*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:JUMB 6000`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:JUMBo?
Description	This query returns the RFC2544 frame loss jumbo frame size for user defined mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:JUMB?` → `6000`
Note	

12.20.4 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:BEGin

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:BEGin <size>
Description	This command sets the RFC2544 frame loss begin frame size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:BEG 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:BEGin?
Description	This query returns the RFC2544 frame loss begin frame size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:BEG?` → `128`
Note	

12.20.5 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:END

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:END <size>
Description	This command sets the RFC2544 frame loss end frame size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 256*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:END 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:END?
Description	This query returns the RFC2544 frame loss end frame size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:END?` → `128`
Note	

12.20.6 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:STEP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:STEP <size>
Description	This command sets the RFC2544 frame loss step frame size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:STEP 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:STEP?
Description	This query returns the RFC2544 frame loss step frame size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:STEP?` → `128`
Note	

12.20.7 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:CONStant

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:CONStant <size>
Description	This command sets the RFC2544 frame loss frame size for constant mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 256*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:CONS 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:CONStant?
Description	This query returns the RFC2544 frame loss frame size for constant mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:CONS?` → `128`
Note	

12.20.8 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:FLEXible:COUNt

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:FLEXible:COUNt <count>
Description	This command sets the RFC2544 flexible frame count.
Parameters	<Pt> = Port number
	<count> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=10, DEFault=1*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:FLEX:COUN 3`
Note	

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:FLEXible:COUNt?
Description	This query returns the RFC2544 flexible frame count.
Parameter	<Pt> = Port number
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:FLEX:COUN?` → `3`
Note	

12.20.9 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:FLEXible<Count>:SIZE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:FLEXible<Count>:SIZE <size>
Description	This command sets the RFC2544 frame loss flexible frame size.
Parameters	<Pt> = Port number
	<Count> = Flexible frame number (1-10)
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:FLEX1:SIZE 384`
Notes	

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:FSIZe:FLEXible<Count>:SIZE?
Description	This query returns the RFC2544 frame loss flexible frame size.
Parameter	<Pt> = Port number
	<Count> = Flexible frame number (1-10)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:FSIZ:FLEX1:SIZE?` → `384`
Note	

12.20.10 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:STOP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:STOP <enable>
Description	This command enables or disables stop on no frame loss at maximum utilization.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:LL:STOP ON`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:STOP?
Description	This query returns the state for stop on no frame loss at maximum utilization.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:LL:STOP?` → `1`
Note	

12.20.11 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:MINimum

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:MINimum <load>
Description	This command sets the minimum line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0008, MAXimum = 100.0000, DEFault = 1.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:LL:MIN 10`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. MINimum, MAXimum and DEFault are all in PCT. `:MINimum` must be smaller or equal to `:MAXimum`.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:MINimum? [<suffix>]
Description	This query returns the minimum line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:LL:MIN? → 10`
Note	

12.20.12 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:MAXimum

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:MAXimum <load>
Description	This command sets the maximum line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0008, MAXimum = 100.0000, DEFault = 100.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:LL:MAX 10`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. MINimum, MAXimum and DEFault are all in PCT. `:MAXimum` must be larger or equal to `:MINimum`.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:MAXimum? [<suffix>]
Description	This query returns the maximum line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:LL:MAX? → 10`
Note	

12.20.13 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:STEP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:STEP <load>
Description	This command sets the step line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0001, MAXimum = 100.0000, DEFault = 10.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:LL:STEP 10`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. MINimum, MAXimum and DEFault are all in PCT.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:STEP? [<suffix>]
Description	This query returns the step line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:LL:STEP? → 10`
Note	

12.20.14 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:ASEarch[:ENABle]

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:ASEarch[:ENABle] <enable>
Description	This command enables or disables the line load Auto Search.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Examples	`ETH:RFC:SET:PORT1:FLOS:LL:ASE ON`
Note	

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:ASEarch[:ENABle]?
Description	This query returns whether or not the line load Auto Search is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:LL:ASE? → 1`
Note	

12.20.15 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:ASEarch:MODE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:ASEarch:MODE <mode>
Description	This command sets the auto search mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> SMARt BINary *DEFault = SMARt*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:LL:ASE:MODE BIN`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:ASEarch:MODE?
Description	This query returns the auto search mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:LL:ASE:MODE?` → `BIN`
Note	

12.20.16 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:ASEarch:RESolution

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:ASEarch:RESolution <res>
Description	This command sets the auto search resolution.
Parameters	<Pt> = Port number
	<res> = <CHARACTER PROGRAM DATA> 01: 0.1% 1: 1% 10: 10% *DEFault = 01*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:LL:ASE:RES 10`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:LLoad:ASEarch:RESolution?
Description	This query returns the auto search resolution.
Parameter	<Pt> = Port number
Response	<res> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:LL:ASE:RES?` → `10`
Note	

12.20.17 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:DURation:STEP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:DURation:STEP <step>
Description	This command sets the step duration. Unit: Seconds.
Parameters	<Pt> = Port number
	<step> = <NUMERIC PROGRAM DATA> *MINimum = 3, MAXimum = 1000000000, DEFault = 10*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:DUR:STEP 5`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:DURation:STEP?
Description	This query returns the step duration. Unit: Seconds.
Parameter	<Pt> = Port number
Response	<step> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:DUR:STEP?` → `5`
Note	

12.20.18 ETHernet:RFC:SETup:PORT<Pt>:FLOSs:DURation:REPeats

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:DURation:REPeats <rep>
Description	This command sets the number of repeats.
Parameters	<Pt> = Port number
	<rep> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1000, DEFault = 0*
Response	None.
Example	`ETH:RFC:SET:PORT1:FLOS:DUR:REP 5`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:FLOSs:DURation:REPeats?
Description	This query returns the number of repeats.
Parameter	<Pt> = Port number
Response	<rep> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:FLOS:DUR:REP?` → `5`
Note	

12.21 RFC2544 - Throughput and Frame Loss

12.21.1 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:MODE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:MODE <mode>
Description	This command sets the RFC2544 throughput and frame loss frame size mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> USER: User defined STEPped: Stepped CONStant: Constant FLEXible: Flexible *DEFault = USER*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:MODE CONS`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:MODE?
Description	This query returns the RFC2544 throughput and frame loss frame size mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:MODE?` → `CONS`
Note	

12.21.2 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:USER

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:USER {<size>}*
Description	This command sets the RFC2544 throughput and frame loss user defined frame size(s) for user defined mode.
Parameters	<Pt> = Port number
	({<size>} ∗ {, }∗) = <EXPRESSION PROGRAM DATA> 64: 64 bytes 128: 128 bytes 256: 256 bytes 512: 512 bytes 768: 768 bytes 1024: 1024 bytes 1280: 1280 bytes 1518: 1518 bytes JUMB: Jumbo frame size use `:JUMBo` to define.
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:USER (64,256,JUMB)`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:USER?
Description	This query returns the RFC2544 throughput and frame loss user defined frame size(s) for user defined mode.
Parameter	<Pt> = Port number
Response	{(<size>), }* = <EXPRESSION RESPONSE DATA> <size> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:USER?` → `(64,256,JUMB)`
Note	

12.21.3 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:JUMBo

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:JUMBo <size>
Description	This command sets the RFC2544 throughput and frame loss jumbo frame size for user defined mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1519, MAXimum = 16000, DEFault = 1582*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:JUMB 6000`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:JUMBo?
Description	This query returns the RFC2544 throughput and frame loss jumbo frame size for user defined mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:JUMB?` → `6000`
Note	

12.21.4 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:BEGin

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:BEGin <size>
Description	This command sets the RFC2544 throughput and frame loss begin frame size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:BEG 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:BEGin?
Description	This query returns the RFC2544 throughput and frame loss begin frame size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:BEG?` → `128`
Note	

12.21.5 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:END

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:END <size>
Description	This command sets the RFC2544 throughput and frame loss end frame size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 256*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:END 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:END?
Description	This query returns the RFC2544 throughput and frame loss end frame size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:END?` → `128`
Note	

12.21.6 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:STEP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:STEP <size>
Description	This command sets the RFC2544 step frame size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:STEP 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:STEP?
Description	This query returns the RFC2544 step frame size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:STEP?` → `128`
Note	

12.21.7 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:CONStant

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:CONStant <size>
Description	This command sets the RFC2544 frame size for constant mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 256*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:CONS 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:CONStant?
Description	This query returns the RFC2544 frame size for constant mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:CONS?` → `128`
Note	

12.21.8 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:FLEXible:COUNt

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:FLEXible:COUNt <count>
Description	This command sets the RFC2544 flexible frame count.
Parameters	<Pt> = Port number
	<count> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=10, DEFault=1*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:FLEX:COUN 3`
Note	

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:FLEXible:COUNt?
Description	This query returns the RFC2544 flexible frame count.
Parameter	<Pt> = Port number
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:FLEX:COUN?` → `3`
Note	

12.21.9 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:FLEXible<Count>:SIZE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:FLEXible<Count>:SIZE <size>
Description	This command sets the RFC2544 flexible frame size.
Parameters	<Pt> = Port number
	<Count> = Flexible frame number (1-10)
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:FLEX1:SIZE 384`
Notes	

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:FSIZe:FLEXible<Count>:SIZE?
Description	This query returns the RFC2544 throughput flexible frame size.
Parameter	<Pt> = Port number
	<Count> = Flexible frame number (1-10)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:FSIZ:FLEX1:SIZE?` → `384`
Note	

12.21.10 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:STOP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:STOP <enable>
Description	This command enables or disables stop on no frame loss at maximum utilization.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:LL:STOP ON`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:STOP?
Description	This query returns the state for stop on no frame loss at maximum utilization.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:LL:STOP?` → `1`
Note	

12.21.11 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:MINimum

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:MINimum <load>
Description	This command sets the minimum line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0008, MAXimum = 100.0000, DEFault = 1.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:LL:MIN 10`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. MINimum, MAXimum and DEFault are all in PCT. `:MINimum` must be smaller or equal to `:MAXimum`.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:MINimum? [<suffix>]
Description	This query returns the minimum line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:LL:MIN? → 10`
Note	

12.21.12 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:MAXimum

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:MAXimum <load>
Description	This command sets the maximum line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0008, MAXimum = 100.0000, DEFault = 100.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:LL:MAX 10`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. MINimum, MAXimum and DEFault are all in PCT. `:MAXimum` must be larger or equal to `:MINimum`.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:MAXimum? [<suffix>]
Description	This query returns the maximum line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:LL:MAX? → 10`
Note	

12.21.13 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:STEP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:STEP <load>
Description	This command sets the step line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0001, MAXimum = 100.0000, DEFault = 10.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:LL:STEP 10`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. MINimum, MAXimum and DEFault are all in PCT.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:STEP? [<suffix>]
Description	This query returns the step line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:LL:STEP? → 10`
Note	

12.21.14 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:ASEarch [:ENABle]

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:ASEarch[:ENABle] <enable>
Description	This command enables or disables the line load Auto Search.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Examples	`ETH:RFC:SET:PORT1:TAFL:LL:ASE ON`
Note	

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:ASEarch[:ENABle]?
Description	This query returns whether or not the line load Auto Search is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:LL:ASE? → 1`
Note	

12.21.15 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:ASEarch:MODE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:ASEarch:MODE <mode>
Description	This command sets the auto search mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> SMARt BINary *DEFault = SMARt*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:LL:ASE:MODE BIN`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:ASEarch:MODE?
Description	This query returns the auto search mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:LL:ASE:MODE? → BIN`
Note	

12.21.16 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:ASEarch :RESolution

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:ASEarch:RESolution <res>
Description	This command sets the auto search resolution.
Parameters	<Pt> = Port number
	<res> = <CHARACTER PROGRAM DATA> 01: 0.1% 1: 1% 10: 10% *DEFault = 01*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:LL:ASE:RES 10`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:LLoad:ASEarch:RESolution?
Description	This query returns the auto search resolution.
Parameter	<Pt> = Port number
Response	<res> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:LL:ASE:RES? → 10`
Note	

12.21.17 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:DURation:STEP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:DURation:STEP <step>
Description	This command sets the step duration. Unit: Seconds.
Parameters	<Pt> = Port number
	<step> = <NUMERIC PROGRAM DATA> *MINimum = 3, MAXimum = 1000000000, DEFault = 10*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:DUR:STEP 5`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:DURation:STEP?
Description	This query returns the step duration. Unit: Seconds.
Parameter	<Pt> = Port number
Response	<step> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:DUR:STEP?` → `5`
Note	

12.21.18 ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:DURation:REPeats

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:DURation:REPeats <rep>
Description	This command sets the number of repeats.
Parameters	<Pt> = Port number
	<rep> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1000, DEFault = 0*
Response	None.
Example	`ETH:RFC:SET:PORT1:TAFL:DUR:REP 5`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:TAFLoss:DURation:REPeats?
Description	This query returns the number of repeats.
Parameter	<Pt> = Port number
Response	<rep> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:TAFL:DUR:REP?` → `5`
Note	

12.22 RFC2544 - Latency

12.22.1 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:MODE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:MODE <mode>
Description	This command sets the RFC2544 latency frame size mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> USER: User defined STEPped: Stepped CONStant: Constant FLEXible: Flexible *DEFault = USER*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:MODE CONS`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:MODE?
Description	This query returns the RFC2544 latency frame size mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:MODE?` → `CONS`
Note	

12.22.2 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:USER

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:USER {<size>}*
Description	This command sets the RFC2544 latency user defined frame size(s) for user defined mode.
Parameters	<Pt> = Port number
	({<size>} ∗ {, }∗) = <EXPRESSION PROGRAM DATA> 64: 64 bytes 128: 128 bytes 256: 256 bytes 512: 512 bytes 768: 768 bytes 1024: 1024 bytes 1280: 1280 bytes 1518: 1518 bytes JUMB: Jumbo frame size use `:JUMBo` to define.
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:USER (64,256,JUMB)`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:USER?
Description	This query returns the RFC2544 latency user defined frame size(s) for user defined mode.
Parameter	<Pt> = Port number
Response	{(<size>), }* = <EXPRESSION RESPONSE DATA> <size> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:USER?` → `(64,256,JUMB)`
Note	

12.22.3 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:JUMBo

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:JUMBo <size>
Description	This command sets the RFC2544 latency the jumbo frame size for user defined mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1519, MAXimum = 16000, DEFault = 1582*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:JUMB 6000`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:JUMBo?
Description	This query returns the RFC2544 latency jumbo frame size for user defined mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:JUMB?` → `6000`
Note	

12.22.4 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:BEGin

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:BEGin <size>
Description	This command sets the RFC2544 latency begin frame size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:BEG 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:BEGin?
Description	This query returns the RFC2544 latency begin frame size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:BEG?` → `128`
Note	

12.22.5 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:END

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:END <size>
Description	This command sets the RFC2544 latency end frame size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 256*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:END 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:END?
Description	This query returns the RFC2544 latency end frame size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:END?` → `128`
Note	

12.22.6 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:STEP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:STEP <size>
Description	This command sets the RFC2544 latency step frame size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:STEP 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:STEP?
Description	This query returns the RFC2544 latency step frame size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:STEP?` → `128`
Note	

12.22.7 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:CONStant

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:CONStant <size>
Description	This command sets the RFC2544 latency frame size for constant mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 256*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:CONS 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:CONStant?
Description	This query returns the RFC2544 latency frame size for constant mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:CONS?` → `128`
Note	

12.22.8 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:FLEXible:COUNt

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:FLEXible:COUNt <count>
Description	This command sets the RFC2544 flexible frame count.
Parameters	<Pt> = Port number
	<count> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=10, DEFault=1*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:FLEX:COUN 3`
Note	

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:FLEXible:COUNt?
Description	This query returns the RFC2544 flexible frame count.
Parameter	<Pt> = Port number
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:FLEX:COUN?` → `3`
Note	

12.22.9 ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:FLEXible<Count>:SIZE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:FLEXible<Count>:SIZE <size>
Description	This command sets the RFC2544 flexible frame size.
Parameters	<Pt> = Port number
	<Count> = Flexible frame number (1-10)
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:FLEX1:SIZE 384`
Notes	

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:FSIZe:FLEXible<Count>:SIZE?
Description	This query returns the RFC2544 flexible frame size.
Parameter	<Pt> = Port number
	<Count> = Flexible frame number (1-10)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:FSIZ:FLEX1:SIZE?` → `384`
Note	

12.22.10 ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:MINimum

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:MINimum <load>
Description	This command sets the minimum line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0008, MAXimum = 100.0000, DEFault = 1.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:LL:MIN 10`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. In Router Latency Mode the maximum allowed line load is 1Mbps. MINimum, MAXimum and DEFault are all in PCT. `:MINimum` must be smaller or equal to `:MAXimum`.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:MINimum? [<suffix>]
Description	This query returns the minimum line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:LL:MIN?` → `10`
Note	

12.22.11 ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:MAXimum

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:MAXimum <load>
Description	This command sets the maximum line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0008, MAXimum = 100.0000, DEFault = 100.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:LL:MAX 10`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. In Router Latency Mode the maximum allowed line load is 1Mbps. MINimum, MAXimum and DEFault are all in PCT. `:MAXimum` must be larger or equal to `:MINimum`.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:MAXimum? [<suffix>]
Description	This query returns the maximum line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:LL:MAX? → 10`
Note	

12.22.12 ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:STEP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:STEP <load>
Description	This command sets the step line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0001, MAXimum = 100.0000, DEFault = 10.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:LL:STEP 10`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. In Router Latency Mode the maximum allowed line load is 1Mbps. MINimum, MAXimum and DEFault are all in PCT.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:STEP? [<suffix>]
Description	This query returns the step line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:LL:STEP? → 10`
Note	

12.22.13 ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:ONLY

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:ONLY <enable>
Description	This command selects whether or not only to run steps where other test passed.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:LL:ONLY ON`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:ONLY?
Description	This query returns 1 if only steps are run where other test passed.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:LL:ONLY?` → `1`
Note	

12.22.14 ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:OTESt

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:OTESt <test>
Description	This command selects the other test to follow.
Parameters	<Pt> = Port number
	<test> = <CHARACTER PROGRAM DATA> THRoughput: Throughput FLOSs: Frame loss TAFLoss: Throughput and frame Loss *DEFault = THR*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:LL:OTES FLOS`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:OTESt?
Description	This query returns the test followed.
Parameter	<Pt> = Port number
Response	<test> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:LL:OTES?` → `FLOS`
Note	

12.22.15 ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:TLEVel

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:TLEVel <tol>
Description	This command sets the Tolerance level percentage (Utilization multiplier). Unit: percentage.
Parameters	<Pt> = Port number
	<tol> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100, DEFault=100*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:LL:TLEV 85`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:LLoad:TLEVel?
Description	This query returns the Tolerance level percentage (Utilization multiplier).
Parameter	<Pt> = Port number
Response	<tol> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:LL:TLEV?` → `85`
Note	

12.22.16 ETHernet:RFC:SETup:PORT<Pt>:LATency:DURation:STEP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:DURation:STEP <step>
Description	This command sets the step duration. Unit: Seconds.
Parameters	<Pt> = Port number
	<step> = <NUMERIC PROGRAM DATA> *MINimum = 3, MAXimum = 1000000000, DEFault = 10*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:DUR:STEP 5`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:DURation:STEP?
Description	This query returns the step duration. Unit: Seconds.
Parameter	<Pt> = Port number
Response	<step> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:DUR:STEP?` → `5`
Note	

12.22.17 ETHernet:RFC:SETup:PORT<Pt>:LATency:DURation:REPeats

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:DURation:REPeats <rep>
Description	This command sets the number of repeats.
Parameters	<Pt> = Port number
	<rep> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1000, DEFault = 0*
Response	None.
Example	`ETH:RFC:SET:PORT1:LAT:DUR:REP 5`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:LATency:DURation:REPeats?
Description	This query returns the number of repeats.
Parameter	<Pt> = Port number
Response	<rep> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:LAT:DUR:REP?` → `5`
Note	

12.23 RFC2544 - Burst

12.23.1 ETHernet:RFC:SETup:PORT<Pt>:BURSt:MODE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:MODE <mode>
Description	This command sets the RFC2544 burst mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> FPBurst: Based on frame SPBurst: Based on seconds *DEFault = FPBurst*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:MODE SPB`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:MODE?
Description	This query returns the RFC2544 burst mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:MODE?` → `SPB`
Note	

12.23.2 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:MODE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:MODE <mode>
Description	This command sets the RFC2544 burst frame size mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> USER: User defined STEPped: Stepped CONStant: Constant FLEXible: Flexible *DEFault = USER*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:MODE CONS`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:MODE?
Description	This query returns the RFC2544 burst frame size mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:MODE?` → `CONS`
Note	

12.23.3 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:USER

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:USER {<size>}*
Description	This command sets the RFC2544 burst user defined frame size(s) for user defined mode.
Parameters	<Pt> = Port number
	({<size>} ∗ {, }∗) = <EXPRESSION PROGRAM DATA> 64: 64 bytes 128: 128 bytes 256: 256 bytes 512: 512 bytes

continued on next page. . .

. . . continued from previous page

	768: 768 bytes 1024: 1024 bytes 1280: 1280 bytes 1518: 1518 bytes JUMB: Jumbo frame size use `:JUMBo` to define.
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:USER (64,256,JUMB)`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:USER?
Description	This query returns the RFC2544 burst user defined frame size(s) for user defined mode.
Parameter	<Pt> = Port number
Response	{(<size>), }* = <EXPRESSION RESPONSE DATA> <size> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:USER?` → `(64,256,JUMB)`
Note	

12.23.4 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:JUMBo

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:JUMBo <size>
Description	This command sets the RFC2544 burst jumbo frame size for user defined mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1519, MAXimum = 16000, DEFault = 1582*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:JUMB 6000`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:JUMBo?
Description	This query returns the RFC2544 burst jumbo frame size for user defined mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:JUMB?` → `6000`
Note	

12.23.5 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:BEGin

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:BEGin <size>
Description	This command sets the RFC2544 burst begin frame size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:BEG 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:BEGin?
Description	This query returns the RFC2544 burst begin frame size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:BEG?` → `128`
Note	

12.23.6 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:END

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:END <size>
Description	This command sets the RFC2544 burst end frame size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 256*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:END 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:END?
Description	This query returns the RFC2544 burst end frame size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:END?` → `128`
Note	

12.23.7 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:STEP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:STEP <size>
Description	This command sets the RFC2544 burst step frame size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:STEP 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:STEP?
Description	This query returns the RFC2544 burst step frame size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:STEP?` → `128`
Note	

12.23.8 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:CONStant

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:CONStant <size>
Description	This command sets the RFC2544 burst frame size for constant mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44* [1]*, MAXimum = 16000, DEFault = 256*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:CONS 128`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. [1] The minimum allowed frame size varies depending on the stream frame setup.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:CONStant?
Description	This query returns the RFC2544 burst frame size for constant mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:CONS?` → `128`
Note	

12.23.9 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:FLEXible:COUNt

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:FLEXible:COUNt <count>
Description	This command sets the RFC2544 flexible frame count.
Parameters	<Pt> = Port number
	<count> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=10, DEFault=1*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:FLEX:COUN 3`
Note	

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:FLEXible:COUNt?
Description	This query returns the RFC2544 flexible frame count.
Parameter	<Pt> = Port number
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:FLEX:COUN?` → `3`
Note	

12.23.10 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:FLEXible<Count>:SIZE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:FLEXible<Count>:SIZE <size>
Description	This command sets the RFC2544 flexible frame size.
Parameters	<Pt> = Port number
	<Count> = Flexible frame number (1-10)
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 44, MAXimum = 16000, DEFault = 64*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:FLEX1:SIZE 384`
Notes	

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FSIZe:FLEXible<Count>:SIZE?
Description	This query returns the RFC2544 flexible frame size.
Parameter	<Pt> = Port number
	<Count> = Flexible frame number (1-10)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FSIZ:FLEX1:SIZE?` → `384`
Note	

12.23.11 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:MODE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:MODE <mode>
Description	This command sets the RFC2544 frames per burst mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> STEPped: Stepped CONStant: Constant *DEFault = STEPped*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FPB:MODE CONS`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:MODE?
Description	This query returns the RFC2544 frames per burst mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FPB:MODE?` → `CONS`
Note	

12.23.12 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:BEGin

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:BEGin <size>
Description	This command sets the RFC2544 burst begin size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 300000000, DEFault = 100*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FPB:BEG 10`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:BEGin?
Description	This query returns the RFC2544 burst begin size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FPB:BEG?` → `10`
Note	

12.23.13 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:END

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:END <size>
Description	This command sets the RFC2544 burst end size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 300000000, DEFault = 2000*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FPB:END 20`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:END?
Description	This query returns the RFC2544 burst end size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FPB:END?` → `20`
Note	

12.23.14 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:STEP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:STEP <size>
Description	This command sets the RFC2544 burst step size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 300000000, DEFault = 100*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FPB:STEP 5`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:STEP?
Description	This query returns the RFC2544 burst step size.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FPB:STEP?` → `5`
Note	

12.23.15 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:CONStant

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:CONStant <size>
Description	This command sets the RFC2544 burst size for constant mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 300000000, DEFault = 2000*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FPB:CONS 500`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:CONStant?
Description	This query returns the RFC2544 burst size for constant mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FPB:CONS?` → `500`
Note	

12.23.16 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:STOP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:STOP <enable>
Description	This command enables or disables stop on no frame loss at maximum burst size.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FPB:STOP ON`
Note	This setting is used in Switch/Router and Single ended network mode.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:STOP?
Description	This query returns whether or not stop on no frame loss at maximum burst size is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FPB:STOP?` → `1`
Note	

12.23.17 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch[:ENABle]

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch[:ENABle] <enable>
Description	This command enables/disables frames per burst auto search.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FPB:ASE ON`
Note	This setting is used in Switch/Router and Single ended network mode. In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch[:ENABle]?
Description	This query returns whether or not frames per burst auto search is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FPB:ASE?` → `0`
Note	

12.23.18 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:MODE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:MODE <mode>
Description	This command sets the frames per burst auto search mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> SMARt: Smart search BINary: Binary search *DEFault = SMARt*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FPB:ASE:MODE SMAR`
Note	This setting is used when `ETH:RFC:SET:PORT1:BURS:FPB:ASE` is enabled.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:MODE?
Description	This query returns the frames per burst auto search mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FPB:ASE:MODE?` → `SMAR,BIN`
Note	

12.23.19 ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:RESolution

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:RESolution <resolution>
Description	This command sets the frames per burst auto search resolution.
Parameters	<Pt> = Port number
	<resolution> = <CHARACTER PROGRAM DATA> PM1: 1 per mille (0.1%) PM10: 10 per mille (1.0%) PM100: 100 per mille (10.0%) *DEFault = PM1*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:FPB:ASE:RES PM10`
Note	This setting is used when `ETH:RFC:SET:PORT1:BURS:FPB:ASE` is enabled.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:RESolution?
Description	This query returns the frames per burst auto search resolution.
Parameter	<Pt> = Port number
Response	<resolution> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:FPB:ASE:RES?` → `PM10`
Note	

12.23.20 ETHernet:RFC:SETup:PORT<Pt>:BURSt:SPBurst:SEConds

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:SPBurst:SEConds <seconds>
Description	This command sets the seconds per burst (Burst length).
Parameters	<Pt> = Port number
	<seconds> = <NUMERIC PROGRAM DATA> *MINimum = 0.001, MAXimum = 8.000, DEFault = 1.000*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:SPB:SEC 2.000`
Notes	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. MINimum, MAXimum and DEFault are all in PCT.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:SPBurst:SEConds?
Description	This query returns the seconds per burst (Burst length).
Parameter	<Pt> = Port number
Response	<seconds> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:SPB:SEC?` → `10`
Note	

12.23.21 ETHernet:RFC:SETup:PORT<Pt>:BURSt:SPBurst:ASEarch:MODE

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:SPBurst:ASEarch:MODE <mode>
Description	This command sets the seconds per burst auto search mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> SMARt: Smart search BINary: Binary search *DEFault = SMARt*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:SPB:ASE:MODE SMAR`
Note	This setting is used when `ETH:RFC:SET:PORT1:BURS:SPB:ASE` is enabled.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:SPBurst:ASEarch:MODE?
Description	This query returns the seconds per burst auto search mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:SPB:ASE:MODE?` → `SMAR,BIN`
Note	

12.23.22 ETHernet:RFC:SETup:PORT<Pt>:BURSt:SPBurst:ASEarch:RESolution

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:SPBurst:ASEarch:RESolution <resolution>
Description	This command sets the seconds per burst auto search resolution.
Parameters	<Pt> = Port number
	<resolution> = <CHARACTER PROGRAM DATA> PM01: 0.1 per mille (0.01%) PM1: 1 per mille (0.1%) PM10: 10 per mille (1.0%) PM100: 100 per mille (10.0%) *DEFault = PM1*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:SPB:ASE:RES PM10`
Note	This setting is used when `ETH:RFC:SET:PORT1:BURS:SPB:ASE` is enabled.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:SPBurst:ASEarch:RESolution?
Description	This query returns the seconds per burst auto search resolution.
Parameter	<Pt> = Port number
Response	<resolution> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:SPB:ASE:RES?` → `PM10`
Note	

12.23.23 ETHernet:RFC:SETup:PORT<Pt>:BURSt:DURation:STEP

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:DURation:STEP <step>
Description	This command sets the RFC2544 burst step duration. Unit: Seconds.
Parameters	<Pt> = Port number
	<step> = <NUMERIC PROGRAM DATA> *MINimum = 3, MAXimum = 1000000000, DEFault = 10*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:DUR:STEP 5`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:DURation:STEP?
Description	This query returns the step duration. Unit: Seconds.
Parameter	<Pt> = Port number
Response	<step> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:DUR:STEP?` → `5`
Note	

12.23.24 ETHernet:RFC:SETup:PORT<Pt>:BURSt:DURation:REPeats

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:DURation:REPeats <rep>
Description	This command sets the number of repeats.
Parameters	<Pt> = Port number
	<rep> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1000, DEFault = 0*
Response	None.
Example	`ETH:RFC:SET:PORT1:BURS:DUR:REP 5`
Note	In Switch/Router mode PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	ETHernet:RFC:SETup:PORT<Pt>:BURSt:DURation:REPeats?
Description	This query returns the number of repeats.
Parameter	<Pt> = Port number
Response	<rep> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:SET:PORT1:BURS:DUR:REP?` → `5`
Note	

12.24 RFC2544 - Result

12.24.1 ETHernet:RFC:RESult:TEST

Syntax	ETHernet:RFC:RESult:TEST <test>
Description	This command selects the test to fetch results from.
Parameter	<test> = <CHARACTER PROGRAM DATA> LATency: Latency test THRoughput: Throughput test FLOSs: Frame loss test TAFLoss: Throughput and frame loss test BURSt: Burst test *DEFault = LATency*
Response	None.
Example	`ETH:RFC:RES:TEST LAT`
Note	This setting is not stored as part of a settings- or result file.

Syntax	ETHernet:RFC:RESult:TEST?
Description	This query returns the selected test to fetch results from.
Parameter	None.
Response	<test> = <CHARACTER RESPONSE DATA>
Example	`ETH:RFC:RES:TEST?` → `LAT`
Note	

12.24.2 ETHernet:RFC:RESult:PORT<Pt>:NSTep?

Syntax	ETHernet:RFC:RESult:PORT<Pt>:NSTep? <dir>
Description	This query returns the number of steps for a given port/direction for the selected test type.
Parameters	<Pt> = Port number
	<dir> = <CHARACTER PROGRAM DATA> TX = Transmitter TXR = Transmitter Remote (only E2E test mode) RX = Receiver RXR = Receiver Remote (only E2E test mode) *DEFault = TX*
Response	<steps> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:RES:PORT1:NST? RX` → `6`
Note	Can only be used while a RFC test is in memory.

12.24.3 ETHernet:RFC:RESult:PORT<Pt>:SELect

Syntax	ETHernet:RFC:RESult:PORT<Pt>:SELect <dir>, <step>
Description	This command selects the direction and step to fetch results from.
Parameters	<Pt> = Port number
	<dir> = <CHARACTER PROGRAM DATA> TX = Transmitter TXR = Transmitter Remote (only E2E test mode) RX = Receiver RXR = Receiver Remote (only E2E test mode) *DEFault = TX*
	<step> = <NUMERIC PROGRAM DATA> *MINimum = 1, DEFault = 1*
Response	None.
Example	`ETH:RFC:RES:PORT1:SEL TX, 2`
Note	Use the `ETHernet:RFC:RESult:PORT<Pt>:NSTep?` command (see section 12.24.2) to get the number of steps. This setting is not stored as part of a settings- or result file.

Syntax	ETHernet:RFC:RESult:PORT<Pt>:SELect?
Description	This query returns the selected direction and step to fetch results from.
Parameter	<Pt> = Port number
Response	<dir> = <CHARACTER RESPONSE DATA>
	<step> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:RFC:RES:PORT1:SEL?` → `TX, 2`
Note	

12.24.4 ETHernet:RFC:RESult:PORT<Pt>:FETCh?

Syntax	ETHernet:RFC:RESult:PORT<Pt>:FETCh? <parameters>
Description	This query fetches the RFC2544 results.
Parameters	<Pt> = Port number
	({<parameter>} + {,}*) = <EXPRESSION PROGRAM DATA> The response format is listed for each parameter. **Common** REP: Repeat. Response: <NR1> STEP: Step. Response: <NR1> TFR: Total frames. Response: <NR1> FSIZ: Frame size. Unit: bytes. Response: <NR1> FRAT: Frame rate[1] (only TX). Unit: fps. Response: <NR1> LL: Line load[1] (only TX). Unit: Mbps[2]. Response: <NR2> LLA: Actual line load[1] (only TX). Unit: Mbps[2]. Response: <NR2> THR: Throughput[1] (only RX). Unit: Mbps[2]. Response: <NR2> UTIL: Utilization[1] (only RX). Unit: %. Response: <NR2> **Latency** LJIT: Latency/Jitter (Min,Max,Avg) (only RX). Unit: us. Response: <NR2> **Throughput** FLOS: Frames lost[3] (only RX). Response: <NR2> **Frame Loss** FLOS: Frames lost[3] (only RX). Response: <NR2> LRAT: Loss rate[3] (only RX). Unit: %. Response: <NR2> **Throughput and Frame Loss** FLOS: Frames lost[3] (only RX). Response: <NR2> LRAT: Loss rate[3] (only RX). Unit: %. Response: <NR2> **Burst** BSIZ: Burst size. Response: <NR1> BSEC: Burst seconds. Response: <NR2> FLOS: Frames lost[3] (only RX). Response: <NR2>
Response	{(<result>),}* = <EXPRESSION RESPONSE DATA> Expression format: Numeric List Each result is formatted according to the specification in the parameter field.
Example	`ETH:RFC:RES:PORT1:FETC? (REP,FSIZ,LL)` → `(0), (64), (90.0000)`
Notes	This query fetches the result from the test, direction and step selected by the `ETHernet:RFC:RESult:TEST` command (see section 12.24.1) and `ETHernet:RFC:RESult:PORT<Pt>:SELect` command (see section 12.24.3). If a requested result is not available, NaN (section 1.6.1) is returned. If there is one or more results, the last "," is always removed. [1] Not available for Burst Test. [2] When running a Router Latency Test the unit is kbps. [3] When Accumulate repeated steps is enabled, see `ETHernet:RFC:SETup:GENeral:ACCumulate`, three values are returned: (Min,Max,Avg).

12.25 Y.1564 Service Activation Test

12.25.1 ETHernet:SATest:STARt

Syntax	ETHernet:SATest:STARt
Description	This command starts a service activation test (described in ITU-T Y.1564).
Parameter	None.
Response	None.
Example	`ETH:SAT:STAR`
Note	

12.25.2 ETHernet:SATest:STOP

Syntax	ETHernet:SATest:STOP
Description	This command stops the service activation test.
Parameter	None.
Response	None.
Example	`ETH:SAT:STOP`
Note	

12.25.3 ETHernet:SATest:SETup:TMODe

Syntax	ETHernet:SATest:SETup:TMODe <mode>
Description	This command sets the service activation test mode.
Parameters	<mode> = <CHARACTER PROGRAM DATA> OWAY: One-way testing RTRip: Round-trip testing *DEFault = RTRip*
Response	None.
Example	`ETH:SAT:SET:TMOD OWA`
Note	For round-trip testing the test setup is done using the local-to-remote commands with the ":LTRemote" node, while the test result is retrieved using the remote-to-local commands with the "RTLocal" node.

Syntax	ETHernet:SATest:SETup:TMODe?
Description	This query returns the service activation test mode.
Parameter	None.
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:TMOD? → OW`
Note	

12.25.4 ETHernet:SATest:SETup:OWTest:LTRemote[:ENABle]

Syntax	ETHernet:SATest:SETup:OWTest:LTRemote[:ENABle] <enable>
Description	This command enables/disables the one-way test in the local to remote direction.
Parameters	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:SAT:SET:OWT:LTR ON`
Note	

Syntax	ETHernet:SATest:SETup:OWTest:LTRemote[:ENABle]?
Description	This query returns whether or not the local to remote direction is enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:SAT:SET:OWT:LTR? → 1`
Note	

12.25.5 ETHernet:SATest:SETup:OWTest:RTLocal[:ENABle]

Syntax	ETHernet:SATest:SETup:OWTest:RTLocal[:ENABle] <enable>
Description	This command enables/disables the one-way test in the remote to local direction.
Parameters	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:SAT:SET:OWT:RTL ON`
Note	

Syntax	ETHernet:SATest:SETup:OWTest:RTLocal[:ENABle]?
Description	This query returns whether or not the remote to local direction is enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:SAT:SET:OWT:RTL?` → `1`
Note	

12.25.6 ETHernet:SATest:SETup:OWTest:BSYMmetry

Syntax	ETHernet:SATest:SETup:OWTest:BSYMmetry <symmetry>
Description	This command sets the one-way test bandwidth symmetry.
Parameters	<symmetry> = <CHARACTER PROGRAM DATA> SYMMetric: Symmetric bandwidth ASYMmetric: Asymmetric bandwidth *DEFault = SYMMetric*
Response	None.
Example	`ETH:SAT:SET:OWT:BSYM SYMM`
Note	The value is significant only for bi-directional one-way tests (both `ETHernet:SATest:SETup:OWTest:RTLocal` and `ETHernet:SATest:SETup:OWTest:LTRemote` are enabled). For bi-directional one-way tests using SYMMetric bandwidth setup, it is the `ETHernet:SATest:SETup:SERVice<no>[:LTRemote]` node that are used to define the traffic in both directions.

Syntax	ETHernet:SATest:SETup:OWTest:BSYMmetry?
Description	This query returns the bandwidth symmetry parameter.
Parameter	None.
Response	<symmetry> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:OWT:BSYM?` → `SYMM`
Note	

12.25.7 ETHernet:SATest:SETup:OWTest:SYNChronization

Syntax	ETHernet:SATest:SETup:OWTest:SYNChronization <mode>
Description	This command sets the one-way test time synchronization mode.
Parameters	<mode> = <CHARACTER PROGRAM DATA> PRETest: Pre-test synchronization GPS: GPS. *DEFault = PRETest*
Response	None.
Example	`ETH:SAT:SET:OWT:SYNC PRET`
Note	

Syntax	ETHernet:SATest:SETup:OWTest:SYNChronization?
Description	This query returns the time synchronization mode.
Parameter	None.
Response	<symmetry> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:OWT:SYNC?` → `GPS`
Note	

12.25.8 ETHernet:SATest:SETup:OWTest:TLSAddresses[:ENABle]

Syntax	ETHernet:SATest:SETup:OWTest:TLSAddresses[:ENABle] <enable>
Description	This command enables/disables transfer of local source addresses to be used as destination addresses on remote side.
Parameters	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:SAT:SET:OWT:TLSA ON`
Note	

Syntax	ETHernet:SATest:SETup:OWTest:TLSAddresses[:ENABle]?
Description	This query returns whether or not local source addresses are used as destination addresses on remote side.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:SAT:SET:OWT:TLSA?` → `1`
Note	

12.25.9 ETHernet:SATest:SETup:CCFC

Syntax	ETHernet:SATest:SETup:CCFC <enable>
Description	This command enables/disables compatible configuration frames with CMA3000 and V2.X or older.
Parameters	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:SAT:SET:CCFC ON`
Note	

Syntax	ETHernet:SATest:SETup:CCFC?
Description	This query returns the state of compatible configuration frames with CMA3000 and V2.X or older.
Parameter	None
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:SAT:SET:CCFC?` → `1`
Note	

12.25.10 ETHernet:SATest:SETup:CBRate

Syntax	ETHernet:SATest:SETup:CBRate <unit>
Description	This command sets the calculated bit rate base.
Parameters	<unit> = <CHARACTER PROGRAM DATA> IRATe: Information rate ULRate: Utilized line rate *DEFault = IRATe*
Response	None.
Example	`ETH:SAT:SET:CBR IRAT`
Note	

Syntax	ETHernet:SATest:SETup:CBRate?
Description	This query returns the calculated bit rate base.
Parameter	None.
Response	<unit> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:CBR?` → `IRAT`
Note	

12.25.11 ETHernet:SATest:SETup:SPTDuration

Syntax	ETHernet:SATest:SETup:SPTDuration <duration>
Description	This command sets the service performance test duration.
Parameters	<duration> = <CHARACTER PROGRAM DATA> 15M: 15 minutes 2H: 2 hours 24H: 24 hours CUSTom: Use the custom value *DEFault = CUSTom*
Response	None.
Example	`ETH:SAT:SET:SPTD 15M`
Note	

Syntax	ETHernet:SATest:SETup:SPTDuration?
Description	This query returns the service performance test duration.
Parameter	None.
Response	<duration> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:SPTD?` → `15M`
Note	

12.25.12 ETHernet:SATest:SETup:SPTDuration:CDURation

Syntax	ETHernet:SATest:SETup:SPTDuration:CDURation <hours>,<minutes>,<seconds>
Description	This command sets the service performance test custom duration.
Parameters	<hours> = <NUMERIC PROGRAM DATA> (0-23)
	<minutes> = <NUMERIC PROGRAM DATA> (0-1439)
	<seconds> = <NUMERIC PROGRAM DATA> (0-86399)
Response	None.
Example	`ETH:SAT:SET:SPTD:CDUR 0,5,0`
Note	The performance test custom duration cannot exceed 23 hours, 59 minutes and 59 seconds.

Syntax	ETHernet:SATest:SETup:SPTDuration:CDURation?
Description	This query returns the service performance test custom duration.
Parameter	None.
Response	<hours> = <NR1 NUMERIC RESPONSE DATA>
	<minutes> = <NR1 NUMERIC RESPONSE DATA>
	<seconds> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SPTD:CDUR?` → `0,5,0`
Note	

12.25.13 ETHernet:SATest:SETup:COFail

Syntax	ETHernet:SATest:SETup:COFail <enable>
Description	This command enables/disables continue on fail. When enabled, all the service configuration tests are performed despite violation of the service acceptance criteria.
Parameters	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:SAT:SET:COF OFF`
Note	Service performance test is never performed if the service acceptance criteria is exceeded.

Syntax	ETHernet:SATest:SETup:COFail?
Description	This query returns whether or not continue on fail is enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:SAT:SET:COF?` → `0`
Note	

12.25.14 ETHernet:SATest:SETup:GENeral:IAFFilter

Syntax	ETHernet:SATest:SETup:GENeral:IAFFilter
Description	This command selects if addresses should be included in frame filter on receiver.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:SAT:SET:GEN:IAFF ON`
Note	

Syntax	ETHernet:SATest:SETup:GENeral:IAFFilter?
Description	This query returns addresses should be included in frame filter on receiver.
Parameter	None.
Response	<enable> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:GEN:IAFF?` → `1`
Note	

12.25.15 ETHernet:SATest:SETup:SERVice<no>[:ENABle]

Syntax	ETHernet:SATest:SETup:SERVice<no>[:ENABle] <enable>
Description	This command enables/disables a service.
Parameters	<no> = Service number (1-8)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:SAT:SET:SERV OFF`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>[:ENABle]?
Description	This query returns whether or not a service is enabled.
Parameter	<no> = Service number (1-8)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:SAT:SET:SERV?` → `0`
Note	

12.25.16 ETHernet:SATest:SETup:SERVice<no>:NAME

Syntax	ETHernet:SATest:SETup:SERVice<no>:NAME <name>
Description	This command changes the name of a service.
Parameters	<no> = Service number (1-8)
	<name> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:SAT:SET:SERV1:NAME "Video"`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:NAME?
Description	This query returns the name of a service.
Parameter	<no> = Service number (1-8)
Response	<name> = <STRING RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:NAME?` → `"Video"`
Note	

12.25.17 ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate] <rate>
Description	This command sets the committed information rate for the specified service. Unit: Mbps.
Parameters	<no> = Service number (1-8)
	<rate> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = see note*[1]*, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1 1.12`
Note	If set to zero, the CIR test steps are excluded. This value is automatically calculated when the profile is Video or Voice. [1] Maximum value depends on the current frame size setup.

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]?
Description	This query returns the committed information rate for the specified service.
Parameter	<no> = Service number (1-8)
Response	<rate> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1?` → `1.12`
Note	

12.25.18 ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate] <rate>
Description	This command sets the committed information rate for the specified service. Unit: Mbps.
Parameters	<no> = Service number (1-8)
	<rate> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = see note*[1]*, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL 1.12`
Note	If set to zero, the CIR test steps are excluded. This value is automatically calculated when the profile is Video or Voice. [1] Maximum value depends on the current frame size setup.

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]?
Description	This query returns the committed information rate for the specified service.
Parameter	<no> = Service number (1-8)
Response	<rate> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL?` → `1.12`
Note	

12.25.19 ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:PROFile

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:PROFile <profile>
Description	This command sets the stream Profile.
Parameters	<no> = Service number (1-8)
	<profile> = <CHARACTER PROGRAM DATA> DATA: Data VIDeo: Video VOICe: Voice *DEFault = DATA*
Response	None.
Example	`ETH:SAT:SET:SERV1:PROF VID`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:PROFile?
Description	This query returns the stream Profile.
Parameter	<no> = Service number (1-8)
Response	<profile> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:PROF? → VID`
Note	

12.25.20 ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:PROFile

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:PROFile <profile>
Description	This command sets the stream Profile.
Parameters	<no> = Service number (1-8)
	<profile> = <CHARACTER PROGRAM DATA> DATA: Data VIDeo: Video VOICe: Voice *DEFault = DATA*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:PROF VID`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:PROFile?
Description	This query returns the stream Profile.
Parameter	<no> = Service number (1-8)
Response	<profile> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:PROF? → VID`
Note	

12.25.21 ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:ENCoding :VIDeo

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:ENCoding:VIDeo <encoding>
Description	This command sets the stream Encoding Video.
Parameters	<no> = Service number (1-8)
	<encoding> = <CHARACTER PROGRAM DATA> SDMPEG2: SDTV (MPEG2) HDMPEG2: HDTV (MPEG2) HDMPEG4: MPEG4 (H.264) SD *DEFault = SDMPEG2*
Response	None.
Example	`ETH:SAT:SET:SERV1:ENC:VID HDMPEG2`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:ENCoding:VIDeo?
Description	This query returns the stream Encoding Video.
Parameter	<no> = Service number (1-8)
Response	<encoding> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:ENC:VID?` → `HDMPEG2`
Note	

12.25.22 ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:ENCoding :VIDeo

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:ENCoding:VIDeo <encoding>
Description	This command sets the stream Encoding Video.
Parameters	<no> = Service number (1-8)
	<encoding> = <CHARACTER PROGRAM DATA> SDMPEG2: SDTV (MPEG2) HDMPEG2: HDTV (MPEG2) HDMPEG4: MPEG4 (H.264) SD *DEFault = SDMPEG2*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:ENC:VID HDMPEG2`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:ENCoding:VIDeo?
Description	This query returns the stream Encoding Video.
Parameter	<no> = Service number (1-8)
Response	<encoding> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:ENC:VID?` → `HDMPEG2`
Note	

12.25.23 ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:ENCoding :VOICe

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:ENCoding:VOICe <encoding>
Description	This command sets the stream Encoding Voice.
Parameters	<no> = Service number (1-8)
	<encoding> = <CHARACTER PROGRAM DATA> G711: VoIP G.711 G7231: VoIP G.723.1 G729: VoIP G.729 *DEFault = G711*
Response	None.
Example	`ETH:SAT:SET:SERV1:ENC:VOIC G7231`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:ENCoding:VOICe?
Description	This query returns the stream Encoding Voice.
Parameter	<no> = Service number (1-8)
Response	<encoding> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:ENC:VOIC?` → `G7231`
Note	

12.25.24 ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:ENCoding :VOICe

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:ENCoding:VOICe <encoding>
Description	This command sets the stream Encoding Voice.
Parameters	<no> = Service number (1-8)
	<encoding> = <CHARACTER PROGRAM DATA> G711: VoIP G.711 G7231: VoIP G.723.1 G729: VoIP G.729 *DEFault = G711*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:ENC:VOIC G7231`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:ENCoding:VOICe?
Description	This query returns the stream Encoding Voice.
Parameter	<no> = Service number (1-8)
Response	<encoding> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:ENC:VOIC? → G7231`
Note	

12.25.25 ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:NCHannels

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:NCHannels <channel>
Description	This command sets the number of channels.
Parameters	<no> = Service number (1-8)
	<channel> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 100000, DEFault = 1*
Response	None.
Example	`ETH:SAT:SET:SERV1:NCH 3`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote][:CIRate]:NCHannels?
Description	This query returns the number of channels.
Parameter	<no> = Service number (1-8)
Response	<channel> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:NCH? → 3`
Note	

12.25.26 ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:NCHannels

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:NCHannels <channel>
Description	This command sets the number of channels.
Parameters	<no> = Service number (1-8)
	<channel> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 100000, DEFault = 1*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:NCH 3`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal[:CIRate]:NCHannels?
Description	This query returns the number of channels.
Parameter	<no> = Service number (1-8)
Response	<channel> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:NCH? → 3`
Note	

12.25.27 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:EIRate

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:EIRate <rate>
Description	This command sets the excess information rate for the specified service. Unit: Mbps.
Parameters	<no> = Service number (1-8)
	<rate> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = see note*[1]*, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:EIR 0.10`
Note	If set to zero, the CIR test steps are excluded. [1] Maximum value depends on the current frame size setup.

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:EIRate?
Description	This query returns the excess information rate for the specified service.
Parameter	<no> = Service number (1-8)
Response	<rate> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:EIR?` → `0.12`
Note	

12.25.28 ETHernet:SATest:SETup:SERVice<no>:RTLocal:EIRate

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:EIRate <rate>
Description	This command sets the excess information rate for the specified service. Unit: Mbps.
Parameters	<no> = Service number (1-8)
	<rate> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = see note*[1]*, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:EIR 0.12`
Note	If set to zero, the CIR test steps are excluded. [1] Maximum value depends on the current frame size setup.

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:EIRate?
Description	This query returns the excess information rate for the specified service.
Parameter	<no> = Service number (1-8)
Response	<rate> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:EIR?` → `0.12`
Note	

12.25.29 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:TPOLicing[:ENABle]

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:TPOLicing[:ENABle] <enable>
Description	This command enables/disables the traffic policing test steps.
Parameters	<no> = Service number (1-8)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:SAT:SET:SERV1:TPOL OFF`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:TPOLicing[:ENABle]?
Description	This query returns whether or not traffic policing test is enabled.
Parameter	<no> = Service number (1-8)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:TPOL?` → `0`
Note	

12.25.30 ETHernet:SATest:SETup:SERVice<no>:RTLocal:TPOLicing[:ENABle]

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:TPOLicing[:ENABle] <enable>
Description	This command enables/disables the traffic policing test steps.
Parameters	<no> = Service number (1-8)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:TPOL OFF`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:TPOLicing[:ENABle]?
Description	This query returns whether or not traffic policing test not is enabled.
Parameter	<no> = Service number (1-8)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:TPOL? → 0`
Note	

12.25.31 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:TPOLicing:MARGin

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:TPOLicing:MARGin <margin>
Description	This command sets the acceptable traffic policing margin. Unit: Mbps.
Parameters	<no> = Service number (1-8)
	<margin> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = see note*[1]*, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:TPOL:MARG 10`
Note	The minimum possible margin is 0.01 Mbps for line speeds up to 1Gbps. The minimum possible margin is 0.10 Mbps for 10Gbps line speeds. [1] Maximum value depends on the current frame size setup.

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:TPOLicing:MARGin?
Description	This query returns the acceptable traffic policing margin.
Parameter	<no> = Service number (1-8)
Response	<margin> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:TPOL:MARG? → 10.15`
Note	

12.25.32 ETHernet:SATest:SETup:SERVice<no>:RTLocal:TPOLicing:MARGin

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:TPOLicing:MARGin <margin>
Description	This command sets the acceptable traffic policing margin. Unit: Mbps.
Parameters	<no> = Service number (1-8)
	<margin> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = see note*[1]*, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:TPOL:MARG 10`
Note	The minimum possible margin is 0.01 Mbps for line speeds up to 1Gbps. The minimum possible margin is 0.10 Mbps for 10Gbps line speeds. [1] Maximum value depends on the current frame size setup.

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:TPOLicing:MARGin?
Description	This query returns the acceptable traffic policing margin.
Parameter	<no> = Service number (1-8)
Response	<margin> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:TPOL:MARG? → 10.15`
Note	

12.25.33 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CBSize

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CBSize <size>
Description	This command sets the committed burst size for the specified service. Unit: Bytes.
Parameters	<no> = Service number (1-8)
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1000000000, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:CBS 10000`
Note	If set to zero, the CBS test is excluded. If set to a value greater than zero but below the MTU size then the MTU size will be applied.

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CBSize?
Description	This query returns the committed burst size for the specified service.
Parameter	<no> = Service number (1-8)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:CBS? → 10000`
Note	

12.25.34 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CBSize

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CBSize <size>
Description	This command sets the committed burst size for the specified service. Unit: Bytes.
Parameters	<no> = Service number (1-8)
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1000000000, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:CBS 10000`
Note	If set to zero, the CBS test is excluded. If set to a value greater than zero but below the MTU size then the MTU size will be applied.

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CBSize?
Description	This query returns the committed burst size for the specified service.
Parameter	<no> = Service number (1-8)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:CBS? → 10000`
Note	

12.25.35 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:EBSize

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:EBSize <size>
Description	This command sets the excess burst size for the specified service. Unit: Bytes.
Parameters	<no> = Service number (1-8)
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1000000000, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:EBS 10000`
Note	If set to zero, the EBS test is excluded. If set to a value greater than zero but below the MTU size then the MTU size will be applied.

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:EBSize?
Description	This query returns the excess burst size for the specified service.
Parameter	<no> = Service number (1-8)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:EBS? → 1`
Note	

12.25.36 ETHernet:SATest:SETup:SERVice<no>:RTLocal:EBSize

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:EBSize <size>
Description	This command sets the excess burst size for the specified service. Unit: Bytes.
Parameters	<no> = Service number (1-8)
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 1000000000, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:EBS 1`
Note	If set to zero, the EBS test is excluded. If set to a value greater than zero but below the MTU size then the MTU size will be applied.

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:EBSize?
Description	This query returns the excess burst size for the specified service.
Parameter	<no> = Service number (1-8)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:EBS?` → `1`
Note	

12.25.37 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare[:ENABle]

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare[:ENABle] <enable>
Description	This command enables/disables the color awareness of a service.
Parameters	<no> = Service number (1-8)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:SAT:SET:SERV1:CAV OFF`
Note	When disabled the color mode is colorblind.

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare[:ENABle]?
Description	This query returns whether or not color awareness of a service service is enabled.
Parameter	<no> = Service number (1-8)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:CAV?` → `0`
Note	

12.25.38 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare[:ENABle]

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare[:ENABle] <enable>
Description	This command enables/disables the color awareness of a service.
Parameters	<no> = Service number (1-8)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:CAV OFF`
Note	When disabled the color mode is colorblind.

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare[:ENABle]?
Description	This query returns whether or not color awareness of a service service is enabled.
Parameter	<no> = Service number (1-8)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:CAV?` → `0`
Note	

12.25.39 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:CMEThod

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:CMEThod <method>
Description	This command sets the color method for the specified service.
Parameters	<no> = Service number (1-8)
	<method> = <CHARACTER PROGRAM DATA> DSCP: Differentiated Services Code Point PCP: Priority Code Point *DEFault = DSCP*
Response	None.
Example	`ETH:SAT:SET:SERV1:CAV:CMET DSCP`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:CMEThod?
Description	This query returns the color method for the specified service.
Parameter	<no> = Service number (1-8)
Response	<method> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:CAV:CMET? → DSCP`
Note	

12.25.40 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:CMEThod

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:CMEThod <method>
Description	This command sets the color method for the specified service.
Parameters	<no> = Service number (1-8)
	<method> = <CHARACTER PROGRAM DATA> DSCP: Differentiated Service Code Point PCP: Priority Code Point *DEFault = DSCP*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:CAV:CMET DSCP`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:CMEThod?
Description	This query returns the color method for the specified service.
Parameter	<no> = Service number (1-8)
Response	<method> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:CAV:CMET? → DSCP`
Note	

12.25.41 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:DSCP:GREen

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:DSCP:GREen <value>
Description	This command sets the green value for the DSCP color method for the specified service.
Parameters	<no> = Service number (1-8)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 63, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:CAV:DSCP:GRE 1`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:DSCP:GREen?
Description	This query returns the green value for the DSCP color method for the specified service.
Parameter	<no> = Service number (1-8)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:CAV:DSCP:GRE? → 1`
Note	

12.25.42 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:DSCP:GREen

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:DSCP:GREen <value>
Description	This command sets the green value for the DSCP color method for the specified service.
Parameters	<no> = Service number (1-8)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 63, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:CAV:DSCP:GRE 1`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:DSCP:GREen?
Description	This query returns the green value for the DSCP color method for the specified service.
Parameter	<no> = Service number (1-8)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:CAV:DSCP:GRE?` → `1`
Note	

12.25.43 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:DSCP :YELLow

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:DSCP:YELLow <value>
Description	This command sets the yellow value for the DSCP color method for the specified service.
Parameters	<no> = Service number (1-8)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 63, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:CAV:DSCP:YELL 1`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:DSCP:YELLow?
Description	This query returns the yellow value for the DSCP color method for the specified service.
Parameter	<no> = Service number (1-8)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:CAV:DSCP:YELL?` → `1`
Note	

12.25.44 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:DSCP:YELLow

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:DSCP:YELLow <value>
Description	This command sets the yellow value for the DSCP color method for the specified service.
Parameters	<no> = Service number (1-8)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 63, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:CAV:DSCP:YELL 1`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:DSCP:YELLow?
Description	This query returns the yellow value for the DSCP color method for the specified service.
Parameter	<no> = Service number (1-8)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:CAV:DSCP:YELL?` → `1`
Note	

12.25.45 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:PCP:GREen

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:PCP:GREen <value>
Description	This command sets the green value for the PCP color method for the specified service.
Parameters	<no> = Service number (1-8)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:CAV:PCP:GRE 1`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:PCP:GREen?
Description	This query returns the green value for the PCP color method for the specified service.
Parameter	<no> = Service number (1-8)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:CAV:PCP:GRE?` → `1`
Note	

12.25.46 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:PCP:GREen

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:PCP:GREen <value>
Description	This command sets the green value for the PCP color method for the specified service.
Parameters	<no> = Service number (1-8)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:CAV:PCP:GRE 1`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:PCP:GREen?
Description	This query returns the green value for the PCP color method for the specified service.
Parameter	<no> = Service number (1-8)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:CAV:PCP:GRE?` → `1`
Note	

12.25.47 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:PCP :YELLow

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:PCP:YELLow <value>
Description	This command sets the yellow value for the PCP color method for the specified service.
Parameters	<no> = Service number (1-8)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:CAV:PCP:YELL 1`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CAVare:PCP:YELLow?
Description	This query returns the yellow value for the PCP color method for the specified service.
Parameter	<no> = Service number (1-8)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:CAV:PCP:YELL?` → `1`
Note	

12.25.48 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:PCP:YELLow

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:PCP:YELLow <value>
Description	This command sets the yellow value for the PCP color method for the specified service.
Parameters	<no> = Service number (1-8)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 0*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:CAV:PCP:YELL 1`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CAVare:PCP:YELLow?
Description	This query returns the yellow value for the PCP color method for the specified service.
Parameter	<no> = Service number (1-8)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:CAV:PCP:YELL? → 1`
Note	

12.25.49 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria:FTDelay

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria:FTDelay <delay>
Description	This command sets the acceptable frame transfer delay value. Unit: ms.
Parameters	<no> = Service number (1-8)
	<delay> = <NUMERIC PROGRAM DATA> *MINimum = 0.000, MAXimum = 2000.000, DEFault = 0.500*
Response	None.
Example	`ETH:SAT:SET:SERV1:SACR:FTD 1.100`
Note	The precision is three decimals.

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria:FTDelay?
Description	This query returns the acceptable frame transfer delay value.
Parameter	<no> = Service number (1-8)
Response	<delay> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:SACR:FTD? → 10.000`
Note	

12.25.50 ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:FTDelay

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:FTDelay <delay>
Description	This command sets the acceptable frame transfer delay value. Unit: ms.
Parameters	<no> = Service number (1-8)
	<delay> = <NUMERIC PROGRAM DATA> *MINimum = 0.000, MAXimum = 2000.000, DEFault = 0.500*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:SACR:FTD 10.000`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:FTDelay?
Description	This query returns the acceptable frame transfer delay value.
Parameter	<no> = Service number (1-8)
Response	<delay> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:SACR:FTD? → 10.000`
Note	

12.25.51 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria :FDVariation

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria:FDVariation <variation>
Description	This command sets the acceptable frame delay variation value. Unit: ms.
Parameters	<no> = Service number (1-8)
	<variation> = <NUMERIC PROGRAM DATA> *MINimum = 0.000, MAXimum = 2000.000, DEFault = 0.050*
Response	None.
Example	`ETH:SAT:SET:SERV1:SACR:FDV 20.000`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria:FDVariation?
Description	This query returns the acceptable frame delay variation value.
Parameter	<no> = Service number (1-8)
Response	<variation> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:SACR:FDV? → 20.000`
Note	

12.25.52 ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:FDVariation

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:FDVariation <variation>
Description	This command sets the acceptable frame delay variation value. Unit: ms.
Parameters	<no> = Service number (1-8)
	<variation> = <NUMERIC PROGRAM DATA> *MINimum = 0.000, MAXimum = 2000.000, DEFault = 0.050*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:SACR:FDV 20.000`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:FDVariation?
Description	This query returns the acceptable frame delay variation value.
Parameter	<no> = Service number (1-8)
Response	<variation> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:SACR:FDV? → 20.000`
Note	

12.25.53 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria:FLRatio

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria:FLRatio <ratio>
Description	This command sets the acceptable frame loss ratio.
Parameters	<no> = Service number (1-8)
	<ratio> = <NUMERIC PROGRAM DATA> *MINimum = 0.00, MAXimum = 1.00, DEFault = 0.00*
Response	None.
Example	`ETH:SAT:SET:SERV1:SACR:FLR 0.000015` `ETH:SAT:SET:SERV1:SACR:FLR 15e-06`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria:FLRatio?
Description	This query returns the acceptable frame loss ratio.
Parameter	<no> = Service number (1-8)
Response	<ratio> = <NR3 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:SACR:FLR? → 1.50e-05`
Note	

12.25.54 ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:FLRatio

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:FLRatio <ratio>
Description	This command sets the acceptable frame loss ratio.
Parameters	<no> = Service number (1-8)
	<ratio> = <NUMERIC PROGRAM DATA> *MINimum = 0.00, MAXimum = 1.00, DEFault = 0.00*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:SACR:FLR 0.000015` `ETH:SAT:SET:SERV1:RTL:SACR:FLR 15e-06`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:FLRatio?
Description	This query returns the acceptable frame loss ratio.
Parameter	<no> = Service number (1-8)
Response	<ratio> = <NR3 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:SACR:FLR?` → `1.50e-05`
Note	

12.25.55 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria :AVAilability

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria:AVAilability <value>
Description	This command sets the acceptable availability. Unit: percentage.
Parameters	<no> = Service number (1-8)
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0.0000, MAXimum=100.0000, DEFault=100.0000*
Response	None.
Example	`ETH:SAT:SET:SERV1:SACR:AVA 99.0000`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:SACRiteria:AVAilability?
Description	This query returns the acceptable availability.
Parameter	<no> = Service number (1-8)
Response	<value> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:SACR:AVA?` → `99.0000`
Note	

12.25.56 ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:AVAilability

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:AVAilability <value>
Description	This command sets the acceptable availability. Unit: percentage.
Parameters	<no> = Service number (1-8)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0.0000, MAXimum = 100.0000, DEFault = 100.0000*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:SACR:AVA 99.0000`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:SACRiteria:AVAilability?
Description	This query returns the acceptable availability.
Parameter	<no> = Service number (1-8)
Response	<value> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:SACR:AVA?` → `99.0000`
Note	

12.25.57 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:FSMode

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:FSMode <mode>
Description	This command sets the configured frame size mode.
Parameters	<no> = Service number (1-8)
	<mode> = <CHARACTER PROGRAM DATA> CONStant: Fixed frame size [1] EMIX: A mix of frame sizes [2] *DEFault = CONStant*
Response	None.
Example	`ETH:SAT:SET:SERV1:FSM CONS`
Note	[1] Uses the frame size defined by the command: `ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:FSIZe`. [2] Uses the frame sizes defined by the command: `ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:EMIX`.

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:FSMode?
Description	This query returns the configured frame size.
Parameter	<no> = Service number (1-8)
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:FSM? → CONS`
Note	

12.25.58 ETHernet:SATest:SETup:SERVice<no>:RTLocal:FSMode

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:FSMode <mode>
Description	This command sets the configured frame size mode.
Parameters	<no> = Service number (1-8)
	<mode> = <CHARACTER PROGRAM DATA> CONStant: Fixed frame size [1] EMIX: A mix of frame sizes [2] *DEFault = CONStant*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:FSM CONS`
Note	[1] Uses the frame size defined by the command: `ETHernet:SATest:SETup:SERVice<no>:RTLocal:FSIZe`. [2] Uses the frame sizes defined by the command: `ETHernet:SATest:SETup:SERVice<no>:RTLocal:EMIX`.

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:FSMode?
Description	This query returns the configured frame size.
Parameter	<no> = Service number (1-8)
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:FSM? → CONS`
Note	

12.25.59 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CFSize

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CFSize <size>
Description	This command sets the constant frame size.
Parameters	<no> = Service number (1-8)
	<size> = <CHARACTER PROGRAM DATA> 64: 64 bytes 128: 128 bytes 256: 256 bytes 512: 512 bytes

continued on next page...

. . . continued from previous page

	1024: 1024 bytes 1280: 1280 bytes 1518: 1518 bytes MTU: Use the MTU value [1] USER: Use the user defined value [2] *DEFault = 512*
Response	None.
Example	`ETH:SAT:SET:SERV1:CFS 256`
Note	[1] The MTU value is modified by the command: `ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:MTU`. [2] The user defined value is modified by the command: `ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:UDFSize`.

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:CFSize?
Description	This query returns the constant frame size.
Parameter	<no> = Service number (1-8)
Response	<size> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:CFS? → 256`
Note	

12.25.60 ETHernet:SATest:SETup:SERVice<no>:RTLocal:CFSize

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CFSize <size>
Description	This command sets the constant frame size.
Parameters	<no> = Service number (1-8)
	<size> = <CHARACTER PROGRAM DATA> 64: 64 bytes 128: 128 bytes 256: 256 bytes 512: 512 bytes 1024: 1024 bytes 1280: 1280 bytes 1518: 1518 bytes MTU: Use the MTU value [1] USER: Use the user defined value [2] *DEFault = 512*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:CFS 256`
Note	[1] The MTU value is modified by the command: `ETHernet:SATest:SETup:SERVice<no>:RTLocal:MTU`. [2] The user defined value is modified by the command: `ETHernet:SATest:SETup:SERVice<no>:RTLocal:UDFSize`.

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:CFSize?
Description	This query returns the constant frame size.
Parameter	<no> = Service number (1-8)
Response	<size> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:CFS? → 256`
Note	

12.25.61 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:EMIX

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:EMIX <mix>
Description	This command defines the EMIX sequence.

continued on next page. . .

. . . continued from previous page

Parameters	<no> = Service number (1-8)
	<mix> = <STRING PROGRAM DATA> This quoted string defines the EMIX sequence by a sequence of the letters 'a'-'h' or 'u'. Specify: 'a' for a 64 bytes frame 'b' for a 128 bytes frame 'c' for a 256 bytes frame 'd' for a 512 bytes frame 'e' for a 1024 bytes frame 'f' for a 1280 bytes frame 'g' for a 1518 bytes frame 'h' for a MTU sized frame [1] 'u' for a user defined sized frame [2] *DEFault = "abceg"*
Response	None.
Example	`ETH:SAT:SET:SERV1:EMIX "ababccdddhu"`
Note	Illegal characters are removed from the EMIX string and if the string is more than 16 characters long, the string will be truncated. [1] The MTU value is modified by the command: `ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:MTU`. [2] The user defined value is modified by the command: `ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:UDFSize`.

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:EMIX?
Description	This query returns the EMIX sequence.
Parameter	<no> = Service number (1-8)
Response	<emix> = <STRING RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:EMIX?` → `"ababccdddhu"`
Note	

12.25.62 ETHernet:SATest:SETup:SERVice<no>:RTLocal:EMIX

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:EMIX <mix>
Description	This command defines the EMIX sequence.
Parameters	<no> = Service number (1-8)
	<mix> = <STRING PROGRAM DATA> This quoted string defines the EMIX sequence by a sequence of the letters 'a'-'h' or 'u'. Specify: 'a' for a 64 bytes frame 'b' for a 128 bytes frame 'c' for a 256 bytes frame 'd' for a 512 bytes frame 'e' for a 1024 bytes frame 'f' for a 1280 bytes frame 'g' for a 1518 bytes frame 'h' for a MTU sized frame [1] 'u' for a user defined sized frame [2] *DEFault = "abceg"*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:EMIX "ababccdddhu"`
Note	Illegal characters are removed from the EMIX string and if the string is more than 16 characters long, the string will be truncated. [1] The MTU value is modified by the command: `ETHernet:SATest:SETup:SERVice<no>:RTLocal:MTU`. [2] The user defined value is modified by the command: `ETHernet:SATest:SETup:SERVice<no>:RTLocal:UDFSize`.

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:EMIX?
Description	This query returns the EMIX sequence.
Parameter	<no> = Service number (1-8)
Response	<emix> = <STRING RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:EMIX?` → `"ababccdddhu"`
Note	

12.25.63 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:MTU

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:MTU <value>
Description	This command sets the MTU value. Unit: bytes.
Parameters	<no> = Service number (1-8)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 1518, MAXimum = 16000, DEFault = 1518*
Response	None.
Example	`ETH:SAT:SET:SERV1:MTU 10101`

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:MTU?
Description	This query returns the MTU value.
Parameter	<no> = Service number (1-8)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:MTU?` → `10101`
Note	

12.25.64 ETHernet:SATest:SETup:SERVice<no>:RTLocal:MTU

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:MTU <value>
Description	This command sets the MTU value. Unit: bytes.
Parameters	<no> = Service number (1-8)
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 1518, MAXimum = 16000, DEFault = 1518*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:MTU 10101`

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:MTU?
Description	This query returns the MTU value.
Parameter	<no> = Service number (1-8)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:MTU?` → `10101`
Note	

12.25.65 ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:UDFSize

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:UDFSize <size>
Description	This command sets the user defined frame size.
Parameters	<no> = Service number (1-8)
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 46* [1]*, MAXimum = 16000, DEFault = 512*
Response	None.
Example	`ETH:SAT:SET:SERV1:UDFS 111`
Note	[1] The minimum allowed frame size varies depending on the frame content setup.

Syntax	ETHernet:SATest:SETup:SERVice<no>[:LTRemote]:UDFSize?
Description	This query returns the user defined frame size.
Parameter	<no> = Service number (1-8)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:UDFS?` → `111`
Note	

12.25.66 ETHernet:SATest:SETup:SERVice<no>:RTLocal:UDFSize

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:UDFSize <size>
Description	This command sets the user defined frame size.
Parameters	<no> = Service number (1-8)
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 46* [1]*, MAXimum = 16000, DEFault = 512*
Response	None.
Example	`ETH:SAT:SET:SERV1:RTL:UDFS 111`
Note	[1] The minimum allowed frame size varies depending on the frame content setup.

Syntax	ETHernet:SATest:SETup:SERVice<no>:RTLocal:UDFSize?
Description	This query returns the user defined frame size.
Parameter	<no> = Service number (1-8)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:RTL:UDFS?` → `111`
Note	

12.25.67 ETHernet:SATest:SETup:SERVice<no>:SCTSteps:SDURation

Syntax	ETHernet:SATest:SETup:SERVice<no>:SCTSteps:SDURation <duration>
Description	This command sets the service configuration test steps duration. Unit: seconds.
Parameters	<no> = Service number (1-8)
	<duration> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 60, DEFault = 3*
Response	None.
Example	`ETH:SAT:SET:SERV1:SCTS:SDUR 5`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:SCTSteps:SDURation?
Description	This query returns the service configuration test steps duration.
Parameter	<no> = Service number (1-8)
Response	<duration> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:SCTS:SDUR?` → `5`
Note	

12.25.68 ETHernet:SATest:SETup:SERVice<no>:SCTSteps:NSTep

Syntax	ETHernet:SATest:SETup:SERVice<no>:SCTSteps:NSTep <duration>
Description	This command sets the service configuration test number of steps.
Parameters	<no> = Service number (1-8)
	<steps> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 10, DEFault = 4*
Response	None.
Example	`ETH:SAT:SET:SERV1:SCTS:NST 2`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:SCTSteps:NSTep?
Description	This query returns the service configuration test number of steps.
Parameter	<no> = Service number (1-8)
Response	<steps> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:SCTS:NST?` → `2`
Note	

12.25.69 ETHernet:SATest:SETup:SERVice<no>:SCTSteps:SLOPe

Syntax	ETHernet:SATest:SETup:SERVice<no>:SCTSteps:SLOPe <slope>
Description	This command sets the service configuration test slope.
Parameters	<no> = Service number (1-8)
	<slope> = <CHARACTER PROGRAM DATA> ASCending: Use an ascending slope (lowest rate first) DESCending: Use a descending slope (CIR rate first) *DEFault = ASCending*
Response	None.
Example	`ETH:SAT:SET:SERV1:SCTS:SLOP ASC`
Note	

Syntax	ETHernet:SATest:SETup:SERVice<no>:SCTSteps:SLOPe?
Description	This query returns the service configuration test slope.
Parameter	<no> = Service number (1-8)
Response	<slope> = <CHARACTER RESPONSE DATA>
Example	`ETH:SAT:SET:SERV1:SCTS:SLOP?` → `ASC`
Note	

12.25.70 ETHernet:SATest:RESult:TSYNchronization?

Syntax	ETHernet:SATest:RESult:TSYNchronization?
Description	Returns the status of the timing synchronization during the service activation test period. For one-way testing the signals from the GPS system are needed to achieve the required accuracy of the frame transfer delay measurement.
Parameter	None.
Response	<local-status> = <STRING RESPONSE DATA> "Pending" or "Fail" or "Pass" denotes the synchronization status on the local side.
	<remote-status> = <STRING RESPONSE DATA> "Pending" or "Fail" or "Pass" denotes the synchronization status on the remote side.
Example	`ETH:SAT:RES:TSYN?` → `Pass,Fail`
Note	

12.25.71 ETHernet:SATest:RESult:SCTest[:LTRemote]?

Syntax	ETHernet:SATest:RESult:SCTest[:LTRemote]?
Description	Returns the overall result of all service configuration tests in the local to remote direction.
Parameter	None.
Response	<status> = <STRING RESPONSE DATA> "Pending" or "Fail" or "Pass" denotes the status of the test step.
Example	`ETH:SAT:RES:SCT?` → `Pass`
Note	

12.25.72 ETHernet:SATest:RESult:SCTest:RTLocal?

Syntax	ETHernet:SATest:RESult:SCTest:RTLocal?
Description	Returns the overall result of all service configuration tests in the remote to local direction.
Parameter	None.
Response	<status> = <STRING RESPONSE DATA> "Pending" or "Fail" or "Pass" denotes the status of the test step.
Example	`ETH:SAT:RES:SCT:RTL?` → `Pass`
Note	

12.25.73 ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:CIRate?

Syntax	ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:CIRate? <step>
Description	Returns the results of the specified step of a service configuration test.
Parameters	<no> = Service number (1-8)
	<step> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 10, DEFault = 1*
Response	<status> = <STRING RESPONSE DATA> "Fail" or "Pass" denotes the status of the test step.
	<IR-min> = <NR2 NUMERIC RESPONSE DATA> Minimum information rate. Unit: Mbps.
	<IR-avg> = <NR2 NUMERIC RESPONSE DATA> Average information rate. Unit: Mbps.
	<IR-max> = <NR2 NUMERIC RESPONSE DATA> Maximum information rate. Unit: Mbps.
	<FL-count> = <NR1 NUMERIC RESPONSE DATA> Frame loss. Unit: Frames.
	<FL-rate> = <NR3 NUMERIC RESPONSE DATA> Frame loss ratio.
	<FTD-min> = <NR1 NUMERIC RESPONSE DATA> Minimum frame transfer delay. Unit: ms.
	<FTD-avg> = <NR1 NUMERIC RESPONSE DATA> Average frame transfer delay. Unit: ms.

continued on next page...

. . . continued from previous page

	<FTD-max> = <NR1 NUMERIC RESPONSE DATA> Maximum frame transfer delay. Unit: ms.
	<FDV-min> = <NR1 NUMERIC RESPONSE DATA> Minimum frame delay variation. Unit: ms.
	<FDV-avg> = <NR1 NUMERIC RESPONSE DATA> Average frame delay variation. Unit: ms.
	<FDV-max> = <NR1 NUMERIC RESPONSE DATA> Maximum frame delay variation. Unit: ms.
Example	`ETH:SAT:RES:SCT:SERV1:CIR? 1` → `Pass,1.00,2.00,3.00,0,0.00E+00,1,2,3,1,2,3`
Note	

12.25.74 ETHernet:SATest:RESult:SCTest:SERVice<no>:RTLocal:CIRate?

Syntax	ETHernet:SATest:RESult:SCTest:SERVice<no>:RTLocal:CIRate? <step>
Description	Returns the results of the specified step of a service configuration test.
Parameters	<no> = Service number (1-8)
	<step> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 10, DEFault = 1*
Response	The same response parameters as for the command: `ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:CIRate`
Example	`ETH:SAT:RES:SCT:SERV1:RTL:CIR? 1` → `Pass,1.00,2.00,3.00,0,0.00E+00,1,2,3,1,2,3`
Note	

12.25.75 ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:EIRate?

Syntax	ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:EIRate? [<color>]
Description	Returns the results of the specified excess information rate test.
Parameters	<no> = Service number (1-8)
	<color> = <CHARACTER PROGRAM DATA> GREen: Green color counters YELLow: Yellow color counters TOTal: In color blind mode this is the only result available. Sum of green and yellow color counters when color aware. *DEFault = TOTal*
Response	The same response parameters as for the command: `ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:CIRate`
Example	`ETH:SAT:RES:SCT:SERV1:EIR? GRE` → `Pass,1.00,2.00,3.00,0,0.00E+00,1,2,3,1,2,3`
Note	

12.25.76 ETHernet:SATest:RESult:SCTest:SERVice<no>:RTLocal:EIRate?

Syntax	ETHernet:SATest:RESult:SCTest:SERVice<no>:RTLocal:EIRate? [<color>]
Description	Returns the results of the specified excess information rate test.
Parameters	<no> = Service number (1-8)
	<color> = <CHARACTER PROGRAM DATA> GREen: Green color counters YELLow: Yellow color counters TOTal: In color blind mode this is the only result available. Sum of green and yellow color counters when color aware. *DEFault = TOTal*
Response	The same response parameters as for the command: `ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:CIRate`
Example	`ETH:SAT:RES:SCT:SERV1:RTL:EIR? GRE` → `Pass,1.00,2.00,3.00,0,0.00E+00,1,2,3,1,2,3`
Note	

12.25.77 ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:TPOLicing?

Syntax	ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:TPOLicing? [<color>]
Description	Returns the results of the specified traffic policing test.
Parameters	<no> = Service number (1-8)
	<color> = <CHARACTER PROGRAM DATA> GREen: Green color counters YELLow: Yellow color counters TOTal: In color blind mode this is the only result available. Sum of green and yellow color counters when color aware. *DEFault = TOTal*
Response	The same response parameters as for the command: `ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:CIRate`
Example	`ETH:SAT:RES:SCT:SERV1:TPOL? GRE` → `Pass,1.00,2.00,3.00,0,0.00E+00,1,2,3,1,2,3`
Note	

12.25.78 ETHernet:SATest:RESult:SCTest:SERVice<no>:RTLocal:TPOLicing?

Syntax	ETHernet:SATest:RESult:SCTest:SERVice<no>:RTLocal:TPOLicing? [<color>]
Description	Returns the results of the specified traffic policing test.
Parameters	<no> = Service number (1-8)
	<color> = <CHARACTER PROGRAM DATA> GREen: Green color counters YELLow: Yellow color counters TOTal: In color blind mode this is the only result available. Sum of green and yellow color counters when color aware. *DEFault = TOTal*
Response	The same response parameters as for the command: `ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:CIRate`
Example	`ETH:SAT:RES:SCT:SERV1:RTL:TPOL? GRE` → `Pass,1.00,2.00,3.00,0,0.00E+00,1,2,3,1,2,3`
Note	

12.25.79 ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:CBSize?

Syntax	ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:CBSize?
Description	Returns the results of the committed burst size test.
Parameter	<no> = Service number (1-8)
Response	The same response parameters as for the command: `ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:CIRate`
Example	`ETH:SAT:RES:SCT:SERV1:CBS?` → `Pass,1.00,2.00,3.00,0,0.00E+00,1,2,3,1,2,3`
Note	

12.25.80 ETHernet:SATest:RESult:SCTest:SERVice<no>:RTLocal:CBSize?

Syntax	ETHernet:SATest:RESult:SCTest:SERVice<no>:RTLocal:CBSize?
Description	Returns the results of the committed burst size test.
Parameter	<no> = Service number (1-8)
Response	The same response parameters as for the command: `ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:CIRate`
Example	`ETH:SAT:RES:SCT:SERV1:RTL:CBS?` → `Pass,1.00,2.00,3.00,0,0.00E+00,1,2,3,1,2,3`
Note	

12.25.81 ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:EBSize?

Syntax	ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:EBSize? [<color>]
Description	Returns the results of the excess burst size test.
Parameters	<no> = Service number (1-8)
	<color> = <CHARACTER PROGRAM DATA> GREen: Green color counters YELLow: Yellow color counters TOTal: In color blind mode this is the only result available. Sum of green and yellow color counters when color aware. *DEFault = TOTal*
Response	The same response parameters as for the command: `ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:CIRate`
Example	`ETH:SAT:RES:SCT:SERV1:EBS? GRE` → `Pass,1.00,2.00,3.00,0,0.00E+00,1,2,3,1,2,3`
Note	

12.25.82 ETHernet:SATest:RESult:SCTest:SERVice<no>:RTLocal:EBSize?

Syntax	ETHernet:SATest:RESult:SCTest:SERVice<no>:RTLocal:EBSize? [<color>]
Description	Returns the results of the excess burst size test.
Parameters	<no> = Service number (1-8)
	<color> = <CHARACTER PROGRAM DATA> GREen: Green color counters YELLow: Yellow color counters TOTal: In color blind mode this is the only result available. Sum of green and yellow color counters when color aware. *DEFault = TOTal*
Response	The same response parameters as for the command: `ETHernet:SATest:RESult:SCTest:SERVice<no>[:LTRemote]:CIRate`
Example	`ETH:SAT:RES:SCT:SERV1:RTL:EBS? GRE` → `Pass,1.00,2.00,3.00,0,0.00E+00,1,2,3,1,2,3`
Note	

12.25.83 ETHernet:SATest:RESult:SPTest[:LTRemote]?

Syntax	ETHernet:SATest:RESult:SPTest[:LTRemote]?
Description	Returns the overall result of all service performance tests in the local to remote direction.
Parameter	None.
Response	<status> = <STRING RESPONSE DATA> "Pending" or "Fail" or "Pass" denotes the status of the test step.
Example	`ETH:SAT:RES:SPT?` → `Pass`
Note	

12.25.84 ETHernet:SATest:RESult:SPTest:RTLocal?

Syntax	ETHernet:SATest:RESult:SPTest:RTLocal?
Description	Returns the overall result of all service performance tests in the remote to local direction.
Parameter	<no> = Service number (1-8)
Response	<status> = <STRING RESPONSE DATA> "Pending" or "Fail" or "Pass" denotes the status of the test step.
Example	`ETH:SAT:RES:SPT:RTL?` → `Pass`
Note	

12.25.85 ETHernet:SATest:RESult:SPTest:SERVice<no>[:LTRemote]?

Syntax	ETHernet:SATest:RESult:SPTest:SERVice<no>[:LTRemote]?
Description	Returns the results of a service performance test of a service.
Parameter	<no> = Service number (1-8)
Response	<status> = <STRING RESPONSE DATA> "Fail" or "Pass" denotes the status of the test step.
	<IR-min> = <NR2 NUMERIC RESPONSE DATA> Minimum information rate. Unit: Mbps.
	<IR-avg> = <NR2 NUMERIC RESPONSE DATA> Average information rate. Unit: Mbps.
	<IR-max> = <NR2 NUMERIC RESPONSE DATA> Maximum information rate. Unit: Mbps.
	<FL-count> = <NR1 NUMERIC RESPONSE DATA> Frame loss. Unit: Frames.
	<FL-rate> = <NR3 NUMERIC RESPONSE DATA> Frame loss ratio.
	<FTD-min> = <NR1 NUMERIC RESPONSE DATA> Minimum frame transfer delay. Unit: ms.
	<FTD-avg> = <NR1 NUMERIC RESPONSE DATA> Average frame transfer delay. Unit: ms.
	<FTD-max> = <NR1 NUMERIC RESPONSE DATA> Maximum frame transfer delay. Unit: ms.
	<FDV-min> = <NR1 NUMERIC RESPONSE DATA> Minimum frame delay variation. Unit: ms.
	<FDV-avg> = <NR1 NUMERIC RESPONSE DATA> Average frame delay variation. Unit: ms.
	<FDV-max> = <NR1 NUMERIC RESPONSE DATA> Maximum frame delay variation. Unit: ms.
	<Availability> = <NR2 NUMERIC RESPONSE DATA> Availability (link). Unit: percentage.
	<Unavailable> = <NR1 NUMERIC RESPONSE DATA> Unavailable time. Unit: seconds.
Example	`ETH:SAT:RES:SPT:SERV1?` → `Pass,1.00,2.00,3.00,0,0.00E+00,1,2,3,1,2,3,100.00,0`
Note	

12.25.86 ETHernet:SATest:RESult:SPTest:SERVice<no>:RTLocal?

Syntax	ETHernet:SATest:RESult:SPTest:SERVice<no>:RTLocal?
Description	Returns the results of a service performance test of a service.
Parameter	<no> = Service number (1-8)
Response	The same response parameters as for the command: `ETHernet:SATest:RESult:SPTest:SERVice<no>[:LTRemote]?`
Example	`ETH:SAT:RES:SPT:SERV1:RTL?` → `Pass,1.00,2.00,3.00,0,0.00E+00,1,2,3,1,2,3,100.00,0`
Note	

12.26 Cable Test

12.26.1 ETHernet:CABLe:STARt

Syntax	ETHernet:CABLe:STARt
Description	This command starts the cable test.
Parameter	None.
Response	None.
Example	`ETH:CABL:STAR`
Note	

12.26.2 ETHernet:CABLe:STOP

Syntax	ETHernet:CABLe:STOP
Description	This command stops the cable test.
Parameter	None.
Response	None.
Example	`ETH:CABL:STOP`
Note	

12.26.3 ETHernet:CABLe:CCStandard

Syntax	ETHernet:CABLe:CCStandard <ccs>
Description	This command sets the color coding standard.
Parameter	<port> = <CHARACTER PROGRAM DATA> T568A: CAT5 T568B: CAT5E T568BCAT6: CAT6, CAT6A *DEFault = T568A*
Response	None.
Example	`ETH:CABL:CCS T568B`
Notes	T568A: Pair 1: Pin 4,5 Pair 2: Pin 3,6 Pair 3: Pin 1,2 Pair 4: Pin 7,8 T568B, T568BCAT6: Pair 1: Pin 4,5

continued on next page...

. . . continued from previous page

	Pair 2: Pin 1,2 Pair 3: Pin 3,6 Pair 4: Pin 7,8

Syntax	ETHernet:CABLe:CCStandard?
Description	This query returns the color coding standard.
Parameter	None.
Response	<ccs> = <CHARACTER RESPONSE DATA>
Example	`ETH:CABL:CCS?` → `T568B`
Note	

12.26.4 ETHernet:CABLe:RESults:PAIR<Pr>?

Syntax	ETHernet:CABLe:RESults:PAIR<Pr>?
Description	This query returns the status of a pair.
Parameter	<Pr> = Pair number (1-4)
Response	<status> = <CHARACTER RESPONSE DATA> OK: Pair is ok SHRT: Pair is shorted OPEN: Pair is open FAIL: Pair is fail(Pair Busy/Invalid) NA: Status not available
	<distance> = <NR2 NUMERIC RESPONSE DATA> Distance to fault. Unit: Meters. If there is no fault it will return 0.0.

Examples	`ETH:CABL:RES:PAIR1?` → `OK,0.0`
	`ETH:CABL:RES:PAIR2?` → `SHRT,20.6`
	`ETH:CABL:RES:PAIR3?` → `OPEN,2.4`

Note	Results cannot be obtained during testing.

12.27 IP Channel Statistics

12.27.1 ETHernet:CSTat:STARt

Syntax	ETHernet:CSTat:STARt
Description	This command starts the IP Channel Statistics test.
Parameter	None.
Response	None.
Example	`ETH:CST:STAR`
Note	This command will make the PHY reset, which will results in link loss for a couple of seconds. This command can be used on V2.00 or later.

12.27.2 ETHernet:CSTat:STOP

Syntax	ETHernet:CSTat:STOP
Description	This command stops the IP Chanel Statistics test.
Parameter	None.
Response	None.
Example	`ETH:CST:STOP`
Note	This command will make the phy reset, which will results in link loss for a couple of seconds. This command can be used on V2.00 or later.

12.27.3 ETHernet:CSTat:SETup:FOLLow

Syntax	ETHernet:CSTat:SETup:FOLLow <enable>
Description	This command enables/disables PORT2 follow PORT1.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:CST:SET:FOLL OFF`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:CSTat:SETup:FOLLow?
Description	This query returns whether or not PORT2 follows PORT1.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:CST:SET:FOLL?` → `0`
Note	This command can be used on V2.00 or later.

12.27.4 ETHernet:CSTat:SETup:PORT<Pt>:FOLLow

Syntax	ETHernet:CSTat:SETup:PORT<Pt>:FOLLow <enable>
Description	This command sets CStat setup for port <Pt> to follow PORT1.
Parameter	<Pt> = Port number (2-4)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:CST:SET:PORT2:FOLL ON`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:CSTat:SETup:PORT<Pt>:FOLLow?
Description	This query returns whether or not CStat setup for port <Pt> follows PORT1.
Parameter	<Pt> = Port number (2-4)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:CST:SET:PORT2:FOLL?` → `1`
Note	This command can be used on V2.00 or later.

12.27.5 ETHernet:CSTat:SETup:PORT<Pt>:CDEFinitions

Syntax	ETHernet:CSTat:SETup:PORT<Pt>:CDEFinitions <definitions>
Description	This command sets the active channel definitions.
Parameters	<Pt> = Port number
	({<definitions>} + {,}*) = <EXPRESSION PROGRAM DATA> SMAC: Source MAC address DMAC: Destination MAC address PINF: Protocol information VLAN: VLAN MPLS: MPLS SIP: Source IP address DIP: Destination IP address IPNH: IP next header SPOR: Source TCP/UDP port DPOR: Destination TCP/UDP port
Response	None.
Example	`ETH:CST:SET:PORT1:CDEF (SMAC,SPOR)`
Note	Changing the channel definitions while a test is running will reset the results for both ports. This command can be used on V2.00 or later.

Syntax	ETHernet:CSTat:SETup:PORT<Pt>:CDEFinitions?
Description	This query returns the active channel definitions.
Parameter	<Pt> = Port number
Response	{(<definition>),}* = <EXPRESSION RESPONSE DATA> <definition> = <CHARACTER RESPONSE DATA>
Example	`ETH:CST:SET:PORT1:CDEF?` → `(SMAC,SPOR)`
Note	This command can be used on V2.00 or later.

12.27.6 ETHernet:CSTat:SETup:PORT<Pt>:MERGe

Syntax	ETHernet:CSTat:SETup:PORT<Pt>:MERGe <definition>
Description	This command sets which channel definition to merge.
Parameters	<Pt> = Port number
	<definition> = <CHARACTER PROGRAM DATA> SMAC: Source MAC address DMAC: Destination MAC address PINF: Protocol information VLAN: VLAN MPLS: MPLS SIP: Source IP address DIP: Destination IP address IPNH: IP next header SPOR: Source TCP/UDP port DPOR: Destination TCP/UDP port SDMP: Source and destination MAC address pairs OFF: Undo merge. *DEFault = OFF*
Response	None.
Example	`ETH:CST:SET:PORT1:MERG SMAC`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:CSTat:SETup:PORT<Pt>:MERGe?
Description	This query returns the currently merged channel definition.
Parameter	<Pt> = Port number
Response	<definition> = <CHARACTER RESPONSE DATA>
Example	`ETH:CST:SET:PORT1:MERG?` → `SMAC`
Note	This command can be used on V2.00 or later.

12.27.7 ETHernet:CSTat:SETup:PORT<Pt>:FRESults

Syntax	ETHernet:CSTat:SETup:PORT<Pt>:FRESults <format>
Description	This command sets the format of the IP Channel Statistics test results.
Parameters	<Pt> = Port number
	<format> = <CHARACTER PROGRAM DATA> UNF: Unformatted SI: SI prefix notation ENG: Engineering exponent notation HEX: Hexadecimal notation *DEFault = SI*
Response	None.
Example	`ETH:CST:SET:PORT1:FRES ENG`
Note	This only affects the GUI and reports. This command can be used on V2.00 or later.

Syntax	ETHernet:CSTat:SETup:PORT<Pt>:FRESults?
Description	This query returns the format of the IP Channel Statistics test results.
Parameter	<Pt> = Port number
Response	<format> = <CHARACTER RESPONSE DATA>
Example	`ETH:CST:SET:PORT1:FRES?` → `ENG`
Note	This command can be used on V2.00 or later.

12.27.8 ETHernet:CSTat:RESults:PORT<Pt>:NCHannels?

Syntax	ETHernet:CSTat:RESults:PORT<Pt>:NCHannels?
Description	This query returns the maximum number of channels.
Parameter	<Pt> = Port number
Response	<max> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:CST:RES:PORT1:NCH?` → `3`
Note	If no channels are available NaN (section 1.6.1) is returned. This command can be used on V2.00 or later.

12.27.9 ETHernet:CSTat:RESults:PORT<Pt>:DFRames?

Syntax	ETHernet:CSTat:RESults:PORT<Pt>:DFRames?
Description	This query returns the number of discarded frames.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:CST:RES:PORT1:DFR?` → `3`
Note	On high speed links the Channel Statistics may sometimes discard frames due to overload. When this happens, the number of frames discarded can be read using this command. This command can be used on V2.00 or later.

12.27.10 ETHernet:CSTat:RESults:PORT<Pt>:FETCh?

Syntax	ETHernet:CSTat:RESults:PORT<Pt>:FETCh? <number>, <parameter>
Description	This query returns Channel Stat results.
Parameters	<Pt> = Port number
	<number> = <NUMERIC PROGRAM DATA> The channel number from which to fetch results from. *MINimum = 1, DEFault = 1*
	({<parameter>} + {,}*) = <EXPRESSION PROGRAM DATA> **Channel Definitions** SMAC: Source MAC address DMAC: Destination MAC address

continued on next page...

. . . continued from previous page

	PINF: Protocol information VLAN: VLAN MPLS: MPLS SIP: Source IP address DIP: Destination IP address IPNH: IP next header SPOR: Source TCP/UDP port DPOR: Destination TCP/UDP port **Frame Statistics** FCT: Frame count FRAT: Frame rate BCT: Byte count THR: Throughput OFR: Over Frames UFR: Under Frames **Size Distribution Statistics** S64: 64-127 S128: 128-255 S256: 256-511 S512: 512-1023 S1024: 1024-1518 S1519: Above 1519 **MPLS Statistics** MPLSF: MPLS frames MPLSB: MPLS bytes **IP Statistics** IPP: IP packets IPPR: IP packet rate IPB: IP bytes IPTH: IP throughput IPHB: IP header bytes IPF: IP fragments TTLV: TTL violations **IPv4 Statistics** IP4P: IPv4 packets IP4PR: IPv4 packet rate IP4B: IPv4 bytes IP4TH: IPv4 Throughput IP4HB: IPv4 header bytes IP4HE: IPv4 header errors **IPv6 Statistics** IP6P: IPv6 packets IP6PR: IPv6 packet rate IP6B: IPv6 bytes IP6TH: IPv6 throughput IP6HB: IPv6 header bytes **TCP Statistics** TPAC: TCP packets TBYT: TCP bytes TPR: TCP packet rate

continued on next page. . .

. . . continued from previous page

	TTHR: TCP throughput **UDP Statistics** UPAC: UDP packets UBYT: UDP bytes UPR: UDP packet rate UTHR: UDP throughput TUEF: TCP/UDP errored frames
Response	{({<result>, }+), }* = <EXPRESSION RESPONSE DATA> **Channel Definitions** <result> = <STRING RESPONSE DATA> **Statistics** <result> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:CST:RES:PORT1:FETC? 1, (SMAC, S64)` → `("00-00-91-E2-15-01"), (45)`
Note	Use NCHannels? to get the maximum channel number. It is not possible to get results from definitions not enabled by the CDEFinitions command. This command can be used on V2.00 or later.

12.27.11 ETHernet:CSTat:RESults:RESet

Syntax	ETHernet:CSTat:RESults:RESet
Description	This command resets the IP Channel Statistics test.
Parameter	None.
Response	None.
Example	`ETH:CST:RES:RES`
Note	This command can be used on V2.00 or later.

12.28 Monitor Generator

12.28.1 ETHernet:MONGen:SETup:GENeral:IAFFilter

Syntax	ETHernet:MONGen:SETup:GENeral:IAFFilter
Description	This command selects if addresses should be included in frame filter on receiver.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:MONG:SET:GEN:IAFF ON`
Note	

Syntax	ETHernet:MONGen:SETup:GENeral:IAFFilter?
Description	This query returns addresses should be included in frame filter on receiver.
Parameter	None.
Response	<enable> = <CHARACTER RESPONSE DATA>
Example	`ETH:MONG:SET:GEN:IAFF? → 1`
Note	

12.29 Sync Test

12.29.1 ETHernet:SYNTest:SETup:MPERiod

Syntax	ETHernet:SYNTest:SETup:MPERiod <period>
Description	This command select the measurement period.
Parameter	<period> = <CHARACTER PROGRAM DATA> DUR100S: 100 seconds DUR1000S: 1000 seconds USERDEFINE: User define
Response	None.
Example	`ETH:SYNT:SET:MPER DUR100S`
Note	

Syntax	ETHernet:SYNTest:SETup:MPERiod?
Description	This query returns the measurement period.
Parameter	None.
Response	<period> = <CHARACTER RESPONSE DATA>
Example	`ETH:SYNT:SET:MPER?` → `DUR100S`
Note	

12.29.2 ETHernet:SYNTest:SETup:MPERiod:USERdefined

Syntax	ETHernet:SYNTest:SETup:MPERiod:USERdefined <period>
Description	This command select the measurement period.
Parameter	<duration> = <NUMERIC PROGRAM DATA> *MINimum = 100, MAXimum = 259200, DEFault = 100*
Response	None.
Example	`ETH:SYNT:SET:MPER:USER 0`
Note	This parameter is used only when MPERiod is USERDEFINE.

Syntax	ETHernet:SYNTest:SETup:MPERiod:USERdefined?
Description	This query returns the measurement period.
Parameter	None.
Response	<period> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SYNT:SET:MPER:USER?` → `0`
Note	This parameter is used only when MPERiod is USERDEFINE.

12.29.3 ETHernet:SYNTest:SETup:MPERiod:USERdefined:TIMe

Syntax	ETHernet:SYNTest:SETup:MPERiod:USERdefined:TIMe <days>, <hours>, <minutes>, <seconds>
Description	This command select the measurement period.
Parameters	<days> = <NUMERIC PROGRAM DATA>
	<hours> = <NUMERIC PROGRAM DATA>
	<minutes> = <NUMERIC PROGRAM DATA>
	<seconds> = <NUMERIC PROGRAM DATA>
Response	None.
Example	`ETH:SYNT:SET:MPER:USER:TIM 0,0,1,40`
Note	This parameter is used only when MPERiod is USERDEFINE.

Syntax	ETHernet:SYNTest:SETup:MPERiod:USERdefined:TIMe?
Description	This query returns the measurement period.
Parameter	None.
Response	<days> = <NR1 NUMERIC RESPONSE DATA>
	<hours> = <NR1 NUMERIC RESPONSE DATA>
	<minutes> = <NR1 NUMERIC RESPONSE DATA>
	<seconds> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SYNT:SET:MPER:USER:TIM?` → `0,0,1,40`
Note	This parameter is used only when MPERiod is USERDEFINE.

12.29.4 ETHernet:SYNTest:SETup:PPS:MODE[:ENABle]

Syntax	ETHernet:SYNTest:SETup:PPS:MODE[:ENABle] <enable>
Description	This command set the PPS mode.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:SYNT:SET:PPS:MODE ON`
Note	

Syntax	ETHernet:SYNTest:SETup:PPS:MODE[:ENABle]?
Description	This query returns the PPS mode.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:SYNT:SET:PPS:MODE?` → `0`
Note	

12.29.5 ETHernet:SYNTest:SETup:PPS:CABLecorrection

Syntax	ETHernet:SYNTest:SETup:PPS:CABLecorrection <correction>
Description	This command set the PPS cable correction.
Parameter	<correction> = <NUMERIC PROGRAM DATA> *MINimum = -100000000, MAXimum = 100000000, DEFault = 0*
Response	None.
Example	`ETH:SYNT:SET:PPS:CABL 0`
Note	

Syntax	ETHernet:SYNTest:SETup:PPS:CABLecorrection?
Description	This query returns the PPS cable correction.
Parameter	None.
Response	<correction> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SYNT:SET:PPS:CABL?` → `0`
Note	

12.29.6 ETHernet:SYNTest:SETup:PACKet:CABLecorrection

Syntax	ETHernet:SYNTest:SETup:PACKet:CABLecorrection <correction>
Description	This command set the Ethernet cable correction.
Parameter	<correction> = <NUMERIC PROGRAM DATA> *MINimum = -100000000, MAXimum = 100000000, DEFault = 0*
Response	None.
Example	`ETH:SYNT:SET:PACK:CABL 0`
Note	

Syntax	ETHernet:SYNTest:SETup:PACKet:CABLecorrection?
Description	This query returns the Ethernet cable correction.
Parameter	None.
Response	<correction> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:SYNT:SET:PACK:CABL?` → `0`
Note	

12.29.7 ETHernet:THResholds:PPS:PHASe[:ENABle]

Syntax	ETHernet:THResholds:PPS:PHASe[:ENABle] <enable>
Description	This command enables or disables the 1PPS phase error thresholds.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:THR:PPS:PHAS OFF`
Note	

Syntax	ETHernet:THResholds:PPS:PHASe[:ENABle]?
Description	This query returns whether or not the 1PPS phase error thresholds enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:THR:PPS:PHAS?` → `0`
Note	

12.29.8 ETHernet:THResholds:PPS:PHASe:MINValue

Syntax	ETHernet:THResholds:PPS:PHASe:MINValue <value>
Description	This command set the minimum 1PPS phase error thresholds.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum = -1500000000, MAXimum = 1500000000, DEFault = 0*
Response	None.
Example	`ETH:THR:PPS:PHAS:MINV 0`
Note	

Syntax	ETHernet:THResholds:PPS:PHASe:MINValue?
Description	This query returns the minimum 1PPS phase error thresholds.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:THR:PPS:PHAS:MINV?` → `0`
Note	

12.29.9 ETHernet:THResholds:PPS:PHASe:MAXValue

Syntax	ETHernet:THResholds:PPS:PHASe:MAXValue <value>
Description	This command set the maximum 1PPS phase error thresholds.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum = -1500000000, MAXimum = 1500000000, DEFault = 0*
Response	None.
Example	`ETH:THR:PPS:PHAS:MAXV 0`
Note	

Syntax	ETHernet:THResholds:PPS:PHASe:MAXValue?
Description	This query returns the maximum 1PPS phase error thresholds.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:THR:PPS:PHAS:MAXV?` → `0`
Note	

12.29.10 ETHernet:THResholds:PPS:FILTered[:ENABle]

Syntax	ETHernet:THResholds:PPS:FILTered[:ENABle] <enable>
Description	This command enables or disables the filtered TE thresholds.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:THR:PPS:FILT OFF`
Note	

Syntax	ETHernet:THResholds:PPS:FILTered[:ENABle]?
Description	This query returns whether or not the filtered TE thresholds enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:THR:PPS:FILT?` → `0`
Note	

12.29.11 ETHernet:THResholds:PPS:FILTered:MINValue

Syntax	ETHernet:THResholds:PPS:FILTered:MINValue <value>
Description	This command set the minimum filtered TE thresholds.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum = -1500000000, MAXimum = 1500000000, DEFault = 0*
Response	None.
Example	`ETH:THR:PPS:FILT:MINV 0`
Note	

Syntax	ETHernet:THResholds:PPS:FILTered:MINValue?
Description	This query returns the minimum filtered TE thresholds.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:THR:PPS:FILT:MINV?` → `0`
Note	

12.29.12 ETHernet:THResholds:PPS:FILTered:MAXValue

Syntax	ETHernet:THResholds:PPS:FILTered:MAXValue <value>
Description	This command set the maximum filtered TE thresholds.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum = -1500000000, MAXimum = 1500000000, DEFault = 0*
Response	None.
Example	`ETH:THR:PPS:FILT:MAXV 0`
Note	

Syntax	ETHernet:THResholds:PPS:FILTered:MAXValue?
Description	This query returns the maximum filtered TE thresholds.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:THR:PPS:FILT:MAXV?` → `0`
Note	

12.29.13 ETHernet:THResholds:PPS:DEViation[:ENABle]

Syntax	ETHernet:THResholds:PPS:DEViation[:ENABle] <enable>
Description	This command enables or disables the 1PPS deviation thresholds.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:THR:PPS:DEV OFF`
Note	

Syntax	ETHernet:THResholds:PPS:DEViation[:ENABle]?
Description	This query returns whether or not the 1PPS deviation thresholds enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:THR:PPS:DEV?` → `0`
Note	

12.29.14 ETHernet:THResholds:PPS:DEViation:MINValue

Syntax	ETHernet:THResholds:PPS:DEViation:MINValue <value>
Description	This command set the minimum 1PPS deviation thresholds.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum = -1500000000, MAXimum = 1500000000, DEFault = 0*
Response	None.
Example	`ETH:THR:PPS:DEV:MINV 0`
Note	

Syntax	ETHernet:THResholds:PPS:DEViation:MINValue?
Description	This query returns the minimum 1PPS deviation thresholds.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:THR:PPS:DEV:MINV?` → `0`
Note	

12.29.15 ETHernet:THResholds:PPS:DEViation:MAXValue

Syntax	ETHernet:THResholds:PPS:DEViation:MAXValue <value>
Description	This command set the maximum 1PPS deviation thresholds.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum = -1500000000, MAXimum = 1500000000, DEFault = 0*
Response	None.
Example	`ETH:THR:PPS:DEV:MAXV 0`
Note	

Syntax	ETHernet:THResholds:PPS:DEViation:MAXValue?
Description	This query returns the maximum 1PPS deviation thresholds.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:THR:PPS:DEV:MAXV?` → `0`
Note	

12.29.16 ETHernet:THResholds:OWD:SYNC[:ENABle]

Syntax	ETHernet:THResholds:OWD:SYNC[:ENABle] <enable>
Description	This command enables or disables the Sync message one way delay thresholds.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:THR:OWD:SYNC OFF`
Note	

Syntax	ETHernet:THResholds:OWD:SYNC[:ENABle]?
Description	This query returns whether or not the Sync message one way delay thresholds enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:THR:OWD:SYNC? → 0`
Note	

12.29.17 ETHernet:THResholds:OWD:SYNC:MINValue

Syntax	ETHernet:THResholds:OWD:SYNC:MINValue <value>
Description	This command set the minimum sync message one way delay thresholds.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum = -1000000000, MAXimum = 1000000000, DEFault = 0*
Response	None.
Example	`ETH:THR:OWD:SYNC:MINV 0`
Note	

Syntax	ETHernet:THResholds:OWD:SYNC:MINValue?
Description	This query returns the minimum sync message one way delay thresholds.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:THR:OWD:SYNC:MINV? → 0`
Note	

12.29.18 ETHernet:THResholds:OWD:SYNC:MAXValue

Syntax	ETHernet:THResholds:OWD:SYNC:MAXValue <value>
Description	This command set the maximum sync message one way delay thresholds.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum = -1000000000, MAXimum = 1000000000, DEFault = 0*
Response	None.
Example	`ETH:THR:OWD:SYNC:MAXV 0`
Note	

Syntax	ETHernet:THResholds:OWD:SYNC:MAXValue?
Description	This query returns the maximum sync message one way delay thresholds.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:THR:OWD:SYNC:MAXV? → 0`
Note	

12.29.19 ETHernet:THResholds:OWD:FOLLow[:ENABle]

Syntax	ETHernet:THResholds:OWD:FOLLow[:ENABle] <enable>
Description	This command enables or disables the follow up message one way delay thresholds.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:THR:OWD:FOLL OFF`
Note	

Syntax	ETHernet:THResholds:OWD:FOLLow[:ENABle]?
Description	This query returns whether or not the follow up message one way delay thresholds enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:THR:OWD:FOLL?` → `0`
Note	

12.29.20 ETHernet:THResholds:OWD:FOLLow:MINValue

Syntax	ETHernet:THResholds:OWD:FOLLow:MINValue <value>
Description	This command set the minimum follow up message one way delay thresholds.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum = -1000000000, MAXimum = 1000000000, DEFault = 0*
Response	None.
Example	`ETH:THR:OWD:FOLL:MINV 0`
Note	

Syntax	ETHernet:THResholds:OWD:FOLLow:MINValue?
Description	This query returns the minimum follow up message one way delay thresholds.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:THR:OWD:FOLL:MINV?` → `0`
Note	

12.29.21 ETHernet:THResholds:OWD:FOLLow:MAXValue

Syntax	ETHernet:THResholds:OWD:FOLLow:MAXValue <value>
Description	This command set the maximum follow up message one way delay thresholds.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum = -1000000000, MAXimum = 1000000000, DEFault = 0*
Response	None.
Example	`ETH:THR:OWD:FOLL:MAXV 0`
Note	

Syntax	ETHernet:THResholds:OWD:FOLLow:MAXValue?
Description	This query returns the maximum follow up message one way delay thresholds.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:THR:OWD:FOLL:MAXV?` → `0`
Note	

12.29.22 ETHernet:THResholds:OWD:DELay[:ENABle]

Syntax	ETHernet:THResholds:OWD:DELay[:ENABle] <enable>
Description	This command enables or disables the delay request message one way delay thresholds.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:THR:OWD:DEL OFF`
Note	

Syntax	ETHernet:THResholds:OWD:DELay[:ENABle]?
Description	This query returns whether or not the delay request message one way delay thresholds enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:THR:OWD:DEL?` → `0`
Note	

12.29.23 ETHernet:THResholds:OWD:DELay:MINValue

Syntax	ETHernet:THResholds:OWD:DELay:MINValue <value>
Description	This command set the minimum delay request message one way delay thresholds.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum = -1000000000, MAXimum = 1000000000, DEFault = 0*
Response	None.
Example	`ETH:THR:OWD:DEL:MINV 0`
Note	

Syntax	ETHernet:THResholds:OWD:DELay:MINValue?
Description	This query returns the minimum delay request message one way delay thresholds.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:THR:OWD:DEL:MINV?` → `0`
Note	

12.29.24 ETHernet:THResholds:OWD:DELay:MAXValue

Syntax	ETHernet:THResholds:OWD:DELay:MAXValue <value>
Description	This command set the maximum delay request message one way delay thresholds.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum = -1000000000, MAXimum = 1000000000, DEFault = 0*
Response	None.
Example	`ETH:THR:OWD:DEL:MAXV 0`
Note	

Syntax	ETHernet:THResholds:OWD:DELay:MAXValue?
Description	This query returns the maximum delay request message one way delay thresholds.
Parameter	None.
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:THR:OWD:DEL:MAXV?` → `0`
Note	

12.29.25 ETHernet:PORT<Pt>:SYNTest:ADVanced:ADAPte[:ENABle]

Syntax	ETHernet:PORT<Pt>:SYNTest:ADVanced:ADAPte[:ENABle] <enable>
Description	This command enables/disables add delay a asymmetry factor to PacketTE.
Parameter	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:SYNT:ADV:ADAP ON`
Note	

Syntax	ETHernet:PORT<Pt>:SYNTest:ADVanced:ADAPte[:ENABle]?
Description	This query returns if add delay a asymmetry factor to PacketTE is enabled/disabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:SYNT:ADV:ADAP?` → `0`
Note	

12.29.26 ETHernet:SYNTest:SETup:ADVanced:FOLLow

Syntax	ETHernet:SYNTest:SETup:ADVanced:FOLLow <enable>
Description	This command sets SyncTest setup advanced for PORT2 to follow PORT1.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:SYNT:SET:ADV:FOLL ON`
Note	

Syntax	ETHernet:SYNTest:SETup:ADVanced:FOLLow?
Description	This query returns whether or not SyncTest setup advanced for PORT2 follows PORT1.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:SYNT:SET:ADV:FOLL?` → `1`
Note	

12.29.27 ETHernet:PORT<Pt>:SYNCtest:RESults:BOUNds?

Syntax	ETHernet:PORT<Pt>:SYNCtest:RESults:BOUNds?
Description	Returns the number of result index lower and upper bounds.
Parameter	<Pt> = Port number
Response	Format: (lower,upper) <lower> = <NR1 NUMERIC RESPONSE DATA> <upper> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:SYNC:RES:BOUN?` → `(1,50)`
Note	

12.29.28 ETHernet:SYNCtest:1PPS:RESults?

Syntax	ETHernet:SYNCtest:1PPS:RESults? <index>
Description	Returns the number of the 1PPS result data(Phase error, Filtered TE, Deviation). Unit: nano Seconds.
Parameter	None.
	<index> = <NR1 NUMERIC RESPONSE DATA>
Response	{(<result>)}* = <EXPRESSION RESPONSE DATA> Format: (<Phase error result>),(<Filtered TE result>),(<Deviation result>)
Example	`ETH:SYNC:1PPS:RES? 10` → `(1.0),(2.0),(3.0)`
Note	

12.29.29 ETHernet:PORT<Pt>:SYNCtest:OWD:RESults?

Syntax	ETHernet:PORT<Pt>:SYNCtest:OWD:RESults? <index>
Description	Returns the number of the OWD result data(Sync, Follow up, Delay req.). Unit: nano Seconds.
Parameter	<Pt> = Port number
	<index> = <NR1 NUMERIC RESPONSE DATA>
Response	{(<result>), }* = <EXPRESSION RESPONSE DATA> Format: (<Sync result>),(<Follow up result>),(<Delay req. result>) (<Sync result>) = (Sync min,Sync max,Sync avg) (<Follow up result>) = (Follow up min,Follow up max,Follow up avg) (<Delay req. result>) = (Delay req. min,Delay req. max,Delay req. avg)
Example	`ETH:PORT1:SYNC:OWD:RES? 10` → `(1.0,2.0,3.0),(1.1,2.1,3.1),(1.3,2.3,3.3)`
Note	

12.29.30 ETHernet:PORT<Pt>:SYNCtest:TERR:RESults?

Syntax	ETHernet:PORT<Pt>:SYNCtest:TERR:RESults? <index>
Description	Returns the number of the Terr result data(TE1, TE4, Terr). Unit: nano Seconds.
Parameter	<Pt> = Port number
	<index> = <NR1 NUMERIC RESPONSE DATA>
Response	{(<result>), }* = <EXPRESSION RESPONSE DATA> Format: (<TE1 result>),(<TE4 result>),(<Terr result>) (<TE1 result>) = (TE1 min,TE1 max,TE1 avg) (<TE4 result>) = (TE4 min,TE4 max,TE4 avg) (<Terr result>) = (Terr min,Terr max,Terr avg)
Example	`ETH:PORT1:SYNC:TERR:RES? 10` → `(1.0,2.0,3.0),(1.1,2.1,3.1),(1.3,2.3,3.3)`
Note	

12.30 Frame Capture

12.30.1 ETHernet:PORT<Pt>:CAPTure:STARt

Syntax	ETHernet:PORT<Pt>:CAPTure:STARt
Description	This command starts the frame capture.
Parameter	<Pt> = Port number
Response	None.
Example	`ETH:PORT1:CAPT:STAR`
Note	This command can be used on V2.00 or later.

12.30.2 ETHernet:PORT<Pt>:CAPTure:STOP

Syntax	ETHernet:PORT<Pt>:CAPTure:STOP
Description	This command stops the frame capture.
Parameter	<Pt> = Port number
Response	None.
Example	`ETH:PORT1:CAPT:STOP`
Note	This command can be used on V2.00 or later.

12.30.3 ETHernet:PORT<Pt>:CAPTure:STATus?

Syntax	ETHernet:PORT<Pt>:CAPTure:STATus?
Description	This query returns the current status of the frame capture.
Parameter	<Pt> = Port number
Response	<status> = <CHARACTER RESPONSE DATA> IDLE WAITING: Waiting for Trigger CAPTURING FINISHED_CAPTURE SAVING FINISHED_SAVE
Example	`ETH:PORT1:CAPT:STAT?` → `IDLE`
Note	This command can be used on V2.00 or later.

12.30.4 ETHernet:PORT<Pt>:CAPTure:BUFSize

Syntax	ETHernet:PORT<Pt>:CAPTure:BUFSize <size>
Description	This command sets the capture buffer size.
Parameters	<Pt> = Port number
	<size> = <CHARACTER PROGRAM DATA> KB512: 512 KByte MB1: 1 MByte MB2: 2 MByte MB4: 4 MByte MB8: 8 MByte MB16: 16 MByte MB32: 32 MByte MB64: 64 MByte MB128: 128 MByte *DEFault = MB1*
Response	None.
Example	`ETH:PORT1:CAPT:BUFS MB1`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:CAPTure:BUFSize?
Description	This query returns the capture buffer size.
Parameter	<Pt> = Port number
Response	<size> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:CAPT:BUFS?` → `MB1`
Note	This command can be used on V2.00 or later.

12.30.5 ETHernet:PORT<Pt>:CAPTure:SLICing

Syntax	ETHernet:PORT<Pt>:CAPTure:SLICing <type>
Description	This command sets the frame slicing type.
Parameters	<type> = <CHARACTER PROGRAM DATA> NONE: Whole frame B64: Top 64 Byte B128: Top 128 Byte *DEFault = NONE*
Response	None.
Example	`ETH:PORT1:CAPT:SLIC B64`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:CAPTure:SLICing?
Description	This query returns the frame slicing type.
Parameter	None.
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:CAPT:SLIC?` → `B64`
Note	This command can be used on V2.00 or later.

12.30.6 ETHernet:PORT<Pt>:CAPTure:ALOCation

Syntax	ETHernet:PORT<Pt>:CAPTure:ALOCation <type>
Description	This command sets the buffer handling allocation type.
Parameters	<type> = <CHARACTER PROGRAM DATA> STOP: Stop when full WRAP: Overwrite *DEFault = STOP*
Response	None.
Example	`ETH:PORT1:CAPT:ALOC WRAP`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:CAPTure:ALOCation?
Description	This query returns the buffer handling allocation type.
Parameter	None.
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:CAPT:ALOC?` → `WRAP`
Note	This command can be used on V2.00 or later.

12.30.7 ETHernet:PORT<Pt>:CAPTure:TXFRame

Syntax	ETHernet:PORT<Pt>:CAPTure:TXFRame <enable>
Description	This command enables/disables capturing transmitted frames.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:CAPT:TXFR ON`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:CAPTure:TXFRame?
Description	This query returns whether or not capturing transmitted frames is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:PORT1:CAPT:TXFR?` → `1`
Note	This command can be used on V2.00 or later.

12.30.8 ETHernet:PORT<Pt>:CAPTure:TRIGger:TYPE

Syntax	ETHernet:PORT<Pt>:CAPTure:TRIGger:TYPE <type>
Description	This command sets the capture trigger type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> MANual ERRor PATTern: Field match *DEFault = MANual*
Response	None.
Example	`ETH:PORT1:CAPT:TRIG:TYPE MAN`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:CAPTure:TRIGger:TYPE?
Description	This query returns the capture trigger type.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:CAPT:TRIG:TYPE?` → `MAN`
Note	This command can be used on V2.00 or later.

12.30.9 ETHernet:PORT<Pt>:CAPTure:TRIGger:POSition

Syntax	ETHernet:PORT<Pt>:CAPTure:TRIGger:POSition <position>
Description	This command sets the capture trigger position.
Parameters	<Pt> = Port number
	<position> = <CHARACTER PROGRAM DATA> TOP MIDDLE *DEFault = TOP*
Response	None.
Example	`ETH:PORT1:CAPT:TRIG:POS MIDDLE`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:CAPTure:TRIGger:POSition?
Description	This query returns the capture trigger position.
Parameter	<Pt> = Port number
Response	<position> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:CAPT:TRIG:POS?` → `MIDDLE`
Note	This command can be used on V2.00 or later.

12.30.10 ETHernet:PORT<Pt>:CAPTure:TRIGger:ERRor

Syntax	ETHernet:PORT<Pt>:CAPTure:TRIGger:ERRor <type>
Description	This command sets the error type of capture trigger.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> ANY: Any type FRAGMENT: Fragment OVERUNDER: Oversized or undersized OVER: Oversized UNDER: Undersized FCSERR: FCS error OVRFCSERR: Oversized & FCS Error IPCHKSUM: IP checksum error *DEFault = ANY*
Response	None.
Example	`ETH:PORT1:CAPT:TRIG:ERR ANY`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:CAPTure:TRIGger:ERRor?
Description	This query returns the error type of capture trigger.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:CAPT:TRIG:ERR?` → `ANY`
Note	This command can be used on V2.00 or later.

12.30.11 ETHernet:PORT<Pt>:CAPTure:TRIGger:OFFSet

Syntax	ETHernet:PORT<Pt>:CAPTure:TRIGger:OFFSet <offset>
Description	This command sets the pattern offset for the capture trigger.
Parameters	<Pt> = Port number
	<offset> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 15999, DEFault = 0*
Response	None.
Example	`ETH:PORT1:CAPT:TRIG:OFFS 12`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:CAPTure:TRIGger:OFFSet?
Description	This query returns the pattern offset for the capture trigger.
Parameter	<Pt> = Port number
Response	<offset> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:CAPT:TRIG:OFFS? → 12`
Note	This command can be used on V2.00 or later.

12.30.12 ETHernet:PORT<Pt>:CAPTure:TRIGger:PATTern

Syntax	ETHernet:PORT<Pt>:CAPTure:TRIGger:PATTern <pattern>
Description	This command sets the pattern for the capture trigger.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 32 characters (one byte resolution).
Response	None.
Example	`ETH:PORT1:CAPT:TRIG:PATT "0800"`
Note	This command can be used on V2.00 or later.

Syntax	ETHernet:PORT<Pt>:CAPTure:TRIGger:PATTern?
Description	This query returns the pattern for the capture trigger.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:CAPT:TRIG:PATT? → "0800"`
Note	This command can be used on V2.00 or later.

12.30.13 ETHernet:PORT<Pt>:CAPTure:STORe

Syntax	ETHernet:PORT<Pt>:CAPTure:STORe [<file>]
Description	This command stores the current capture data to a given file on the instrument.
Parameter	<Pt> = Port number
	<file> = <STRING PROGRAM DATA> The path and name of the file to store the data.
Response	None.
Example	`ETH:PORT1:CAPT:STOR "Internal/my-frame-capture.pcap"`
Note	This command can be used on V2.00 or later.
	The application server must be in the idle state.
	Files must be saved to the `Internal/` directory or a sub-directory hereof. When a USB storage device is mounted, files can stored via the Usb/ directory. Default File Path is `Internal/`.

12.30.14 ETHernet:PORT<Pt>:CAPTure:BUFFer?

Syntax	ETHernet:PORT<Pt>:CAPTure:BUFFer?
Description	This query returns the buffer usage percentage (0-100).
Parameter	<Pt> = Port number
Response	<percentage> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:CAPT:BUFF?` → `30`
Note	This command can be used on V2.00 or later.

12.31 Discovery

Please refer to Example Scripts for how to create Discovery Test programs (see section A.8).

12.31.1 ETHernet:DISCovery:SETup:TYPE

Syntax	ETHernet:DISCovery:SETup:TYPE <mode>
Description	This command sets the search mode.
Parameter	<mode> = <CHARACTER RESPONSE DATA> MULTI: Inside local IP network (muliticast search) UNI: Any IP address (unicast search) *DEFault = MULTI*
Response	None.
Example	`ETH:DISC:SET:TYPE MULTI`
Note	

Syntax	ETHernet:DISCovery:SETup:TYPE?
Description	This query returns the search mode.
Parameter	None.
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:DISC:SET:TYPE?` → `MULTI`
Note	

12.31.2 ETHernet:DISCovery:SETup:TARGet:COUNt

Syntax	ETHernet:DISCovery:SETup:TARGet:COUNt <number>
Description	This command sets the search range.
Parameters	<number> = <NR1 NUMERIC RESPONSE DATA> *MINimum = 1, MAXimum = 255, DEFault = 1*
Response	None.
Example	`ETH:DISC:SET:TARG:COUN 100`
Note	

Syntax	ETHernet:DISCovery:SETup:TARGet:COUNt?
Description	This query returns the search range.
Parameter	None.
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:DISC:SET:TARG:COUN?` → `100`
Note	

12.31.3 ETHernet:DISCovery:SETup:TARGet:IPV4[:BEGin]

Syntax	ETHernet:DISCovery:SETup:TARGet:IPV4[:BEGin] <address>
Description	This command sets the IP address of the starting point of the search.
Parameter	<address> = <STRING RESPONSE DATA>
Response	None.
Example	`ETH:DISC:SET:TARG:IPV4 "172.29.2.36"`
Note	

Syntax	ETHernet:DISCovery:SETup:TARGet:IPV4[:BEGin]?
Description	This query returns the IP address of the starting point of the search.
Parameter	None.
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:DISC:SET:TARG:IPV4?` → `"172.29.2.36"`
Note	

12.31.4 ETHernet:DISCovery:RESults:NREQuest?

Syntax	ETHernet:DISCovery:RESults:NREQuest?
Description	This query returns the number of detected Network Master.
Parameter	None.
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:DISC:RES:NREQ?` → `1`
Note	

12.31.5 ETHernet:DISCovery:RESults:SREQuest?

Syntax	ETHernet:DISCovery:RESults:SREQuest? <index>
Description	This query returns the result of selected Network Master.
Parameter	<index> = <NR1 NUMERIC RESPONSE DATA> *MINimum = 1, DEFault = 1*
Response	<result> = <EXPRESSION RESPONSE DATA> Format: String List Items in the list are as follows. Nickname, Board number, Port number, MAC address of in-band control, IP address of in-band control, IPv4 netmask of in-band control, IPv4 gateway of in-band control, IPv4 default gateway address of in-band control, VLAN levels of in-band control, VLAN ID1 of in-band control, VLAN ID2 of in-band control, In-band control TCP port, Controller serial number, Controller type, Module type, Software version, Current application of in-band control, Password lock state of in-band control
Example	`ETH:DISC:RES:SREQ? 1` → `NetworkMaster,1,1,00-50-C2-35-D2-EF,172.29.2.36,255.255.255.0,1,172.29.2.1,0,0,0,56002,1234567890,MT1000A,MU100010A,7.0,RFC2544,0`
Note	

12.31.6 ETHernet:DISCovery:RESults:SREQuest:DETail?

Syntax	ETHernet:DISCovery:RESults:SREQuest:DETail? <index>,(<parameter>)
Description	This query returns the detailed result of selected Network Master.
Parameter	<index> = <NR1 NUMERIC RESPONSE DATA> *MINimum = 1, DEFault = 1* (<parameter>) = <STRING RESPONSE DATA> NICN: Nickname BOARD: Board number PORT: Port number MAC: MAC address of in-band control IPV4: IP address of in-band control NMV4: IPv4 netmask of in-band control GWV4E: IPv4 gateway of in-band control GWV4: IPv4 default gateway address of in-band control VLAN: The number of active VLAN levels of in-band control VLAN1: VLAN ID1 of in-band control VLAN2: VLAN ID2 of in-band control TCPPORT: In-band control TCP port SEID: Controller serial number MODEL: Controller type MODULE: Module type SVER: Software version CAPP: Current application of in-band control LOCK: Password lock state of in-band control
Response	<result> is the following format. NICN: <CHARACTER RESPONSE DATA>

continued on next page...

. . . continued from previous page

	BOARD: <NR1 NUMERIC RESPONSE DATA> PORT: <NR1 NUMERIC RESPONSE DATA> MAC: MAC address with no quotes IPV4: IPv4 address with no quotes NMV4: IPv4 netmask with no quotes GWV4E: <BOOLEAN RESPONSE DATA> GWV4: IPv4 address with no quotes VLAN: <NR1 NUMERIC RESPONSE DATA> VLAN1: <NR1 NUMERIC RESPONSE DATA> VLAN2: <NR1 NUMERIC RESPONSE DATA> TCPPORT: <NR1 NUMERIC RESPONSE DATA> SEID: <NR1 NUMERIC RESPONSE DATA> MODEL: MT1000A MODULE: Any string of MU100010A and MU100011A SVER: String of (Magor version number).(Minor version number) CAPP: Any string of RFC 2544, RFC 6349, SAT (Y.1564)", BERT, Mon./Gen., Reflector, Channel Stat., Ping, Traceroute, Sync Test, Discovery LOCK: <BOOLEAN RESPONSE DATA>
Example	`ETH:DISC:RES:SREQ:DET? 1,(MAC)` → `00-50-C2-35-D2-EF`
Note	

12.31.7 ETHernet:DISCovery:RESults:UPDate

Syntax	ETHernet:DISCovery:RESults:UPDate <index>
Description	This command updates the result.
Parameter	<index> = <NR1 NUMERIC RESPONSE DATA>
Response	None.
Example	`ETH:DISC:RES:UPD 1`
Note	

12.31.8 ETHernet:DISCovery:RESults:UPDate:RESult?

Syntax	ETHernet:DISCovery:RESults:UPDate:RESult?
Description	This query returns the last update condition.
Response	<status> = <CHARACTER RESPONSE DATA> SUCCESS: Update succeeded. TIMEOUT: Update timed out.
Example	`ETH:DISC:RES:UPD:RES?` → `SUCCESS`
Note	

12.31.9 ETHernet:DISCovery:CAPPlication[:EXECute]

Syntax	ETHernet:DISCovery:CAPPlication[:EXECute] <index>
Description	This command executes the application change.
Parameter	<index> = <NR1 NUMERIC RESPONSE DATA>
Response	None.
Example	`ETH:DISC:CAPP 1`
Note	

12.31.10 ETHernet:DISCovery:CAPPlication:PASSword

Syntax	ETHernet:DISCovery:CAPPlication:PASSword <value>
Description	This command sets the password in application change dialog.
Parameter	<value> = <STRING RESPONSE DATA>
Response	None.
Example	`ETH:DISC:CAPP:PASS "Anritsu"`
Note	

Syntax	ETHernet:DISCovery:CAPPlication:PASSword?
Description	This query returns the password in application change dialog.
Parameter	None.
Response	<value> = <STRING RESPONSE DATA>
Example	`ETH:DISC:CAPP:PASS?` → `"Anritsu"`
Note	

12.31.11 ETHernet:DISCovery:CAPPlication:SAPPlication

Syntax	ETHernet:DISCovery:CAPPlication:SAPPlication <app name>
Description	This command sets the application name to be changed in application change dialog.
Parameter	<value> = <STRING RESPONSE DATA> RFC: Ethernet RFC-2544 test application. SAT: Ethernet Service Activation Test application. RFC6349: Ethernet RFC-6349 test application. REFL: Ethernet reflector application. DISC: Ethernet discovery application. *DEFault = RFC*
Response	None.
Example	`ETH:DISC:CAPP:SAPP RFC`
Note	

Syntax	ETHernet:DISCovery:CAPPlication:SAPPlication?
Description	This query returns the application name to be changed in application change dialog.
Parameter	None.
Response	<app name> = <STRING RESPONSE DATA>
Example	`ETH:DISC:CAPP:SAPP?` → `RFC`
Note	

12.31.12 ETHernet:DISCovery:CAPPlication:TARGet:MAC:SOURce

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:MAC:SOURce <address>
Description	This command sets the MAC source address in application change dialog.
Parameters	<address> = <STRING PROGRAM DATA> MAC address
Response	None.
Example	`ETH:DISC:CAPP:TARG:MAC:SOUR "00-50-C2-35-D2-EF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:MAC:SOURce?
Description	This query returns the MAC source address in application change dialog.
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:DISC:CAPP:TARG:MAC:SOUR?` → `"00-50-C2-35-D2-EF"`
Note	

12.31.13 ETHernet:DISCovery:CAPPlication:TARGet:IPV4:SOURce

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:IPV4:SOURce <address>
Description	This command sets the IPv4 source address in application change dialog.
Parameters	<address> = <STRING PROGRAM DATA> IPv4 address
Response	None.
Example	`ETH:DISC:CAPP:TARG:IPV4:SOUR "172.29.2.36"`
Note	Only the character '.' is accepted as separator.

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:IPV4:SOURce?
Description	This query returns the IPv4 source address in application change dialog.
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:DISC:CAPP:TARG:IPV4:SOUR?` → `"172.29.2.36"`
Note	

12.31.14 ETHernet:DISCovery:CAPPlication:TARGet:IPV4:DESTination

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:IPV4:DESTination <address>
Description	This command sets the IPv4 destination address in application change dialog.
Parameters	<address> = <STRING PROGRAM DATA> IPv4 address
Response	None.
Example	`ETH:DISC:CAPP:TARG:IPV4:DEST "172.29.2.36"`
Note	

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:IPV4:DESTination?
Description	This query returns the IPv4 destination address in application change dialog.
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:DISC:CAPP:TARG:IPV4:DEST?` → `"172.29.2.36"`
Note	

12.31.15 ETHernet:DISCovery:CAPPlication:TARGet:IPV4:NETMask

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:IPV4:NETMask <address>
Description	This command sets the IPv4 netmask in application change dialog.
Parameters	<mask> = <STRING PROGRAM DATA> IPv4 netmask
Response	None.
Example	`ETH:DISC:CAPP:TARG:IPV4:NETM "255.255.255.0"`
Note	

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:IPV4:NETMask?
Description	This query returns the IPv4 netmask in application change dialog.
Response	<mask> = <STRING RESPONSE DATA>
Example	`ETH:DISC:CAPP:TARG:IPV4:NETM?` → `"255.255.255.0"`
Note	

12.31.16 ETHernet:DISCovery:CAPPlication:TARGet:IPV4:GATeway[:ENABle]

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:IPV4:GATeway[:ENABle] <enable>
Description	This command sets the IPv4 gateway in application change dialog.
Parameters	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:DISC:CAPP:TARG:IPV4:GAT ON`
Note	

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:IPV4:GATeway[:ENABle]?
Description	This query returns the IPv4 gateway in application change dialog.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:DISC:CAPP:TARG:IPV4:GAT?` → `1`
Note	

12.31.17 ETHernet:DISCovery:CAPPlication:TARGet:IPV4:GATeway:ADDRess

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:IPV4:GATeway:ADDRess <address>
Description	This command sets the IPv4 default gateway in application change dialog.
Parameters	<address> = <STRING PROGRAM DATA> IPv4 address
Response	None.
Example	`ETH:DISC:CAPP:TARG:IPV4:GAT:ADDR "172.29.2.36"`
Note	

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:IPV4:GATeway:ADDRess?
Description	This query returns the IPv4 default gateway in application change dialog.
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:DISC:CAPP:TARG:IPV4:GAT:ADDR?` → `"172.29.2.36"`
Note	

12.31.18 ETHernet:DISCovery:CAPPlication:TARGet:VLAN:LCount

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:VLAN:LCount <levels>
Description	This command sets the number of active VLAN levels in application change dialog.
Parameters	<levels> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8, DEFault=1*
Response	None.
Example	`ETH:DISC:CAPP:TARG:VLAN:LC 2`
Note	

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:VLAN:LCount?
Description	This query returns the number of active VLAN levels in application change dialog.
Response	<levels> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:DISC:CAPP:TARG:VLAN:LC?` → `2`
Note	

12.31.19 ETHernet:DISCovery:CAPPlication:TARGet:VLAN:LEVel<Lv>:ID

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:VLAN:LEVel<Lv>:ID <number>
Description	This command sets the VLAN ID in application change dialog.
Parameter	<Lv> = VLAN level (1-N[1])
	<number> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=4095, DEFault=0*
Response	None.
Example	`ETH:DISC:CAPP:TARG:VLAN:LEV1:ID 1024`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels (see section 12.7.29). Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame).

Syntax	ETHernet:DISCovery:CAPPlication:TARGet:VLAN:LEVel<Lv>:ID?
Description	This query returns the VLAN ID in application change dialog.
Parameter	<Lv> = VLAN level (1-N)
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:DISC:CAPP:TARG:VLAN:LEV1:ID?` → `1024`
Note	

12.31.20 ETHernet:DISCovery:CAPPlication:IMULti:MAC:SOURce

Syntax	ETHernet:DISCovery:CAPPlication:IMULti:MAC:SOURce <enable>
Description	This command enables/disables use of the setting to increment MAC source address of all streams.
Parameter	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:DISC:CAPP:IMUL:MAC:SOUR ON`
Note	

Syntax	ETHernet:DISCovery:CAPPlication:IMULti:MAC:SOURce?
Description	This query returns whether or not the setting to increment MAC source address of all streams is used.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`ETH:DISC:CAPP:IMUL:MAC:SOUR?` → `ON`
Note	

12.31.21 ETHernet:DISCovery:CAPPlication:IMULti:IPV4:SOURce

Syntax	ETHernet:DISCovery:CAPPlication:IMULti:IPV4:SOURce <enable>
Description	This command enables/disables use of the setting to increment IPv4 source address of all streams.
Parameter	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:DISC:CAPP:IMUL:IPV4:SOUR ON`
Note	

Syntax	ETHernet:DISCovery:CAPPlication:IMULti:IPV4:SOURce?
Description	This query returns whether or not the setting to increment IPv4 source address of all streams is used.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`ETH:DISC:CAPP:IMUL:IPV4:SOUR?` → `ON`
Note	

12.31.22 ETHernet:DISCovery:CAPPlication:IMULti:VLAN:LEVel<Lv>:ID

Syntax	ETHernet:DISCovery:CAPPlication:IMULti:VLAN:LEVel<Lv>:ID <enable>
Description	This command enables/disables use of the setting to increment the VLAN ID of all streams.
Parameter	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:DISC:CAPP:IMUL:VLAN:LEV1:ID ON`
Note	

Syntax	ETHernet:DISCovery:CAPPlication:IMULti:VLAN:LEVel<Lv>:ID?
Description	This query returns whether or not the setting to increment the VLAN ID of all streams is used.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`ETH:DISC:CAPP:IMUL:VLAN:LEV1:ID?` → `ON`
Note	

12.32 OAM

The commands in this section is available to all Ethernet applications but the following: Cable Test, Reflector and Pass Through.

Use the general `ETHernet:PORT<Pt>:IFETch?` command to retrieve OAM results.

12.32.1 ETHernet:PORT<Pt>:OAM:PROTocol

Syntax	ETHernet:PORT<Pt>:OAM:PROTocol <protocol>
Description	This command sets the OAM protocol to use.
Parameters	<Pt> = Port number
	<protocol> = <CHARACTER PROGRAM DATA> AH: OAM according to IEEE 802.3ah AG: OAM according to IEEE 802.1AG Y: OAM according to IEEE Y.1731 *DEFault = AH*
Response	None.
Example	`ETH:PORT1:OAM:PROT AH`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:PROTocol?
Description	This query returns the OAM protocol setting.
Parameter	<Pt> = Port number
Response	<protocol> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:PROT?` → `AH`
Note	

12.32.2 ETHernet:PORT<Pt>:OAM:SMAC

Syntax	ETHernet:PORT<Pt>:OAM:SMAC <address>
Description	This command sets the OAM source MAC address.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> MAC address
Response	None.
Example	`ETH:PORT1:OAM:SMAC "00-50-C2-35-D2-EF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:OAM:SMAC?
Description	This query returns the OAM source MAC address.
Parameter	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:SMAC?` → `"00-50-C2-35-D2-EF"`
Note	

12.32.3 ETHernet:PORT<Pt>:OAM:MAC:ETYPe

Syntax	ETHernet:PORT<Pt>:OAM:MAC:ETYPe <type>
Description	This command sets the encapsulated customer Ethernet Type.
Parameters	<Pt> = Port number
	<type> = <NUMERIC PROGRAM DATA> Acceptable values: #H8100 #H88A8 #H9100 #H9200
Response	None.
Example	`ETH:PORT1:OAM:MAC:ETYP #H8100`
Note	This setting is used when `ETH:PORT1:OAM:VLAN` is enabled only.

Syntax	ETHernet:PORT<Pt>:OAM:MAC:ETYPe?
Description	This query returns the encapsulated customer Ethernet Type.
Parameters	<Pt> = Port number
Response	<type> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:MAC:ETYP?` → `#H8100`
Note	

12.32.4 ETHernet:PORT<Pt>:OAM:AH:DISCovery[:ENABle]

Syntax	ETHernet:PORT<Pt>:OAM:AH:DISCovery[:ENABle] <enable>
Description	This command enables/disables IEEE 802.3ah protocol activity.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:AH:DISC OFF`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AH:DISCovery[:ENABle]?
Description	This query returns whether or not IEEE 802.3ah protocol activity is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:AH:DISC?` → `0`
Note	

12.32.5 ETHernet:PORT<Pt>:OAM:AH:DISCovery:LMODe

Syntax	ETHernet:PORT<Pt>:OAM:AH:DISCovery:LMODe <mode>
Description	This command sets the link mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> PASSive: ACTive: *DEFault = PASSive*
Response	None.
Example	`ETH:PORT1:OAM:AH:DISC:LMOD PASS`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AH:DISCovery:LMODe?
Description	This query returns the link mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:AH:DISC:LMOD?` → `PASS`
Note	

12.32.6 ETHernet:PORT<Pt>:OAM:AH:DISCovery:VOUI

Syntax	ETHernet:PORT<Pt>:OAM:AH:DISCovery:VOUI <value>
Description	This command sets the vendor OUI.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=16777215, DEFault=145*
Response	None.
Example	`ETH:PORT1:OAM:AH:DISC:VOUI #H91`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AH:DISCovery:VOUI?
Description	This query returns the vendor OUI.
Parameter	<Pt> = Port number
Response	<value> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:AH:DISC:VOUI?` → `145`
Note	

12.32.7 ETHernet:PORT<Pt>:OAM:AH:DISCovery:VSINfo

Syntax	ETHernet:PORT<Pt>:OAM:AH:DISCovery:VSINfo <value>
Description	This command sets the vendor specific information.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=268435455, DEFault=0*
Response	None.
Example	`ETH:PORT1:OAM:AH:DISC:VSIN 1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AH:DISCovery:VSINfo?
Description	This query returns the vendor specific information.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:AH:DISC:VSIN?` → `1`
Note	

12.32.8 ETHernet:PORT<Pt>:OAM:AH:DISCovery:UNIDirectional

Syntax	ETHernet:PORT<Pt>:OAM:AH:DISCovery:UNIDirectional <enable>
Description	This command enables/disables unidirectional operation.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:AH:DISC:UNID OFF`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AH:DISCovery:UNIDirectional?
Description	This query returns whether or not unidirectional operation is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:AH:DISC:UNID?` → `0`
Note	

12.32.9 ETHernet:PORT<Pt>:OAM:AH:DISCovery:VRETrival

Syntax	ETHernet:PORT<Pt>:OAM:AH:DISCovery:VRETrival <enable>
Description	This command enables/disables variable retrieval.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:AH:DISC:VRET OFF`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AH:DISCovery:VRETrival?
Description	This query returns whether or not variable retrieval is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:AH:DISC:VRET?` → `0`
Note	

12.32.10 ETHernet:PORT<Pt>:OAM:AH:LDEFects:LFAults

Syntax	ETHernet:PORT<Pt>:OAM:AH:LDEFects:LFAults <enable>
Description	This command enables/disables detection of local link faults.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:AH:LDEF:LFA OFF`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AH:LDEFects:LFAults?
Description	This query returns whether or not detection of local link faults is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:AH:LDEF:LFA?` → `0`
Note	

12.32.11 ETHernet:PORT<Pt>:OAM:AH:LDEFects:DGASp

Syntax	ETHernet:PORT<Pt>:OAM:AH:LDEFects:DGASp <enable>
Description	This command enables/disables detection of dying gasp.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:AH:LDEF:DGAS OFF`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AH:LDEFects:DGASp?
Description	This query returns whether or not detection of dying gasp is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:AH:LDEF:DGAS?` → `0`
Note	

12.32.12 ETHernet:PORT<Pt>:OAM:AH:LDEFects:CEvent

Syntax	ETHernet:PORT<Pt>:OAM:AH:LDEFects:CEvent <enable>
Description	This command enables/disables detection of critical events.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:AH:LDEF:CE OFF`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AH:LDEFects:CEvent?
Description	This query returns whether or not detection of critical events is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:AH:LDEF:CE?` → `0`
Note	

12.32.13 ETHernet:PORT<Pt>:OAM:AG[:ENABle]

Syntax	ETHernet:PORT<Pt>:OAM:AG[:ENABle] <enable>
Description	This command enables/disables IEEE 802.1ah protocol activity.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:AG OFF`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AG[:ENABle]?
Description	This query returns whether or not IEEE 802.1ah protocol activity is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:AG?` → `0`
Note	

12.32.14 ETHernet:PORT<Pt>:OAM:AG:MEPid

Syntax	ETHernet:PORT<Pt>:OAM:AG:MEPid <id>
Description	This command sets the management end point ID.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191, DEFault=1*
Response	None.
Example	`ETH:PORT1:OAM:AG:MEP 1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AG:MEPid?
Description	This query returns the management end point ID.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:AG:MEP?` → `1`
Note	

12.32.15 ETHernet:PORT<Pt>:OAM:AG:MDLevel

Syntax	ETHernet:PORT<Pt>:OAM:AG:MDLevel <level>
Description	This command sets the maintenance domain level.
Parameters	<Pt> = Port number
	<level> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=7, DEFault=1*
Response	None.
Example	`ETH:PORT1:OAM:AG:MDL 1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AG:MDLevel?
Description	This query returns the maintenance domain level.
Parameter	<Pt> = Port number
Response	<level> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:AG:MDL?` → `1`
Note	

12.32.16 ETHernet:PORT<Pt>:OAM:AG:DOMain

Syntax	ETHernet:PORT<Pt>:OAM:AG:DOMain <name>
Description	This command sets the domain name.
Parameters	<Pt> = Port number
	<name> = <STRING PROGRAM DATA> Domain name string.
Response	None.
Example	`ETH:PORT1:OAM:AG:DOM "Domain"`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AG:DOMain?
Description	This query returns the domain name.
Parameter	<Pt> = Port number
Response	<name> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:AG:DOM?` → `"Domain"`
Note	

12.32.17 ETHernet:PORT<Pt>:OAM:AG:MAIDentifier

Syntax	ETHernet:PORT<Pt>:OAM:AG:MAIDentifier <name>
Description	This command sets the maintenance association identifier.
Parameters	<Pt> = Port number
	<name> = <STRING PROGRAM DATA> Association string.
Response	None.
Example	`ETH:PORT1:OAM:AG:MAID "Anritsu"`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AG:MAIDentifier?
Description	This query returns the maintenance association identifier.
Parameter	<Pt> = Port number
Response	<name> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:AG:MAID?` → `"Anritsu"`
Note	

12.32.18 ETHernet:PORT<Pt>:OAM:AG:CCM:INTerval

Syntax	ETHernet:PORT<Pt>:OAM:AG:CCM:INTerval <interval>
Description	This command sets the continuity check message interval.
Parameters	<Pt> = Port number
	<interval> = <CHARACTER PROGRAM DATA> MS10: 10 milli seconds MS100: 100 milli seconds S1: 1 second S10: 10 seconds S60: 1 minute S600: 10 minutes *DEFault = S1*
Response	None.
Example	`ETH:PORT1:OAM:AG:CCM:INT S1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AG:CCM:INTerval?
Description	This query returns the continuity check message interval.
Parameter	<Pt> = Port number
Response	<interval> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:AG:CCM:INT?` → `S1`
Note	

12.32.19 ETHernet:PORT<Pt>:OAM:AG:LBM:OTLV

Syntax	ETHernet:PORT<Pt>:OAM:AG:LBM:OTLV <tlv>
Description	This command sets the loop-back message optional TLV type.
Parameters	<Pt> = Port number
	<tlv> = <CHARACTER PROGRAM DATA> NONE: No TLV DATA: Data TLV TEST: Test TLV *DEFault = NONE*
Response	None.
Example	`ETH:PORT1:OAM:AG:LBM:OTLV NONE`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AG:LBM:OTLV?
Description	This query returns the loop-back message optional TLV type.
Parameter	<Pt> = Port number
Response	<tlv> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:AG:LBM:OTLV?` → `NONE`
Note	

12.32.20 ETHernet:PORT<Pt>:OAM:AG:LBM:TLVLength

Syntax	ETHernet:PORT<Pt>:OAM:AG:LBM:TLVLength <length>
Description	This command sets the loop-back message TLV length.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum=32, MAXimum=1480, DEFault=32*
Response	None.
Example	`ETH:PORT1:OAM:AG:LBM:TLVL 60`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AG:LBM:TLVLength?
Description	This query returns the loop-back message TLV length.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:AG:LBM:TLVL? → 60`
Note	

12.32.21 ETHernet:PORT<Pt>:OAM:AG:LBM:DTLV:VALue

Syntax	ETHernet:PORT<Pt>:OAM:AG:LBM:DTLV:VALue <value>
Description	This command sets the data loop-back message TLV value.
Parameters	<Pt> = Port number
	<value> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:OAM:AG:LBM:DTLV:VAL ""`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AG:LBM:DTLV:VALue?
Description	This query returns the data loop-back message TLV value.
Parameter	<Pt> = Port number
Response	<value> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:AG:LBM:DTLV:VAL? → ""`
Note	

12.32.22 ETHernet:PORT<Pt>:OAM:AG:LBM:TTLV:VALue

Syntax	ETHernet:PORT<Pt>:OAM:AG:LBM:TTLV:VALue <value>
Description	This command sets the loop-back message test TLV value.
Parameters	<Pt> = Port number
	<value> = <CHARACTER PROGRAM DATA> ZERos: Zeros without CRC-32 ZCRC: Zeros with CRC-32 *DEFault = ZERos*
Response	None.
Example	`ETH:PORT1:OAM:AG:LBM:TTLV:VAL ZER`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AG:LBM:TTLV:VALue?
Description	This query returns the loop-back message test TLV value.
Parameter	<Pt> = Port number
Response	<value> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:AG:LBM:TTLV:VAL? → ZER`
Note	

12.32.23 ETHernet:PORT<Pt>:OAM:AG:LTM:TTL

Syntax	ETHernet:PORT<Pt>:OAM:AG:LTM:TTL <ttl>
Description	This command sets the link trace message TTL.
Parameters	<Pt> = Port number
	<ttl> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=255, DEFault=65*
Response	None.
Example	`ETH:PORT1:OAM:AG:LTM:TTL 65`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AG:LTM:TTL?
Description	This query returns the link trace message TTL.
Parameter	<Pt> = Port number
Response	<ttl> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:AG:LTM:TTL? → 65`
Note	

12.32.24 ETHernet:PORT<Pt>:OAM:AG:LTM:TID

Syntax	ETHernet:PORT<Pt>:OAM:AG:LTM:TID <id>
Description	This command sets the link trace message transmission ID.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=65535, DEFault=1*
Response	None.
Example	`ETH:PORT1:OAM:AG:LTM:TID 1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:AG:LTM:TID?
Description	This query returns the link trace message transmission ID.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:AG:LTM:TID? → 1`
Note	

12.32.25 ETHernet:PORT<Pt>:OAM:AG:MEPList:ADD

Syntax	ETHernet:PORT<Pt>:OAM:AG:MEPList:ADD <mepId>,<mac>[,<level>[,<association>[,<domain>]]]
Description	This command adds a management end point to the MEP list.
Parameters	<Pt> = Port number
	<mepId> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191*
	<mac> = <STRING PROGRAM DATA>
	<level> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=7, DEFault=0*
	<association> = <STRING PROGRAM DATA>
	<domain> = <STRING PROGRAM DATA>
Response	None
Example	`ETH:PORT1:OAM:AG:MEPL:ADD 234,"00-00-00-00-00-11"`
Note	

12.32.26 ETHernet:PORT<Pt>:OAM:AG:MEPList:DELete[:MEP]

Syntax	ETHernet:PORT<Pt>:OAM:AG:MEPList:DELete[:MEP] <mepId>
Description	This command deletes a management end point from the MEP list.
Parameters	<Pt> = Port number
	<mepId> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191*
Response	None
Example	`ETH:PORT1:OAM:AG:MEPL:DEL 234`
Note	

12.32.27 ETHernet:PORT<Pt>:OAM:AG:MEPList:CATalog?

Syntax	ETHernet:PORT<Pt>:OAM:AG:MEPList:CATalog?
Description	This query returns the items on the management end point list.
Parameter	<Pt> = Port number
Response	{(<endPoint>), }* = <EXPRESSION RESPONSE DATA> <endPoint> is split into five separate results: (<mepId>,<mac>,<level>,<association>,<domain>)

Example	`ETH:PORT1:OAM:AG:MEPL:CAT?` →
	`(234,00-00-00-00-00-11,1,Association-A,Domain1),`
	`(345,00-00-00-00-00-22,1,Association-B,Domain2)`

Note	If the list is empty an execution error will be reported.

12.32.28 ETHernet:PORT<Pt>:OAM:Y[:ENABle]

Syntax	ETHernet:PORT<Pt>:OAM:Y[:ENABle] <enable>
Description	This command enables/disables Y.1731 protocol activity.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:Y OFF`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y[:ENABle]?
Description	This query returns whether or not Y.1731 protocol activity is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y?` → `0`
Note	

12.32.29 ETHernet:PORT<Pt>:OAM:Y:MEPid

Syntax	ETHernet:PORT<Pt>:OAM:Y:MEPid <id>
Description	This command sets the management end point ID.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191, DEFault=1*
Response	None.
Example	`ETH:PORT1:OAM:Y:MEP 1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:MEPid?
Description	This query returns the management end point ID.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:MEP? → 1`
Note	

12.32.30 ETHernet:PORT<Pt>:OAM:Y:MEGid

Syntax	ETHernet:PORT<Pt>:OAM:Y:MEGid <id>
Description	This command sets the MEG ID.
Parameters	<Pt> = Port number
	<id> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:OAM:Y:MEG "A"`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:MEGid?
Description	This query returns the MEG ID.
Parameter	<Pt> = Port number
Response	<id> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:MEG? → "A"`
Note	

12.32.31 ETHernet:PORT<Pt>:OAM:Y:MEGLevel

Syntax	ETHernet:PORT<Pt>:OAM:Y:MEGLevel <level>
Description	This command sets the MEG level.
Parameters	<Pt> = Port number
	<level> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=7, DEFault=1*
Response	None.
Example	`ETH:PORT1:OAM:Y:MEGL 1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:MEGLevel?
Description	This query returns the MEG level.
Parameter	<Pt> = Port number
Response	<level> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:MEGL? → 1`
Note	

12.32.32 ETHernet:PORT<Pt>:OAM:Y:CCM:INTerval

Syntax	ETHernet:PORT<Pt>:OAM:Y:CCM:INTerval <interval>
Description	This command sets the continuity check messages interval.
Parameters	<Pt> = Port number
	<interval> = <CHARACTER PROGRAM DATA> MS10: 10 milli seconds MS100: 100 milli seconds S1: 1 second S10: 10 seconds S60: 1 minute S600: 10 minutes *DEFault = S1*
Response	None.
Example	`ETH:PORT1:OAM:Y:CCM:INT S1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:CCM:INTerval?
Description	This query returns the continuity check messages interval.
Parameter	<Pt> = Port number
Response	<interval> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:CCM:INT?` → `S1`
Note	

12.32.33 ETHernet:PORT<Pt>:OAM:Y:LBM:OTLV

Syntax	ETHernet:PORT<Pt>:OAM:Y:LBM:OTLV <tlv>
Description	This command sets the loop-back message optional TLV type.
Parameters	<Pt> = Port number
	<tlv> = <CHARACTER PROGRAM DATA> NONE: No TLV DATA: Data TLV TEST: Test TLV *DEFault = NONE*
Response	None.
Example	`ETH:PORT1:OAM:Y:LBM:OTLV NON`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:LBM:OTLV?
Description	This query returns the loop-back message optional TLV type.
Parameter	<Pt> = Port number
Response	<tlv> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:LBM:OTLV?` → `NON`
Note	

12.32.34 ETHernet:PORT<Pt>:OAM:Y:LBM:TLVLength

Syntax	ETHernet:PORT<Pt>:OAM:Y:LBM:TLVLength <length>
Description	This command sets the loop-back message TLV length.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum=32, MAXimum=1480, DEFault=32*
Response	None.
Example	`ETH:PORT1:OAM:Y:LBM:TLVL 60`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:LBM:TLVLength?
Description	This query returns the loop-back message TLV length.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:LBM:TLVL?` → `60`
Note	

12.32.35 ETHernet:PORT<Pt>:OAM:Y:LBM:DTLV:VALue

Syntax	ETHernet:PORT<Pt>:OAM:Y:LBM:DTLV:VALue <value>
Description	This command sets the loop-back message data TLV value.
Parameters	<Pt> = Port number
	<value> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:OAM:Y:LBM:DTLV:VAL ""`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:LBM:DTLV:VALue?
Description	This query returns the loop-back message data TLV value.
Parameter	<Pt> = Port number
Response	<value> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:LBM:DTLV:VAL?` → `""`
Note	

12.32.36 ETHernet:PORT<Pt>:OAM:Y:LBM:TTLV:VALue

Syntax	ETHernet:PORT<Pt>:OAM:Y:LBM:TTLV:VALue <value>
Description	This command sets the loop-back message test TLV value.
Parameters	<Pt> = Port number
	<value> = <CHARACTER PROGRAM DATA> ZERos: Zeros without CRC-32 ZCRC: Zeros with CRC-32 *DEFault = ZERos*
Response	None.
Example	`ETH:PORT1:OAM:Y:LBM:TTLV:VAL ZER`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:LBM:TTLV:VALue?
Description	This query returns the loop-back message test TLV value.
Parameter	<Pt> = Port number
Response	<value> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:LBM:TTLV:VAL?` → `ZER`
Note	

12.32.37 ETHernet:PORT<Pt>:OAM:Y:LTM:TTL

Syntax	ETHernet:PORT<Pt>:OAM:Y:LTM:TTL <ttl>
Description	This command sets the link trace message TTL.
Parameters	<Pt> = Port number
	<ttl> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=255, DEFault=65*
Response	None.
Example	`ETH:PORT1:OAM:Y:LTM:TTL 65`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:LTM:TTL?
Description	This query returns the link trace message TTL.
Parameter	<Pt> = Port number
Response	<ttl> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:LTM:TTL?` → `65`
Note	

12.32.38 ETHernet:PORT<Pt>:OAM:Y:LTM:TID

Syntax	ETHernet:PORT<Pt>:OAM:Y:LTM:TID <id>
Description	This command sets the link trace message transaction ID.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=65535, DEFault=1*
Response	None.
Example	`ETH:PORT1:OAM:Y:LTM:TID 1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:LTM:TID?
Description	This query returns the link trace message transaction ID.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:LTM:TID?` → `1`
Note	

12.32.39 ETHernet:PORT<Pt>:OAM:Y:TST:FTSend

Syntax	ETHernet:PORT<Pt>:OAM:Y:TST:FTSend <count>
Description	This command sets the number of test frames to send.
Parameters	<Pt> = Port number
	<count> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=100, DEFault=1*
Response	None.
Example	`ETH:PORT1:OAM:Y:TST:FTS 1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:TST:FTSend?
Description	This query returns the number of test frames to send.
Parameter	<Pt> = Port number
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:TST:FTS?` → `1`
Note	

12.32.40 ETHernet:PORT<Pt>:OAM:Y:TST:RATE

Syntax	ETHernet:PORT<Pt>:OAM:Y:TST:RATE <interval>
Description	This command sets the test rate.
Parameters	<Pt> = Port number
	<interval> = <CHARACTER PROGRAM DATA> MS100: 100 milli seconds S1: 1 second S10: 10 seconds *DEFault = S1*
Response	None.
Example	`ETH:PORT1:OAM:Y:TST:RATE S1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:TST:RATE?
Description	This query returns the test rate.
Parameter	<Pt> = Port number
Response	<interval> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:TST:RATE?` → `S1`
Note	

12.32.41 ETHernet:PORT<Pt>:OAM:Y:TST:OTLV

Syntax	ETHernet:PORT<Pt>:OAM:Y:TST:OTLV <tlv>
Description	This command sets the test optional TLV type.
Parameters	<Pt> = Port number
	<tlv> = <CHARACTER PROGRAM DATA> NONE: No TLV TEST: Test TLV *DEFault = NONE*
Response	None.
Example	`ETH:PORT1:OAM:Y:TST:OTLV NONE`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:TST:OTLV?
Description	This query returns the test optional TLV type.
Parameter	<Pt> = Port number
Response	<tlv> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:TST:OTLV?` → `NONE`
Note	

12.32.42 ETHernet:PORT<Pt>:OAM:Y:TST:TLVLength

Syntax	ETHernet:PORT<Pt>:OAM:Y:TST:TLVLength <length>
Description	This command sets the test TLV length.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum=32, MAXimum=1480, DEFault=32*
Response	None.
Example	`ETH:PORT1:OAM:Y:TST:TLVL 0`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:TST:TLVLength?
Description	This query returns the test TLV length.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:TST:TLVL?` → `0`
Note	

12.32.43 ETHernet:PORT<Pt>:OAM:Y:TST:TTLV:VALue

Syntax	ETHernet:PORT<Pt>:OAM:Y:TST:TTLV:VALue <value>
Description	This command sets the test TLV value.
Parameters	<Pt> = Port number
	<value> = <CHARACTER PROGRAM DATA> ZERos: Zeros without CRC-32 ZCRC: Zeros with CRC-32 *DEFault = ZERos*
Response	None.
Example	`ETH:PORT1:OAM:Y:TST:TTLV:VAL ZER`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:TST:TTLV:VALue?
Description	This query returns the test TLV value.
Parameter	<Pt> = Port number
Response	<value> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:TST:TTLV:VAL?` → `ZER`
Note	

12.32.44 ETHernet:PORT<Pt>:OAM:Y:MCC:OUI

Syntax	ETHernet:PORT<Pt>:OAM:Y:MCC:OUI <value>
Description	This command sets the MCC OUI.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=16777215, DEFault=145*
Response	None.
Example	`ETH:PORT1:OAM:Y:MCC:OUI #H91`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:MCC:OUI?
Description	This query returns the MCC OUI.
Parameter	<Pt> = Port number
Response	<value> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:MCC:OUI?` → `145`
Note	

12.32.45 ETHernet:PORT<Pt>:OAM:Y:MCC:DATA

Syntax	ETHernet:PORT<Pt>:OAM:Y:MCC:DATA <pattern>
Description	This command sets the MCC data.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:OAM:Y:MCC:DATA "12345AB"`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:MCC:DATA?
Description	This query returns the MCC data.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:MCC:DATA?` → `"12345AB"`
Note	

12.32.46 ETHernet:PORT<Pt>:OAM:Y:LCK:CMEG

Syntax	ETHernet:PORT<Pt>:OAM:Y:LCK:CMEG <level>
Description	This command sets the lock condition client MEG level.
Parameters	<Pt> = Port number
	<level> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=7, DEFault=0*
Response	None.
Example	`ETH:PORT1:OAM:Y:LCK:CMEG 0`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:LCK:CMEG?
Description	This query returns the lock condition client MEG level
Parameter	<Pt> = Port number
Response	<level> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:LCK:CMEG?` → `0`
Note	

12.32.47 ETHernet:PORT<Pt>:OAM:Y:LCK:RATE

Syntax	ETHernet:PORT<Pt>:OAM:Y:LCK:RATE <rate>
Description	This command sets the lock condition rate.
Parameters	<Pt> = Port number
	<rate> = <CHARACTER PROGRAM DATA> S1: 1 second S60: 1 minute *DEFault = S1*
Response	None.
Example	`ETH:PORT1:OAM:Y:LCK:RATE S1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:LCK:RATE?
Description	This query returns the lock condition rate.
Parameter	<Pt> = Port number
Response	<rate> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:LCK:RATE?` → `S1`
Note	

12.32.48 ETHernet:PORT<Pt>:OAM:Y:AIS:CMEG

Syntax	ETHernet:PORT<Pt>:OAM:Y:AIS:CMEG <level>
Description	This command sets the alarm indication signal client MEG level.
Parameters	<Pt> = Port number
	<level> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=7, DEFault=0*
Response	None.
Example	`ETH:PORT1:OAM:Y:AIS:CMEG 0`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:AIS:CMEG?
Description	This query returns the alarm indication signal client MEG level.
Parameter	<Pt> = Port number
Response	<level> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:AIS:CMEG?` → `0`
Note	

12.32.49 ETHernet:PORT<Pt>:OAM:Y:AIS:RATE

Syntax	ETHernet:PORT<Pt>:OAM:Y:AIS:RATE <rate>
Description	This command sets the alarm indication signal rate.
Parameters	<Pt> = Port number
	<rate> = <CHARACTER PROGRAM DATA> S1: 1 second S60: 1 minute *DEFault = S1*
Response	None.
Example	`ETH:PORT1:OAM:Y:AIS:RATE S1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:AIS:RATE?
Description	This query returns the alarm indication signal rate.
Parameter	<Pt> = Port number
Response	<rate> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:AIS:RATE?` → `S1`
Note	

12.32.50 ETHernet:PORT<Pt>:OAM:Y:DMONe:TYPE

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMONe:TYPE <mode>
Description	This command sets the one-way delay measurement mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> ODEMand: On-demand mode PROactive: Proactive mode *DEFault = PROactive*
Response	None.
Example	`ETH:PORT1:OAM:Y:DMON:TYPE PRO`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMONe:TYPE?
Description	This query returns the one-way delay measurement mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:DMON:TYPE?` → `PR`
Note	

12.32.51 ETHernet:PORT<Pt>:OAM:Y:DMONe:RATE

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMONe:RATE <interval>
Description	This command sets the one-way delay measurement rate.
Parameters	<Pt> = Port number
	<interval> = <CHARACTER PROGRAM DATA> MS100: 100 milli seconds S1: 1 second S10: 10 seconds *DEFault = S1*
Response	None.
Example	`ETH:PORT1:OAM:Y:DMON:RATE S1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMONe:RATE?
Description	This query returns the one-way delay measurement rate.
Parameter	<Pt> = Port number
Response	<interval> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:DMON:RATE?` → `S1`
Note	

12.32.52 ETHernet:PORT<Pt>:OAM:Y:DMONe:FTSend

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMONe:FTSend <count>
Description	This command sets the number of one-way delay measurement frames to send.
Parameters	<Pt> = Port number
	<count> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=1000, DEFault=1*
Response	None.
Example	`ETH:PORT1:OAM:Y:DMON:FTS 1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMONe:FTSend?
Description	This query returns the number of one-way delay measurement frames to send.
Parameter	<Pt> = Port number
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:DMON:FTS? → 1`
Note	

12.32.53 ETHernet:PORT<Pt>:OAM:Y:DMONe:OTLV

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMONe:OTLV <tlv>
Description	This command sets the one-way delay measurement optional TLV type.
Parameters	<Pt> = Port number
	<tlv> = <CHARACTER PROGRAM DATA> NONE: No TLV DATA: Data TLV TEST: Test TLV *DEFault = NONE*
Response	None.
Example	`ETH:PORT1:OAM:Y:DMON:OTLV NONE`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMONe:OTLV?
Description	This query returns the one-way delay measurement optional TLV type.
Parameter	<Pt> = Port number
Response	<tlv> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:DMON:OTLV? → NONE`
Note	

12.32.54 ETHernet:PORT<Pt>:OAM:Y:DMONe:TLVLength

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMONe:TLVLength <length>
Description	This command sets the one-way delay measurement TLV length.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum=32, MAXimum=1480, DEFault=32*
Response	None.
Example	`ETH:PORT1:OAM:Y:DMON:TLVL 0`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMONe:TLVLength?
Description	This query returns the one-way delay measurement TLV length.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:DMON:TLVL? → 0`
Note	

12.32.55 ETHernet:PORT<Pt>:OAM:Y:DMONe:DTLV:VALue

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMONe:DTLV:VALue <value>
Description	This command sets the data TLV value.
Parameters	<Pt> = Port number
	<value> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:OAM:Y:DMON:DTLV:VAL ""`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMONe:DTLV:VALue?
Description	This query returns the data TLV value.
Parameter	<Pt> = Port number
Response	<value> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:DMON:DTLV:VAL? → ""`
Note	

12.32.56 ETHernet:PORT<Pt>:OAM:Y:DMONe:TTLV:VALue

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMONe:TTLV:VALue <value>
Description	This command sets the test TLV value.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=500, DEFault=1*
Response	None.
Example	`ETH:PORT1:OAM:Y:DMON:TTLV:VAL 1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMONe:TTLV:VALue?
Description	This query returns the test TLV value.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:DMON:TTLV:VAL? → 1`
Note	

12.32.57 ETHernet:PORT<Pt>:OAM:Y:DMM:TYPE

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:TYPE <mode>
Description	This command sets the delay measurement message type.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> ODEMand: On-demand mode PROactive: Proactive mode *DEFault = PROactive*
Response	None.
Example	`ETH:PORT1:OAM:Y:DMM:TYPE PRO`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:TYPE?
Description	This query returns the delay measurement message type.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:DMM:TYPE? → PR`
Note	

12.32.58 ETHernet:PORT<Pt>:OAM:Y:DMM:RATE

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:RATE <interval>
Description	This command sets the delay measurement message rate.
Parameters	<Pt> = Port number
	<interval> = <CHARACTER PROGRAM DATA> MS100: 100 milli seconds S1: 1 second S10: 10 seconds *DEFault = S1*
Response	None.
Example	`ETH:PORT1:OAM:Y:DMM:RATE S1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:RATE?
Description	This query returns the delay measurement message rate.
Parameter	<Pt> = Port number
Response	<interval> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:DMM:RATE?` → `S1`
Note	

12.32.59 ETHernet:PORT<Pt>:OAM:Y:DMM:FTSend

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:FTSend <count>
Description	This command sets the number of delay measurement message frames to send.
Parameters	<Pt> = Port number
	<count> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=1000, DEFault=1*
Response	None.
Example	`ETH:PORT1:OAM:Y:DMM:FTS 1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:FTSend?
Description	This query returns the number of delay measurement message frames to send.
Parameter	<Pt> = Port number
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:DMM:FTS?` → `1`
Note	

12.32.60 ETHernet:PORT<Pt>:OAM:Y:DMM:FVTHreshold

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:FVTHreshold <count>
Description	This command sets the delay measurement message frame variation threshold.
Parameters	<Pt> = Port number
	<count> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100, DEFault=0*
Response	None.
Example	`ETH:PORT1:OAM:Y:DMM:FVTH 0`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:FVTHreshold?
Description	This query returns the frame variation.
Parameter	<Pt> = Port number
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:DMM:FVTH?` → `0`
Note	

12.32.61 ETHernet:PORT<Pt>:OAM:Y:DMM:OTLV

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:OTLV <tlv>
Description	This command sets the delay measurement message optional TLV type.
Parameters	<Pt> = Port number
	<tlv> = <CHARACTER PROGRAM DATA> NONE: No TLV DATA: Data TLV TEST: Test TLV *DEFault = NONE*
Response	None.
Example	`ETH:PORT1:OAM:Y:DMM:OTLV NONE`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:OTLV?
Description	This query returns the delay measurement message optional TLV type.
Parameter	<Pt> = Port number
Response	<tlv> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:DMM:OTLV? → NONE`
Note	

12.32.62 ETHernet:PORT<Pt>:OAM:Y:DMM:TLVLength

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:TLVLength <length>
Description	This command sets the delay measurement message TLV length.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum=32, MAXimum=1480, DEFault=32*
Response	None.
Example	`ETH:PORT1:OAM:Y:DMM:TLVL 0`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:TLVLength?
Description	This query returns the delay measurement message TLV length.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:DMM:TLVL? → 0`
Note	

12.32.63 ETHernet:PORT<Pt>:OAM:Y:DMM:DTLV:VALue

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:DTLV:VALue <value>
Description	This command sets the delay measurement message data TLV value.
Parameters	<Pt> = Port number
	<value> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:OAM:Y:DMM:DTLV:VAL ""`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:DTLV:VALue?
Description	This query returns the delay measurement message data TLV value.
Parameter	<Pt> = Port number
Response	<value> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:DMM:DTLV:VAL? → ""`
Note	

12.32.64 ETHernet:PORT<Pt>:OAM:Y:DMM:TTLV:VALue

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:TTLV:VALue <value>
Description	This command sets the delay measurement message test TLV value.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=500, DEFault=1*
Response	None.
Example	`ETH:PORT1:OAM:Y:DMM:TTLV:VAL 1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:DMM:TTLV:VALue?
Description	This query returns the delay measurement message test TLV value.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:DMM:TTLV:VAL? → 1`
Note	

12.32.65 ETHernet:PORT<Pt>:OAM:Y:LMM:RATE

Syntax	ETHernet:PORT<Pt>:OAM:Y:LMM:RATE <interval>
Description	This command sets the loss measurement message interval length.
Parameters	<Pt> = Port number
	<interval> = <CHARACTER PROGRAM DATA> MS100: 100 milli seconds S1: 1 second S10: 10 seconds *DEFault = S1*
Response	None.
Example	`ETH:PORT1:OAM:Y:LMM:RATE S1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:LMM:RATE?
Description	This query returns the loss measurement message interval length.
Parameter	<Pt> = Port number
Response	<interval> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:LMM:RATE? → S1`
Note	

12.32.66 ETHernet:PORT<Pt>:OAM:Y:LMM:FTSend

Syntax	ETHernet:PORT<Pt>:OAM:Y:LMM:FTSend <count>
Description	This command sets the number of loss measurement message frames to send.
Parameters	<Pt> = Port number
	<count> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=1000, DEFault=1*
Response	None.
Example	`ETH:PORT1:OAM:Y:LMM:FTS 1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:LMM:FTSend?
Description	This query returns the number of loss measurement message frames to send.
Parameter	<Pt> = Port number
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:LMM:FTS? → 1`
Note	

12.32.67 ETHernet:PORT<Pt>:OAM:Y:LMM:FLTHreshold

Syntax	ETHernet:PORT<Pt>:OAM:Y:LMM:FLTHreshold <threshold>
Description	This command sets the loss measurement message frame loss threshold.
Parameters	<Pt> = Port number
	<threshold> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=5, DEFault=0*
Response	None.
Example	`ETH:PORT1:OAM:Y:LMM:FLTH 0`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:LMM:FLTHreshold?
Description	This query returns the loss measurement message frame loss threshold.
Parameter	<Pt> = Port number
Response	<threshold> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:LMM:FLTH?` → `0`
Note	

12.32.68 ETHernet:PORT<Pt>:OAM:Y:SLM:RATE

Syntax	ETHernet:PORT<Pt>:OAM:Y:SLM:RATE <interval>
Description	This command sets the synthetic loss measurement rate.
Parameters	<Pt> = Port number
	<interval> = <CHARACTER PROGRAM DATA> MS100: 100 milli seconds S1: 1 second S10: 10 seconds *DEFault = S1*
Response	None.
Example	`ETH:PORT1:OAM:Y:SLM:RATE S1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:SLM:RATE?
Description	This query returns the synthetic loss measurement rate.
Parameter	<Pt> = Port number
Response	<interval> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:SLM:RATE?` → `S1`
Note	

12.32.69 ETHernet:PORT<Pt>:OAM:Y:SLM:FTSend

Syntax	ETHernet:PORT<Pt>:OAM:Y:SLM:FTSend <count>
Description	This command sets the number of synthetic loss measurement frames to send.
Parameters	<Pt> = Port number
	<count> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=1000, DEFault=1*
Response	None.
Example	`ETH:PORT1:OAM:Y:SLM:FTS 1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:SLM:FTSend?
Description	This query returns the number of synthetic loss measurement frames to send.
Parameter	<Pt> = Port number
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:SLM:FTS?` → `1`
Note	

12.32.70 ETHernet:PORT<Pt>:OAM:Y:SLM:FLTHreshold

Syntax	ETHernet:PORT<Pt>:OAM:Y:SLM:FLTHreshold <threshold>
Description	This command sets the synthetic loss measurement frame loss threshold.
Parameters	<Pt> = Port number
	<threshold> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=5, DEFault=0*
Response	None.
Example	`ETH:PORT1:OAM:Y:SLM:FLTH 0`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:SLM:FLTHreshold?
Description	This query returns the frame loss threshold.
Parameter	<Pt> = Port number
Response	<threshold> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:SLM:FLTH? → 0`
Note	

12.32.71 ETHernet:PORT<Pt>:OAM:Y:EXM:DATA

Syntax	ETHernet:PORT<Pt>:OAM:Y:EXM:DATA <pattern>
Description	This command sets the experimental message data.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:OAM:Y:EXM:DATA "12345AB"`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:EXM:DATA?
Description	This query returns the experimental message data.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:EXM:DATA? → "12345AB"`
Note	

12.32.72 ETHernet:PORT<Pt>:OAM:Y:VSM:DATA

Syntax	ETHernet:PORT<Pt>:OAM:Y:VSM:DATA <pattern>
Description	This command sets the vendor specific message data.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:OAM:Y:VSM:DATA "12345AB"`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:Y:VSM:DATA?
Description	This query returns the vendor specific message data.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:Y:VSM:DATA? → "12345AB"`
Note	

12.32.73 ETHernet:PORT<Pt>:OAM:Y:MEPList:ADD

Syntax	ETHernet:PORT<Pt>:OAM:Y:MEPList:ADD <mepId>,<mac>[,<level>[,<megId>]]
Description	This command adds a management end point to the MEP list.
Parameters	<Pt> = Port number
	<mepId> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191*
	<mac> = <STRING PROGRAM DATA>
	<level> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=7, DEFault=0*
	<association> = <STRING PROGRAM DATA>
	<domain> = <STRING PROGRAM DATA>
Response	None
Example	`ETH:PORT1:OAM:Y:MEPL:ADD 234,"00-00-00-00-00-11"`
Note	

12.32.74 ETHernet:PORT<Pt>:OAM:Y:MEPList:DELete[:MEP]

Syntax	ETHernet:PORT<Pt>:OAM:Y:MEPList:DELete[:MEP] <mepId>
Description	This command deletes a management end point from the MEP list.
Parameters	<Pt> = Port number
	<mepId> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191*
Response	None
Example	`ETH:PORT1:OAM:Y:MEPL:DEL 234`
Note	

12.32.75 ETHernet:PORT<Pt>:OAM:Y:MEPList:CATalog?

Syntax	ETHernet:PORT<Pt>:OAM:Y:MEPList:CATalog?
Description	This query returns the items on the management end point list.
Parameter	<Pt> = Port number
Response	{(<endPoint>), }* = <EXPRESSION RESPONSE DATA> <endPoint> is split into five separate results: (<mepId>,<mac>,<level>,<association>,<domain>)

Example	`ETH:PORT1:OAM:Y:MEPL:CAT?` →
	`(234,00-00-00-00-00-11,1,Association-A,Domain1),`
	`(345,00-00-00-00-00-22,1,Association-B,Domain2)`

Note	If the list is empty an execution error will be reported.

12.32.76 ETHernet:PORT<Pt>:OAM:DISCovery:TYPE

Syntax	ETHernet:PORT<Pt>:OAM:DISCovery:TYPE <mode>
Description	This command sets the discovery type.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> CCM: LBM: *DEFault = CCM*
Response	None.
Example	`ETH:PORT1:OAM:DISC:TYPE CCM`
Note	This setting applies to the 802.1ag and Y.1731 protocols.

Syntax	ETHernet:PORT<Pt>:OAM:DISCovery:TYPE?
Description	This query returns the discovery type.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:DISC:TYPE?` → `CCM`
Note	

12.32.77 ETHernet:PORT<Pt>:OAM:DISCovery:INTerval

Syntax	ETHernet:PORT<Pt>:OAM:DISCovery:INTerval <interval>
Description	This command sets the discovery interval.
Parameters	<Pt> = Port number
	<interval> = <CHARACTER PROGRAM DATA> S5: 5 seconds S15: 15 seconds S60: 1 minute *DEFault = S5*
Response	None.
Example	`ETH:PORT1:OAM:DISC:INT S5`
Note	This setting applies to the 802.1ag and Y.1731 protocols.

Syntax	ETHernet:PORT<Pt>:OAM:DISCovery:INTerval?
Description	This query returns the discovery interval.
Parameter	<Pt> = Port number
Response	<interval> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:DISC:INT?` → `S5`
Note	

12.32.78 ETHernet:PORT<Pt>:OAM:DISCovery:DMAX

Syntax	ETHernet:PORT<Pt>:OAM:DISCovery:DMAX <value>
Description	This command sets the maximum number of devices to discover.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=32, DEFault=10*
Response	None.
Example	`ETH:PORT1:OAM:DISC:DMAX 1`
Note	This setting applies to the 802.1ag and Y.1731 protocols.

Syntax	ETHernet:PORT<Pt>:OAM:DISCovery:DMAX?
Description	This query returns the maximum number of devices to discover.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:DISC:DMAX?` → `1`
Note	

12.32.79 ETHernet:PORT<Pt>:OAM:DISCovery:CATalog?

Syntax	ETHernet:PORT<Pt>:OAM:DISCovery:CATalog?
Description	This query returns the list of discovered OAM MEPs.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Response	{(<endPoint>), }* = <EXPRESSION RESPONSE DATA> <endPoint> is split into five separate results: (<mepId>,<mac>,<level>,<associationOrMeg>,<domain>)

Example	`ETH:PORT1:OAM:DISC:CAT?` →
	`(234,00-00-00-00-00-11,1,Association-A,Domain1),`
	`(345,00-00-00-00-00-22,1,Association-B,Domain2)`

Note	The <domain> string is not returned when Y.1731 is active. The time for this command to return can extend to the value in the `ETH:PORT1:OAM:DISC:INT` setting.

12.32.80 ETHernet:PORT<Pt>:OAM:MPLS[:ENABle]

Syntax	ETHernet:PORT<Pt>:OAM:MPLS[:ENABle] <enable>
Description	This command enables/disables MPLS.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:MPLS ON`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:MPLS[:ENABle]?
Description	This query returns the state of MPLS.
Parameters	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:MPLS?` → `1`
Note	

12.32.81 ETHernet:PORT<Pt>:OAM:MPLS:LCOunt

Syntax	ETHernet:PORT<Pt>:OAM:MPLS:LCOunt <levels>
Description	This command sets the number of active MPLS levels.
Parameters	<Pt> = Port number
	<levels> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8, DEFault=1*
Response	None.
Example	`ETH:PORT1:OAM:MPLS:LCO 2`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:MPLS:LCOunt?
Description	This query returns the number of active MPLS levels.
Parameters	<Pt> = Port number
Response	<levels> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:MPLS:LCO?` → `2`
Note	

12.32.82 ETHernet:PORT<Pt>:OAM:MPLS:LEVel<Lv>:LABel

Syntax	ETHernet:PORT<Pt>:OAM:MPLS:LEVel<Lv>:LABel <label>
Description	This command sets the MPLS label.
Parameters	<Pt> = Port number
	<Lv> = MPLS level (1-N[1])
	<label> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1048575, DEFault=0*
Response	None.
Example	`ETH:PORT1:OAM:MPLS:LEV1:LAB 1048575`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active MPLS Levels. Level 1 corresponds to the level at the top of the label stack and N is the level at the bottom of the stack (bottom of stack flag set).

Syntax	ETHernet:PORT<Pt>:OAM:MPLS:LEVel<Lv>:LABel?
Description	This query returns the MPLS label.
Parameters	<Pt> = Port number
	<Lv> = MPLS level (1-N)
Response	<label> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:MPLS:LEV1:LAB? → 1048575`
Note	

12.32.83 ETHernet:PORT<Pt>:OAM:MPLS:LEVel<Lv>:TCLass

Syntax	ETHernet:PORT<Pt>:OAM:MPLS:LEVel<Lv>:TCLass <value>
Description	This command sets the MPLS traffic class.
Parameters	<Pt> = Port number
	<Lv> = MPLS level (1-N[1])
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=7, DEFault=0*
Response	None.
Example	`ETH:PORT1:OAM:MPLS:LEV1:TCL 5`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active MPLS Levels. Level 1 corresponds to the level at the top of the label stack and N is the level at the bottom of the stack (bottom of stack flag set).

Syntax	ETHernet:PORT<Pt>:OAM:MPLS:LEVel<Lv>:TCLass?
Description	This query returns the MPLS traffic class.
Parameters	<Pt> = Port number
	<Lv> = MPLS level (1-N)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:MPLS:LEV1:TCL? → 5`
Note	

12.32.84 ETHernet:PORT<Pt>:OAM:MPLS:LEVel<Lv>:TTL

Syntax	ETHernet:PORT<Pt>:OAM:MPLS:LEVel<Lv>:TTL <value>
Description	This command sets the MPLS time to live.
Parameters	<Pt> = Port number
	<Lv> = MPLS level (1-N[1])
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=32*
Response	None.
Example	`ETH:PORT1:OAM:MPLS:LEV1:TTL 32`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active MPLS Levels. Level 1 corresponds to the level at the top of the label stack and N is the level at the bottom of the stack (bottom of stack flag set).

Syntax	ETHernet:PORT<Pt>:OAM:MPLS:LEVel<Lv>:TTL?
Description	This query returns the MPLS time to live.
Parameters	<Pt> = Port number
	<Lv> = MPLS level (1-N)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:MPLS:LEV1:TTL?` → `32`
Note	

12.32.85 ETHernet:PORT<Pt>:OAM:MPLS:TPRofile[:ENABle]

Syntax	ETHernet:PORT<Pt>:OAM:MPLS:TPRofile[:ENABle] <enable>
Description	This command enables/disables MPLS transport profile.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:MPLS:TPR ON`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:MPLS:TPRofile[:ENABle]?
Description	This query returns the state of MPLS transport profile.
Parameters	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:MPLS:TPR?` → `1`
Note	

12.32.86 ETHernet:PORT<Pt>:OAM:MIM[:ENABle]

Syntax	ETHernet:PORT<Pt>:OAM:MIM[:ENABle] <enable>
Description	This command enables/disables MAC in MAC (alias PBB).
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:OAM:MIM ON`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:MIM[:ENABle]?
Description	This query returns whether or not MAC in MAC is enabled.
Parameters	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:MIM?` → `1`
Note	

12.32.87 ETHernet:PORT<Pt>:OAM:MIM:BTAG:DEI

Syntax	ETHernet:PORT<Pt>:OAM:MIM:BTAG:DEI <enable>
Description	This command sets the B-TAG DEI bit.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:OAM:MIM:BTAG:DEI ON`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:MIM:BTAG:DEI?
Description	This query returns the B-TAG DEI bit.
Parameters	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:MIM:BTAG:DEI?` → `1`
Note	

12.32.88 ETHernet:PORT<Pt>:OAM:MIM:BTAG:PRIority

Syntax	ETHernet:PORT<Pt>:OAM:MIM:BTAG:PRIority <priority>
Description	This command sets the B-TAG Priority (PCP).
Parameters	<Pt> = Port number
	<priority> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 0*
Response	None.
Example	`ETH:PORT1:OAM:MIM:BTAG:PRI 7`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:MIM:BTAG:PRIority?
Description	This query returns the B-TAG Priority (PCP).
Parameters	<Pt> = Port number
Response	<priority> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:MIM:BTAG:PRI?` → `7`
Note	

12.32.89 ETHernet:PORT<Pt>:OAM:MIM:BTAG:VID

Syntax	ETHernet:PORT<Pt>:OAM:MIM:BTAG:VID <vid>
Description	This command sets the B-TAG Backbone VLAN ID.
Parameters	<Pt> = Port number
	<vid> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 4095, DEFault = 0*
Response	None.
Example	`ETH:PORT1:OAM:MIM:BTAG:VID 1024`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:MIM:BTAG:VID?
Description	This query returns the B-TAG Backbone VLAN ID.
Parameters	<Pt> = Port number
Response	<vid> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:MIM:BTAG:VID?` → `1024`
Note	

12.32.90 ETHernet:PORT<Pt>:OAM:MIM:ITAG:UCA

Syntax	ETHernet:PORT<Pt>:OAM:MIM:ITAG:UCA <enable>
Description	This command sets the I-TAG UCA bit.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:OAM:MIM:ITAG:UCA ON`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:MIM:ITAG:UCA?
Description	This query returns the I-TAG UCA bit.
Parameters	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:MIM:ITAG:UCA?` → `1`
Note	

12.32.91 ETHernet:PORT<Pt>:OAM:MIM:ITAG:DEI

Syntax	ETHernet:PORT<Pt>:OAM:MIM:ITAG:DEI <enable>
Description	This command sets the I-TAG DEI bit.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault=OFF*
Response	None.
Example	`ETH:PORT1:OAM:MIM:ITAG:DEI ON`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:MIM:ITAG:DEI?
Description	This query returns the I-TAG DEI bit.
Parameters	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:MIM:ITAG:DEI?` → `1`
Note	

12.32.92 ETHernet:PORT<Pt>:OAM:MIM:ITAG:PRIority

Syntax	ETHernet:PORT<Pt>:OAM:MIM:ITAG:PRIority <priority>
Description	This command sets the I-TAG Priority (PCP).
Parameters	<Pt> = Port number
	<priority> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 7, DEFault = 0*
Response	None.
Example	`ETH:PORT1:OAM:MIM:ITAG:PRI 7`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:MIM:ITAG:PRIority?
Description	This query returns the I-TAG Priority (PCP).
Parameters	<Pt> = Port number
Response	<priority> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:MIM:ITAG:PRI?` → `7`
Note	

12.32.93 ETHernet:PORT<Pt>:OAM:MIM:ITAG:SID

Syntax	ETHernet:PORT<Pt>:OAM:MIM:ITAG:SID <sid>
Description	This command sets the I-TAG SID.
Parameters	<Pt> = Port number
	<sid> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 16777215, DEFault = 0*
Response	None.
Example	`ETH:PORT1:OAM:MIM:ITAG:SID 1`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:MIM:ITAG:SID?
Description	This query returns the I-TAG SID.
Parameters	<Pt> = Port number
Response	<sid> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:MIM:ITAG:SID?` → `1`
Note	

12.32.94 ETHernet:PORT<Pt>:OAM:MIM:SMAC

Syntax	ETHernet:PORT<Pt>:OAM:MIM:SMAC <address>
Description	This command sets the MAC in MAC source MAC address.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> MAC address
Response	None.
Example	`ETH:PORT1:OAM:MIM:SMAC "00-50-C2-35-D2-EF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:OAM:MIM:SMAC?
Description	This query returns the MAC in MAC source MAC address.
Parameter	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:MIM:SMAC?` → `"00-50-C2-35-D2-EF"`
Note	

12.32.95 ETHernet:PORT<Pt>:OAM:MIM:ETHernet:DMAC

Syntax	ETHernet:PORT<Pt>:OAM:MIM:ETHernet:DMAC <address>
Description	This command sets the MAC in MAC destination MAC address.
Parameters	<Pt> = Port number
	<address> = <STRING PROGRAM DATA> MAC address
Response	None.
Example	`ETH:PORT1:OAM:MIM:ETH:DMAC "00-50-C2-35-D2-EF"`
Note	Only the character '-' is accepted as separator.

Syntax	ETHernet:PORT<Pt>:OAM:MIM:ETHernet:DMAC?
Description	This query returns the MAC in MAC destination MAC address.
Parameter	<Pt> = Port number
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:MIM:ETH:DMAC?` → `"00-50-C2-35-D2-EF"`
Note	

12.32.96 ETHernet:PORT<Pt>:OAM:VLAN[:ENABle]

Syntax	ETHernet:PORT<Pt>:OAM:VLAN[:ENABle] <enable>
Description	This command enables/disables VLAN.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:VLAN ON`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:VLAN[:ENABle]?
Description	This query returns whether or not VLAN is enabled.
Parameters	<Pt> = Port number
Response	<boolean> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:VLAN?` → `1`
Note	

12.32.97 ETHernet:PORT<Pt>:OAM:VLAN:LCOunt

Syntax	ETHernet:PORT<Pt>:OAM:VLAN:LCOunt <levels>
Description	This command sets the number of active VLAN levels.
Parameters	<Pt> = Port number
	<levels> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8, DEFault=1*
Response	None.
Example	`ETH:PORT1:OAM:VLAN:LCO 2`
Note	

Syntax	ETHernet:PORT<Pt>:OAM:VLAN:LCOunt?
Description	This query returns the number of active VLAN levels.
Parameters	<Pt> = Port number
Response	<levels> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:VLAN:LCO?` → `2`
Note	

12.32.98 ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:ID

Syntax	ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:ID <number>
Description	This command sets the VLAN ID.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N[1])
	<number> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=4095, DEFault=0*
Response	None.
Example	`ETH:PORT1:OAM:VLAN:LEV1:ID 1024`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels. Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame).

Syntax	ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:ID?
Description	This query returns the VLAN ID.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N)
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:VLAN:LEV1:ID?` → `1024`
Note	

12.32.99 ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:CFI

Syntax	ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:CFI <enable>
Description	This command enables/disables the VLAN canonical format indicator.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N[1])
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:VLAN:LEV1:CFI ON`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels. Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame). [2] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition [3] This command is as same as `ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:DEI`

Syntax	ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:CFI?
Description	This query returns the state of the VLAN canonical format indicator.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N)
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:VLAN:LEV1:CFI?` → `1`
Note	[1] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition [2] This command is as same as `ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:DEI?`

12.32.100 ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:DEI

Syntax	ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:DEI <enable>
Description	This command enables/disables the VLAN canonical format indicator.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N[1])
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:VLAN:LEV1:DEI ON`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels. Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame). [2] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition [3] This command is as same as `ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:CFI`

Syntax	ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:DEI?
Description	This query returns the state of the VLAN canonical format indicator.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N)
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:VLAN:LEV1:DEI?` → `1`
Note	[1] CFI bit in the VLAN tag was renamed to 'DEI' in IEEE802.1Q 2014 edition [2] This command is as same as `ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:CFI?`

12.32.101 ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:PRIority

Syntax	ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:PRIority <priority>
Description	This command sets the VLAN priority.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N[1])
	<priority> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=7, DEFault=0*
Response	None.
Example	`ETH:PORT1:OAM:VLAN:LEV1:PRI 7`
Notes	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels. Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame).

Syntax	ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:PRIority?
Description	This query returns the VLAN priority.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N)
Response	<priority> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:VLAN:LEV1:PRI? → 7`
Note	

12.32.102 ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:ETYPe

Syntax	ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:ETYPe <type>
Description	This command sets the VLAN Ethertype.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N)[1]
	<type> = <NUMERIC PROGRAM DATA> Acceptable values: #H8100 #H88A8 #H9100 #H9200 *DEFault=#H8100*
Response	None.
Example	`ETH:PORT1:OAM:VLAN:LEV1:ETYP #H8100`
Notes	[1] It is only possible to use this command for levels (1-M), where M is the number of active VLAN Levels minus one. Level 1 corresponds to the outer tag (closest to the Ethernet header) and N is the inner tag (closest to the payload portion of the frame). It is not possible to use this command for VLAN level N because Ethertype at this level is automatically set. MAC level Ethertype is set by :ETH:PORT1:OAM:MAC:ETYPe command.

Syntax	ETHernet:PORT<Pt>:OAM:VLAN:LEVel<Lv>:ETYPe?
Description	This query returns the OAM VLAN Ethertype.
Parameters	<Pt> = Port number
	<Lv> = VLAN level (1-N[1])
Response	<type> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:VLAN:LEV1:ETYP? → #H8100`
Note	[1] It is only possible to use this command for levels 1-N, where N is the number of active VLAN Levels.

12.32.103 ETHernet:PORT<Pt>:OAM:OUTPut:ULBM

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:ULBM <mepId>
Description	This command activates OAM unicast loop-back message event.
Parameters	<Pt> = Port number
	<mepId> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191*
Response	None.
Example	`ETH:PORT1:OAM:OUTP:ULBM 10`
Note	Only used when 802.ag or Y.1731 is active.

12.32.104 ETHernet:PORT<Pt>:OAM:OUTPut:MLBM

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:MLBM <mepId>
Description	This command activates OAM multicast loop-back message event.
Parameters	<Pt> = Port number
	<mepId> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191*
Response	None.
Example	`ETH:PORT1:OAM:OUTP:MLBM 10`
Note	Only used when 802.ag or Y.1731 is active.

12.32.105 ETHernet:PORT<Pt>:OAM:OUTPut:LTM

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:LTM <mepId>
Description	This command activates the link trace message event.
Parameters	<Pt> = Port number
	<mepId> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191*
Response	None.
Example	`ETH:PORT1:OAM:OUTP:LTM 10`
Note	Only used when 802.ag or Y.1731 is active.

12.32.106 ETHernet:PORT<Pt>:OAM:OUTPut:MCC

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:MCC <mepId>
Description	This command activates the MCC event.
Parameters	<Pt> = Port number
	<mepId> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191*
Response	None.
Example	`ETH:PORT1:OAM:OUTP:MCC 10`
Note	Only used when Y.1731 is active.

12.32.107 ETHernet:PORT<Pt>:OAM:OUTPut:EXM

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:EXM <mepId>
Description	This command activates the experimental OAM message event.
Parameters	<Pt> = Port number
	<mepId> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191*
Response	None.
Example	`ETH:PORT1:OAM:OUTP:EXM`
Note	Only used when Y.1731 is active.

12.32.108 ETHernet:PORT<Pt>:OAM:OUTPut:VSM

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:VSM <mepId>
Description	This command activates the vendor specific OAM message event.
Parameters	<Pt> = Port number
	<mepId> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191*
Response	None.
Example	`ETH:PORT1:OAM:OUTP:VSM 10`
Note	Only used when Y.1731 is active.

12.32.109 ETHernet:PORT<Pt>:OAM:OUTPut:TST

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:TST <mepId>
Description	This command activates the OAM test event.
Parameters	<Pt> = Port number
	<mepId> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191*
Response	None.
Example	`ETH:PORT1:OAM:OUTP:TST 10`
Note	Only used when Y.1731 is active.

12.32.110 ETHernet:PORT<Pt>:OAM:OUTPut:LMM

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:LMM <mepId>
Description	This command activates the loss measurement message event.
Parameters	<Pt> = Port number
	<mepId> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191*
Response	None.
Example	`ETH:PORT1:OAM:OUTP:LMM 10`
Note	Only used when Y.1731 is active.

12.32.111 ETHernet:PORT<Pt>:OAM:OUTPut:SLM

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:SLM <mepId>
Description	This command activates the synthetic loss measurement event.
Parameters	<Pt> = Port number
	<mepId> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191*
Response	None.
Example	`ETH:PORT1:OAM:OUTP:SLM 10`
Note	Only used when Y.1731 is active.

12.32.112 ETHernet:PORT<Pt>:OAM:OUTPut:DMONe

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:DMONe <mepId>
Description	This command activates the one-way delay measurement event.
Parameters	<Pt> = Port number
	<mepId> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191*
Response	None.
Example	`ETH:PORT1:OAM:OUTP:DMON 10`
Note	Only used when Y.1731 is active.

12.32.113 ETHernet:PORT<Pt>:OAM:OUTPut:DMM

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:DMM <mepId>
Description	This command activates the delay measurement message event.
Parameters	<Pt> = Port number
	<mepId> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=8191*
Response	None.
Example	`ETH:PORT1:OAM:OUTP:DMM 10`
Note	Only used when Y.1731 is active.

12.32.114 ETHernet:PORT<Pt>:OAM:OUTPut:CCM

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:CCM <enable>
Description	This command enables/disables continuity check messages.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:OUTP:CCM OFF`
Note	Only used when 802.ag or Y.1731 is active.

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:CCM?
Description	This query returns whether or not continuity check messages are enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:OUTP:CCM?` → `0`
Note	

12.32.115 ETHernet:PORT<Pt>:OAM:OUTPut:AIS

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:AIS <enable>
Description	This command enables/disables AIS alarm.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:OUTP:AIS OFF`
Note	Only used when Y.1731 is active.

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:AIS?
Description	This query returns whether or not AIS alarm is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:OUTP:AIS?` → `0`
Note	

12.32.116 ETHernet:PORT<Pt>:OAM:OUTPut:LCK

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:LCK <enable>
Description	This command enables/disables lock condition.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:OUTP:LCK OFF`
Note	Only used when Y.1731 is active.

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:LCK?
Description	This query returns whether or not lock condition is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:OUTP:LCK?` → `0`
Note	

12.32.117 ETHernet:PORT<Pt>:OAM:OUTPut:LOOP

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:LOOP <enable>
Description	This command enables/disables 802.3ah loop.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:PORT1:OAM:OUTP:LOOP OFF`
Note	Only used when 802.3ah is active.

Syntax	ETHernet:PORT<Pt>:OAM:OUTPut:LOOP?
Description	This query returns whether or not 802.3ah loop is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:OUTP:LOOP?` → `0`
Note	

12.32.118 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:MODE?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:MODE?
Description	This query returns the remote mode string.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:REM:MODE?` → `"N/A"`
Note	

12.32.119 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:PARSer?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:PARSer?
Description	This query returns the remote parser string.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:REM:PARS?` → `"N/A"`
Note	

12.32.120 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:MUXer?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:MUXer?
Description	This query returns the remote muxer string.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:REM:MUX?` → `"N/A"`
Note	

12.32.121 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:OUI?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:OUI?
Description	This query returns the remote OUI string.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:REM:OUI?` → `"N/A"`
Note	

12.32.122 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:VSI?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:VSI?
Description	This query returns the remote VSI string.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:REM:VSI?` → `"N/A"`
Note	

12.32.123 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:UNIDirectional?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:UNIDirectional?
Description	This query returns the remote unidirectional support status.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:REM:UNID?` → `"N/A"`
Note	

12.32.124 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:LEVents?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:LEVents?
Description	This query returns the remote link events support status.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:REM:LEV?` → `"Supported"`
Note	

12.32.125 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:LBACk?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:LBACk?
Description	This query returns the remote loop-back support status.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:REM:LBAC?` → `"Unsupported"`
Note	

12.32.126 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:VRETrieval?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:VRETrieval?
Description	This query returns the remote variable retrieval support status.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:REM:VRET?` → `"N/A"`
Note	

12.32.127 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:REVision?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:REVision?
Description	This query returns the remote revision number.
Parameter	<Pt> = Port number
Response	<NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:REM:REV?` → `0`
Note	

12.32.128 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:MAC?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:MAC?
Description	This query returns the remote MAC address.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:REM:MAC?` → `"N/A"`
Note	

12.32.129 ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:MPDU?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:REMote:MPDU?
Description	This query returns the remote maximum PDU size.
Parameter	<Pt> = Port number
Response	<NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:REM:MPDU?` → `0`
Note	

12.32.130 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:MODE?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:MODE?
Description	This query returns the local mode string.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:LOC:MODE?` → `"N/A"`
Note	

12.32.131 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:PARSer?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:PARSer?
Description	This query returns the local parser string.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:LOC:PARS?` → `"N/A"`
Note	

12.32.132 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:MUXer?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:MUXer?
Description	This query returns the local muxer string.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:LOC:MUX?` → `"N/A"`
Note	

12.32.133 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:UNIDirectional?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:UNIDirectional?
Description	This query returns the local unidirectional support status.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:LOC:UNID?` → `"N/A"`
Note	

12.32.134 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:LEVents?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:LEVents?
Description	This query returns the local link events support status.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:LOC:LEV?` → `"Supported"`
Note	

12.32.135 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:LBACk?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:LBACk?
Description	This query returns the local loop-back support status.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:LOC:LBAC?` → `"Unsupported"`
Note	

12.32.136 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:VRETrieval?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:VRETrieval?
Description	This query returns the local variable retrieval support status.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:LOC:VRET?` → `"N/A"`
Note	

12.32.137 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:REVision?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:REVision?
Description	This query returns the local revision number.
Parameter	<Pt> = Port number
Response	<NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:LOC:REV?` → `0`
Note	

12.32.138 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:MPDU?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:MPDU?
Description	This query returns the local maximum PDU size.
Parameter	<Pt> = Port number
Response	<NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:LOC:MPDU?` → `0`
Note	

12.32.139 ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:STATe?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:LOCal:STATe?
Description	This query returns the local state string.
Parameter	<Pt> = Port number
Response	<STRING RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:LOC:STAT?` → `"N/A"`
Note	

12.32.140 ETHernet:PORT<Pt>:OAM:STATus:AH:VARiable:REQuest

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:VARiable:REQuest <variable>
Description	This command requests the specified variable from the remote side.
Parameters	<Pt> = Port number
	<variable> = <CHARACTER PROGRAM DATA> TUCodes: Tx Unsupported codes RUCodes: Rx Unsupported codes TINFormation: Tx Information RINFormation: Rx Information TUEVent: Tx Unique Event RUEVent: Rx Unique Event TDEVent: Tx Duplicate Event RDEVent: Rx Duplicate Event TLBControl: Tx Loop-back control RLBControl: Rx Loop-back control TVREQuest: Tx Variable request RVREQuest: Rx Variable request TVResponse: Tx Variable response RVResponse: Rx Variable response TOSPecific: Tx Organizational specific ROSPecific: Rx Organizational specific LSYConfig: Local errored symbol config LSYevent: Local errored symbol event LFConfig: Local errored frame config LFEVent: Local errored frame event LPConfig: Local errored period config LPEVent: Local errored period event LSConfig: Local errored seconds config LSEVent: Local errored seconds event RSYevent: Remote errored symbol event RFEVent: Remote errored frame event RPEVent: Remote errored period event RSEVent: Remote errored seconds event FLOerror: Frames lost to OAM error *DEFault = TUC*
Response	None.
Example	`ETH:PORT1:OAM:STAT:AH:VAR:REQ TUC`
Note	The requested parameter is fetched with the command `ETH:PORT1:OAM:STAT:AH:VAR:RESP?` I may take a few seconds before the response is obtainable.

12.32.141 ETHernet:PORT<Pt>:OAM:STATus:AH:VARiable:RESPonse?

Syntax	ETHernet:PORT<Pt>:OAM:STATus:AH:VARiable:RESPonse?
Description	This command fetches the latest variable response from the remote side.
Parameters	<Pt> = Port number
Response	<variable> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:OAM:STAT:AH:VAR:RESP?` → `"Unsupported Codes Tx :  0"`
Note	

12.32.142 ETHernet:PORT<Pt>:OAM:LOG?

Syntax	ETHernet:PORT<Pt>:OAM:LOG?
Description	This query returns a lists of logged OAM events.
Parameter	<Pt> = Port number
Response	<log> = <STRING RESPONSE DATA>

Example	`ETH:PORT1:OAM:LOG?` →
	`"16:15:54 LTM transaction ID 1 passed`
	`16:16:01 MCC message sent`
	`"`

Note	

12.33 TCP Throughput Test - RFC-6349

The SCPI commands in this section are available in the following application:
- TP-RFC6349-ETH: Ethernet RFC-6349 test application.
This applications supports one port only.

12.33.1 ETHernet:TTTest:SETup:TCPPort

Syntax	ETHernet:TTTest:SETup:TCPPort <port>
Description	This command sets the TCP port number.
Parameter	<port> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=65535, DEFault=5001*
Response	None.
Example	`ETH:TTT:SET:TCPP 5001`
Note	

Syntax	ETHernet:TTTest:SETup:TCPPort?
Description	This query returns the TCP port number.
Parameter	None.
Response	<port> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:TCPP? → 5001`
Note	

12.33.2 ETHernet:TTTest:SETup:FATSequence

Syntax	ETHernet:TTTest:SETup:FATSequence <enable>
Description	This command enables/disables full auto test sequence.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:TTT:SET:FATS OFF`
Note	

Syntax	ETHernet:TTTest:SETup:FATSequence?
Description	This query returns whether or not full auto test sequence is enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:FATS? → 0`
Note	

12.33.3 ETHernet:TTTest:SETup:CIPerf

Syntax	ETHernet:TTTest:SETup:CIPerf <enable>
Description	This command enables/disables connection to iPerf server.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:TTT:SET:CIP OFF`
Note	

Syntax	ETHernet:TTTest:SETup:CIPerf?
Description	This query returns whether or not connection to iPerf server is enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:CIP? → 0`
Note	

12.33.4 ETHernet:TTTest:SETup:SLSName

Syntax	ETHernet:TTTest:SETup:SLSName <name>
Description	This command sets the short local site name.
Parameter	<name> = <STRING PROGRAM DATA> The supplied name is truncated to a maximum of 4 characters.
Response	None.
Example	`ETH:TTT:SET:SLSN "LOC"`
Note	

Syntax	ETHernet:TTTest:SETup:SLSName?
Description	This query returns the short local site name.
Parameter	None.
Response	<name> = <STRING RESPONSE DATA>
Example	`ETH:TTT:SET:SLSN? → "LOC"`
Note	

12.33.5 ETHernet:TTTest:SETup:LLSName

Syntax	ETHernet:TTTest:SETup:LLSName <name>
Description	This command sets the long local site name.
Parameter	<name> = <STRING PROGRAM DATA> The supplied name is truncated to a maximum of 12 characters.
Response	None.
Example	`ETH:TTT:SET:LLSN "LOCAL"`
Note	

Syntax	ETHernet:TTTest:SETup:LLSName?
Description	This query returns the long local site name.
Parameter	None.
Response	<name> = <STRING RESPONSE DATA>
Example	`ETH:TTT:SET:LLSN? → "LOCAL"`
Note	

12.33.6 ETHernet:TTTest:SETup:SRSName

Syntax	ETHernet:TTTest:SETup:SRSName <name>
Description	This command sets the short remote site name.
Parameter	<name> = <STRING PROGRAM DATA> The supplied name is truncated to a maximum of 4 characters.
Response	None.
Example	`ETH:TTT:SET:SRSN "S123"`
Note	

Syntax	ETHernet:TTTest:SETup:SRSName?
Description	This query returns the short remote site name.
Parameter	None.
Response	<name> = <STRING RESPONSE DATA>
Example	`ETH:TTT:SET:SRSN? → "S123"`
Note	

12.33.7 ETHernet:TTTest:SETup:LRSName

Syntax	ETHernet:TTTest:SETup:LRSName <name>
Description	This command sets the long remote site name.
Parameter	<name> = <STRING PROGRAM DATA> The supplied name is truncated to a maximum of 12 characters.
Response	None.
Example	`ETH:TTT:SET:LRSN "SITE-1-23"`
Note	

Syntax	ETHernet:TTTest:SETup:LRSName?
Description	This query returns the long remote site name.
Parameter	None.
Response	<name> = <STRING RESPONSE DATA>
Example	`ETH:TTT:SET:LRSN? → "SITE-1-23"`
Note	

12.33.8 ETHernet:TTTest:SETup:TDLR

Syntax	ETHernet:TTTest:SETup:TDLR <enable>
Description	This command enables/disables the test direction local to remote.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:TTT:SET:TDLR ON`
Note	

Syntax	ETHernet:TTTest:SETup:TDLR?
Description	This query returns whether or not the test direction local to remote is enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:TDLR? → 1`
Note	

12.33.9 ETHernet:TTTest:SETup:TDRL

Syntax	ETHernet:TTTest:SETup:TDRL <enable>
Description	This command enables/disables the test direction remote to local.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:TTT:SET:TDRL OFF`
Note	

Syntax	ETHernet:TTTest:SETup:TDRL?
Description	This query returns whether or not the test direction remote to local is enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:TDRL? → 0`
Note	

12.33.10 ETHernet:TTTest:SETup:TDSimultaneos

Syntax	ETHernet:TTTest:SETup:TDSimultaneos <enable>
Description	This command enables/disables the test direction both simultaneous.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:TTT:SET:TDS OFF`
Note	

Syntax	ETHernet:TTTest:SETup:TDSimultaneos?
Description	This query returns whether or not the test direction simultaneous is enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:TDS?` → `0`
Note	

12.33.11 ETHernet:TTTest:SETup[:LTRemote]:CIRate

Syntax	ETHernet:TTTest:SETup[:LTRemote]:CIRate <rate>
Description	This command sets the local to remote CIR rate (upstream). Unit: Mbps.
Parameter	<rate> = <NUMERIC PROGRAM DATA> *MINimum=0.010, MAXimum=10000, DEFault=1000*
Response	None.
Example	`ETH:TTT:SET:CIR 1000`
Note	

Syntax	ETHernet:TTTest:SETup[:LTRemote]:CIRate?
Description	This query returns the local to remote CIR rate.
Parameter	None.
Response	<rate> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:CIR?` → `1000`
Note	

12.33.12 ETHernet:TTTest:SETup[:LTRemote]:MTUSize

Syntax	ETHernet:TTTest:SETup[:LTRemote]:MTUSize <size>
Description	This command sets the local to remote MTU size.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum=64, MAXimum=15982, DEFault=1500*
Response	None.
Example	`ETH:TTT:SET:MTUS 1500`
Note	This MTU size is used when `ETHernet:TTTest:SETup:PMTU:DISCover:ENABle` is disabled.

Syntax	ETHernet:TTTest:SETup[:LTRemote]:MTUSize?
Description	This query returns the local to remote MTU size.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:MTUS?` → `1500`
Note	

12.33.13 ETHernet:TTTest:SETup:RTLocal:CIRate

Syntax	ETHernet:TTTest:SETup:RTLocal:CIRate <rate>
Description	This command sets the remote to local CIR rate (downstream). Unit: Mbps.
Parameter	<rate> = <NUMERIC PROGRAM DATA> *MINimum=0.010, MAXimum=10000, DEFault=1000*
Response	None.
Example	`ETH:TTT:SET:RTL:CIR 1000`
Note	

Syntax	ETHernet:TTTest:SETup:RTLocal:CIRate?
Description	This query returns the remote to local CIR rate.
Parameter	None.
Response	<rate> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:RTL:CIR?` → `1000`
Note	

12.33.14 ETHernet:TTTest:SETup:RTLocal:MTUSize

Syntax	ETHernet:TTTest:SETup:RTLocal:MTUSize <size>
Description	This command sets the remote to local MTU size.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum=64, MAXimum=15982, DEFault=1500*
Response	None.
Example	`ETH:TTT:SET:RTL:MTUS 1500`
Note	This MTU size is used when `ETHernet:TTTest:SETup:PMTU:DISCover:ENABle` is disabled.

Syntax	ETHernet:TTTest:SETup:RTLocal:MTUSize?
Description	This query returns the remote to local MTU size.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:RTL:MTUS?` → `1500`
Note	

12.33.15 ETHernet:TTTest:SETup:PMTU:DISCover[:ENABle]

Syntax	ETHernet:TTTest:SETup:PMTU:DISCover[:ENABle] <enable>
Description	This command enables/disables path MTU discovery.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:TTT:SET:PMTU:DISC ON`
Note	

Syntax	ETHernet:TTTest:SETup:PMTU:DISCover[:ENABle]?
Description	This query returns whether or not path MTU discovery is enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:PMTU:DISC?` → `1`
Note	

12.33.16 ETHernet:TTTest:SETup:PMTU:DISCover:MINimum

Syntax	ETHernet:TTTest:SETup:PMTU:DISCover:MINimum <size>
Description	This command sets the path MTU discovery minimum size.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum=64, MAXimum=15982, DEFault=512*
Response	None.
Example	`ETH:TTT:SET:PMTU:DISC:MIN 512`
Note	

Syntax	ETHernet:TTTest:SETup:PMTU:DISCover:MINimum?
Description	This query returns the path MTU discovery minimum size.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:PMTU:DISC:MIN? → 512`
Note	

12.33.17 ETHernet:TTTest:SETup:PMTU:DISCover:MAXimum

Syntax	ETHernet:TTTest:SETup:PMTU:DISCover:MAXimum <size>
Description	This command sets the path MTU discovery maximum size.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum=64, MAXimum=15982, DEFault=1500*
Response	None.
Example	`ETH:TTT:SET:PMTU:DISC:MAX 1500`
Note	

Syntax	ETHernet:TTTest:SETup:PMTU:DISCover:MAXimum?
Description	This query returns the path MTU discovery maximum size.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:PMTU:DISC:MAX? → 1500`
Note	

12.33.18 ETHernet:TTTest:SETup:BRTT

Syntax	ETHernet:TTTest:SETup:BRTT <size>
Description	This command sets the baseline RTT. Unit: ms.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum=0.001, MAXimum=5000, DEFault=2*
Response	None.
Example	`ETH:TTT:SET:BRTT 2`
Note	This value is used when `ETHernet:TTTest:SETup:BRTT:AUTO` is disabled.

Syntax	ETHernet:TTTest:SETup:BRTT?
Description	This query returns the baseline RTT.
Parameter	None.
Response	<size> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:BRTT? → 0.020`
Note	

12.33.19 ETHernet:TTTest:SETup:BRTT:AUTO

Syntax	ETHernet:TTTest:SETup:BRTT:AUTO <enable>
Description	This command enables/disables a test to determine the baseline RTT.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:TTT:SET:BRTT:AUTO ON`
Note	

Syntax	ETHernet:TTTest:SETup:BRTT:AUTO?
Description	This query returns whether or not the test to determine the baseline RTT is enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:BRTT:AUTO? → 1`
Note	

12.33.20 ETHernet:TTTest:SETup:BRTT:AUTO:DURation

Syntax	ETHernet:TTTest:SETup:BRTT:AUTO:DURation <duration>
Description	This command sets the duration of the RTT discovery test. Unit: seconds.
Parameter	<duration> = <NUMERIC PROGRAM DATA> *MINimum=5, MAXimum=86399, DEFault=10*
Response	None.
Example	`ETH:TTT:SET:BRTT:AUTO:DUR 10`
Note	

Syntax	ETHernet:TTTest:SETup:BRTT:AUTO:DURation?
Description	This query returns the RTT discovery test duration.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:BRTT:AUTO:DUR? → 10`
Note	

12.33.21 ETHernet:TTTest:SETup:WSTTest[:ENABle]

Syntax	ETHernet:TTTest:SETup:WSTTest[:ENABle] <enable>
Description	This command enables/disables the window scan and throughput test.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:TTT:SET:WSTT ON`
Note	

Syntax	ETHernet:TTTest:SETup:WSTTest[:ENABle]?
Description	This query returns whether or not the window scan and throughput test is enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:WSTT? → 1`
Note	

12.33.22 ETHernet:TTTest:SETup:WSTTest:MODE

Syntax	ETHernet:TTTest:SETup:WSTTest:MODE <mode>
Description	This command sets the window scan and throughput test mode.
Parameter	<mode> = <CHARACTER PROGRAM DATA> AUTO: Scans the window size in four predefined steps. EXPert: Scans the window size in a number of user defined steps. *DEFault = AUTO*
Response	None.
Example	`ETH:TTT:SET:WSTT:MODE AUTO`
Note	

Syntax	ETHernet:TTTest:SETup:WSTTest:MODE?
Description	This query returns the window scan and throughput test mode.
Parameter	None.
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`ETH:TTT:SET:WSTT:MODE? → AUTO`
Note	

12.33.23 ETHernet:TTTest:SETup:WSCan[:LTRemote]:SDURation

Syntax	ETHernet:TTTest:SETup:WSCan[:LTRemote]:SDURation <duration>
Description	This command sets the window scan step duration in the local to remote direction. Unit: seconds.
Parameter	<duration> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=86399, DEFault=10*
Response	None.
Example	`ETH:TTT:SET:WSC:SDUR 10`
Note	

Syntax	ETHernet:TTTest:SETup:WSCan[:LTRemote]:SDURation?
Description	This query returns the window scan step duration in the local to remote direction.
Parameter	None.
Response	<duration> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:WSC:SDUR? → 10`
Note	

12.33.24 ETHernet:TTTest:SETup:WSCan[:LTRemote]:AMODe:MWSize

Syntax	ETHernet:TTTest:SETup:WSCan[:LTRemote]:AMODe:MWSize <size>
Description	This command sets the auto mode max window size in the local to remote direction.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum=16384, MAXimum=8388607, DEFault=65535*
Response	None.
Example	`ETH:TTT:SET:WSC:AMOD:MWS 65535`
Note	

Syntax	ETHernet:TTTest:SETup:WSCan[:LTRemote]:AMODe:MWSize?
Description	This query returns the auto mode max window size in the local to remote direction.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:WSC:AMOD:MWS? → 65535`
Note	

12.33.25 ETHernet:TTTest:SETup:WSCan:RTLocal:SDURation

Syntax	ETHernet:TTTest:SETup:WSCan:RTLocal:SDURation <duration>
Description	This command sets the window scan step duration. in the remote to local direction. Unit: seconds.
Parameter	<duration> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=86399, DEFault=10*
Response	None.
Example	`ETH:TTT:SET:WSC:RTL:SDUR 10`
Note	

Syntax	ETHernet:TTTest:SETup:WSCan:RTLocal:SDURation?
Description	This query returns the window scan step duration in the remote to local direction.
Parameter	None.
Response	<duration> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:WSC:RTL:SDUR? → 10`
Note	

12.33.26 ETHernet:TTTest:SETup:WSCan:RTLocal:AMODe:MWSize

Syntax	ETHernet:TTTest:SETup:WSCan:RTLocal:AMODe:MWSize <size>
Description	This command sets the auto mode max window size in the remote to local direction.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum=16384, MAXimum=8388607, DEFault=65535*
Response	None.
Example	`ETH:TTT:SET:WSC:RTL:AMOD:MWS 65535`
Note	

Syntax	ETHernet:TTTest:SETup:WSCan:RTLocal:AMODe:MWSize?
Description	This query returns the auto mode max window size in the remote to local direction.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:WSC:RTL:AMOD:MWS? → 65535`
Note	

12.33.27 ETHernet:TTTest:SETup:WSCan:AMODe[:LTRemote]:STEP<no> [:ENABle]

Syntax	ETHernet:TTTest:SETup:WSCan:AMODe[:LTRemote]:STEP<no>[:ENABle] <enable>
Description	This command enables/disables the auto mode steps in the local to remote direction.
Parameters	<no> = Step number (1-3)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:TTT:SET:WSC:AMOD:STEP1 ON`
Note	

Syntax	ETHernet:TTTest:SETup:WSCan:AMODe[:LTRemote]:STEP<no>[:ENABle]?
Description	This query returns whether or not the auto mode step is enabled in the local to remote direction.
Parameter	<no> = Step number (1-3)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:WSC:AMOD:STEP1? → 1`
Note	

12.33.28 ETHernet:TTTest:SETup:WSCan:AMODe:RTLocal:STEP<no>[:ENABle]

Syntax	ETHernet:TTTest:SETup:WSCan:AMODe:RTLocal:STEP<no>[:ENABle] <enable>
Description	This command enables/disables the auto mode steps in the remote to local direction.
Parameters	<no> = Step number (1-3)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:TTT:SET:WSC:AMOD:RTL:STEP1 ON`
Note	

Syntax	ETHernet:TTTest:SETup:WSCan:AMODe:RTLocal:STEP<no>[:ENABle]?
Description	This query returns whether or not the auto mode step is enabled in the remote to local direction.
Parameter	<no> = Step number (1-3)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:WSC:AMOD:RTL:STEP1?` → `1`
Note	

12.33.29 ETHernet:TTTest:SETup:WSCan:EMODe[:LTRemote]:STEP<no> [:ENABle]

Syntax	ETHernet:TTTest:SETup:WSCan:EMODe[:LTRemote]:STEP<no>[:ENABle] <enable>
Description	This command enables/disables the expert mode steps in the local to remote direction.
Parameters	<no> = Step number (1-5)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:TTT:SET:WSC:EMOD:STEP1 ON`
Note	

Syntax	ETHernet:TTTest:SETup:WSCan:EMODe[:LTRemote]:STEP<no>[:ENABle]?
Description	This query returns whether or not the expert mode step is enabled in the local to remote direction.
Parameter	<no> = Step number (1-5)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:WSC:EMOD:STEP1?` → `1`
Note	

12.33.30 ETHernet:TTTest:SETup:WSCan:EMODe[:LTRemote]:STEP<no>:WSIZe

Syntax	ETHernet:TTTest:SETup:WSCan:EMODe[:LTRemote]:STEP<no>:WSIZe <size>
Description	This command sets the expert mode step window size in the local to remote direction.
Parameters	<no> = Step number (1-5)
	<size> = <NUMERIC PROGRAM DATA> *MINimum=8, MAXimum=8388607, DEFault=65535*
Response	None.
Example	`ETH:TTT:SET:WSC:EMOD:STEP1:WSIZ 65535`
Note	

Syntax	ETHernet:TTTest:SETup:WSCan:EMODe[:LTRemote]:STEP<no>:WSIZe?
Description	This query returns the expert mode step window size in the local to remote direction.
Parameter	<no> = Step number (1-5)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:WSC:EMOD:STEP1:WSIZ?` → `65535`
Note	

12.33.31 ETHernet:TTTest:SETup:WSCan:EMODe[:LTRemote]:STEP<no> CONNections

Syntax	ETHernet:TTTest:SETup:WSCan:EMODe[:LTRemote]:STEP<no>:CONNections <count>
Description	This command sets the expert mode step connections count in the local to remote direction.
Parameters	<no> = Step number (1-5)
	<count> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=16, DEFault=1*
Response	None.
Example	`ETH:TTT:SET:WSC:EMOD:STEP1:CONN 1`
Note	

Syntax	ETHernet:TTTest:SETup:WSCan:EMODe[:LTRemote]:STEP<no>:CONNections?
Description	This query returns the expert mode step connections count in the local to remote direction.
Parameter	<no> = Step number (1-5)
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:WSC:EMOD:STEP1:CONN? → 1`
Note	

12.33.32 ETHernet:TTTest:SETup:WSCan:EMODe:RTLocal:STEP<no>[:ENABle]

Syntax	ETHernet:TTTest:SETup:WSCan:EMODe:RTLocal:STEP<no>[:ENABle] <enable>
Description	This command enables/disables the expert mode steps in the remote to local direction.
Parameters	<no> = Step number (1-5)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:TTT:SET:WSC:EMOD:RTL:STEP1 ON`
Note	

Syntax	ETHernet:TTTest:SETup:WSCan:EMODe:RTLocal:STEP<no>[:ENABle]?
Description	This query returns whether or not the expert mode step is enabled in the remote to local direction.
Parameter	<no> = Step number (1-5)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:WSC:EMOD:RTL:STEP1? → 1`
Note	

12.33.33 ETHernet:TTTest:SETup:WSCan:EMODe:RTLocal:STEP<no>:WSIZe

Syntax	ETHernet:TTTest:SETup:WSCan:EMODe:RTLocal:STEP<no>:WSIZe <size>
Description	This command sets the expert mode step window size in the remote to local direction.
Parameters	<no> = Step number (1-5)
	<size> = <NUMERIC PROGRAM DATA> *MINimum=8, MAXimum=8388607, DEFault=65535*
Response	None.
Example	`ETH:TTT:SET:WSC:EMOD:RTL:STEP1:WSIZ 65535`
Note	

Syntax	ETHernet:TTTest:SETup:WSCan:EMODe:RTLocal:STEP<no>:WSIZe?
Description	This query returns the expert mode step window size in the remote to local direction.
Parameter	<no> = Step number (1-5)
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:WSC:EMOD:RTL:STEP1:WSIZ? → 65535`
Note	

12.33.34 ETHernet:TTTest:SETup:WSCan:EMODe:RTLocal:STEP<no> :CONNections

Syntax	ETHernet:TTTest:SETup:WSCan:EMODe:RTLocal:STEP<no>:CONNections <count>
Description	This command sets the expert mode step connections count in the remote to local direction.
Parameters	<no> = Step number (1-5)
	<count> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=16, DEFault=1*
Response	None.
Example	`ETH:TTT:SET:WSC:EMOD:RTL:STEP1:CONN 1`
Note	

Syntax	ETHernet:TTTest:SETup:WSCan:EMODe:RTLocal:STEP<no>:CONNections?
Description	This query returns the expert mode step connections count in the remote to local direction.
Parameter	<no> = Step number (1-5)
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:WSC:EMOD:RTL:STEP1:CONN? → 1`
Note	

12.33.35 ETHernet:TTTest:SETup:TTESt[:LTRemote]:THReshold[:ENABle]

Syntax	ETHernet:TTTest:SETup:TTESt[:LTRemote]:THReshold[:ENABle] <enable>
Description	This command enables/disables throughput test threshold in the local to remote direction.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:TTT:SET:TTES:THR ON`
Note	

Syntax	ETHernet:TTTest:SETup:TTESt[:LTRemote]:THReshold[:ENABle]?
Description	This query returns whether or not throughput test threshold is enabled in the local to remote direction.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:TTES:THR? → 1`
Note	

12.33.36 ETHernet:TTTest:SETup:TTESt[:LTRemote]:THReshold:PERCentage

Syntax	ETHernet:TTTest:SETup:TTESt[:LTRemote]:THReshold:PERCentage <pct>
Description	This command sets the throughput test threshold value in the local to remote direction. Unit: percentage.
Parameter	<pct> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100.00, DEFault=95*
Response	None.
Example	`ETH:TTT:SET:TTES:THR:PERC 95`
Note	

Syntax	ETHernet:TTTest:SETup:TTESt[:LTRemote]:THReshold:PERCentage?
Description	This query returns the throughput test threshold value in the local to remote direction.
Parameter	None.
Response	<pct> = <NR2 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:TTES:THR:PERC? → 95.00`
Note	

12.33.37 ETHernet:TTTest:SETup:TTESt[:LTRemote]:SDURation

Syntax	ETHernet:TTTest:SETup:TTESt[:LTRemote]:SDURation <duration>
Description	This command sets the throughput test step duration in the local to remote direction. Unit: seconds.
Parameter	<duration> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=86399, DEFault=10*
Response	None.
Example	`ETH:TTT:SET:TTES:SDUR 10`
Note	

Syntax	ETHernet:TTTest:SETup:TTESt[:LTRemote]:SDURation?
Description	This query returns the throughput test step duration in the local to remote direction.
Parameter	None.
Response	<duration> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:TTES:SDUR?` → `10`
Note	

12.33.38 ETHernet:TTTest:SETup:TTESt:RTLocal:THReshold[:ENABle]

Syntax	ETHernet:TTTest:SETup:TTESt:RTLocal:THReshold[:ENABle] <enable>
Description	This query returns whether or not throughput test threshold is enabled in the remote to local direction.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:TTT:SET:TTES:RTL:THR ON`
Note	

Syntax	ETHernet:TTTest:SETup:TTESt:RTLocal:THReshold[:ENABle]?
Description	This query returns whether or not throughput test threshold is enabled in the remote to local direction.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:TTES:RTL:THR?` → `1`
Note	

12.33.39 ETHernet:TTTest:SETup:TTESt:RTLocal:THReshold:PERCentage

Syntax	ETHernet:TTTest:SETup:TTESt:RTLocal:THReshold:PERCentage <pct>
Description	This command sets the throughput test threshold value in the remote to local direction. Unit: percentage.
Parameter	<pct> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100.00, DEFault=95*
Response	None.
Example	`ETH:TTT:SET:TTES:RTL:THR:PERC 95`
Note	

Syntax	ETHernet:TTTest:SETup:TTESt:RTLocal:THReshold:PERCentage?
Description	This query returns the throughput test threshold value in the remote to local direction.
Parameter	None.
Response	<pct> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:TTES:RTL:THR:PERC?` → `95`
Note	

12.33.40 ETHernet:TTTest:SETup:TTESt:RTLocal:SDURation

Syntax	ETHernet:TTTest:SETup:TTESt:RTLocal:SDURation <duration>
Description	This command sets the throughput test step duration in the remote to local direction. Unit: seconds.
Parameter	<duration> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=86399, DEFault=10*
Response	None.
Example	`ETH:TTT:SET:TTES:RTL:SDUR 10`
Note	

Syntax	ETHernet:TTTest:SETup:TTESt:RTLocal:SDURation?
Description	This query returns the throughput test step duration in the remote to local direction.
Parameter	None.
Response	<duration> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:TTES:RTL:SDUR? → 10`
Note	

12.33.41 ETHernet:TTTest:SETup:TTESt:EMODe[:LTRemote][:ENABle]

Syntax	ETHernet:TTTest:SETup:TTESt:EMODe[:LTRemote][:ENABle] <enable>
Description	This command enables/disables expert mode throughput test in the local to remote direction.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:TTT:SET:TTES:EMOD ON`
Note	

Syntax	ETHernet:TTTest:SETup:TTESt:EMODe[:LTRemote][:ENABle]?
Description	This query returns whether or not expert mode throughput test is enabled in the local to remote direction.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:TTES:EMOD? → 1`
Note	

12.33.42 ETHernet:TTTest:SETup:TTESt:EMODe[:LTRemote]:WSIZe

Syntax	ETHernet:TTTest:SETup:TTESt:EMODe[:LTRemote]:WSIZe <size>
Description	This command sets the expert mode throughput test window size in the local to remote direction.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum=8, MAXimum=8388607, DEFault=65535*
Response	None.
Example	`ETH:TTT:SET:TTES:EMOD:WSIZ 65535`
Note	

Syntax	ETHernet:TTTest:SETup:TTESt:EMODe[:LTRemote]:WSIZe?
Description	This query returns the expert mode throughput test window size in the local to remote direction.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:TTES:EMOD:WSIZ? → 65535`
Note	

12.33.43 ETHernet:TTTest:SETup:TTESt:EMODe[:LTRemote]:CONNections

Syntax	ETHernet:TTTest:SETup:TTESt:EMODe[:LTRemote]:CONNections <count>
Description	This command sets the expert mode throughput test connection count in the local to remote direction.
Parameter	<count> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=16, DEFault=1*
Response	None.
Example	`ETH:TTT:SET:TTES:EMOD:CONN 1`
Note	

Syntax	ETHernet:TTTest:SETup:TTESt:EMODe[:LTRemote]:CONNections?
Description	This query returns the expert mode throughput test connection count in the local to remote direction.
Parameter	None.
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:TTES:EMOD:CONN? → 1`
Note	

12.33.44 ETHernet:TTTest:SETup:TTESt:EMODe:RTLocal[:ENABle]

Syntax	ETHernet:TTTest:SETup:TTESt:EMODe:RTLocal[:ENABle] <enable>
Description	This command enables/disables expert mode throughput test in the remote to local direction.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`ETH:TTT:SET:TTES:EMOD:RTL ON`
Note	

Syntax	ETHernet:TTTest:SETup:TTESt:EMODe:RTLocal[:ENABle]?
Description	This query returns whether or not expert mode throughput test is enabled in the remote to local direction.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:TTES:EMOD:RTL? → 1`
Note	

12.33.45 ETHernet:TTTest:SETup:TTESt:EMODe:RTLocal:WSIZe

Syntax	ETHernet:TTTest:SETup:TTESt:EMODe:RTLocal:WSIZe <size>
Description	This command sets the expert mode throughput test window size in the remote to local direction.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum=8, MAXimum=8388607, DEFault=65535*
Response	None.
Example	`ETH:TTT:SET:TTES:EMOD:RTL:WSIZ 65535`
Note	

Syntax	ETHernet:TTTest:SETup:TTESt:EMODe:RTLocal:WSIZe?
Description	This query returns the expert mode throughput test window size in the remote to local direction.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:TTES:EMOD:RTL:WSIZ? → 65535`
Note	

12.33.46 ETHernet:TTTest:SETup:TTESt:EMODe:RTLocal:CONNections

Syntax	ETHernet:TTTest:SETup:TTESt:EMODe:RTLocal:CONNections <count>
Description	This command sets the expert mode throughput test connection count in the remote to local direction.
Parameter	<count> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=16, DEFault=1*
Response	None.
Example	`ETH:TTT:SET:TTES:EMOD:RTL:CONN 1`
Note	

Syntax	ETHernet:TTTest:SETup:TTESt:EMODe:RTLocal:CONNections?
Description	This query returns the expert mode throughput test connection count in the remote to local direction.
Parameter	None.
Response	<count> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:TTES:EMOD:RTL:CONN?` → `1`
Note	

12.33.47 ETHernet:TTTest:SETup:MSERvice[:ENABle]

Syntax	ETHernet:TTTest:SETup:MSERvice[:ENABle] <enable>
Description	This command enables/disables multi service.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:TTT:SET:MSER OFF`
Note	

Syntax	ETHernet:TTTest:SETup:MSERvice[:ENABle]?
Description	This query returns whether or not multi service is enabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:MSER?` → `0`
Note	

12.33.48 ETHernet:TTTest:SETup:MSERvice[:LTRemote]:TDURation

Syntax	ETHernet:TTTest:SETup:MSERvice[:LTRemote]:TDURation <duration>
Description	This command sets the multi service test duration in the local to remote direction. Unit: seconds.
Parameter	<duration> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=86399, DEFault=10*
Response	None.
Example	`ETH:TTT:SET:MSER:TDUR 10`
Note	

Syntax	ETHernet:TTTest:SETup:MSERvice[:LTRemote]:TDURation?
Description	This query returns the multi service test duration in the local to remote direction.
Parameter	None.
Response	<duration> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:MSER:TDUR?` → `10`
Note	

12.33.49 ETHernet:TTTest:SETup:MSERvice[:LTRemote]:WSIZe

Syntax	ETHernet:TTTest:SETup:MSERvice[:LTRemote]:WSIZe <size>
Description	This command sets the multi service window size in the local to remote direction.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum=8, MAXimum=8388607, DEFault=65535*
Response	None.
Example	`ETH:TTT:SET:MSER:WSIZ 65535`
Note	

Syntax	ETHernet:TTTest:SETup:MSERvice[:LTRemote]:WSIZe?
Description	This query returns the multi service window size in the local to remote direction.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:MSER:WSIZ?` → `65535`
Note	This value is used when `ETHernet:TTTest:SETup:MSERvice[:LTRemote]:WSIZe:AUTO` is disabled.

12.33.50 ETHernet:TTTest:SETup:MSERvice[:LTRemote]:WSIZe:AUTO

Syntax	ETHernet:TTTest:SETup:MSERvice[:LTRemote]:WSIZe:AUTO <enable>
Description	This command enables/disables multi service window size automatic calculation in the local to remote direction. The window size is calculated from the bandwidth delay product (BDP).
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:TTT:SET:MSER:WSIZ:AUTO OFF`
Note	

Syntax	ETHernet:TTTest:SETup:MSERvice[:LTRemote]:WSIZe:AUTO?
Description	This query returns whether or not multi service window size automatic calculation is enabled in the local to remote direction.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:MSER:WSIZ:AUTO?` → `0`
Note	

12.33.51 ETHernet:TTTest:SETup:MSERvice:RTLocal:TDURation

Syntax	ETHernet:TTTest:SETup:MSERvice:RTLocal:TDURation <duration>
Description	This command sets the multi service test duration in the remote to local direction. Unit: seconds.
Parameter	<duration> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=86399, DEFault=10*
Response	None.
Example	`ETH:TTT:SET:MSER:RTL:TDUR 10`
Note	

Syntax	ETHernet:TTTest:SETup:MSERvice:RTLocal:TDURation?
Description	This query returns the multi service test duration in the remote to local direction.
Parameter	None.
Response	<duration> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:MSER:RTL:TDUR?` → `10`
Note	

12.33.52 ETHernet:TTTest:SETup:MSERvice:RTLocal:WSIZe

Syntax	ETHernet:TTTest:SETup:MSERvice:RTLocal:WSIZe <size>
Description	This command sets the multi service window size in the remote to local direction.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum=8, MAXimum=8388607, DEFault=65535*
Response	None.
Example	`ETH:TTT:SET:MSER:RTL:WSIZ 65535`
Note	

Syntax	ETHernet:TTTest:SETup:MSERvice:RTLocal:WSIZe?
Description	This query returns the multi service window size in the remote to local direction.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:MSER:RTL:WSIZ?` → `65535`
Note	

12.33.53 ETHernet:TTTest:SETup:MSERvice:RTLocal:WSIZe:AUTO

Syntax	ETHernet:TTTest:SETup:MSERvice:RTLocal:WSIZe:AUTO <enable>
Description	This command enables/disables multi service window size automatic calculation in the remote to local direction.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:TTT:SET:MSER:RTL:WSIZ:AUTO OFF`
Note	

Syntax	ETHernet:TTTest:SETup:MSERvice:RTLocal:WSIZe:AUTO?
Description	This query returns whether or not multi service window size automatic calculation is enabled in the remote to local direction.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:MSER:RTL:WSIZ:AUTO?` → `0`
Note	

12.33.54 ETHernet:TTTest:SETup:MSERvice:SERVice<no>[:ENABle]

Syntax	ETHernet:TTTest:SETup:MSERvice:SERVice<no>[:ENABle] <enable>
Description	This command enables/disables the specified multi service service.
Parameters	<no> = Service number (1-16)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`ETH:TTT:SET:MSER:SERV1 OFF`
Note	

Syntax	ETHernet:TTTest:SETup:MSERvice:SERVice<no>[:ENABle]?
Description	This query returns whether or not a specific multi service service is enabled.
Parameter	<no> = Service number (1-16)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`ETH:TTT:SET:MSER:SERV1?` → `0`
Note	

12.33.55 ETHernet:TTTest:SETup:MSERvice:SERVice<no>:DPORt

Syntax	ETHernet:TTTest:SETup:MSERvice:SERVice<no>:DPORt <no>
Description	This command sets the destination port of a specific service.
Parameters	<no> = Service number (1-16)
	<no> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=65535, DEFault=5001*
Response	None.
Example	`ETH:TTT:SET:MSER:SERV1:DPOR 5001`
Note	

Syntax	ETHernet:TTTest:SETup:MSERvice:SERVice<no>:DPORt?
Description	This query returns the destination port of a specific service.
Parameter	<no> = Service number (1-16)
Response	<no> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:MSER:SERV1:DPOR?` → `5001`
Note	

12.33.56 ETHernet:TTTest:SETup:MSERvice:SERVice<no>:DOT

Syntax	ETHernet:TTTest:SETup:MSERvice:SERVice<no>:DOT <value>
Description	This command sets the DSCP or TOS value of a specific service.
Parameters	<no> = Service number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=0*
Response	None.
Example	`ETH:TTT:SET:MSER:SERV1:DOT 0`
Note	

Syntax	ETHernet:TTTest:SETup:MSERvice:SERVice<no>:DOT?
Description	This query returns the DSCP or TOS value of a specific service.
Parameter	<no> = Service number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:MSER:SERV1:DOT?` → `0`
Note	

12.33.57 ETHernet:TTTest:SETup:MSERvice:SERVice<no>:TCLass

Syntax	ETHernet:TTTest:SETup:MSERvice:SERVice<no>:TCLass <value>
Description	This command sets the IPv6 traffic class value of a specific service.
Parameters	<no> = Service number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=0*
Response	None.
Example	`ETH:TTT:SET:MSER:SERV1:TCL 0`
Note	

Syntax	ETHernet:TTTest:SETup:MSERvice:SERVice<no>:TCLass?
Description	This query returns the IPv6 traffic class value of a specific service.
Parameter	<no> = Service number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:MSER:SERV1:TCL?` → `0`
Note	

12.33.58 ETHernet:TTTest:SETup:MSERvice:SERVice<no>:FLABel

Syntax	ETHernet:TTTest:SETup:MSERvice:SERVice<no>:FLABel <value>
Description	This command sets the IPv6 flow label value of a specific service.
Parameters	<no> = Service number (1-16)
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1048575, DEFault=0*
Response	None.
Example	`ETH:TTT:SET:MSER:SERV1:FLAB 0`
Note	

Syntax	ETHernet:TTTest:SETup:MSERvice:SERVice<no>:FLABel?
Description	This query returns the IPv6 flow label value of a specific service.
Parameter	<no> = Service number (1-16)
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:TTT:SET:MSER:SERV1:FLAB?` → `0`
Note	

12.33.59 ETHernet:TTTest:SETup:MSERvice:SERVice<no>:NAME

Syntax	ETHernet:TTTest:SETup:MSERvice:SERVice<no>:NAME <name>
Description	This command sets the name of a specific service.
Parameters	<no> = Service number (1-16)
	<name> = <STRING PROGRAM DATA> The supplied name is truncated to a maximum of 7 characters.
Response	None.
Example	`ETH:TTT:SET:MSER:SERV1:NAME "Video"`
Note	

Syntax	ETHernet:TTTest:SETup:MSERvice:SERVice<no>:NAME?
Description	This query returns the name of a specific service.
Parameter	<no> = Service number (1-16)
Response	<name> = <STRING RESPONSE DATA>
Example	`ETH:TTT:SET:MSER:SERV1:NAME?` → `"Video"`
Note	

12.33.60 ETHernet:TTTest:RESult:PARameters?

Syntax	ETHernet:TTTest:RESult:PARameters? <direction>
Description	Returns the result of measured parameter values during the test.
Parameter	<direction> = <CHARACTER PROGRAM DATA> LTRemote: Results for the local to remote direction. RTLocal: Results for the remote to local direction. *DEFault = LTRemote*
Response	<mtu> = <NR1 NUMERIC RESPONSE DATA> MTU value. Unit: Octets.
	<rtt> = <NR2 NUMERIC RESPONSE DATA> RTT. Unit: ms.
	<bdp> = <NR1 NUMERIC RESPONSE DATA> Calculated BDP value.
	<cir0> = <NR1 NUMERIC RESPONSE DATA> Layer 0 CIR. Unit: Bytes.
	<cir4> = <NR1 NUMERIC RESPONSE DATA> Layer 4 CIR. Unit: Bytes.
Example	`ETH:TTT:RES:PAR? RTL` → `1500,0.005,261,1000000000,949284785`
Note	

12.33.61 ETHernet:TTTest:RESult:WSCan?

Syntax	ETHernet:TTTest:RESult:WSCan? <direction>
Description	Returns the results of the window scan in the specified direction.
Parameter	<direction> = <CHARACTER PROGRAM DATA> LTRemote: Results for the local to remote direction. RTLocal: Results for the remote to local direction. LTRBoth: Results for the local to remote direction in the simultaneous test step. RTLBoth: Results for the remote to local direction in the simultaneous test step. *DEFault = LTRemote*
Response	A list of results for all window scans and throughput steps.
	<status> = <STRING RESPONSE DATA> "Pending", "Done", "Pass" or "Fail" denotes the status of the test.
	<window-size> = <NR1 NUMERIC RESPONSE DATA> Window size. Unit: Bytes.
	<connections> = <NR1 NUMERIC RESPONSE DATA> Number of connections.
	<avg-throughput> = <NR1 NUMERIC RESPONSE DATA> Average throughput rate. Unit: bps.
	<ideal-throughput> = <NR1 NUMERIC RESPONSE DATA> Ideal throughput rate. Unit: bps.
	<avg-RTT> = <NR2 NUMERIC RESPONSE DATA> Average RTT. Unit: ms.
	<tcp-eff> = <NR2 NUMERIC RESPONSE DATA> TCP Efficiency. Unit: percentage.
Example	`ETH:TTT:RES:WSC? RTL` → `(Pass,1460,1,419779200,94928478,0.002,100.00)`
Note	There are no window scan results for `LTRBoth` and `RTLBoth`

12.33.62 ETHernet:TTTest:RESult:THRoughput?

Syntax	ETHernet:TTTest:RESult:THRoughput? <direction>
Description	Returns the results of the throughput test in the specified direction.
Parameter	<direction> = <CHARACTER PROGRAM DATA> LTRemote: Results for the local to remote direction.

continued on next page...

. . . continued from previous page

	RTLocal: Results for the remote to local direction. LTRBoth: Results for the local to remote direction in the simultaneous test step. RTLBoth: Results for the remote to local direction in the simultaneous test step. *DEFault = LTRemote*
Response	<> = <NR1 NUMERIC RESPONSE DATA> Average throughput. Unit: bps.
	<> = <NR1 NUMERIC RESPONSE DATA> Ideal throughput. Unit: bps.
	<> = <NR1 NUMERIC RESPONSE DATA> Actual transfer time. Unit: seconds.
	<> = <NR1 NUMERIC RESPONSE DATA> Ideal transfer time. Unit: seconds.
	<> = <NR1 NUMERIC RESPONSE DATA> Transfer time ratio.
	<> = <NR1 NUMERIC RESPONSE DATA> Window size. Unit: bytes.
	<> = <NR1 NUMERIC RESPONSE DATA> Number of services/connections.
	<> = <NR1 NUMERIC RESPONSE DATA> Transmitted bytes. Unit: bytes.
	<> = <NR1 NUMERIC RESPONSE DATA> Retransmitted bytes. Unit: bytes.
	<> = <NR1 NUMERIC RESPONSE DATA> Retransmitted percentage.
	<> = <NR1 NUMERIC RESPONSE DATA> TCP Efficiency. Unit: percentage.
	<> = <NR1 NUMERIC RESPONSE DATA> Baseline RTT. Unit: ms.
	<> = <NR1 NUMERIC RESPONSE DATA> Minimum RTT. Unit: ms.
	<> = <NR1 NUMERIC RESPONSE DATA> Average RTT. Unit: ms.
	<> = <NR1 NUMERIC RESPONSE DATA> Maximum RTT. Unit: ms.
	<> = <NR1 NUMERIC RESPONSE DATA> Buffer delay. Unit: percentage.

Example	`ETH:TTT:RES:THR? RTL` →
	`4197792000,949284785,10.00,44.22,0.23,1460,1,`
	`5247240000,0,0.00,100.00,0.002,0.002,0.002,0.002,0.00`

Note	

12.33.63 ETHernet:TTTest:RESult:THRoughput:EXPort

Syntax	ETHernet:TTTest:RESult:THRoughput:EXPort <direction> <file> [<delimiter>]
Description	This command writes the data of the throughput graph to the specified file in a CSV format. The file can be written to the internal disk or to a connected USB memory device. The specified delimiter is used as field delimiter. Lines are terminated with CR-LF (0x0d,0x10).
Parameters	<direction> = <CHARACTER PROGRAM DATA> LTRemote: Results for the local to remote direction. RTLocal: Results for the remote to local direction. LTRBoth: Results for the local to remote direction in the simultaneous test step. RTLBoth: Results for the remote to local direction in the simultaneous test step.
	<file> = <STRING PROGRAM DATA> The path and name of the file to store the data.
	<delimiter> = <CHARACTER PROGRAM DATA> COMMa: Use ',' as field separator. SEMicolon: Use ';' as field separator.

continued on next page. . .

. . . continued from previous page

	TABulator: Use a tabulator character (0x09) as field separator. *DEFault = COMMa*
Response	None.
Example	`ETH:TTT:RES:THR:EXP RTL,"Internal/graph-data.csv",TAB`
Note	Files must be saved to the `Internal/` directory or a sub-directory hereof.
	When a USB storage device is mounted, files are stored via the Usb/ directory.

12.33.64 ETHernet:TTTest:RESult:MSERvice?

Syntax	ETHernet:TTTest:RESult:MSERvice? <direction>
Description	Returns the results of all multi service services in the specified direction.
Parameter	<direction> = <CHARACTER PROGRAM DATA> LTRemote: Results for the local to remote direction. RTLocal: Results for the remote to local direction. LTRBoth: Results for the local to remote direction in the simultaneous test step. RTLBoth: Results for the remote to local direction in the simultaneous test step. *DEFault = LTRemote*
Response	A list of results for all services.
	<min-RTT> = <NR2 NUMERIC RESPONSE DATA> Minimum RTT. Unit: ms.
	<avg-RTT> = <NR2 NUMERIC RESPONSE DATA> Average RTT. Unit: ms.
	<max-RTT> = <NR2 NUMERIC RESPONSE DATA> Maximum RTT. Unit: ms.
	<avg-throughput> = <NR1 NUMERIC RESPONSE DATA> Average throughput rate. Unit: bps.
	<tcp-eff> = <NR2 NUMERIC RESPONSE DATA> TCP efficiency. Unit: percentage.
Example	`ETH:TTT:RES:MSER? RTL` → `(0.010,0.020,0,050,10100,50.00),(0.010,0.020,0,040,20200,100.00)`
Note	

12.33.65 ETHernet:TTTest:RESult:MSERvice:TOTal?

Syntax	ETHernet:TTTest:RESult:MSERvice:TOTal? <direction>
Description	Returns the result totals of all multi service services in the specified direction.
Parameter	<direction> = <CHARACTER PROGRAM DATA> LTRemote: Results for the local to remote direction. RTLocal: Results for the remote to local direction. LTRBoth: Results for the local to remote direction in the simultaneous test step. RTLBoth: Results for the remote to local direction in the simultaneous test step. *DEFault = LTRemote*
Response	A list results for all services.
	<min-RTT> = <NR2 NUMERIC RESPONSE DATA> Minimum RTT. Unit: ms.
	<avg-RTT> = <NR2 NUMERIC RESPONSE DATA> Average RTT. Unit: ms.
	<max-RTT> = <NR2 NUMERIC RESPONSE DATA> Maximum RTT. Unit: ms.
	<avg-throughput> = <NR1 NUMERIC RESPONSE DATA> Average throughput rate. Unit: bps.
	<tcp-eff> = <NR2 NUMERIC RESPONSE DATA> TCP efficiency. Unit: percentage.
Example	`ETH:TTT:RES:MSER:TOT? RTL` → `(0.010,0.020,0,050,1.100,50.00),(0.010,0.020,0,040,2.200,100.00)`
Note	

12.33.66 ETHernet:TTTest:RESult:MSERvice:EXPort

Syntax	ETHernet:TTTest:RESult:MSERvice:EXPort <direction> <file> [<delimiter>]
Description	This command writes the data of the multi service throughput graph to the specified file in a CSV format. The file can be written to the internal disk or to a connected USB memory device. The specified delimiter is used as field delimiter. Lines are terminated with CR-LF (0x0d,0x10).
Parameters	<direction> = <CHARACTER PROGRAM DATA> LTRemote: Results for the local to remote direction. RTLocal: Results for the remote to local direction. LTRBoth: Results for the local to remote direction in the simultaneous test step. RTLBoth: Results for the remote to local direction in the simultaneous test step.
	<file> = <STRING PROGRAM DATA> The path and name of the file to store the data.
	<delimiter> = <CHARACTER PROGRAM DATA> COMMa: Use ',' as field separator. SEMicolon: Use ';' as field separator. TABulator: Use a tabulator character (0x09) as field separator. *DEFault = COMMa*
Response	None.
Example	`ETH:TTT:RES:MSER:EXP RTL,"Internal/graph-data.csv",TAB`
Note	Files must be saved to the `Internal/` directory or a sub-directory hereof.
	When a USB storage device is mounted, files are stored via the Usb/ directory.

Chapter 13

eCPRI/RoE BERT

Almost all eCPRI/RoE BERT SCPI commands are same as Ethernet commands. This chapter only describes the differences from Ethernet. For the other commands, please refer to Ethernet.(see Chapter 12).

13.1 Port Setup

For detail of the SCPI command, please refer to Ethernet.(see section 12.1).

13.1.1 ETHernet:PORT<Pt>:DPIReference

Syntax	ETHernet:PORT<Pt>:DPIConstraint <interface>
Description	This command sets the dual port interface constraint.
Parameters	<Pt> = Port number
	<interface> = <CHARACTER PROGRAM DATA> 10G10G : Port1=10M-10G : Port2=10M-10G 10G25G : Port1=10M-10G : Port2=25G 25G10G : Port1=25G : Port2=10M-10G 25G25G : Port1=25G : Port2=25G *DEFault = 25G25G*
Response	None.
Example	`ETH:PORT1:DPIC 10G10G`
Note	This command can be used on eCPRI/RoE application with "25G eCPRI/RoE Dual Ability".

Syntax	ETHernet:PORT<Pt>:DPIConstraint?
Description	This query returns the dual port interface constraint.
Parameter	<Pt> = Port number
Response	<interface> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:DPIC?` → `10G10G`
Note	

13.2 PCS

For detail of the SCPI command, please refer to Ethernet.(see section 12.4).

13.3 Traffic

RoE BERT application uses stream number one only. Setting other streams has no effect in these applications. For detail of the SCPI command, please refer to Ethernet.(see section 12.5).

13.4 Settings

For detail of the SCPI command, please refer to Ethernet.(see section 12.7).

13.5 Thresholds

For detail of the SCPI command, please refer to Ethernet.(see section 12.9).

13.6 SyncE

For detail of the SCPI command, please refer to Ethernet.(see section 12.10).

13.7 Precision Time Protocol - IEEE 1588v2

For detail of the SCPI command, please refer to Ethernet.(see section 12.11).

13.8 Follow

For detail of the SCPI command, please refer to Ethernet.(see section 12.12).

13.9 Stimuli

For detail of the SCPI command, please refer to Ethernet.(see section 12.13).

13.10 Result

For detail of the SCPI command, please refer to Ethernet.(see section 12.14).

13.11 Status

For detail of the SCPI command, please refer to Ethernet.(see section 12.15).

13.12 Frame Content

This section only describes the differences from Ethernet. For the other commands, please refer to Ethernet.(see section 12.6).

13.12.1 ETHernet:PORT<Pt>:STReam<St>:RFRame

Syntax	ETHernet:PORT<Pt>:STReam<St>:RFRame <mode>
Description	This command sets the Radio frame type.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<mode> = <CHARACTER PROGRAM DATA> ECPRI: eCPRI IEEE1914: IEEE 1914.3 *DEFault = ECPRI*
Response	None.
Example	`ETH:PORT1:STR1:RFR ECPRI`
Notes	

Syntax	ETHernet:PORT<Pt>:STReam<St>:RFRame?
Description	This command sets the Radio frame type
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<interface> = <CHARACTER RESPONSE DATA>
Example	`ETH:PORT1:STR1:RFR?` → `ECPRI`
Note	

13.12.2 ETHernet:PORT<Pt>:STReam<St>:ECPRi:MESSage

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:MESSage <message>
Description	This command sets the Message type of eCPRI.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<message> = <CHARACTER PROGRAM DATA> IQD: IQ Data BITS: Bit Sequence RTCD: Real-Time Control Data GDT: Generic Data Transfer RMA: Remote Memory Access OWDM: One-way Delay Measurement RRST: Remote Reset EVIN: Event Indication UDEF: User Defined
Response	None.
Example	`ETH:PORT1:STR1:ECPR:MESS IQD`
Notes	

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:MESSage? <message>
Description	This query returns the Message type of eCPRI.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<message> = <CHARACTER PROGRAM DATA>
Example	`ETH:PORT1:STR1:ECPR:MESS?` → `IQD`
Note	

13.12.3 ETHernet:PORT<Pt>:STReam<St>:ECPRi:MVAL

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:MVAL <message>
Description	This command sets the Message type value of eCPRI.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<message> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:ECPR:MVAL 0`
Notes	This command valid in Message Type is User defined.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:MVAL? <message>
Description	This query returns the Message type value of eCPRI.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<message> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:MVAL? → 0`
Note	

13.12.4 ETHernet:PORT<Pt>:STReam<St>:ECPRi:LENGth

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:LENGth? <length>
Description	This query returns the Payload length of eCPRI.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:LENG? → 64`
Note	

13.12.5 ETHernet:PORT<Pt>:STReam<St>:ECPRi:IQData:PCID

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:IQData:PCID <data>
Description	This command sets the PC_ID of IQ Data.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:IQD:PCID "0000"`
Notes	2Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:IQData:PCID? <data>
Description	This query returns the PC_ID of IQ Data.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:IQD:PCID? → 0000`
Note	2Byte hex data.

13.12.6 ETHernet:PORT<Pt>:STReam<St>:ECPRi:IQData:SEQId

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:IQData:SEQId <data>
Description	This command sets the SEQ_ID of IQ Data.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:IQD:SEQI "0000"`
Notes	2Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:IQData:SEQId? <data>
Description	This query returns the SEQ_ID of IQ Data.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:IQD:SEQI?` → `0000`
Note	2Byte hex data.

13.12.7 ETHernet:PORT<Pt>:STReam<St>:ECPRi:BITSeq:PCID

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:BITSeq:PCID <data>
Description	This command sets the PC_ID of Bit Sequence.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:BITS:PCID "0000"`
Notes	2Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:BITSeq:PCID? <data>
Description	This query returns the PC_ID of Bit Sequence.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:BITS:PCID?` → `0000`
Note	2Byte hex data.

13.12.8 ETHernet:PORT<Pt>:STReam<St>:ECPRi:BITSeq:SEQId

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:BITSeq:SEQId <data>
Description	This command sets the SEQ_ID of Bit Sequence.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:BITS:SEQI "0000"`
Notes	2Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:BITSeq:SEQId? <data>
Description	This query returns the SEQ_ID of Bit Sequence.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:BITS:SEQI?` → `0000`
Note	2Byte hex data.

13.12.9 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RTCData:PCID

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RTCData:PCID <data>
Description	This command sets the RTC_ID of Real-Time Control Data.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:RTCD:PCID "0000"`
Notes	2Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RTCData:PCID? <data>
Description	This query returns the RTC_ID of Real-Time Control Data.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:RTCD:PCID?` → `0000`
Note	2Byte hex data.

13.12.10 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RTCData:SEQId

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RTCData:SEQId <data>
Description	This command sets the SEQ_ID of Real-Time Control Data.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:RTCD:SEQI "0000"`
Notes	2Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RTCData:SEQId? <data>
Description	This query returns the SEQ_ID of Real-Time Control Data.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:RTCD:SEQI?` → `0000`
Note	2Byte hex data.

13.12.11 ETHernet:PORT<Pt>:STReam<St>:ECPRi:GDTRans:PCID

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:GDTRans:PCID <data>
Description	This command sets the PC_ID of Generic Data Transfer.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:GDTR:PCID "00000000"`
Notes	4Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:GDTRans:PCID? <data>
Description	This query returns the PC_ID of Generic Data Transfer.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:GDTR:PCID?` → `00000000`
Note	4Byte hex data.

13.12.12 ETHernet:PORT<Pt>:STReam<St>:ECPRi:GDTRans:SEQId

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:GDTRans:SEQId <data>
Description	This command sets the SEQ_ID of Generic Data Transfer.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:GDTR:SEQI "00000000"`
Notes	4Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:GDTRans:SEQId? <data>
Description	This query returns the SEQ_ID of Generic Data Transfer.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:GDTR:SEQI?` → `00000000`
Note	4Byte hex data.

13.12.13 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:ACID

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:ACID <data>
Description	This command sets the Remote Memory Access ID of Remote Memory Access
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:RMAC:ACID "00"`
Notes	1Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:ACID? <data>
Description	This query returns the Remote Memory Access ID of Remote Memory Access
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:RMAC:ACID?` → `00`
Note	1Byte hex data.

13.12.14 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:RW

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:RW <data>
Description	This command sets the Read/Write of Remote Memory Access
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 15, DEFault = 0* 0: Read 1: Write 2: Write No Resp Others: Reserved
Response	None.
Example	`ETH:PORT1:STR1:ECPR:RMAC:RW 0`
Notes	

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:RW? <data>
Description	This query returns the Read/Write of Remote Memory Access
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <BINARY NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:RMAC:RW?` → `#B0`
Note	

13.12.15 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:RRESp

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:RRESp <data>
Description	This command sets the Req/Resp of Remote Memory Access
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 15, DEFault = 0* 0: Request 1: Response 2: Failure Others: Reserved
Response	None.
Example	`ETH:PORT1:STR1:ECPR:RMAC:RRES 0`
Notes	

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:RRESp? <data>
Description	This query returns the Req/Resp of Remote Memory Access
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <BINARY NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:RMAC:RRES?` → `#B0`
Note	

13.12.16 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:ELID

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:ELID <data>
Description	This command sets the Element ID of Remote Memory Access
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:RMAC:ELID "0000"`
Notes	2Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:ELID? <data>
Description	This query returns the Element ID of Remote Memory Access
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:RMAC:ELID?` → `0000`
Note	2Byte hex data.

13.12.17 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:ADDRess

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:ADDRess <address>
Description	This command sets the Address of Remote Memory Access
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<address> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:RMAC:ADDR "000000000000"`
Notes	6Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:ADDRess? <address>
Description	This query returns the Address of Remote Memory Access
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<address> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:RMAC:ADDR?` → `000000000000`
Note	6Byte hex data.

13.12.18 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:LENGth

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:LENGth <length>
Description	This command sets the Length of Remote Memory Access
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 65535, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:ECPR:RMAC:LENG 64`
Notes	

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RMACess:LENGth? <length>
Description	This query returns the Length of Remote Memory Access
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:RMAC:LENG?` → `64`
Note	

13.12.19 ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:MESId

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:MESId <data>
Description	This command sets the Measurement ID of One-Way Delay Measurement
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:OWDM:MESI "00"`
Notes	

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:MESId? <data>
Description	This query returns the Measurement ID of One-Way Delay Measurement
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:OWDM:MESI?` → `00`
Note	

13.12.20 ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:ACTType

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:ACTType <data>
Description	This command sets the Action Type of One-Way Delay Measurement
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255, DEFault = 0* 0: Request 1: Request with Follow_Up 2: Response 3: Remote Request 4: Remote Request with Follow_Up 5: Follow_Up Others: Reserved
Response	None.
Example	`ETH:PORT1:STR1:ECPR:OWDM:ACTT 0`
Notes	

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:ACTType? <data>
Description	This query returns the Action Type of One-Way Delay Measurement
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:OWDM:ACTT?` → `#H0`
Note	

13.12.21 ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:TSTamp

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:TSTamp <data>
Description	This command sets the TimeStamp of One-Way Delay Measurement
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:OWDM:TST "00000000000000000000"`
Notes	10Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:TSTamp? <data>
Description	This query returns the TimeStamp of One-Way Delay Measurement
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:OWDM:TST?` → `00000000000000000000`
Note	10Byte hex data.

13.12.22 ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:CVALue

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:CVALue <data>
Description	This command sets the Compensation Value of One-Way Delay Measurement
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:OWDM:CVAL "0000000000000000"`
Notes	8Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:OWDMeas:CVALue? <data>
Description	This query returns the Compensation Value of One-Way Delay Measurement
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:OWDM:CVAL?` → `0000000000000000`
Note	8Byte hex data.

13.12.23 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RRESet:REID

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RRESet:REID <data>
Description	This command sets the Reset ID of Remote Reset
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:RRES:REID "0000"`
Notes	2Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RRESet:REID? <data>
Description	This query returns the Reset ID of Remote Reset
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:RRES:REID?` → `0000`
Note	2Byte hex data.

13.12.24 ETHernet:PORT<Pt>:STReam<St>:ECPRi:RRESet:COPtion

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RRESet:COPtion <data>
Description	This command sets the Reset Code Op of Remote Reset
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255, DEFault = 0* 1: Remote reset request 2: Remote reset response Others: Reserved
Response	None.
Example	`ETH:PORT1:STR1:ECPR:RRES:COP 0`
Notes	

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:RRESet:COPtion? <data>
Description	This query returns the Reset Code Op of Remote Reset
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:RRES:COP?` → `#H0`
Note	

13.12.25 ETHernet:PORT<Pt>:STReam<St>:ECPRi:CDATa

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:CDATa <data>
Description	This command sets the Custom pattern data.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:CDAT "00001122"`
Notes	

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:CDATa? <data>
Description	This query returns the Custom pattern data.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:CDAT? → 00001122`
Note	

13.12.26 ETHernet:PORT<Pt>:STReam<St>:ECPRi:CLENgth

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:CLENgth <length>
Description	This command sets the Custom pattern length.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 256, DEFault = 1*
Response	None.
Example	`ETH:PORT1:STR1:ECPR:CLEN 4`
Notes	

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:CLENgth? <length>
Description	This query returns the Custom pattern length.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:ECPR:CLEN? → 4`
Note	

13.12.27 ETHernet:PORT<Pt>:STReam<St>:ECPRi:CENable

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:CENable <enable>
Description	This command sets the Enable of Custom pattern.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<enable> = <BOOLEAN PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ECPR:CEN ON`
Notes	

Syntax	ETHernet:PORT<Pt>:STReam<St>:ECPRi:CENable? <enable>
Description	This query returns the Enable of Custom pattern.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`ETH:PORT1:STR1:ECPR:CEN? → 1`
Note	

13.12.28 ETHernet:PORT<Pt>:STReam<St>:ROE:SUBType

Syntax	ETHernet:PORT<Pt>:STReam<St>:ROE:SUBType <subtype>
Description	This command sets the Sub type of IEEE 1914.3.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<subtype> = <CHARACTER PROGRAM DATA> CONT: Control Sub type SAD: Structure-agnostic data sub type SACD: Structure-aware CPRI data sub type SCMC: Slow C&M CORI data sub type NTDD: Native time domain data sub type NFDD: Native frequency domain data sub type NPD: Native PRACH data sub type UDEF: User Defined
Response	None.
Example	`ETH:PORT1:STR1:ROE:SUBT CONT`
Notes	

Syntax	ETHernet:PORT<Pt>:STReam<St>:ROE:SUBType? <subtype>
Description	This query returns the Sub type of IEEE 1914.3.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<subtype> = <CHARACTER PROGRAM DATA>
Example	`ETH:PORT1:STR1:ROE:SUBT?` → `CONT`
Note	

13.12.29 ETHernet:PORT<Pt>:STReam<St>:ROE:SVALue

Syntax	ETHernet:PORT<Pt>:STReam<St>:ROE:SVALue <subtype>
Description	This command sets the Sub type value of IEEE 1914.4.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<subtype> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255, DEFault = 0*
Response	None.
Example	`ETH:PORT1:STR1:ROE:SVAL 0`
Notes	This command valid in Message Type is User defined.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ROE:SVALue? <subtype>
Description	This query returns the Sub type value of IEEE 1914.4.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<subtype> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:ROE:SVAL?` → `0`
Note	

13.12.30 ETHernet:PORT<Pt>:STReam<St>:ROE:FLID

Syntax	ETHernet:PORT<Pt>:STReam<St>:ROE:FLID <data>
Description	This command sets the Flow type of IEEE 1914.3.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ROE:FLID "00"`
Notes	1Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ROE:FLID? <data>
Description	This query returns the Flow type of IEEE 1914.3.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ROE:FLID?` → `00`
Note	1Byte hex data.

13.12.31 ETHernet:PORT<Pt>:STReam<St>:ROE:LENGth

Syntax	ETHernet:PORT<Pt>:STReam<St>:ROE:LENGth? <length>
Description	This query returns the Payload length of IEEE 1914.3.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:ROE:LENG?` → `64`
Note	

13.12.32 ETHernet:PORT<Pt>:STReam<St>:ROE:OINFo

Syntax	ETHernet:PORT<Pt>:STReam<St>:ROE:OINFo <data>
Description	This command sets the Ordering information of IEEE 1914.3.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ROE:OINF "00000000"`
Notes	4Byte hex data.

Syntax	ETHernet:PORT<Pt>:STReam<St>:ROE:OINFo? <data>
Description	This query returns the Ordering information of IEEE 1914.3.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ROE:OINF?` → `00000000`
Note	4Byte hex data.

13.12.33 ETHernet:PORT<Pt>:STReam<St>:ROE:CONTrol:OPCode

Syntax	ETHernet:PORT<Pt>:STReam<St>:ROE:CONTrol:OPCode <data>
Description	This command sets the OpCode of Control sub type.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 255, DEFault = 0* #H00: RoE OAM TLV write #H01: Ctrl_AxC words #H02: VSD #H03: Timing control packet #H80: RoE OAM TLV read Others: Reserved
Response	None.
Example	`ETH:PORT1:STR1:ROE:CONT:OPC #H0`
Notes	

Syntax	ETHernet:PORT<Pt>:STRream<St>:ROE:CONTrol:OPCode? <data>
Description	This query returns the OpCode of Control sub type.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:ROE:CONT:OPC?` → `#H0`
Note	

13.12.34 ETHernet:PORT<Pt>:STRream<St>:ROE:CDATa

Syntax	ETHernet:PORT<Pt>:STRream<St>:ROE:CDATa <data>
Description	This command sets the Custom pattern data of IEEE 1914.3.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<data> = <STRING PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ROE:CDAT "00001122"`
Notes	

Syntax	ETHernet:PORT<Pt>:STRream<St>:ROE:CDATa? <data>
Description	This query returns the Custom pattern data of IEEE 1914.3.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<data> = <STRING RESPONSE DATA>
Example	`ETH:PORT1:STR1:ROE:CDAT?` → `00001122`
Note	

13.12.35 ETHernet:PORT<Pt>:STRream<St>:ROE:CLENgth

Syntax	ETHernet:PORT<Pt>:STRream<St>:ROE:CLENgth <length>
Description	This command sets the Custom pattern length of IEEE 1914.3.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 256, DEFault = 1*
Response	None.
Example	`ETH:PORT1:STR1:ROE:CLEN 4`
Notes	

Syntax	ETHernet:PORT<Pt>:STRream<St>:ROE:CLENgth? <length>
Description	This query returns the Custom pattern length of IEEE 1914.3.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`ETH:PORT1:STR1:ROE:CLEN?` → `4`
Note	

13.12.36 ETHernet:PORT<Pt>:STRream<St>:ROE:CENable

Syntax	ETHernet:PORT<Pt>:STRream<St>:ROE:CENable <enable>
Description	This command sets the Enable of Custom pattern of IEEE 1914.3.
Parameters	<Pt> = Port number
	<St> = Stream number (1)
	<enable> = <BOOLEAN PROGRAM DATA>
Response	None.
Example	`ETH:PORT1:STR1:ROE:CEN ON`
Notes	

Syntax	ETHernet:PORT<Pt>:STReam<St>:ROE:CENable? <enable>
Description	This query returns the Enable of Custom pattern of IEEE 1914.3.
Parameter	<Pt> = Port number
	<St> = Stream number (1)
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`ETH:PORT1:STR1:ROE:CEN?` → `1`
Note	

Chapter 14

Fibre Channel

14.1 Port

14.1.1 FCHannel:PORT<Pt>:MODE

Syntax	FCHannel:PORT<Pt>:MODE <mode>
Description	This command sets the port mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: Port off FC100: 1Gb/s 100MBytes/s FC200: 2Gb/s 200MBytes/s FC400: 4Gb/s 400MBytes/s FC800: 8Gb/s 800MBytes/s FC1200: 10Gb/s 1200MBytes/s FC1600: 16Gb/s 1600MBytes/s *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:MODE FC100`
Note	

Syntax	FCHannel:PORT<Pt>:MODE?
Description	This query returns the port mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`FCH:PORT1:MODE?` → `FC100`
Note	

14.1.2 FCHannel:TX<Pt>:TIMing

Syntax	FCHannel:TX<Pt>:TIMing <timing>
Description	This command sets the timing source.
Parameters	<Pt> = Port number
	<timing> = <CHARACTER PROGRAM DATA> INTernal: Internal clock EXTernal: External clock GPS: GPS signal RX: Received clock *DEFault = INT*
Response	None
Example	`FCH:TX1:TIM INT`
Note	

Syntax	FCHannel:TX<Pt>:TIMing?
Description	This query returns the timing source.
Parameter	<Pt> = Port number
Response	<timing> = <CHARACTER RESPONSE DATA>
Example	`FCH:TX1:TIM?` → `INT`
Note	

14.2 Interface

14.2.1 FCHannel:PORT<Pt>:TOPology

Syntax	FCHannel:PORT<Pt>:TOPology <topology>
Description	This command sets the topology
Parameters	<Pt> = Port number
	<topology> = <CHARACTER PROGRAM DATA> PTPoint: Point to Point FABRic: Fabric EPORt: E-Port *DEFault = PTPoint*
Response	None.
Example	`FCH:PORT1:TOP FABR`
Note	

Syntax	FCHannel:PORT<Pt>:TOPology?
Description	This query returns the topology
Parameter	<Pt> = Port number
Response	<topology> = <CHARACTER RESPONSE DATA>
Example	`FCH:PORT1:TOP?` → `FABR`
Note	

14.2.2 FCHannel:PORT<Pt>:PTP:LOGin

Syntax	FCHannel:PORT<Pt>:PTP:LOGin <enable>
Description	This command enables/disables point to point login
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:PTP:LOG OFF`
Note	

Syntax	FCHannel:PORT<Pt>:PTP:LOGin?
Description	This query returns whether or not point to point login is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:PTP:LOG?` → `0`
Note	

14.2.3 FCHannel:PORT<Pt>:FABRic:LOGin

Syntax	FCHannel:PORT<Pt>:FABRic:LOGin
Description	This command activates the fabric login.
Parameters	<Pt> = Port number
Response	None.
Example	`FCH:PORT1:FABR:LOG`
Note	

14.2.4 FCHannel:PORT<Pt>:EPORt:ELP

Syntax	FCHannel:PORT<Pt>:EPORt:ELP
Description	This command activates the ELP.
Parameters	<Pt> = Port number
Response	None.
Example	`FCH:PORT1:EPOR:ELP`
Note	

14.2.5 FCHannel:PORT<Pt>:FCONtrol[:ENABle]

Syntax	FCHannel:PORT<Pt>:FCONtrol[:ENABle] <enable>
Description	This command enables/disables flow control.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:FCON OFF`
Note	

Syntax	FCHannel:PORT<Pt>:FCONtrol[:ENABle]?
Description	This query returns whether or not flow control is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:FCON?` → `0`
Note	

14.2.6 FCHannel:PORT<Pt>:FCONtrol:LCRedit

Syntax	FCHannel:PORT<Pt>:FCONtrol:LCRedit <credit>
Description	This command sets the local credit.
Parameters	<Pt> = Port number
	<credit> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=65535, DEFault=64*
Response	None.
Example	`FCH:PORT1:FCON:LCR 64`
Note	

Syntax	FCHannel:PORT<Pt>:FCONtrol:LCRedit?
Description	This query returns the local credit.
Parameter	<Pt> = Port number
Response	<credit> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:FCON:LCR?` → `64`
Note	

14.2.7 FCHannel:PORT<Pt>:FCONtrol:RCRedit

Syntax	FCHannel:PORT<Pt>:FCONtrol:RCRedit <credit>
Description	This command sets the remote credit.
Parameters	<Pt> = Port number
	<credit> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=65535, DEFault=64*
Response	None.
Example	`FCH:PORT1:FCON:RCR 64`
Note	

Syntax	FCHannel:PORT<Pt>:FCONtrol:RCRedit?
Description	This query returns the remote credit.
Parameter	<Pt> = Port number
Response	<credit> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:FCON:RCR?` → `64`
Note	

14.2.8 FCHannel:PORT<Pt>:SOURce:WWN

Syntax	FCHannel:PORT<Pt>:SOURce:WWN <wwn>
Description	This command sets the source port WWN.
Parameters	<Pt> = Port number
	<wwn> = <STRING PROGRAM DATA> Port WWN
Response	None.
Example	`FCH:PORT1:SOUR:WWN "00-00-00-00-00-00-00-00"`
Note	Only the character '-' is accepted as separator.

Syntax	FCHannel:PORT<Pt>:SOURce:WWN?
Description	This query returns the source port WWN.
Parameter	<Pt> = Port number
Response	<wwn> = <STRING RESPONSE DATA>
Example	`FCH:PORT1:SOUR:WWN?` → `"00-00-00-00-00-00-00-00"`
Note	

14.2.9 FCHannel:PORT<Pt>:SOURce:WWN:DEFault

Syntax	FCHannel:PORT<Pt>:SOURce:WWN:DEFault <enable>
Description	This command enables/disables default port WWN.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`FCH:PORT1:SOUR:WWN:DEF ON`
Note	

Syntax	FCHannel:PORT<Pt>:SOURce:WWN:DEFault?
Description	This query returns whether or not default port WWN is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:SOUR:WWN:DEF?` → `1`
Note	

14.2.10 FCHannel:PORT<Pt>:SOURce:ID

Syntax	FCHannel:PORT<Pt>:SOURce:ID <id>
Description	This command sets the source ID.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=16777215, DEFault=0*
Response	None.
Example	`FCH:PORT1:SOUR:ID 0`
Note	

Syntax	FCHannel:PORT<Pt>:SOURce:ID?
Description	This query returns the source ID.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:SOUR:ID?` → `0`
Note	

14.2.11 FCHannel:PORT<Pt>:DESTination:WWN

Syntax	FCHannel:PORT<Pt>:DESTination:WWN <wwn>
Description	This command sets the destination port WWN.
Parameters	<Pt> = Port number
	<wwn> = <STRING PROGRAM DATA> Port WWN
Response	None.
Example	`FCH:PORT1:DEST:WWN "00-00-00-00-00-00-00-00"`
Note	Only the character '-' is accepted as separator.

Syntax	FCHannel:PORT<Pt>:DESTination:WWN?
Description	This query returns the destination port WWN.
Parameter	<Pt> = Port number
Response	<wwn> = <STRING RESPONSE DATA>
Example	`FCH:PORT1:DEST:WWN?` → `"00-00-00-00-00-00-00-00"`
Note	

14.2.12 FCHannel:PORT<Pt>:DESTination:ID

Syntax	FCHannel:PORT<Pt>:DESTination:ID <id>
Description	This command sets the destination ID.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=16777215, DEFault=0*
Response	None.
Example	`FCH:PORT1:DEST:ID 0`
Note	

Syntax	FCHannel:PORT<Pt>:DESTination:ID?
Description	This query returns the destination ID.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:DEST:ID?` → `0`
Note	

14.2.13 FCHannel:PORT<Pt>:DESTination:LOGin

Syntax	FCHannel:PORT<Pt>:DESTination:LOGin
Description	This command activates port login.
Parameters	<Pt> = Port number
Response	None.
Example	`FCH:PORT1:DEST:LOG`
Note	

14.3 Frame

14.3.1 FCHannel:PORT<Pt>:FRAMe:FRAMing

Syntax	FCHannel:PORT<Pt>:FRAMe:FRAMing <type>
Description	This command sets the frame structure.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> FT0: SOF:Data:EOF FT1: SOF:Header:Data:CRC:EOF *DEFault = FT1*
Response	None.
Example	`FCH:PORT1:FRAM:FRAM FT1`
Note	

Syntax	FCHannel:PORT<Pt>:FRAMe:FRAMing?
Description	This query returns the frame structure.
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`FCH:PORT1:FRAM:FRAM?` → `FT1`
Note	

14.3.2 FCHannel:PORT<Pt>:FRAMe:CONTent

Syntax	FCHannel:PORT<Pt>:FRAMe:CONTent <content>
Description	This command sets the data content.
Parameters	<Pt> = Port number
	<content> = <CHARACTER PROGRAM DATA> FOX 5555 PRBS9 PRBS11 PRBS15 PRBS20 PRBS23 PRBS29 PRBS31 BHFTest BCRPat BJTPat BSPat USER32BIT ZERO *DEFault = FOX*
Response	None.
Example	`FCH:PORT1:FRAM:CONT 5555`
Note	

Syntax	FCHannel:PORT<Pt>:FRAMe:CONTent?
Description	This query returns the data content.
Parameter	<Pt> = Port number
Response	<content> = <CHARACTER RESPONSE DATA>
Example	`FCH:PORT1:FRAM:CONT?` → `5555`
Note	

14.3.3 FCHannel:PORT<Pt>:FRAMe:UP32

Syntax	FCHannel:PORT<Pt>:FRAMe:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `CONTent` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters (one bit resolution).
Response	None.
Example	`FCH:PORT1:FRAM:UP32 "0110"`
Note	

Syntax	FCHannel:PORT<Pt>:FRAMe:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`FCH:PORT1:FRAM:UP32?` → `"0110"`
Note	

14.3.4 FCHannel:PORT<Pt>:FRAMe:HEADer:SID

Syntax	FCHannel:PORT<Pt>:FRAMe:HEADer:SID <id>
Description	This command sets the Seq ID in the frame header.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=0*
Response	None.
Example	`FCH:PORT1:FRAM:HEAD:SID #HAE` `FCH:PORT1:FRAM:HEAD:SID 174`
Note	

Syntax	FCHannel:PORT<Pt>:FRAMe:HEADer:SID?
Description	This query returns the Seq ID in the frame header.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:FRAM:HEAD:SID?` → `174`
Note	

14.3.5 FCHannel:PORT<Pt>:FRAMe:HEADer:OID

Syntax	FCHannel:PORT<Pt>:FRAMe:HEADer:OID <id>
Description	This command sets the OX ID in the frame header.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=65535, DEFault=0*
Response	None.
Example	`FCH:PORT1:FRAM:HEAD:OID #HAE` `FCH:PORT1:FRAM:HEAD:OID 174`
Note	

Syntax	FCHannel:PORT<Pt>:FRAMe:HEADer:OID?
Description	This query returns the OX ID in the frame header.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:FRAM:HEAD:OID?` → `174`
Note	

14.3.6 FCHannel:PORT<Pt>:FRAMe:HEADer:RID

Syntax	FCHannel:PORT<Pt>:FRAMe:HEADer:RID <id>
Description	This command sets the Rx ID in the frame header.
Parameters	<Pt> = Port number
	<id> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=65535, DEFault=0*
Response	None.
Example	`FCH:PORT1:FRAM:HEAD:RID #HAE` `FCH:PORT1:FRAM:HEAD:RID 174`
Note	

Syntax	FCHannel:PORT<Pt>:FRAMe:HEADer:RID?
Description	This query returns the Rx ID in the frame header.
Parameter	<Pt> = Port number
Response	<id> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:FRAM:HEAD:RID? → 174`
Note	

14.4 Generator

14.4.1 FCHannel:PORT<Pt>:TRAFfic:DMODe

Syntax	FCHannel:PORT<Pt>:TRAFfic:DMODe <mode>
Description	This command sets the duration mode for the traffic generator.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> CONTinuous: Continuous FRAMes: Frames SEConds: Seconds *DEFault = CONTinuous*
Response	None.
Example	`FCH:PORT1:TRAF:DMOD CONT`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:DMODe?
Description	This query returns the duration mode for the traffic generator.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`FCH:PORT1:TRAF:DMOD? → CONT`
Note	

14.4.2 FCHannel:PORT<Pt>:TRAFfic:DSEConds

Syntax	FCHannel:PORT<Pt>:TRAFfic:DSEConds <seconds>
Description	This command sets the duration for the traffic generator in seconds.
Parameters	<Pt> = Port number
	<seconds> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=2000000000, DEFault=1*
Response	None.
Example	`FCH:PORT1:TRAF:DSEC 1`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:DSEConds?
Description	This query returns the duration for the traffic generator in seconds.
Parameter	<Pt> = Port number
Response	<seconds> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:TRAF:DSEC?` → `1`
Note	

14.4.3 FCHannel:PORT<Pt>:TRAFfic:DFRames

Syntax	FCHannel:PORT<Pt>:TRAFfic:DFRames <frames>
Description	This command sets the duration for the traffic generator in number of frames.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=2000000000, DEFault=1*
Response	None.
Example	`FCH:PORT1:TRAF:DFR 1`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:DFRames?
Description	This query returns the duration for the traffic generator in number of frames.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:TRAF:DFR?` → `1`
Note	

14.4.4 FCHannel:PORT<Pt>:TRAFfic:ASTGenerator

Syntax	FCHannel:PORT<Pt>:TRAFfic:ASTGenerator <enable>
Description	This command enables/disables automatic start of the transmitter when running a BER measurement.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:TRAF:ASTG OFF`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:ASTGenerator?
Description	This query returns whether or not transmitter automatically starts when running a BER measurement.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:TRAF:ASTG?` → `0`
Note	

14.4.5 FCHannel:TRAFfic:GENerator:STARt

Syntax	FCHannel:TRAFfic:GENerator:STARt
Description	This command starts the traffic generator.
Parameters	<Pt> = Port number
Response	None.
Example	`FCH:TRAF:GEN:STAR`
Note	This command applies to all ports.

14.4.6 FCHannel:TRAFfic:GENerator:STOP

Syntax	FCHannel:TRAFfic:GENerator:STOP
Description	This command stops the traffic generator.
Parameters	<Pt> = Port number
Response	None.
Example	`FCH:TRAF:GEN:STOP`
Note	This command applies to all ports.

14.4.7 FCHannel:PORT<Pt>:TRAFfic:GENerator:STARt

Syntax	FCHannel:PORT<Pt>:TRAFfic:GENerator:STARt
Description	This command starts the traffic generator.
Parameters	<Pt> = Port number
Response	None.
Example	`FCH:PORT1:TRAF:GEN:STAR`
Note	This command requires that the Port is enabled (`FCHannel:PORT<Pt>:MODE`).

14.4.8 FCHannel:PORT<Pt>:TRAFfic:GENerator:STOP

Syntax	FCHannel:PORT<Pt>:TRAFfic:GENerator:STOP
Description	This command stops the traffic generator.
Parameters	<Pt> = Port number
Response	None.
Example	`FCH:PORT1:TRAF:GEN:STOP`
Note	

14.4.9 FCHannel:PORT<Pt>:TRAFfic:GENerator:STATus?

Syntax	FCHannel:PORT<Pt>:TRAFfic:GENerator:STATus?
Description	This query returns the current status of the traffic generator.
Parameter	<Pt> = Port number
Response	<stat> = <BOOLEAN RESPONSE DATA> 0: Traffic generator inactive 1: Traffic generator active
Example	`FCH:PORT1:TRAF:GEN:STAT?` → `0`
Note	

14.5 Stream

14.5.1 FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:PROFile

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:PROFile <profile>
Description	This command sets the stream Line Load Profile.
Parameters	<Pt> = Port number
	<profile> = <CHARACTER PROGRAM DATA> CONStant: Constant line load profile RAMP: Ramp line load profile *DEFault = CONStant*
Response	None.
Example	`FCH:PORT1:TRAF:STR:LL:PROF RAMP`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:PROFile?
Description	This query returns the stream Line Load Profile.
Parameter	<Pt> = Port number
Response	<profile> = <CHARACTER RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:LL:PROF?` → `RAMP`
Note	

14.5.2 FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad[:CONStant]

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad[:CONStant] <load>
Description	This command sets the stream Line Load in constant mode. Unit: Percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum=0.0000, MAXimum=100.0000, DEFault=100.0000*
Response	None.
Example	`FCH:PORT1:TRAF:STR:LL 50`
Note	The minimum possible line load is 0.0008.

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad[:CONStant]?
Description	This query returns the stream Line Load in constant mode. Unit: Percentage.
Parameter	<Pt> = Port number
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:LL?` → `50`
Note	

14.5.3 FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP[:MODE]

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP[:MODE] <mode>
Description	This command sets the ramp mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> KEEPend: Maintain line load level at ramp end REPeat: Repeat ramp INVert: Invert ramp *DEFault = INVert*
Response	None.
Example	`FCH:PORT1:TRAF:STR:LL:RAMP KEEP`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP[:MODE]?
Description	This query returns the ramp mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:LL:RAMP?` → `KEEP`
Note	

14.5.4 FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:STARt

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:STARt <load>
Description	This command sets the stream initial line load in ramp mode. Unit: Percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum=0.0000, MAXimum=100.0000, DEFault=0.0000*
Response	None.
Example	`FCH:PORT1:TRAF:STR:LL:RAMP:STAR 20`
Note	The minimum possible line load is 0.0008.

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:STARt?
Description	This query returns the stream initial line load in ramp mode. Unit: Percentage.
Parameter	<Pt> = Port number
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:LL:RAMP:STAR?` → `20`
Note	

14.5.5 FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:END

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:END <load>
Description	This command sets the stream end line load in ramp mode. Unit: Percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum=0.0000, MAXimum=100.0000, DEFault=0.0000*
Response	None.
Example	`FCH:PORT1:TRAF:STR:LL:RAMP:END 50`
Note	The minimum possible line load is 0.0008.

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:END?
Description	This query returns the stream end line load in ramp mode. Unit: Percentage.
Parameter	<Pt> = Port number
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:LL:RAMP:END?` → `50`
Note	

14.5.6 FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:STEP

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:STEP <load>
Description	This command sets the stream line load step size, in ramp mode. Unit: Percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum=0.0000, MAXimum=100.0000, DEFault=1.0000*
Response	None.
Example	`FCH:PORT1:TRAF:STR:LL:RAMP:STEP 2`
Note	The minimum possible line load is 0.0008.

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:STEP?
Description	This query returns the stream line load step size, in ramp mode. Unit: Percentage.
Parameter	<Pt> = Port number
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:LL:RAMP:STEP?` → `2`
Note	

14.5.7 FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:DURation

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:DURation <dur>
Description	This command sets the stream step line load duration in ramp mode. Unit: Seconds.
Parameters	<Pt> = Port number
	<dur> = <NUMERIC PROGRAM DATA> *MINimum=3, MAXimum=3600, DEFault=10*
Response	None.
Example	`FCH:PORT1:TRAF:STR:LL:RAMP:DUR 5`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LLoad:RAMP:DURation?
Description	This query returns the stream step line load duration in ramp mode. Unit: Seconds.
Parameter	<Pt> = Port number
Response	<dur> = <NR2 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:LL:RAMP:DUR?` → `5`
Note	

14.5.8 FCHannel:PORT<Pt>:TRAFfic:STReam:FSIZe

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:FSIZe <dur>
Description	This command sets the frame size. Unit: Bytes
Parameters	<Pt> = Port number
	<dur> = <NUMERIC PROGRAM DATA> *MINimum=8* [1], *MAXimum=3240, DEFault=8*
Response	None.
Example	`FCH:PORT1:TRAF:STR:FSIZ 8`
Note	The frame size must be a multiple of 4. [1] The minimum allowed frame size varies depending on the port mode. FC400: *MINimum = 64* FC800: *MINimum = 256* FC1200: *MINimum = 256*

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:FSIZe?
Description	This query returns the frame size. Unit: Bytes
Parameter	<Pt> = Port number
Response	<dur> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:FSIZ?` → `8`
Note	

14.5.9 FCHannel:PORT<Pt>:TRAFfic:STReam:LATency[:ENABle]

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LATency[:ENABle] <enable>
Description	This command enables/disables stream latency measurement.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:TRAF:STR:LAT OFF`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LATency[:ENABle]?
Description	This query returns whether or not stream latency measurement is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:LAT?` → `0`
Note	

14.5.10 FCHannel:PORT<Pt>:TRAFfic:STReam:LATency:THResholds[:ENABle]

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LATency:THResholds[:ENABle] <enable>
Description	This command enables/disables stream latency threshold.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:TRAF:STR:LAT:THR OFF`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LATency:THResholds[:ENABle]?
Description	This query returns whether or not stream latency threshold is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:LAT:THR?` → `0`
Note	

14.5.11 FCHannel:PORT<Pt>:TRAFfic:STReam:LATency:THResholds:VALue

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LATency:THResholds:VALue <value>
Description	This command sets the Latency thresholds value. Unit: Microseconds.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=429496729.5, DEFault=0*
Response	None.
Example	`FCH:PORT1:TRAF:STR:LAT:THR:VAL 10.5`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:LATency:THResholds:VALue?
Description	This query returns the Latency thresholds value. Unit: Microseconds.
Parameter	<Pt> = Port number
Response	<value> = <NR2 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:LAT:THR:VAL?` → `10.5`
Note	

14.5.12 FCHannel:PORT<Pt>:TRAFfic:STReam:JITTer[:ENABle]

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:JITTer[:ENABle] <enable>
Description	This command enables/disables stream jitter measurement.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:TRAF:STR:JITT OFF`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:JITTer[:ENABle]?
Description	This query returns whether or not stream jitter measurement is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:JITT?` → `0`
Note	

14.5.13 FCHannel:PORT<Pt>:TRAFfic:STReam:JITTer:THResholds[:ENABle]

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:JITTer:THResholds[:ENABle] <enable>
Description	This command enables/disables stream jitter threshold.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:TRAF:STR:JITT:THR OFF`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:JITTer:THResholds[:ENABle]?
Description	This query returns whether or not stream jitter threshold is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:JITT:THR?` → `0`
Note	

14.5.14 FCHannel:PORT<Pt>:TRAFfic:STReam:JITTer:THResholds:VALue

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:JITTer:THResholds:VALue <value>
Description	This command sets the stream jitter threshold value. Unit: Microseconds.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=429496792.5, DEFault=0*
Response	None.
Example	`FCH:PORT1:TRAF:STR:JITT:THR:VAL 10.5`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:JITTer:THResholds:VALue?
Description	This query returns the stream jitter threshold value. Unit: Microseconds.
Parameter	<Pt> = Port number
Response	<value> = <NR2 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:JITT:THR:VAL?` → `10.5`
Note	

14.5.15 FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure[:ENABle]

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure[:ENABle] <enable>
Description	This command enables/disables Service Disruption Measurement.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:TRAF:STR:BER:SDM OFF`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure[:ENABle]?
Description	This query returns whether or not Service Disruption Measurement is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:BER:SDM?` → `0`
Note	

14.5.16 FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure:MDISruption

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure:MDISruption <value>
Description	This command sets the Min Disruption Frames.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 5000, DEFault = 10*
Response	None.
Example	`FCH:PORT1:TRAF:STR:BER:SDM:MDIS 5`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure:MDISruption?
Description	This query return the Min Disruption Frames.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:BER:SDM:MDIS?` → `5`
Note	

14.5.17 FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure:THResholds [:ENABle]

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure:THResholds[:ENABle] <enable>
Description	This command enables/disables Service Disruption threshold.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:TRAF:STR:BER:SDM:THR OFF`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure:THResholds[:ENABle]?
Description	This query returns whether or not Service Disruption threshold is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:BER:SDM:THR?` → `0`
Note	

14.5.18 FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure:THResholds :VALue

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure:THResholds:VALue <value>
Description	This command sets the Service Disruption threshold value. Unit: Microseconds.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 5000000, DEFault = 50000*
Response	None.
Example	`FCH:PORT1:TRAF:STR:BER:SDM:THR:VAL 10`
Note	

Syntax	FCHannel:PORT<Pt>:TRAFfic:STReam:BER:SDMeasure:THResholds:VALue?
Description	This query returns the Service Disruption threshold value. Unit: Microseconds.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:TRAF:STR:BER:SDM:THR:VAL?` → `10`
Note	This only applies to Ethernet and Fibre Channel.

14.6 Settings

14.6.1 FCHannel:PORT<Pt>:SETTings:BER:OBAMeasuring

Syntax	FCHannel:PORT<Pt>:SETTings:BER:OBAMeasuring <enable>
Description	This command enables/disables only show BER alarms when measuring.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:SETT:BER:OBAM ON`
Note	This setting applies to all ports.

Syntax	FCHannel:PORT<Pt>:SETTings:BER:OBAMeasuring?
Description	This query returns if BER alarms should only be shown when measuring.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:SETT:BER:OBAM?` → `1`
Note	This setting applies to all ports.

14.6.2 FCHannel:PORT<Pt>:SETTings:BER:IAFFilter

Syntax	FCHannel:PORT<Pt>:SETTings:BER:IAFFilter <enable>
Description	This command enables/disables include addresses in frame filter on receiver.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`FCH:PORT1:SETT:BER:IAFF OFF`
Note	This setting applies to all ports.

Syntax	FCHannel:PORT<Pt>:SETTings:BER:IAFFilter?
Description	This query returns if include addresses in frame filter on receiver is enabled/disabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:SETT:BER:IAFF?` → `0`
Note	This setting applies to all ports.

14.6.3 FCHannel:PORT<Pt>:SETTings:BER:CLFrames

Syntax	FCHannel:PORT<Pt>:SETTings:BER:CLFrames <enable>
Description	This command enables/disables count lost frames as pattern errors.
Parameter	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:SETT:BER:CLF ON`
Note	This setting applies to all ports.

Syntax	FCHannel:PORT<Pt>:SETTings:BER:CLFrames?
Description	This query return if count lost frames as pattern errors is enabled/disabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:SETT:BER:CLF?` → `1`
Note	This setting applies to all ports.

14.6.4 FCHannel:PORT<Pt>:SETTings:BER:HIDeframeloss

Syntax	FCHannel:PORT<Pt>:SETTings:BER:HIDeframeloss <enable>
Description	This command show/hide Frame/Pattern loss secs. count.
Parameter	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:SETT:BER:HID ON`
Note	This setting applies to all ports.

Syntax	FCHannel:PORT<Pt>:SETTings:BER:HIDeframeloss?
Description	This query return if Frame/Pattern loss secs. count is show/hide.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:SETT:BER:HID?` → `1`
Note	This setting applies to all ports.

14.7 Thresholds

14.7.1 FCHannel:PORT<Pt>:THResholds[:ENABle]

Syntax	FCHannel:PORT<Pt>:THResholds[:ENABle] <enable>
Description	This command enables/disables thresholds.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:THR OFF`
Note	

Syntax	FCHannel:PORT<Pt>:THResholds[:ENABle]?
Description	This query returns whether or not thresholds is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:THR? → 0`
Note	

14.7.2 FCHannel:PORT<Pt>:THResholds:SELect

Syntax	FCHannel:PORT<Pt>:THResholds:SELect <type>, <enable>
Description	This command enables/disables the specific threshold type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> ALL: All thresholds UTIL: Utilization THR: Throughput ERR: Errored frames UNDer: Undersized frames OVER: Oversized frames CRC: CRC errored frames IFG: IFG violations SYMBol: Symbol errors CRD: CRD errors *DEFault = ALL*
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:THR:SEL UTIL, ON`
Note	

Syntax	FCHannel:PORT<Pt>:THResholds:SELect? <type>
Description	This query returns whether or not the specific threshold type is enabled.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA>
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT1:THR:SEL? UTIL → 1`
Note	

14.7.3 FCHannel:PORT<Pt>:THResholds:VALue

Syntax	FCHannel:PORT<Pt>:THResholds:VALue <type>, <compare>, <value>
Description	This command sets the level for the specific threshold type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> UTIL: Utilization[1] THR: Throughput[2] ERR: Errored frames UNDer: Undersized frames OVER: Oversized frames CRC: CRC errored frames IFG: IFG violations SYMBol: Symbol errors CRD: CRD errors
	<compare> = <CHARACTER PROGRAM DATA> GT: Greater than LT: Less than GTEQ: Greater than or equal to LTEQ: Less than or equal to *DEFault = LT*
	<value> = <NUMERIC PROGRAM DATA> Absolute values: Decimals are rounded. Maximum is 4000000000 Percentage values: Allows one decimal, Maximum is 100.0 *DEFault = 0, MINimum = 0* *Allowed Suffixes = PCT*
Response	None.
Example	`FCH:PORT1:THR:VAL UTIL, GT, 10.5PCT`
Note	[1] Utilization only supports percentage values. [2] Throughput only supports absolute values, the unit is MBPS, and has one decimal. Maximum = 10000.0

Syntax	FCHannel:PORT<Pt>:THResholds:VALue? <type>
Description	This query returns the level for the specific threshold type.
Parameter	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA>
Response	<compare> = <CHARACTER RESPONSE DATA>
	<value> = <NR2 NUMERIC RESPONSE DATA> Count or percentage.
Example	`FCH:PORT1:THR:VAL? UTIL → GT, 10.5PCT`
Note	

14.8 Follow

14.8.1 FCHannel:PORT<Pt>:FOLLow:GENerator

Syntax	FCHannel:PORT<Pt>:FOLLow:GENerator <enable>
Description	This command enables/disables generator setup for port <Pt> to follow PORT1.
Parameter	<Pt> = Port number (2-4)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT2:FOLL:GEN OFF`
Note	

Syntax	FCHannel:PORT<Pt>:FOLLow:GENerator?
Description	This query returns whether or not generator setup for port <Pt> to follow PORT1 is enabled.
Parameter	<Pt> = Port number (2-4)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT2:FOLL:GEN?` → `0`
Note	

14.8.2 FCHannel:PORT<Pt>:FOLLow:THResholds

Syntax	FCHannel:PORT<Pt>:FOLLow:THResholds <enable>
Description	This command enables/disables thresholds setup for port <Pt> to follow PORT1.
Parameter	<Pt> = Port number (2-4)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT2:FOLL:THR OFF`
Note	

Syntax	FCHannel:PORT<Pt>:FOLLow:THResholds?
Description	This query returns whether or not thresholds setup for port <Pt> to follow PORT1 is enabled.
Parameter	<Pt> = Port number (2-4)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT2:FOLL:THR?` → `0`
Note	

14.8.3 FCHannel:PORT<Pt>:FOLLow:FRAMe

Syntax	FCHannel:PORT<Pt>:FOLLow:FRAMe <enable>
Description	This command enables/disables frame setup for port <Pt> to follow PORT1.
Parameter	<Pt> = Port number (2-4)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT2:FOLL:FRAM OFF`
Note	

Syntax	FCHannel:PORT<Pt>:FOLLow:FRAMe?
Description	This query returns whether or not frame setup for port <Pt> to follow PORT1 is enabled.
Parameter	<Pt> = Port number (2-4)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT2:FOLL:FRAM?` → `0`
Note	

14.8.4 FCHannel:PORT<Pt>:FOLLow:STReam

Syntax	FCHannel:PORT<Pt>:FOLLow:STReam <enable>
Description	This command enables/disables stream setup for port <Pt> to follow PORT1
Parameter	<Pt> = Port number (2-4)
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`FCH:PORT2:FOLL:STR OFF`
Note	

Syntax	FCHannel:PORT<Pt>:FOLLow:STReam?
Description	This query returns whether or not stream setup for port <Pt> to follow PORT1 is enabled.
Parameter	<Pt> = Port number (2-4)
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`FCH:PORT2:FOLL:STR?` → `0`
Note	

14.9 Stimuli

14.9.1 FCHannel:PORT<Pt>:STIMuli:ALARm

Syntax	FCHannel:PORT<Pt>:STIMuli:ALARm <alarm>
Description	This command sets the stimuli alarm state.
Parameters	<Pt> = Port number
	<alarm> = <CHARACTER PROGRAM DATA> NONE: No alarm LRESet: List reset LRResponse: List reset response NOPerational: Not operational OFFLine: Offline *DEFault = NONE*
Response	None.
Example	`FCH:PORT1:STIM:ALAR LRES`
Note	

Syntax	FCHannel:PORT<Pt>:STIMuli:ALARm?
Description	This query returns the stimuli alarm state.
Parameter	<Pt> = Port number
Response	<alarm> = <CHARACTER RESPONSE DATA>
Example	`FCH:PORT1:STIM:ALAR?` → `LRES`
Note	

14.9.2 FCHannel:PORT<Pt>:STIMuli:ERRor

Syntax	FCHannel:PORT<Pt>:STIMuli:ERRor <error>
Description	This command sets the stimuli error state.
Parameters	<Pt> = Port number
	<error> = <CHARACTER PROGRAM DATA> NONE: No error BIT: Bit SYMBol: Symbol RRDY: R_RDY CRC: CRC BLOCk: Block *DEFault = NONE*
Response	None.
Example	`FCH:PORT1:STIM:ERR BIT`
Note	

Syntax	FCHannel:PORT<Pt>:STIMuli:ERRor?
Description	This query returns the stimuli error state.
Parameter	<Pt> = Port number
Response	<error> = <CHARACTER RESPONSE DATA>
Example	`FCH:PORT1:STIM:ERR?` → `BIT`
Note	

14.9.3 FCHannel:PORT<Pt>:STIMuli:EINSertion

Syntax	FCHannel:PORT<Pt>:STIMuli:EINSertion <mode>
Description	This command sets the stimuli error insertion mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: Stimuli disabled MANual B1: Burst every 1 second B10: Burst every 10 seconds BE2: Burst · 1E-02 BE3: Burst · 1E-03 BE4: Burst · 1E-04 BE5: Burst · 1E-05 BE6: Burst · 1E-06 BE7: Burst · 1E-07 *DEFault = OFF*
Response	None.
Example	`FCH:PORT1:STIM:EINS MAN`
Note	

Syntax	FCHannel:PORT<Pt>:STIMuli:EINSertion?
Description	This query returns the stimuli error insertion mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`FCH:PORT1:STIM:EINS?` → `MAN`
Note	

14.9.4 FCHannel:PORT<Pt>:STIMuli:EBLength

Syntax	FCHannel:PORT<Pt>:STIMuli:EBLength <length>
Description	This command sets the error burst length.
Parameters	<Pt> = Port number
	<length> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 4000*[1], *DEFault = 1*
Response	None.
Example	`FCH:PORT1:STIM:EBL 128`
Note	[1] The following error type have other MAXimum value. RRDY: *MAXimum = 500* SYMBol: *MAXimum = 1000* BLOCk: *MAXimum = 256*

Syntax	FCHannel:PORT<Pt>:STIMuli:EBLength?
Description	This query returns the error burst length.
Parameter	<Pt> = Port number
Response	<length> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:STIM:EBL?` → `128`
Note	

14.9.5 FCHannel:PORT<Pt>:STIMuli:FOFFset

Syntax	FCHannel:PORT<Pt>:STIMuli:FOFFset <offset>
Description	This command sets the frequency offset for the clock source. Unit: ppm.
Parameters	<Pt> = Port number
	<offset> = <NUMERIC PROGRAM DATA> MT1000A: *MINimum=-100, MAXimum=100, DEFault=0* MT1100A: *MINimum=-200.0, MAXimum=200.0, DEFault = 0*
Response	None
Example	`FCH:PORT1:STIM:FOFF 0`
Note	The offset is applied to the internal clock source only.

Syntax	FCHannel:PORT<Pt>:STIMuli:FOFFset?
Description	This query returns the frequency offset for the clock source. Unit: ppm.
Parameter	<Pt> = Port number
Response	MT1000A: <offset> = <NR1 NUMERIC RESPONSE DATA> MT1100A: <offset> = <NR2 NUMERIC RESPONSE DATA>
Example	MT1100A: `FCH:PORT1:STIM:FOFF?` → `0.0`
Note	

14.10 Results

14.10.1 FCHannel:PORT<Pt>:IFETch?

Syntax	FCHannel:PORT<Pt>:IFETch? <parameter>
Description	This query fetches an Fibre Channel interval if available.
Parameters	<Pt> = Port number
	({<parameter>} + {,}∗) = <EXPRESSION PROGRAM DATA> The response format is listed for each parameter. **BERT** BPBC: Pattern bit count. Response: <Count> BPE: Pattern errors. Response: <Count>,<Ratio> BFL: Frame loss. Response: <Count> BFLS: Frame loss seconds. Response: <Count> BPLS: Pattern loss seconds. Response: <Count> BL: Latency (microseconds). Response: <Min>,<Max>,<Avg> BJ: Jitter (microseconds). Response: <Min>,<Max>,<Avg> BSD: Service disruption (microseconds). Response: <Max>,<Avg>,<Count> **Reflector** RFFRM: Reflected frames. Response: <Count>,<Ratio> RFBYT: Reflected bytes. Response: <Count> NRFFR: Not reflected frames. Response: <Count>,<Ratio> NRFBY: Not reflected bytes. Response: <Count> TOFRM: Total frames. Response: <Count>,<Ratio> TOBYT: Reflected bytes. Response: <Count> **Link** SER: Symbol errors. Response: <Count> BLER: Block errors. Response: <Count> RRDY: R_RDY. Response: <CountTx>,<CountRx> LR: Link reset. Response: <Count> LRR: Link reset response. Response: <Count> NOS: Not operational seconds. Response: <Count> OLS: Offline seconds. Response: <Count> **Frame** FRM: Frames. Response: <CountTx>,<CountRx> BYT: Bytes. Response: <CountTx>,<CountRx> IV: IFG violations. Response: <Count>,<Ratio> CRC: CRC errors. Response: <Count>,<Ratio> SEFR: Symbol error frames. Response: <Count>,<Ratio> CRD: CRD errors. Response: <Count>,<Ratio> TER: Terminate errors. Response: <Count>,<Ratio> EFRM: Errored frames. Response: <Count>,<Ratio> RFRM: Reflected frames. Response: <Count>,<Ratio> RBYT: Reflected bytes. Response: <Count> NRFRM: Not reflected frames. Response: <Count>,<Ratio> NRBYT: Not reflected bytes. Response: <Count> TFRM: Total frames. Response: <Count>,<Ratio> TBYT: Total bytes. Response: <Count> NCR: No credit. Response: <Count> **Performance** LRAT: Line rate (bps). Response: <Min>,<Max>,<Avg> FRAT: Frame rate (fps). Response: <Min>,<Max>,<Avg> UTIL: Utilization. Response: <Min%>,<Max%>,<Avg%> THR: Throughput (bps). Response: <Min>,<Max>,<Avg> **Size Distribution** UNFR: Undersized frames. Response: <Count>,<Ratio> R28: 28-196. Response: <Count>,<Ratio> R200: 200-396. Response: <Count>,<Ratio>

continued on next page...

. . . continued from previous page

	R400: 400-596. Response: <Count>,<Ratio> R600: 600-796. Response: <Count>,<Ratio> R800: 800-996. Response: <Count>,<Ratio> R1000: 1000-1196. Response: <Count>,<Ratio> R1200: 1200-1396. Response: <Count>,<Ratio> R1400: 1400-1596. Response: <Count>,<Ratio> R1600: 1600-1796. Response: <Count>,<Ratio> R1800: 1800-1996. Response: <Count>,<Ratio> R2000: 2000-2196. Response: <Count>,<Ratio> OVFR: Oversized frames. Response: <Count>,<Ratio> FSIZ: Frame size (bytes). Response: <Avg>
Response	{(<result>), }* = <EXPRESSION RESPONSE DATA> Expression format: Numeric List Each result is formatted according to the specification in the parameter field. Values that are not relevant or applicable for the current measurement return NaN (section 1.6.1).
Examples	`FCH:PORT1:IFET? (UTIL)` → `(0.0000,50.0000,41.1117)` `FCH:PORT1:IFET? (BPE,BPLS)` → `(13944487,0.000979),(104)`
Notes	This command fetches the results from the interval selected using the `MEASurement:SETup:SELect` command (see section 18.2.2). If the requested result is not available, NaN (section 1.6.1) is returned. If there is one or more results, the last "," is always removed.

14.10.2 FCHannel:PORT<Pt>:TFETch?

Syntax	FCHannel:PORT<Pt>:TFETch? <parameter>
Description	This query fetches thresholds for a Fibre Channel interval if available.
Parameters	<Pt> = Port number
	({<parameter>} + {, }∗) = <EXPRESSION PROGRAM DATA> **Frame** IV: IFG violations. CRC: CRC errors. SEFR: Symbol error frames. CRD: CRD errors. EFRM: Errored frames **Performance** UTIL: Utilization (pct). THR: Throughput (bps). **Size Distribution** UNFR: Undersized frames. OVFR: Oversized frames.
Response	{(<result>), }* = <EXPRESSION RESPONSE DATA> Expression format: Numeric List <result> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PORT1:TFET? (UTIL,UNFR)` → `(1),(0)`
Note	This query fetches from the interval selected using the `MEASurement:SETup:SELect` command (see section 18.2.2).

14.11 Status

14.11.1 FCHannel:STATus:PORT<Pt>:AESummary[:EVENt]?

Syntax	FCHannel:STATus:PORT<Pt>:AESummary[:EVENt]?
Description	This query returns the Fibre Channel alarms and errors summary event register. The content of this event register is summarized in DB12 of the STATus:INTerface:PORT<Pt>:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 (2) = Error summary DB3 - DB16 = NOT USED
Example	`FCH:STAT:PORT1:AES?` → `1`
Note	

14.11.2 FCHannel:STATus:PORT<Pt>:AESummary:CONDition?

Syntax	FCHannel:STATus:PORT<Pt>:AESummary:CONDition?
Description	This query returns the Fibre Channel alarms and errors summary condition register. DB1 is summary of FCHannel:STATus:PORT<Pt>:ALARm[:EVENt]? DB2 is summary of FCHannel:STATus:PORT<Pt>:ERRor[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 (2) = Error summary DB3 - DB16 = NOT USED
Example	`FCH:STAT:PORT1:AES:COND?` → `2`
Note	

14.11.3 FCHannel:STATus:PORT<Pt>:ALARm[:EVENt]?

Syntax	FCHannel:STATus:PORT<Pt>:ALARm[:EVENt]?
Description	This query returns the alarms event register. The content of this register is summarized in DB1 of the FCHannel:STATus:PORT<Pt>:AESummary:CONDition register.
Parameters	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Signal Present DB2 (2) = Sync Acquired DB3 (4) = Link DB4 (8) = Fabric Login DB5 (16) = Port Login DB6 (32) = Pattern Sync DB7 (64) = ELP DB8 - DB16 = NOT USED
Example	`FCH:STAT:PORT1:ALAR?` → `2`
Notes	

14.11.4 FCHannel:STATus:PORT<Pt>:ALARm:CONDition?

Syntax	FCHannel:STATus:PORT<Pt>:ALARm:CONDition?
Description	This query returns the alarms condition register.
Parameters	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Signal Present DB2 (2) = Sync Acquired DB3 (4) = Link DB4 (8) = Fabric Login DB5 (16) = Port Login DB6 (32) = Pattern Sync DB7 (64) = ELP DB8 - DB16 = NOT USED
Example	`FCH:STAT:PORT1:ALAR:COND?` → `4`
Notes	

14.11.5 FCHannel:STATus:PORT<Pt>:ERRor[:EVENt]?

Syntax	FCHannel:STATus:PORT<Pt>:ERRor[:EVENt]?
Description	This query returns the errors event register. The content of this register is summarized in DB2 of the FCHannel:STATus:PORT<Pt>:AESummary:CONDition register.
Parameters	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Pattern error DB2 (2) = Traffic error DB3 - DB16 = NOT USED
Example	`FCH:STAT:PORT1:ERR?` → `1`
Notes	

14.11.6 FCHannel:STATus:PORT<Pt>:ERRor:CONDition?

Syntax	FCHannel:STATus:PORT<Pt>:ERRor:CONDition?
Description	This query returns the errors condition register.
Parameters	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Pattern error DB2 (2) = Traffic error DB3 - DB16 = NOT USED
Example	`FCH:STAT:PORT1:ERR:COND?` → `1`
Notes	

14.11.7 FCHannel:STATus:PORT<Pt>:SID?

Syntax	FCHannel:STATus:PORT<Pt>:SID?
Description	This query returns the source ID if Login is enabled.
Parameter	<Pt> = Port number
Response	<link> = <HEXADECIMAL NUMERIC RESPONSE DATA¿>
Example	`FCH:STAT:PORT1:SID?` → `#H11`
Note	NaN (section 1.6.1) is returned if Login is not enabled or port is off.

14.11.8 FCHannel:STATus:PORT<Pt>:DID?

Syntax	FCHannel:STATus:PORT<Pt>:DID?
Description	This query returns the destination ID if Login is enabled.
Parameter	<Pt> = Port number
Response	<link> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`FCH:STAT:PORT1:DID?` → `#H22`
Note	NaN (section 1.6.1) is returned if Login is not enabled or port is off.

14.11.9 FCHannel:STATus:PORT<Pt>:PDEViation?

Syntax	FCHannel:STATus:PORT<Pt>:PDEViation?
Description	This query returns physical deviation. Units: ppm and bps.
Parameter	<Pt> = Port number
Response	<ppm> = <NR1 NUMERIC RESPONSE DATA>
	<bps> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:STAT:PORT1:PDEV?` → `0, 0`
Note	

14.12 Performance Test

14.12.1 FCHannel:PERF:STARt

Syntax	FCHannel:PERF:STARt
Description	This command starts the Performance Test measurement.
Parameter	None.
Response	None.
Example	`FCH:PERF:STAR`
Note	

14.12.2 FCHannel:PERF:STOP

Syntax	FCHannel:PERF:STOP
Description	This command stops the Performance Test measurement.
Parameter	None.
Response	None.
Example	`FCH:PERF:STOP`
Note	

14.12.3 FCHannel:PERF:SETup:GENeral:MODE

Syntax	FCHannel:PERF:SETup:GENeral:MODE <mode>
Description	This command sets the Performance Test test mode.
Parameter	<mode> = <CHARACTER PROGRAM DATA> PTP: Port-to-Port LOOP: Loopback *DEFault = PTP*
Response	None.
Examples	`FCH:PERF:SET:GEN:MODE PTP` `FCH:PERF:SET:GEN:MODE LOOP`
Note	

Syntax	FCHannel:PERF:SETup:GENeral:MODE?
Description	This query returns the Performance Test test mode.
Parameter	None.
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`FCH:PERF:SET:GEN:MODE?` → `PTP`
Note	

14.12.4 FCHannel:PERF:SETup:GENeral:TSELection

Syntax	FCHannel:PERF:SETup:GENeral:TSELection <test>
Description	This command sets the test selection to be executed.
Parameter	({<test>} ∗ {,}∗) = <EXPRESSION PROGRAM DATA> THRoughput: Throughput TRPRofile: Traffic Profile LATency: Latency BURSt: Burst CREDit: Credit
Response	None.
Example	`FCH:PERF:SET:GEN:TSEL (THR,LAT,BURS)`
Note	

Syntax	FCHannel:PERF:SETup:GENeral:TSELection?
Description	This query returns the test selection that will be executed.
Parameter	None.
Response	{(<test>),}* = <EXPRESSION RESPONSE DATA>
Example	`FCH:PERF:SET:GEN:TSEL?` → `(THR,LAT,BURS)`
Note	

14.12.5 FCHannel:PERF:SETup:GENeral:TCLayer

Syntax	FCHannel:PERF:SETup:GENeral:TCLayer <layer>
Description	This command selects the throughput calculation layer.
Parameter	<layer> = <CHARACTER PROGRAM DATA> LR: At line rate L1: At frame layer L2: At data layer *DEFault = LR*
Response	None.
Example	`FCH:PERF:SET:GEN:TCL LR`
Note	

Syntax	FCHannel:PERF:SETup:GENeral:TCLayer?
Description	This query returns the throughput calculation layer.
Parameter	None.
Response	<layer> = <CHARACTER RESPONSE DATA>
Example	`FCH:PERF:SET:GEN:TCL?` → `LR`
Note	

14.12.6 FCHannel:PERF:SETup:GENeral:TTYPe

Syntax	FCHannel:PERF:SETup:GENeral:TTYPe <layer>
Description	This command sets the throughput calculation type.
Parameter	<type> = <CHARACTER PROGRAM DATA> AVG: Average MAX: Maximum *DEFault = MAX*
Response	None.
Example	`FCH:PERF:SET:GEN:TTYP AVG`
Note	

Syntax	FCHannel:PERF:SETup:GENeral:TTYPe?
Description	This query returns the throughput calculation type.
Parameter	None.
Response	<layer> = <CHARACTER RESPONSE DATA>
Example	`FCH:PERF:SET:GEN:TTYP?` → `AVG`
Note	

14.12.7 FCHannel:PERF:SETup:FSIZe:MODE

Syntax	FCHannel:PERF:SETup:FSIZe:MODE <mode>
Description	This command sets the Performance Test frame size mode.
Parameter	None.
	<mode> = <CHARACTER PROGRAM DATA> USER: User defined STEPped: Stepped CONStant: Constant *DEFault = USER*
Response	None.
Example	`FCH:PERF:SET:FSIZ:MODE CONS`
Note	

Syntax	FCHannel:PERF:SETup:FSIZe:MODE?
Description	This query returns the Performance Test frame size mode.
Parameter	None.
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`FCH:PERF:SET:FSIZ:MODE?` → `CONS`
Note	

14.12.8 FCHannel:PERF:SETup:FSIZe:USER

Syntax	FCHannel:PERF:SETup:FSIZe:USER {<size>}*
Description	This command sets the Performance Test user defined frame size(s).
Parameter	({<size>} ∗ {, }∗) = <EXPRESSION PROGRAM DATA> 64: 64 bytes 128: 128 bytes 256: 256 bytes 512: 512 bytes 768: 768 bytes 1024: 1024 bytes 1280: 1280 bytes 2140: 2140 bytes JUMB: Jumbo frame size use `:JUMBo` to define.
Response	None.
Example	`FCH:PERF:SET:FSIZ:USER (64,256,JUMB)`
Note	

Syntax	FCHannel:PERF:SETup:FSIZe:USER?
Description	This query returns the Performance Test user defined frame size(s).
Parameter	None.
Response	{(<size>), }* = <EXPRESSION RESPONSE DATA> <size> = <CHARACTER RESPONSE DATA>
Example	`FCH:PERF:SET:FSIZ:USER?` → `(64,256,JUMB)`
Note	

14.12.9 FCHannel:PERF:SETup:FSIZe:JUMBo

Syntax	FCHannel:PERF:SETup:FSIZe:JUMBo <size>
Description	This command sets the Performance Test jumbo frame size.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum = 2168, MAXimum = 3240, DEFault = 2168*
Response	None.
Example	`FCH:PERF:SET:FSIZ:JUMB 2168`
Note	

Syntax	FCHannel:PERF:SETup:FSIZe:JUMBo?
Description	This query returns the Performance Test jumbo frame size.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:FSIZ:JUMB?` → `2168`
Note	

14.12.10 FCHannel:PERF:SETup:FSIZe:BEGin

Syntax	FCHannel:PERF:SETup:FSIZe:BEGin <size>
Description	This command sets the Performance Test begin frame size for stepped mode.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum = 40, MAXimum = 3240, DEFault = 64*
Response	None.
Example	`FCH:PERF:SET:FSIZ:BEG 128`
Note	

Syntax	FCHannel:PERF:SETup:FSIZe:BEGin?
Description	This query returns the Performance Test begin frame size for stepped mode.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:FSIZ:BEG?` → `128`
Note	

14.12.11 FCHannel:PERF:SETup:FSIZe:END

Syntax	FCHannel:PERF:SETup:FSIZe:END <size>
Description	This command sets the Performance Test end frame size for stepped mode.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum = 40, MAXimum = 3240, DEFault = 2140*
Response	None.
Example	`FCH:PERF:SET:FSIZ:END 128`
Note	

Syntax	FCHannel:PERF:SETup:FSIZe:END?
Description	This query returns the Performance Test end frame size for stepped mode.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:FSIZ:END?` → `128`
Note	

14.12.12 FCHannel:PERF:SETup:FSIZe:STEP

Syntax	FCHannel:PERF:SETup:FSIZe:STEP <size>
Description	This command sets the Performance Test step frame size for stepped mode.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 3240, DEFault = 256*
Response	None.
Example	`FCH:PERF:SET:FSIZ:STEP 128`
Note	

Syntax	FCHannel:PERF:SETup:FSIZe:STEP?
Description	This query returns the Performance Test step frame size for stepped mode.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:FSIZ:STEP?` → `128`
Note	

14.12.13 FCHannel:PERF:SETup:FSIZe:CONStant

Syntax	FCHannel:PERF:SETup:FSIZe:CONStant <size>
Description	This command sets the Performance Test frame size for constant mode.
Parameter	<size> = <NUMERIC PROGRAM DATA> *MINimum = 40, MAXimum = 3240, DEFault = 2140*
Response	None.
Example	`FCH:PERF:SET:FSIZ:CONS 128`
Note	

Syntax	FCHannel:PERF:SETup:FSIZe:CONStant?
Description	This query returns the Performance Test frame size for constant mode.
Parameter	None.
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:FSIZ:CONS?` → `128`
Note	

14.12.14 FCHannel:PERF:SETup:PORT<Pt>:THRoughput:LLoad:MAXimum

Syntax	FCHannel:PERF:SETup:PORT<Pt>:THRoughput:LLoad:MAXimum <load>
Description	This command sets the maximum line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0008, MAXimum = 100.0000, DEFault = 100.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`FCH:PERF:SET:PORT1:THR:LL:MAX 10`
Notes	MINimum, MAXimum and DEFault are all in PCT. `:MAXimum` must be larger or equal to `:MINimum`.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:THRoughput:LLoad:MAXimum? [<suffix>]
Description	This query returns the maximum line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:THR:LL:MAX?` → `10`
Note	

14.12.15 FCHannel:PERF:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch :MODE

Syntax	FCHannel:PERF:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch:MODE <mode>
Description	This command sets the auto search mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> SMARt: Smart search BINary: Binary search *DEFault = SMARt*
Response	None.
Example	`FCH:PERF:SET:PORT1:THR:LL:ASE:MODE BIN`
Note	

Syntax	FCHannel:PERF:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch:MODE?
Description	This query returns the auto search mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:THR:LL:ASE:MODE?` → `BIN`
Note	

14.12.16 FCHannel:PERF:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch :RESolution

Syntax	FCHannel:PERF:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch:RESolution <res>
Description	This command sets the auto search resolution.
Parameters	<Pt> = Port number
	<res> = <CHARACTER PROGRAM DATA> 01: 0.1% 1: 1% 10: 10% *DEFault = 01*
Response	None.
Example	`FCH:PERF:SET:PORT1:THR:LL:ASE:RES 10`
Note	In Port-to-Port test PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:THRoughput:LLoad:ASEarch:RESolution?
Description	This query returns the auto search resolution.
Parameter	<Pt> = Port number
Response	<res> = <CHARACTER RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:THR:LL:ASE:RES?` → `10`
Note	

14.12.17 FCHannel:PERF:SETup:PORT<Pt>:THRoughput:DURation:STEP

Syntax	FCHannel:PERF:SETup:PORT<Pt>:THRoughput:DURation:STEP <step>
Description	This command sets the step duration. Unit: seconds.
Parameters	<Pt> = Port number
	<step> = <NUMERIC PROGRAM DATA> *MINimum = 3, MAXimum = 100000000, DEFault = 10*
Response	None.
Example	`FCH:PERF:SET:PORT1:THR:DUR:STEP 5`
Note	In Port-to-Port test PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:THRoughput:DURation:STEP?
Description	This query returns the step duration. Unit: Seconds. Unit: seconds.
Parameter	<Pt> = Port number
Response	<step> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:THR:DUR:STEP?` → `5`
Note	

14.13 Performance Test - Traffic Profile

14.13.1 FCHannel:PERF:SETup:PORT<Pt>:PROFile:LLoad:MINimum

Syntax	FCHannel:PERF:SETup:PORT<Pt>:PROFile:LLoad:MINimum <load>
Description	This command sets the minimum line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0008, MAXimum = 100.0000, DEFault = 10.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`FCH:PERF:SET:PORT1:PROF:LL:MIN 10`
Notes	In Port-to-Port test PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. MINimum, MAXimum and DEFault are all in PCT. `:MINimum` must be smaller or equal to `:MAXimum`.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:PROFile:LLoad:MINimum? [<suffix>]
Description	This query returns the minimum line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:PROF:LL:MIN? → 10`
Note	

14.13.2 FCHannel:PERF:SETup:PORT<Pt>:PROFile:LLoad:MAXimum

Syntax	FCHannel:PERF:SETup:PORT<Pt>:PROFile:LLoad:MAXimum <load>
Description	This command sets the maximum line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0008, MAXimum = 100.0000, DEFault = 100.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`FCH:PERF:SET:PORT1:PROF:LL:MAX 10`
Notes	In Port-to-Port test PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. MINimum, MAXimum and DEFault are all in PCT. `:MAXimum` must be larger or equal to `:MINimum`.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:PROFile:LLoad:MAXimum? [<suffix>]
Description	This query returns the maximum line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:PROF:LL:MAX? → 10`
Note	

14.13.3 FCHannel:PERF:SETup:PORT<Pt>:PROFile:LLoad:STEP

Syntax	FCHannel:PERF:SETup:PORT<Pt>:PROFile:LLoad:STEP <load>
Description	This command sets the step line load percentage.
Parameters	<Pt> = Port number
	<load> = <NUMERIC PROGRAM DATA> *MINimum = 0.0001, MAXimum = 100.0000, DEFault = 10.0000* *Allowed suffixes = PCT, MBPS, KBPS. Default = PCT*
Response	None.
Example	`FCH:PERF:SET:PORT1:PROF:LL:STEP 10`
Notes	In Port-to-Port test PORT1 setup is used. PORT2 setup can be set but will be ignored during the test. MINimum, MAXimum and DEFault are all in PCT.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:PROFile:LLoad:STEP? [<suffix>]
Description	This query returns the step line load percentage.
Parameters	<Pt> = Port number
	<suffix> = <CHARACTER PROGRAM DATA> PCT: Percent KBPS: Kilo bit per second MBPS: Mega bit per second *DEFault = PCT*
Response	<load> = <NR2 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:PROF:LL:STEP? → 10`
Note	

14.13.4 FCHannel:PERF:SETup:PORT<Pt>:PROFile:DURation:STEP

Syntax	FCHannel:PERF:SETup:PORT<Pt>:PROFile:DURation:STEP <step>
Description	This command sets the step duration. Unit: Seconds.
Parameters	<Pt> = Port number
	<step> = <NUMERIC PROGRAM DATA> *MINimum = 3, MAXimum = 100000000, DEFault = 10*
Response	None.
Example	`FCH:PERF:SET:PORT1:PROF:DUR:STEP 5`
Note	In Port-to-Port test PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:PROFile:DURation:STEP?
Description	This query returns the step duration. Unit: Seconds.
Parameter	<Pt> = Port number
Response	<step> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:PROF:DUR:STEP? → 5`
Note	

14.14 Performance Test - Latency

14.14.1 FCHannel:PERF:SETup:PORT<Pt>:LATency:DURation:STEP

Syntax	FCHannel:PERF:SETup:PORT<Pt>:LATency:DURation:STEP <step>
Description	This command sets the step duration. Unit: Seconds.
Parameters	<Pt> = Port number
	<step> = <NUMERIC PROGRAM DATA> *MINimum = 3, MAXimum = 100000000, DEFault = 10*
Response	None.
Example	`FCH:PERF:SET:PORT1:LAT:DUR:STEP 5`
Note	In Port-to-Port test PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:LATency:DURation:STEP?
Description	This query returns the step duration. Unit: Seconds.
Parameter	<Pt> = Port number
Response	<step> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:LAT:DUR:STEP?` → `5`
Note	

14.15 Performance Test - Burst

14.15.1 FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:MODE

Syntax	FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:MODE <mode>
Description	This command sets the Performance Test frames per burst mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> ASEarch: Auto Search STEPped: Stepped *DEFault = ASEarch*
Response	None.
Example	`FCH:PERF:SET:PORT1:BURS:FPB:MODE ASE`
Note	In Port-to-Port test PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:MODE?
Description	This query returns the Performance Test frames per burst mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:BURS:FPB:MODE?` → `ASE`
Note	

14.15.2 FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:BEGin

Syntax	FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:BEGin <size>
Description	This command sets the Performance Test burst begin size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 2000000000, DEFault = 100*
Response	None.
Example	`FCH:PERF:SET:PORT1:BURS:FPB:BEG 10`
Note	In Port-to-Port test PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:BEGin?
Description	This query returns the Performance Test burst begin size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:BURS:FPB:BEG?` → `10`
Note	

14.15.3 FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:END

Syntax	FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:END <size>
Description	This command sets the Performance Test burst end size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 2000000000, DEFault = 2000*
Response	None.
Example	`FCH:PERF:SET:PORT1:BURS:FPB:END 20`
Note	In Port-to-Port test PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:END?
Description	This query returns the Performance Test burst end size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:BURS:FPB:END?` → `20`
Note	

14.15.4 FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:STEP

Syntax	FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:STEP <size>
Description	This command sets the Performance Test burst step size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 2000000000, DEFault = 100*
Response	None.
Example	`FCH:PERF:SET:PORT1:BURS:FPB:STEP 5`
Note	In Port-to-Port test PORT1 setup is used. PORT2 setup can be set but will be ignored during the test.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:STEP?
Description	This query returns the Performance Test burst step size.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:BURS:FPB:STEP?` → `5`
Note	

14.15.5 FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:MODE

Syntax	FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:MODE <mode>
Description	This command sets the frames per burst auto search mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> SMARt: Smart search BINary: Binary search *DEFault = SMARt*
Response	None.
Example	`FCH:PERF:SET:PORT1:BURS:FPB:ASE:MODE SMAR`
Note	This setting is used when frames per burst mode is auto search.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:MODE?
Description	This query returns the frames per burst auto search mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:BURS:FPB:ASE:MODE?` → `SMAR`
Note	

14.15.6 FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:RESolution

Syntax	FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:RESolution <resolution>
Description	This command sets the frames per burst auto search resolution.
Parameters	<Pt> = Port number
	<resolution> = <CHARACTER PROGRAM DATA> PM1: 1 per mille (0.1%) PM10: 10 per mille (1.0%) PM100: 100 per mille (10.0%) *DEFault = PM1*
Response	None.
Example	`FCH:PERF:SET:PORT1:BURS:FPB:ASE:RES PM10`
Note	This setting is used when frames per burst mode is auto search.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:BURSt:FPBurst:ASEarch:RESolution?
Description	This query returns the frames per burst auto search resolution.
Parameter	<Pt> = Port number
Response	<resolution> = <CHARACTER RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:BURS:FPB:ASE:RES?` → `PM10`
Note	

14.16 Performance Test - Credit

14.16.1 FCHannel:PERF:SETup:PORT<Pt>:CREDit:MODE

Syntax	FCHannel:PERF:SETup:PORT<Pt>:CREDit:MODE <mode>
Description	This command sets the Performance Test frames per credit mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> ASEarch: Auto Search STEPped: Stepped *DEFault = ASEarch*
Response	None.
Example	`FCH:PERF:SET:PORT1:CRED:MODE ASE`
Note	This setting is not available when Port-to-Port test selected.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:CREDit:MODE?
Description	This query returns the Performance Test frames per credit mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:CRED:MODE?` → `ASE`
Note	

14.16.2 FCHannel:PERF:SETup:PORT<Pt>:CREDit:BEGin

Syntax	FCHannel:PERF:SETup:PORT<Pt>:CREDit:BEGin <size>
Description	This command sets the Performance Test credit begin size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 65535, DEFault = 1*
Response	None.
Example	`FCH:PERF:SET:PORT1:CRED:BEG 10`
Note	This setting is not available when Port-to-Port test selected.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:CREDit:BEGin?
Description	This query returns the Performance Test credit begin size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:CRED:BEG?` → `10`
Note	

14.16.3 FCHannel:PERF:SETup:PORT<Pt>:CREDit:END

Syntax	FCHannel:PERF:SETup:PORT<Pt>:CREDit:END <size>
Description	This command sets the Performance Test credit end size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 65535, DEFault = 2000*
Response	None.
Example	`FCH:PERF:SET:PORT1:CRED:END 20`
Note	This setting is not available when Port-to-Port test selected.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:CREDit:END?
Description	This query returns the Performance Test credit end size for stepped mode.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:CRED:END?` → `20`
Note	

14.16.4 FCHannel:PERF:SETup:PORT<Pt>:CREDit:STEP

Syntax	FCHannel:PERF:SETup:PORT<Pt>:CREDit:STEP <size>
Description	This command sets the Performance Test credit step size for stepped mode.
Parameters	<Pt> = Port number
	<size> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 65535, DEFault = 100*
Response	None.
Example	`FCH:PERF:SET:PORT1:CRED:STEP 5`
Note	This setting is not available when Port-to-Port test selected.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:CREDit:STEP?
Description	This query returns the Performance Test credit step size.
Parameter	<Pt> = Port number
Response	<size> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:CRED:STEP? → 5`
Note	

14.16.5 FCHannel:PERF:SETup:PORT<Pt>:CREDit:ASEarch:MODE

Syntax	FCHannel:PERF:SETup:PORT<Pt>:CREDit:ASEarch:MODE <mode>
Description	This command sets the credit auto search mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> SMARt: Smart search BINary: Binary search *DEFault = BINary*
Response	None.
Example	`FCH:PERF:SET:PORT1:CRED:ASE:MODE SMAR`
Notes	This setting is not available when Port-to-Port test selected. This setting is used when frames per burst mode is auto search.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:CREDit:ASEarch:MODE?
Description	This query returns the credit auto search mode.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:CRED:ASE:MODE? → SMAR`
Note	

14.16.6 FCHannel:PERF:SETup:PORT<Pt>:CREDit:ASEarch:RESolution

Syntax	FCHannel:PERF:SETup:PORT<Pt>:CREDit:ASEarch:RESolution <resolution>
Description	This command sets the credit auto search resolution.
Parameters	<Pt> = Port number
	<resolution> = <CHARACTER PROGRAM DATA> 1: 1 credits 10: 10 credits 100: 100 credits *DEFault = 1*
Response	None.
Example	`FCH:PERF:SET:PORT1:CRED:ASE:RES 1`
Note	This setting is not available when Port-to-Port test selected.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:CREDit:ASEarch:RESolution?
Description	This query returns the credit auto search resolution.
Parameter	<Pt> = Port number
Response	<resolution> = <CHARACTER RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:CRED:ASE:RES?` → `1`
Note	

14.16.7 FCHannel:PERF:SETup:PORT<Pt>:CREDit:DURation:STEP

Syntax	FCHannel:PERF:SETup:PORT<Pt>:CREDit:DURation:STEP <step>
Description	This command sets the Performance Test credit step duration. Unit: Seconds.
Parameters	<Pt> = Port number
	<step> = <NUMERIC PROGRAM DATA> *MINimum = 3, MAXimum = 100000000, DEFault = 10*
Response	None.
Example	`FCH:PERF:SET:PORT1:CRED:DUR:STEP 5`
Note	This setting is not available when Port-to-Port test selected.

Syntax	FCHannel:PERF:SETup:PORT<Pt>:CREDit:DURation:STEP?
Description	This query returns the step duration. Unit: Seconds.
Parameter	<Pt> = Port number
Response	<step> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:SET:PORT1:CRED:DUR:STEP?` → `5`
Note	

14.17 Performance Test - Result

14.17.1 FCHannel:PERF:RESult:TEST

Syntax	FCHannel:PERF:RESult:TEST <test>
Description	This command selects the test to fetch results from.
Parameter	<test> = <CHARACTER PROGRAM DATA> THRoughput: Throughput test TRPRofile: Traffic Profile LATency: Latency test BURSt: Burst test CREDit: Credit test *DEFault = THRoughput*
Response	None.
Example	`FCH:PERF:RES:TEST LAT`
Note	This setting is not stored as part of a settings- or result file.

Syntax	FCHannel:PERF:RESult:TEST?
Description	This query returns the selected test to fetch results from.
Parameter	None.
Response	<test> = <CHARACTER RESPONSE DATA>
Example	`FCH:PERF:RES:TEST?` → `LAT`
Note	

14.17.2 FCHannel:PERF:RESult:PORT<Pt>:NSTep?

Syntax	FCHannel:PERF:RESult:PORT<Pt>:NSTep? <dir>
Description	This query returns the number of steps for a given port/direction for the selected test type.
Parameters	<Pt> = Port number
	<dir> = <CHARACTER PROGRAM DATA> TX = Transmitter RX = Receiver *DEFault = TX*
Response	<steps> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:RES:PORT1:NST? RX` → `6`
Note	Can only be used while a Performance test is in memory.

14.17.3 FCHannel:PERF:RESult:PORT<Pt>:SELect

Syntax	FCHannel:PERF:RESult:PORT<Pt>:SELect <dir>, <step>
Description	This command selects the direction and step to fetch results from.
Parameters	<Pt> = Port number
	<dir> = <CHARACTER PROGRAM DATA> TX = Transmitter RX = Receiver *DEFault = TX*
	<step> = <NUMERIC PROGRAM DATA> *MINimum = 1, DEFault = 1*
Response	None.
Example	`FCH:PERF:RES:PORT1:SEL TX, 2`
Note	Use the `FCHannel:PERF:RESult:PORT<Pt>:NSTep?` command (see section 14.17.2) to get the number of steps. This setting is not stored as part of a settings- or result file.

Syntax	FCHannel:PERF:RESult:PORT<Pt>:SELect?
Description	This query returns the selected direction and step to fetch results from.
Parameter	<Pt> = Port number
Response	<dir> = <CHARACTER RESPONSE DATA>
	<step> = <NR1 NUMERIC RESPONSE DATA>
Example	`FCH:PERF:RES:PORT1:SEL?` → `TX, 2`
Note	

14.17.4 FCHannel:PERF:RESult:PORT<Pt>:FETCh?

Syntax	FCHannel:PERF:RESult:PORT<Pt>:FETCh? <parameters>
Description	This query fetches the Performance Test results.
Parameters	<Pt> = Port number
	({<parameter>} + {,}∗) = <EXPRESSION PROGRAM DATA> The response format is listed for each parameter. **Common** STEP: Step. Response: <NR1> TFR: Total frames. Response: <NR1> FSIZ: Frame size. Unit: bytes. Response: <NR1> FRAT: Frame rate[1] (only TX). Unit: fps. Response: <NR1> LL: Line load[1] (only TX). Unit: Mbps. Response: <NR2> LLA: Actual line load[1] (only TX). Unit: Mbps. Response: <NR2> THR: Throughput[1] (only RX). Unit: Mbps. Response: <NR2> UTIL: Utilization[1] (only RX). Unit: %. Response: <NR2> **Traffic Profile** FLOS: Frames lost (only RX). Response: <NR2> LRAT: Loss Rate (only RX). Response: <NR2> **Latency** LAT: Latency (Min,Max,Avg) (only RX). Unit: us. Response: <NR2> JITT: Jitter (Min,Max,Avg) (only RX). Unit: us. Response: <NR2> **Throughput** FLOS: Frames lost (only RX). Response: <NR2> **Burst** BSIZ: Burst size FLOS: Frames lost (only RX). Response: <NR2> BPR: Back Pressure **Credit** CSIZ: Credit size FLOS: Frames lost (only RX). Response: <NR2>
Response	{(<result>), }* = <EXPRESSION RESPONSE DATA> Expression format: Numeric List Each result is formatted according to the specification in the parameter field.
Example	`FCH:PERF:RES:PORT1:FETC? (REP,FSIZ,LL)` → `(0), (64), (90.0000)`
Notes	This query fetches the result from the test, direction and step selected by the `FCHannel:PERF:RESult:TEST` command (see section 14.17.1) and `FCHannel:PERF:RESult:PORT<Pt>:SELect` command (see section 14.17.3). If a requested result is not available, NaN (section 1.6.1) is returned. If there is one or more results, the last ”,” is always removed. [1] Not available for Burst Test.

Chapter 15

OTN

15.1 Transmitter

15.1.1 OTN:TX<Pt>:INTerface

Syntax	OTN:TX<Pt>:INTerface <mode>
Description	This command sets the used interface of the transmitter.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> SFP: SFP/SFP+ optical interface QSFP: QSFP+ optical interface CFP: CFP optical interface CFP2: CFP2 optical interface CFP4: CFP4 optical interface QSFP28ADpt: CFP2-QSFP28 Adaptor interface QSFP28: QSFP28 optical interface *DEFault = SFP*
Response	None
Example	`OTN:TX1:INT SFP`
Note	

Syntax	OTN:TX<Pt>:INTerface?
Description	This query returns the used interface.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`OTN:TX1:INT?` → `SFP`
Note	

15.1.2 OTN:TX<Pt>[:ENABled]

Syntax	OTN:TX<Pt>[:ENABled] <mode>
Description	This command sets the mode of the optical transmitters.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: OFF NORMal: Normal (Optical) THRough: Through Rx (pass through mode) OVERwrite: OH Overwrite *DEFault = OFF*
Response	None
Example	`OTN:TX1 OFF`
Note	

Syntax	OTN:TX<Pt>[:ENABled]?
Description	This query returns the mode of the optical transmitters.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`OTN:TX1?` → `OFF`
Note	

15.1.3 OTN:TX<Pt>:TIMing

Syntax	OTN:TX<Pt>:TIMing <timing>
Description	This command sets the timing source.
Parameters	<Pt> = Port number
	<timing> = <CHARACTER PROGRAM DATA> INTernal: Internal clock EXTernal: External clock GPS: GPS signal RX: Received clock *DEFault = INT*
Response	None
Example	`OTN:TX1:TIM INT`
Note	

Syntax	OTN:TX<Pt>:TIMing?
Description	This query returns the timing source.
Parameter	<Pt> = Port number
Response	<timing> = <CHARACTER RESPONSE DATA>
Example	`OTN:TX1:TIM?` → `INT`
Note	

15.1.4 OTN:TX<Pt>:MAPPing:CSIGnal

Syntax	OTN:TX<Pt>:MAPPing:CSIGnal <signal>
Description	This command sets the expected client signal structure.
Parameters	<Pt> = Port number
	<signal> = <CHARACTER PROGRAM DATA> PRBS NUL STM256 STS768 STM64 STS192 STM16 STS48 STM4 STS12 STM1 STS3 STM64S STS192S STM16S STS48S PRBSTRANS NULLTRANS GBE100 GBE40 GBE10

continued on next page...

. . . continued from previous page

	GBE ETHERNET FC1200 FC800 FC400 FC200 FC100 10137M 9830M 6144M 4915M 3072M 2457M 1228M 614M MPLS IPV4PDU IPV6PDU *DEFault = PRBS*
Response	None
Example	`OTN:TX2:MAPP:CSIG PRBS`
Note	Available for SDH/SONET option: STM256/STS768 STM64/STM64S/STS192/STS192S/STM16/STM16S/STS48/STS48S STM4/STS12/STM1/STS3 PRBSTRANS/NULLTRANS Available for Ethernet option: GBE100/GBE40 GBE10/GBE Available for FC option: FC1200/FC800 FC400/FC200/FC100 Available for CPRI option: 10137M/9830M/6144M/4915M/3072M/2457M/1228M/614M

Syntax	OTN:TX<Pt>:MAPPing:CSIGnal?
Description	This query returns the expected client signal structure.
Parameter	<Pt> = Port number
Response	<signal> = <CHARACTER RESPONSE DATA>
Example	`OTN:TX2:MAPP:CSIG?` → `PRBS`
Note	

15.1.5 OTN:TX<Pt>:PATTern

Syntax	OTN:TX<Pt>:PATTern <pattern>
Description	This command sets the pattern type when CSIGnal is PRBS.
Parameters	<Pt> = Port number
	<pattern> = <CHARACTER PROGRAM DATA> PRBS9:PRBS9 pattern PRBS11:PRBS11 pattern PRBS15:PRBS15 pattern PRBS20:PRBS20 pattern PRBS23:PRBS23 pattern PRBS29:PRBS29 pattern PRBS31:PRBS31 pattern USER32BIT: User Pattern (32bit) USER2048BIT: User Pattern (2048bit) *DEFault = PRBS31*
Response	None
Example	`OTN:TX2:PATT PRBS31`
Note	

Syntax	OTN:TX<Pt>:PATTern?
Description	This query returns the pattern type.
Parameter	<Pt> = Port number
Response	<pattern> = <CHARACTER RESPONSE DATA>
Example	`OTN:TX2:PATT?` → `PRBS31`
Note	

15.1.6 OTN:TX<Pt>:PINVersion

Syntax	OTN:TX<Pt>:PINVersion <inverted>
Description	This command enables or disables PRBS pattern inversion.
Parameters	<Pt> = Port number
	<inverted> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None
Example	`OTN:TX1:PINV ON`
Note	

Syntax	OTN:TX<Pt>:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the PRBS pattern.
Parameter	<Pt> = Port number
Response	<inverted> = <BOOLEAN RESPONSE DATA>
Example	`OTN:TX1:PINV?` → `1`
Note	

15.1.7 OTN:TX<Pt>:UP32

Syntax	OTN:TX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters (one bit resolution).
Response	None
Example	`OTN:TX1:UP32 "01101"`
Note	

Syntax	OTN:TX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`OTN:TX1:UP32?` → `"01101"`
Note	

15.1.8 OTN:TX<Pt>:UP2K

Syntax	OTN:TX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`OTN:TX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `OTN:TX1:PATT` is `USER2048BIT`

Syntax	OTN:TX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`OTN:TX1:UP2K?` → `"12DF"`
Note	

15.1.9 OTN:TX<Pt>:MAPPing:OUTPutsignal

Syntax	OTN:TX<Pt>:MAPPing:OUTPutsignal <signal>
Description	This command sets OTN output signal.
Parameters	<Pt> = Port number
	<signal> = <CHARACTER PROGRAM DATA> OTU1 OTU2 OTU2EXTOPU2 OTU1E OTU2E OTU1F OTU2F OTU3 OTU3E1 OTU3E2 OTU4 *DEFault = OTU2*
Response	None
Example	`OTN:TX1:MAPP:OUTP OTU2`
Note	

Syntax	OTN:TX<Pt>:MAPPing:OUTPutsignal?
Description	This query returns OTN output signal.
Parameter	<Pt> = Port number
Response	<signal> = <CHARACTER RESPONSE DATA>
Example	`OTN:TX1:MAPP:OUTP?` → `OTU2`
Note	

15.1.10 OTN:TX<Pt>:FEC

Syntax	OTN:TX<Pt>:FEC <fec>
Description	This command sets OTU FEC control.
Parameters	<Pt> = Port number
	<fec> = <CHARACTER PROGRAM DATA> NOFec: No forward error correction RS: RS (255,239) *DEFault = RS*
Response	None
Example	`OTN:TX1:FEC RS`
Note	

Syntax	OTN:TX<Pt>:FEC?
Description	This query returns OTU FEC control.
Parameter	<Pt> = Port number
Response	<fec> = <CHARACTER RESPONSE DATA>
Example	`OTN:TX1:FEC?` → `RS`
Note	

15.1.11 OTN:TX<Pt>:MAPPing:MULTiplexing<stage>

Syntax	OTN:TX<Pt>:MAPPing:MULTiplexing<stage> <odutype>
Description	This command sets the ODU type of each multiplexing stage.
Parameters	<Pt> = Port number
	<stage> = Multiplexing stage (1-3)
	<odutype> = <CHARACTER PROGRAM DATA> NONE: None ODTU01: ODTU01 ODTU12PT20: ODTU12 (PT=20) ODTU12PT21: ODTU12 (PT=21) ODTU21: ODTU2.1 ODTU2TS: ODTU2.ts ODTU13PT20: ODTU13 (PT=20) ODTU13PT21: ODTU13 (PT=21) ODTU23PT20: ODTU23 (PT=20) ODTU23PT20EXTOPU2: ODTU23 (PT=20)(Ext. OPU2) ODTU23PT21: ODTU23 (PT=21) ODTU23PT21EXTOPU2: ODTU23 (PT=21)(Ext. OPU2) ODTU31: ODTU3.1 ODTU3E28: ODTU3e2.8 ODTU2E3E1: ODTU2e3e1 ODTU3TS: ODTU3.ts ODTU41: ODTU4.1 ODTU42: ODTU4.2 ODTU48ODU2: ODTU4.8 (ODU2) ODTU48EXTOPU2: ODTU4.8 (Ext. OPU2) ODTU48ODU2E: ODTU4.8 (ODU2e) ODTU431: ODTU4.31 ODTU4TS: ODTU4.ts *DEFault = NONE*
Response	None
Example	`OTN:TX1:MAPP:MULT1 NONE`
Note	

Syntax	OTN:TX<Pt>:MAPPing:MULTiplexing<stage>?
Description	This query returns the ODU type of each multiplexing stage.
Parameters	<Pt> = Port number
	<stage> = Multiplexing stage (1-3)
Response	<odutype> = <CHARACTER RESPONSE DATA>
Example	`OTN:TX1:MAPP:MULT1?` → `NONE`
Note	

15.1.12 OTN:TX<Pt>:MAPPing:LANE

Syntax	OTN:TX<Pt>:MAPPing:LANE <value>
Description	This command sets the LLD lane marker assignment.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List List consist of the value of the lane marker. OTU3: 0 to 3 OTU4: 0 to 19
Response	None.
Example	`OTN:TX1:MAPP:LANE (0,1,2,3)` `OTN:TX1:MAPP:LANE (0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19)`
Note	This command can be used on 40/100G

Syntax	OTN:TX<Pt>:MAPPing:LANE?
Description	This query returns the LLD lane marker assignment.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION RESPONSE DATA> Format: Numeric List
Example	`OTN:TX1:MAPP:LANE?` → `(0,1,2,3)` → `(0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19)`
Note	This command can be used on 40/100G

15.1.13 OTN:TX<Pt>:COPY

Syntax	OTN:TX<Pt>:COPY <odutype>,<mode>
Description	This command sets the copy mode of dummy CH.
Parameters	<Pt> = Port number
	<odutype> = <CHARACTER PROGRAM DATA> ODTU01 ODTU12 ODTU21 ODTU2TS ODTU13 ODTU23 ODTU31 ODTU3E28 ODTU2E3E1 ODTU3TS ODTU41 ODTU42 ODTU48 ODTU431 ODTU4TS
	<mode> = <CHARACTER PROGRAM DATA> COPY

continued on next page...

. . . continued from previous page

	UNUSe *DEFault = COPY*
Response	None
Example	`OTN:TX1:COPY ODTU01,COPY`
Note	

Syntax	OTN:TX<Pt>:COPY? <odutype>
Description	This query returns the copy mode of dummy CH.
Parameters	<Pt> = Port number
	<odutype> = <CHARACTER PROGRAM DATA> ODTU01 ODTU12 ODTU21 ODTU2TS ODTU13 ODTU23 ODTU31 ODTU3E28 ODTU2E3E1 ODTU3TS ODTU41 ODTU42 ODTU48 ODTU431 ODTU4TS
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`OTN:TX1:COPY? ODTU01` → `COPY`
Note	

15.1.14 OTN:TX<Pt>:CSF

Syntax	OTN:TX<Pt>:CSF <csf>
Description	This command sets the insertion mode of CSF.
Parameters	<Pt> = Port number
	<csf> = <CHARACTER PROGRAM DATA> GFPT: GFP-T CSF Replacement ETHer: Ethernet Block Replacement *DEFault = ETH*
Response	None
Example	`OTN:TX1:CSF ETH`
Note	

Syntax	OTN:TX<Pt>:CSF?
Description	This query returns the insertion mode of CSF.
Parameter	<Pt> = Port number
Response	<csf> = <CHARACTER RESPONSE DATA>
Example	`OTN:TX1:CSF?` → `ETH`
Note	

15.1.15 OTN:TX<Pt>:FOLLow

Syntax	OTN:TX<Pt>:FOLLow <mode>
Description	This command sets the transmitter setup to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> NONE: Do not follow. TX1: Follows setup of the Tx port1. *DEFault = NONE*
Response	None.
Example	`OTN:TX1:FOLL NONE`
Note	

Syntax	OTN:TX<Pt>:FOLLow?
Description	This query returns the transmitter setup to follow another setup or not to follow.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`OTN:TX1:FOLL?` → `NONE`
Note	

15.1.16 OTN:OH:TX<Pt>:OTN:OH

Syntax	OTN:OH:TX<Pt>:OTN:OH <odutype>,<row>,<column>,<data>
Description	This command sets the value of the specified bytes in the OH.
Parameters	<Pt> = Port number
	<odutype> = <CHARACTER PROGRAM DATA> ODU2: ODU2 ODU1: ODU1 ODU0: ODU0 ODU2E: ODU2e ODU1E: ODU1e ODU2F: ODU2f ODU1F: ODU1f ODUFLEX: ODUflex ODU3: ODU3 ODU3E1: ODU3e1 ODU3E2: ODU3e2 ODU4: ODU4 ODUC: ODUC
	<row> = <NUMERIC PROGRAM DATA> OH Row Number (1-4)
	<column> = <NUMERIC PROGRAM DATA> OH Column Number (1-16)
	<data> = <NUMERIC PROGRAM DATA> Hexadecimal format Multi frame sequence data. Data length depends on the position of OH. The pattern is padded with zeros until multi frame sequence data length.
Response	None
Example	`OTN:OH:TX1:OTN:OH ODU2,3,3,#H1`
Note	The parameter error will occur when you specify a position that can not overwrite.

Syntax	OTN:OH:TX<Pt>:OTN:OH? <odutype>,<row>,<column>
Description	This query returns the value of the specified bytes in the OH.
Parameters	<Pt> = Port number
	<odutype> = <CHARACTER PROGRAM DATA> ODU2: ODU2 ODU1: ODU1 ODU0: ODU0 ODU2E: ODU2e ODU1E: ODU1e ODU2F: ODU2f ODU1F: ODU1f ODUFLEX: ODUflex ODU3: ODU3 ODU3E1: ODU3e1 ODU3E2: ODU3e2 ODU4: ODU4 ODUC: ODUC
	<row> = <NUMERIC PROGRAM DATA> OH Row Number (1-4)
	<column> = <NUMERIC PROGRAM DATA> OH Column Number (1-16)
Response	<data> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`OTN:OH:TX1:OTN:OH? ODU2,3,3` → `#1`
Note	

15.1.17 OTN:OH:TX<Pt>:OTN:OH:DEFault

Syntax	OTN:OH:TX<Pt>:OTN:OH:DEFault <route>
Description	This command sets all overhead bytes to their default value.
Parameters	<Pt> = Port number
	<route> = <CHARACTER PROGRAM DATA> ODU2: ODU2 ODU1: ODU1 ODU0: ODU0 ODU1E: ODU1e ODU2E: ODU2e ODU1F: ODU1f ODU2F: ODU2f ODUFLEX: ODUflex ODU3: ODU3 ODU3E1: ODU3e1 ODU3E2: ODU3e2 ODU4: ODU4 ODUC: ODUC
Response	None
Example	`OTN:OH:TX1:OTN:OH:DEF ODU2`
Note	

15.1.18 OTN:TX<Pt>:TP

Syntax	OTN:TX<Pt>:TP <route>,<tp>
Description	This command sets the TP to be used as the main transmission channel.
Parameters	<Pt> = Port number
	<route> = <CHARACTER PROGRAM DATA> ODTU01: ODTU01 ODTU12: ODTU12 ODTU21: ODTU2.1 ODTU2TS: ODTU2.ts ODTU13: ODTU13

continued on next page...

. . . continued from previous page

	ODTU23: ODTU23 ODTU31: ODTU3.1 ODTU3E28: ODTU3e2.8 ODTU2E3E1: ODTU2e3e1 ODTU3TS: ODTU3.ts ODTU41: ODTU4.1 ODTU42: ODTU4.2 ODTU48: ODTU4.8 ODTU431: ODTU4.31 ODTU4TS: ODTU4.ts *DEFault = ODTU21*
	<tp> = <NUMERIC PROGRAM DATA> ODTU01: TP: 1-2 ODTU12(PT=0x20) TP: 1-4 ODTU12(PT=0x21) TP: 1-4 ODTU21 TP: 1-8 ODTU2TS TP: 1-8 ODTU13(PT=0x20) TP: 1-16 ODTU13(PT=0x21) TP: 1-16 ODTU23(PT=0x20) TP: 1-4 ODTU23(PT=0x21) TP: 1-4 ODTU31 TP: 1-32 ODTU3E28 TP: 1-32 ODTU2E3E1 TP: 1-4 ODTU3TS TP: 1-32 ODTU41 TP: 1-80 ODTU42 TP: 1-80 ODTU48 TP: 1-80 ODTU431 TP: 1-80 ODTU4TS TP: 1-80
Response	None
Example	`OTN:TX1:TP ODTU01,1`
Note	

Syntax	OTN:TX<Pt>:TP? <route>
Description	This query returns the TP to be used as the main transmission channel.
Parameter	<Pt> = Port number
	<route> = <CHARACTER PROGRAM DATA> ODTU01: ODTU01 ODTU12: ODTU12 ODTU21: ODTU2.1 ODTU2TS: ODTU2.ts ODTU13: ODTU13 ODTU23: ODTU23 ODTU31: ODTU3.1 ODTU3E28: ODTU3e2.8 ODTU2E3E1: ODTU2e3e1 ODTU3TS: ODTU3.ts ODTU41: ODTU4.1 ODTU42: ODTU4.2 ODTU48: ODTU4.8 ODTU431: ODTU4.31 ODTU4TS: ODTU4.ts
Response	<tp> = <NUMERIC PROGRAM DATA>
Example	`OTN:TX1:TP? ODTU01` → `1`
Note	

15.1.19 OTN:TX<Pt>:TS

Syntax	OTN:TX<Pt>:TS <route>,<ts>
Description	This command sets a set of TS(s) to be used as the main transmission channel.
Parameters	<Pt> = Port number
	<route> = <CHARACTER PROGRAM DATA> ODTU01: ODTU01 ODTU12: ODTU12 ODTU21: ODTU2.1 ODTU2TS: ODTU2.ts ODTU13: ODTU13 ODTU23: ODTU23 ODTU31: ODTU3.1 ODTU3E28: ODTU3e2.8 ODTU2E3E1: ODTU2e3e1 ODTU3TS: ODTU3.ts ODTU41: ODTU4.1 ODTU42: ODTU4.2 ODTU48: ODTU4.8 ODTU431: ODTU4.31 ODTU4TS: ODTU4.ts *DEFault = ODTU21*
	(<ts> {,<ts>}) = <EXPRESSION PROGRAM DATA> The format is listed for each parameter.
Response	None
Example	`OTN:TX1:TS ODTU12,(1,2)`
Note	ODTU01 TS: will become the same value as TP. ODTU12 (PT=0x20) TS: will become the same value as TP. ODTU12 (PT=0x21) TS1: 1-8, TS2: 1-8. Two different TS numbers must be specified. The two values should not overlap. ODTU2.1 TS: 1-8 ODTU2TS TS: 1-8 ODTU13(PT=0x20) TS: will become the same value as TP. ODTU13(PT=0x21) TS: 1-32 ODTU23(PT=0x20) TS: 1-16 ODTU23(PT=0x21) TS: 1-32 ODTU31 TS: 1-32 ODTU3E28 TS: 1-32 ODTU2E3E1 TS: 1-16 ODTU3TS TS: 1-32 ODTU41 TS: 1-80 ODTU42 TS: 1-80 ODTU48 TS: 1-80 ODTU431 TS: 1-80 ODTU4TS TS: 1-80

Syntax	OTN:TX<Pt>:TS? <route>
Description	This command returns a set of TS(s) to be used as the main transmission channel.
Parameters	<Pt> = Port number
	<route> = <CHARACTER PROGRAM DATA> ODTU01: ODTU01 ODTU12: ODTU12 ODTU21: ODTU2.1 ODTU2TS: ODTU2.ts ODTU13: ODTU13 ODTU23: ODTU23 ODTU31: ODTU3.1

continued on next page...

. . . continued from previous page

	ODTU3E28: ODTU3e2.8 ODTU2E3E1: ODTU2e3e1 ODTU3TS: ODTU3.ts ODTU41: ODTU4.1 ODTU42: ODTU4.2 ODTU48: ODTU4.8 ODTU431: ODTU4.31 ODTU4TS: ODTU4.ts
Response	(<ts> {,<ts>}) = <EXPRESSION RESPONSE DATA> The response format is listed for each parameter.
Example	`OTN:TX1:TS? ODTU12` → `(1)`
Note	

15.1.20 OTN:TX<Pt>:ODUFlex:TS

Syntax	OTN:TX<Pt>:ODUFlex:TS <ts>
Description	This command sets the size of a TS set to be used in ODUflex.
Parameters	<Pt> = Port number
	<ts> = <NUMERIC PROGRAM DATA> ODTU2.ts - PRBS/Ethernet: 1-8 ODTUk.ts - FC400: 4 ODTUk.ts - FC800: 7 ODTUk.ts - 3072M: 3 ODTUk.ts - 4915M: 4 ODTUk.ts - 6144M: 5 ODTUk.ts - 9830M: 8 ODTU3.ts - PRBS: 1-32 ODTU4.ts - PRBS: 1-80 ODTU3.ts - Ethernet: 1-32 ODTU4.ts - Ethernet: 1-80 ODTU3.ts - 10137M: 9 ODTU4.ts - 10137M: 8
Response	None
Example	`OTN:TX1:ODUF:TS 1`

Syntax	OTN:TX<Pt>:ODUFlex:TS?
Description	This command returns the size of a TS set to be used in ODUflex.
Parameter	<Pt> = Port number
Response	<ts> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:TX1:ODUF:TS?` → `1`
Note	

15.1.21 OTN:TX<Pt>:ALLocated

Syntax	OTN:TX<Pt>:ALLocated <route>,<allocated>
Description	This command sets the channel allocation.
Parameters	<Pt> = Port number
	<route> = <CHARACTER PROGRAM DATA> ODTU12: ODTU12 ODTU21: ODTU2.1 ODTU1321: ODTU13 (PT=0x21) ODTU2320: ODTU23 (PT=0x20) ODTU2321: ODTU23 (PT=0x21) ODTU31: ODTU3.1 ODTU3E28: ODTU3e2.8 ODTU3TS: ODTU3.ts ODTU41: ODTU4.1 ODTU42: ODTU4.2 ODTU48: ODTU4.8 ODTU431: ODTU4.31 ODTU4TS: ODTU4.ts
	<allocated> = <STRING PROGRAM DATA> ”0” : All Allocated ”80” : CH1 Unallocated ”40” : CH2 Unallocated ”20” : CH3 Unallocated ”10” : CH4 Unallocated ”8” : CH5 Unallocated ”4” : CH6 Unallocated ”2” : CH7 Unallocated ”1” : CH8 Unallocated *DEFault = 0*
Response	None
Example	`OTN:TX1:ALL ODTU12,"0"`
Note	

Syntax	OTN:TX<Pt>:ALLocated? <route>
Description	This query returns the channel allocation.
Parameters	<Pt> = Port number
	<route> = <CHARACTER PROGRAM DATA> ODTU12: ODTU12 ODTU21: ODTU2.1 ODTU1321: ODTU13 (PT=0x21) ODTU2320: ODTU23 (PT=0x20) ODTU2321: ODTU23 (PT=0x21) ODTU31: ODTU3.1 ODTU3E28: ODTU3e2.8 ODTU3TS: ODTU3.ts ODTU41: ODTU4.1 ODTU42: ODTU4.2 ODTU48: ODTU4.8 ODTU431: ODTU4.31 ODTU4TS: ODTU4.ts
Response	<allocated> = <STRING RESPONSE DATA>
Example	`OTN:TX1:ALL? ODTU12` → `"0"`
Note	

15.1.22 OTN:TX<Pt>:GFP:PTI

Syntax	OTN:TX<Pt>:GFP:PTI <pti>
Description	This command sets GFP-F Payload Type Identifier.
Parameters	<Pt> = Port number
	<pti> = <NUMERIC PROGRAM DATA> *MINimum=#B000, MAXimum=#B111, DEFault=#B000*
Response	None
Example	`OTN:TX1:GFP:PTI #B000`
Note	This command can be used on V2.00 or later

Syntax	OTN:TX<Pt>:GFP:PTI?
Description	This query returns GFP-F Payload Type Identifier.
Parameters	<Pt> = Port number
Response	<pti> = <BINARY NUMERIC RESPONSE DATA>
Example	`OTN:TX1:GFP:PTI?` → `#B000`
Note	This command can be used on V2.00 or later

15.1.23 OTN:TX<Pt>:GFP:PFI

Syntax	OTN:TX<Pt>:GFP:PFI <pfi>
Description	This command sets GFP-F Payload Frame check sequence Indicator.
Parameters	<Pt> = Port number
	<pfi> = <BOOLEAN PROGRAM DATA> *DEFault=0*
Response	None
Example	`OTN:TX1:GFP:PFI 0`
Note	This command can be used on V2.00 or later

Syntax	OTN:TX<Pt>:GFP:PFI?
Description	This query returns GFP-F Payload Frame check sequence Indicator.
Parameters	<Pt> = Port number
Response	<pfi> = <BOOLEAN RESPONSE DATA>
Example	`OTN:TX1:GFP:PFI?` → `0`
Note	This command can be used on V2.00 or later

15.1.24 OTN:TX<Pt>:GFP:EXI

Syntax	OTN:TX<Pt>:GFP:EXI <exi>
Description	This command sets GFP-F Extension header Identifier.
Parameters	<Pt> = Port number
	<exi> = <NUMERIC PROGRAM DATA> *MINimum=#B0000, MAXimum=#B0001, DEFault=#B0000*
Response	None
Example	`OTN:TX1:GFP:EXI #B0000`
Note	This command can be used on V2.00 or later

Syntax	OTN:TX<Pt>:GFP:EXI?
Description	This query returns GFP-F Extension header Identifier.
Parameters	<Pt> = Port number
Response	<exi> = <BINARY NUMERIC RESPONSE DATA>
Example	`OTN:TX1:GFP:EXI?` → `#B0000`
Note	This command can be used on V2.00 or later

15.1.25 OTN:TX<Pt>:GFP:UPI

Syntax	OTN:TX<Pt>:GFP:UPI <upi>
Description	This command sets GFP-F User Payload Identifier.
Parameters	<Pt> = Port number
	<upi> = <NUMERIC PROGRAM DATA> *MINimum=#B00000000, MAXimum=#B11111111, DEFault=#B00000001*
Response	None
Example	`OTN:TX1:GFP:UPI #B00000001`
Note	This command can be used on V2.00 or later

Syntax	OTN:TX<Pt>:GFP:UPI?
Description	This query returns GFP-F User Payload Identifier.
Parameters	<Pt> = Port number
Response	<upi> = <BINARY NUMERIC RESPONSE DATA>
Example	`OTN:TX1:GFP:UPI?` → `#B00000001`
Note	This command can be used on V2.00 or later

15.1.26 OTN:TX<Pt>:GFP:EXTHeader:CID

Syntax	OTN:TX<Pt>:GFP:EXTHeader:CID <cid>
Description	This command sets Channel ID of GFP-F extension header.
Parameters	<Pt> = Port number
	<cid> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=0*
Response	None
Example	`OTN:TX1:GFP:EXTH:CID 0`
Note	This command can be used on V2.00 or later

Syntax	OTN:TX<Pt>:GFP:EXTHeader:CID?
Description	This query returns Channel ID of GFP-F extension header.
Parameters	<Pt> = Port number
Response	<cid> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:TX1:GFP:EXTH:CID?` → `0`
Note	This command can be used on V2.00 or later

15.1.27 OTN:TX<Pt>:GFP:EXTHeader:SPARe

Syntax	OTN:TX<Pt>:GFP:EXTHeader:SPARe <spare>
Description	This command sets spare field of GFP-F extension header.
Parameters	<Pt> = Port number
	<spare> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=255, DEFault=0*
Response	None
Example	`OTN:TX1:GFP:EXTH:SPAR 0`
Note	This command can be used on V2.00 or later

Syntax	OTN:TX<Pt>:GFP:EXTHeader:SPARe?
Description	This query returns spare field of GFP-F extension header.
Parameters	<Pt> = Port number
Response	<spare> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:TX1:GFP:EXTH:SPAR?` → `0`
Note	This command can be used on V2.00 or later

15.1.28 OTN:TX<Pt>:GFP:PTIMes

Syntax	OTN:TX<Pt>:GFP:PTIMes <ptimes>
Description	This command sets GFP-F cHEC Presync Times.
Parameters	<Pt> = Port number
	<ptimes> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=14, DEFault=1*
Response	None
Example	`OTN:TX1:GFP:PTIM 1`
Note	This command can be used on V2.00 or later

Syntax	OTN:TX<Pt>:GFP:PTIMes?
Description	This query returns GFP-F cHEC Presync Times.
Parameters	<Pt> = Port number
Response	<ptimes> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:TX1:GFP:PTIM?` → `1`
Note	This command can be used on V2.00 or later

15.2 Receiver

15.2.1 OTN:RX<Pt>:INTerface

Syntax	OTN:RX<Pt>:INTerface <mode>
Description	This command sets the used interface or switches off the receiver.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> OFF: Off SFP: SFP / SFP+ optical interface QSFP: QSFP+ optical interface CFP: CFP optical interface CFP2: CFP2 optical interface CFP4: CFP4 optical interface QSFP28ADpt: CFP2-QSFP28 Adaptor interface QSFP28: QSFP28 optical interface *DEFault = OFF*
Response	None
Example	`OTN:RX1:INT OFF`
Note	

Syntax	OTN:RX<Pt>:INTerface?
Description	This query returns the mode of the optical receiver.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`OTN:RX1:INT?` → `OFF`
Note	

15.2.2 OTN:RX<Pt>:MAPPing:CSIGnal

Syntax	OTN:RX<Pt>:MAPPing:CSIGnal <signal>
Description	This command sets the client signal.
Parameters	<Pt> = Port number
	<signal> = <CHARACTER PROGRAM DATA> PRBS NUL STM256 STS768 STM64 STS192 STM16 STS48 STM4 STS12 STM1 STS3 STM64S STS192S STM16S STS48S PRBSTRANS NULLTRANS GBE100 GBE40 GBE10 GBE ETHERNET

continued on next page...

. . . continued from previous page

	FC1200 FC800 FC400 FC200 FC100 10137M 9830M 6144M 4915M 3072M 2457M 1228M 614M MPLS IPV4PDU IPV6PDU *DEFault = PRBS*
Response	None
Example	`OTN:RX1:MAPP:CSIG PRBS`
Note	Available for SDH/SONET option: STM256/STS768 STM64/STM64S/STS192/STS192S/STM16/STM16S/STS48/STS48S STM4/STS12/STM1/STS3 PRBSTRANS/NULLTRANS Available for Ethernet option: GBE100/GBE40 GBE10/GBE Available for FC option: FC1200/FC800 FC400/FC200/FC100 Available for CPRI option: 10137M/9830M/6144M/4915M/3072M/2457M/1228M/614M

Syntax	OTN:RX<Pt>:MAPPing:CSIGnal?
Description	This query returns the client signal.
Parameter	<Pt> = Port number
Response	<signal> = <CHARACTER RESPONSE DATA>
Example	`OTN:RX1:MAPP:CSIG?` → `PRBS`
Note	

15.2.3 OTN:RX<Pt>:PATTern

Syntax	OTN:RX<Pt>:PATTern <pattern>
Description	This command sets the pattern type when CSIGnal is PRBS.
Parameters	<Pt> = Port number
	<pattern> = <CHARACTER PROGRAM DATA> PRBS9:PRBS9 pattern PRBS11:PRBS11 pattern PRBS15:PRBS15 pattern PRBS20:PRBS20 pattern PRBS23:PRBS23 pattern PRBS29:PRBS29 pattern PRBS31:PRBS31 pattern USER32BIT: User Pattern (32bit) USER2048BIT: User Pattern (2048bit) *DEFault = PRBS31*
Response	None
Example	`OTN:RX1:PATT PRBS31`
Note	

Syntax	OTN:RX<Pt>:PATTern?
Description	This query returns the pattern type.
Parameter	<Pt> = Port number
Response	<pattern> = <CHARACTER RESPONSE DATA>
Example	`OTN:RX1:PATT?` → `PRBS31`
Note	

15.2.4 OTN:RX<Pt>:PINVersion

Syntax	OTN:RX<Pt>:PINVersion <inverted>
Description	This command enables or disables PRBS pattern inversion.
Parameters	<Pt> = Port number
	<inverted> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None
Example	`OTN:RX1:PINV ON`
Note	

Syntax	OTN:RX<Pt>:PINVersion?
Description	This query returns the inversion state (enabled/disabled) of the PRBS pattern.
Parameter	<Pt> = Port number
Response	<inverted> = <BOOLEAN RESPONSE DATA>
Example	`OTN:RX1:PINV?` → `1`
Note	

15.2.5 OTN:RX<Pt>:UP32

Syntax	OTN:RX<Pt>:UP32 <pattern>
Description	This command sets the 32 bit wide variable length user defined pattern used when `PATTern` is `USER32BIT`.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use binary digits '0' and '1' to describe the pattern. The string must consist of 1 to 32 characters (one bit resolution).
Response	None
Example	`OTN:RX1:UP32 "01101"`
Note	

Syntax	OTN:RX<Pt>:UP32?
Description	This query returns the 32 bit wide user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`OTN:RX1:UP32?` → `"01101"`
Note	

15.2.6 OTN:RX<Pt>:UP2K

Syntax	OTN:RX<Pt>:UP2K <pattern>
Description	This command sets the 2048 bit wide user defined pattern.
Parameters	<Pt> = Port number
	<pattern> = <STRING PROGRAM DATA> Use hexadecimal upper or lower case. The string must consist of 2 to 512 characters (one byte resolution).
Response	None.
Example	`OTN:RX1:UP2K "12DF"`
Notes	The pattern is padded with zeros until it is a multiple of eight bits long. In effect when `OTN:RX1:PATT` is `USER2048BIT`

Syntax	OTN:RX<Pt>:UP2K?
Description	This query returns the 2048 bit user defined pattern.
Parameter	<Pt> = Port number
Response	<pattern> = <STRING RESPONSE DATA>
Example	`OTN:RX1:UP2K?` → `"12DF"`
Note	

15.2.7 OTN:RX<Pt>:MAPPing:INPutsignal

Syntax	OTN:RX<Pt>:MAPPing:INPutsignal <signal>
Description	This command sets OTN input signal.
Parameters	<Pt> = Port number
	<signal> = <CHARACTER PROGRAM DATA> OTU1 OTU2 OTU2EXTOPU2 OTU1E OTU2E OTU1F OTU2F OTU3 OTU3E1 OTU3E2 OTU4 *DEFault = OTU2*
Response	None
Example	`OTN:RX1:MAPP:INP OTU2`
Note	

Syntax	OTN:RX<Pt>:MAPPing:INPutsignal?
Description	This query returns OTN input signal.
Parameter	<Pt> = Port number
Response	<signal> = <CHARACTER RESPONSE DATA>
Example	`OTN:RX1:MAPP:INP?` → `OTU2`
Note	

15.2.8 OTN:RX<Pt>:MAPPing:MULTiplexing<stage>

Syntax	OTN:RX<Pt>:MAPPing:MULTiplexing<stage> <odutype>
Description	This command sets the ODU type of each multiplexing stage.
Parameters	<Pt> = Port number
	<stage> = Multiplexing stage (1-3)
	<odutype> = <CHARACTER PROGRAM DATA> NONE: None ODTU01: ODTU01 ODTU12PT20: ODTU12 (PT=20) ODTU12PT21: ODTU12 (PT=21) ODTU21: ODTU2.1 ODTU2TS: ODTU2.ts ODTU13PT20: ODTU13 (PT=20) ODTU13PT21: ODTU13 (PT=21) ODTU23PT20: ODTU23 (PT=20) ODTU23PT20EXTOPU2: ODTU23 (PT=20)(Ext. OPU2) ODTU23PT21: ODTU23 (PT=21) ODTU23PT21EXTOPU2: ODTU23 (PT=21)(Ext. OPU2) ODTU31: ODTU3.1 ODTU3E28: ODTU3e2.8 ODTU2E3E1: ODTU2e3e1 ODTU3TS: ODTU3.ts ODTU41: ODTU4.1 ODTU42: ODTU4.2 ODTU48ODU2: ODTU4.8 (ODU2) ODTU48EXTOPU2: ODTU4.8 (Ext. OPU2) ODTU48ODU2E: ODTU4.8 (ODU2e) ODTU431: ODTU4.31 ODTU4TS: ODTU4.ts *DEFault = NONE*
Response	None
Example	`OTN:RX1:MAPP:MULT1 NONE`
Note	

Syntax	OTN:RX<Pt>:MAPPing:MULTiplexing<stage>?
Description	This query returns the ODU type of each multiplexing stage.
Parameter	<Pt> = Port number
	<stage> = Multiplexing stage (1-3)
Response	<odutype> = <CHARACTER RESPONSE DATA>
Example	`OTN:RX1:MAPP:MULT1?` → `NONE`
Note	

15.2.9 OTN:RX<Pt>:FEC

Syntax	OTN:RX<Pt>:FEC <fec>
Description	This command sets OTU FEC control.
Parameters	<Pt> = Port number
	<fec> = <CHARACTER PROGRAM DATA> NOFec: No FEC RS: RS(255,239) *DEFault = RS*
Response	None
Example	`OTN:RX1:FEC RS`
Note	

Syntax	OTN:RX<Pt>:FEC?
Description	This query returns OTU FEC control.
Parameter	<Pt> = Port number
Response	<fec> = <CHARACTER RESPONSE DATA>
Example	`OTN:RX1:FEC?` → `RS`
Note	

15.2.10 OTN:RX<Pt>:FOLLow

Syntax	OTN:RX<Pt>:FOLLow <follow>
Description	This command sets the receiver setup to follow another setup or not to follow.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> NONE: Do not follow. TX: Follows setup of the Tx port. RX1: Follows setup of the Rx port1. *DEFault = TX*
Response	None.
Example	`OTN:RX1:FOLL TX`
Note	

Syntax	OTN:RX<Pt>:FOLLow?
Description	This query returns the receiver setup to follow another setup or not to follow.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`OTN:RX1:FOLL?` → `TX`
Note	

15.2.11 OTN:RX<Pt>:TP

Syntax	OTN:RX<Pt>:TP <route>,<tp>
Description	This command sets the TP to be used as the main received channel.
Parameters	<Pt> = Port number
	<route> = <CHARACTER PROGRAM DATA> ODTU01: ODTU01 ODTU12: ODTU12 ODTU21: ODTU2.1 ODTU2TS: ODTU2.ts ODTU13: ODTU13 ODTU23: ODTU23 ODTU31: ODTU3.1 ODTU3E28: ODTU3e2.8 ODTU2E3E1: ODTU2e3e1 ODTU3TS: ODTU3.ts ODTU41: ODTU4.1 ODTU42: ODTU4.2 ODTU48: ODTU4.8 ODTU431: ODTU4.31 ODTU4TS: ODTU4.ts
	<tp> = <NUMERIC PROGRAM DATA> ODTU01: TP: 1-2 ODTU12(PT=0x20) TP: 1-4 ODTU12(PT=0x21) TP: 1-4 ODTU21: TP: 1-8 ODTU2TS: TP: 1-8 ODTU13(PT=0x20) TP: 1-16

continued on next page...

. . . continued from previous page

	ODTU13(PT=0x21) TP: 1-16 ODTU23(PT=0x20) TP: 1-4 ODTU23(PT=0x21) TP: 1-4 ODTU31 TP: 1-32 ODTU3E28 TP: 1-4 ODTU2E3E1 TP: 1-32 ODTU3TS TP: 1-32 ODTU41 TP: 1-80 ODTU42 TP: 1-80 ODTU48 TP: 1-80 ODTU431 TP: 1-80 ODTU4TS TP: 1-80
Response	None
Example	`OTN:RX1:TP ODTU01,1`
Note	

Syntax	OTN:RX<Pt>:TP? <route>
Description	This query returns the TP to be used as the main received channel.
Parameters	<Pt> = Port number
	<route> = <CHARACTER PROGRAM DATA> ODTU01: ODTU01 ODTU12: ODTU12 ODTU21: ODTU2.1 ODTU2TS: ODTU2.ts ODTU13: ODTU13 ODTU23: ODTU23 ODTU31: ODTU3.1 ODTU3E28: ODTU3e2.8 ODTU2E3E1: ODTU2e3e1 ODTU3TS: ODTU3.ts ODTU41: ODTU4.1 ODTU42: ODTU4.2 ODTU48: ODTU4.8 ODTU431: ODTU4.31 ODTU4TS: ODTU4.ts
Response	<tp> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:RX1:TP? ODTU01` → `1`
Note	

15.2.12 OTN:RX<Pt>:TS

Syntax	OTN:RX<Pt>:TS <route>,<ts>
Description	This command sets a set of TS(s) to be used as the main received channel.
Parameters	<Pt> = Port number
	<route> = <CHARACTER PROGRAM DATA> ODTU01: ODTU01 ODTU12: ODTU12 ODTU21: ODTU2.1 ODTU2TS: ODTU2.ts ODTU13: ODTU13 ODTU23: ODTU23 ODTU31: ODTU3.1 ODTU3E28: ODTU3e2.8 ODTU2E3E1: ODTU2e3e1 ODTU3TS: ODTU3.ts ODTU41: ODTU4.1

continued on next page. . .

. . . continued from previous page

	ODTU42: ODTU4.2 ODTU48: ODTU4.8 ODTU431: ODTU4.31 ODTU4TS: ODTU4.ts
	(<ts> {,<ts>}) = <EXPRESSION PROGRAM DATA> The format is listed for each parameter.
Response	None
Example	`OTN:RX1:TS ODTU12,(1,2)`
Note	ODTU01 TS : 1-2 will become the same value as TP. ODTU12 (PT=0x20) TS : 1-4 will become the same value as TP. ODTU12 (PT=0x21) TS1: 1-8, TS2: 1-8 TS2 must not overlap and TS1. ODTU2.1 TS: 1-8 ODTU2TS TS: 1-8 ODTU13(PT=0x20) TS: will become the same value as TP. ODTU13(PT=0x21) TS: 1-32 ODTU23(PT=0x20) TS: 1-16 ODTU23(PT=0x21) TS: 1-32 ODTU31 TS: 1-32 ODTU3E28 TS: 1-32 ODTU2E3E1 TS: 1-16 ODTU3TS TS: 1-32 ODTU41 TS: 1-80 ODTU42 TS: 1-80 ODTU48 TS: 1-80 ODTU431 TS: 1-80 ODTU4TS TS: 1-80

Syntax	OTN:RX<Pt>:TS? <route>
Description	This command returns a set of TS(s) to be used as the main received channel.
Parameters	<Pt> = Port number
	<route> = <CHARACTER PROGRAM DATA> ODTU01: ODTU01 ODTU12: ODTU12 ODTU21: ODTU2.1 ODTU2TS: ODTU2.ts ODTU13: ODTU13 ODTU23: ODTU23 ODTU31: ODTU3.1 ODTU3E28: ODTU3e2.8 ODTU2E3E1: ODTU2e3e1 ODTU3TS: ODTU3.ts ODTU41: ODTU4.1 ODTU42: ODTU4.2 ODTU48: ODTU4.8 ODTU431: ODTU4.31 ODTU4TS: ODTU4.ts
Response	(<ts> {,<ts>}) = <EXPRESSION RESPONSE DATA> The response format is listed for each parameter.
Example	`OTN:RX1:TS? ODTU12` → `1,2`
Note	

15.2.13 OTN:RX<Pt>:ODUFlex:TS

Syntax	OTN:RX<Pt>:ODUFlex:TS <ts>
Description	This command sets the size of a TS set to be used in ODUflex.
Parameters	<Pt> = Port number

continued on next page. . .

. . . continued from previous page

	<ts> = <NUMERIC PROGRAM DATA> ODTU2.ts: PRBS/Ethernet: 1-8 ODTUk.ts: FC400: 4 ODTUk.ts: FC800: 7 ODTUk.ts - 3072M: 3 ODTUk.ts - 4915M: 4 ODTUk.ts - 6144M: 5 ODTUk.ts - 9830M: 8 ODTU3.ts - PRBS: 1-32 ODTU4.ts - PRBS: 1-80 ODTU3.ts - Ethernet: 1-32 ODTU4.ts - Ethernet: 1-80 ODTU3.ts - 10137M: 9 ODTU4.ts - 10137M: 8
Response	None
Example	`OTN:RX1:ODUF:TS 1`
Note	

Syntax	OTN:RX<Pt>:ODUFlex:TS?
Description	This command returns the size of a TS set to be used in ODUflex.
Parameter	<Pt> = Port number
Response	<ts> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:RX1:ODUF:TS?` → `1`
Note	

15.2.14 OTN:RX<Pt>:TPTS:DETect

Syntax	OTN:RX<Pt>:TPTS:DETect <route>,<mode>
Description	This commands sets the TP and TS detection mode.
Parameters	<Pt> = Port number
	<route> = <CHARACTER PROGRAM DATA> ODTU01: ODTU01 ODTU12: ODTU12 ODTU21: ODTU2.1 ODTU2TS: ODTU2.ts ODTU13: ODTU13 ODTU23: ODTU23 ODTU31: ODTU3.1 ODTU3E28: ODTU3e2.8 ODTU2E3E1: ODTU2e3e1 ODTU3TS: ODTU3.ts ODTU41: ODTU4.1 ODTU42: ODTU4.2 ODTU48: ODTU4.8 ODTU431: ODTU4.31 ODTU4TS: ODTU4.ts
	<mode> = <CHARACTER PROGRAM DATA> MANual: Manual TPDetect: Auto Detect TP TSDetect: Auto Detect TS *DEFault = MANual*
Response	None
Example	`OTN:RX1:TPTS:DET ODTU12,MAN`
Note	

Syntax	OTN:RX<Pt>:TPTS:DETect? <route>
Description	This query returns the TP and TS detection mode.
Parameters	<Pt> = Port number
	<route> = <CHARACTER PROGRAM DATA> ODTU01: ODTU01 ODTU12: ODTU12 ODTU21: ODTU2.1 ODTU2TS: ODTU2.ts ODTU13: ODTU13 ODTU23: ODTU23 ODTU31: ODTU3.1 ODTU3E28: ODTU3e2.8 ODTU2E3E1: ODTU2e3e1 ODTU3TS: ODTU3.ts ODTU41: ODTU4.1 ODTU42: ODTU4.2 ODTU48: ODTU4.8 ODTU431: ODTU4.31 ODTU4TS: ODTU4.ts
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`OTN:RX1:TPTS:DET? ODTU12` → `MAN`
Note	

15.2.15 OTN:RX<Pt>:MEAS:TTI:DET

Syntax	OTN:RX<Pt>:MEAS:TTI:DET <type>,<detection>
Description	This command sets TTI detection type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> SM PM TCM1 TCM2 TCM3 TCM4 TCM5 TCM6
	<detection> = <CHARACTER PROGRAM DATA> OFF: OFF SAPI: SAPI DAPI: DAPI SDAPI: SAPI and DAPI ALL: SAPI, DAPI and Operator Code *DEFault = OFF*
Response	None
Example	`OTN:RX1:MEAS:TTI:DET TCM1,SDAPI`
Note	

Syntax	OTN:RX<Pt>:MEAS:TTI:DET? <type>
Description	This query returns TTI detection type.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> SM PM TCM1 TCM2 TCM3 TCM4 TCM5 TCM6
Response	<detection> = <CHARACTER RESPONSE DATA>
Example	`OTN:RX1:MEAS:TTI:DET? TCM1` → `SDAPI`
Note	

15.3 Measurement Setup

15.3.1 OTN:TX<Pt>:MEAS:TTI:ENCoding

Syntax	OTN:TX<Pt>:MEAS:TTI:ENCoding <encoding>
Description	This command sets the TTI encoding standard.
Parameters	<Pt> = Port number
	<encoding> = <CHARACTER PROGRAM DATA> ITUT: ITU-T ANSI: ANSI *DEFault = ITUT*
Response	None
Example	`OTN:TX1:MEAS:TTI:ENC ITUT`
Note	

Syntax	OTN:TX<Pt>:MEAS:TTI:ENCoding?
Description	This query returns the TTI encoding standard.
Parameter	<Pt> = Port number
Response	<encoding> = <CHARACTER PROGRAM DATA>
Example	`OTN:TX1:MEAS:TTI:ENC?` → `ITUT`
Note	

15.3.2 OTN:RX<Pt>:MEAS:TCM:ENABle

Syntax	OTN:RX<Pt>:MEAS:TCM:ENABle <enable>
Description	This command enables or disables TCM measurement.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None
Example	`OTN:RX1:MEAS:TCM:ENAB ON`
Note	

Syntax	OTN:RX<Pt>:MEAS:TCM:ENABle?
Description	This query returns whether or not TCM measurement is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`OTN:RX1:MEAS:TCM:ENAB?` → `1`
Note	

15.3.3 OTN:RX<Pt>:MEASurement:PLM

Syntax	OTN:RX<Pt>:MEASurement:PLM <enable>
Description	This command enables or disables PLM measurement.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None
Example	`OTN:RX1:MEAS:PLM ON`
Note	

Syntax	OTN:RX<Pt>:MEASurement:PLM?
Description	This query returns whether or not PLM measurement is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`OTN:RX1:MEAS:PLM? → 1`
Note	

15.3.4 MEASurement:SETup:OTN:RX<Pt>:MSIMdetection<stage>:SETup

Syntax	MEASurement:SETup:OTN:RX<Pt>:MSIMdetection<stage>:SETup <type>
Description	This command sets the MSIM detection type.
Parameters	<Pt> = Port number
	<stage> = Multiplexing stage (1-3)
	<type> = <CHARACTER PROGRAM DATA> NONE: None TXDATA: Use Tx OH preset PSI data RECEIVED: Use received PSI data *DEFault = NONE*
Response	None
Example	`MEAS:SET:OTN:RX1:MSIM1:SET NONE`
Note	

Syntax	MEASurement:SETup:OTN:RX<Pt>:MSIMdetection<stage>:SETup?
Description	This query returns the MSIM detection type.
Parameters	<Pt> = Port number
	<stage> = Multiplexing stage (1-3)
Response	<type> = <CHARACTER PROGRAM DATA>
Example	`MEAS:SET:OTN:RX1:MSIM1:SET? → NONE`
Note	

15.3.5 MEASurement:SETup:OTN:RX<Pt>:MSIMdetection<stage>:MSIM?

Syntax	MEASurement:SETup:OTN:RX<Pt>:MSIMdetection<stage>:MSIM?
Description	This query returns the expected MSIM detection pattern.
Parameters	<Pt> = Port number
	<stage> = Multiplexing stage (1-3)
Response	<value> = <HEXADECIMAL NUMERIC RESPONSE DATA> The MSIM detection pattern varies depending on the :MSIMdetection<stage>:SETup and the ODU type. ODTU01: PSI[2:3] ODTU12(PT20): PSI[2:5] ODTU12(PT21): PSI[2:9] ODTU21: PSI[2:9] ODTU13(PT20): PSI[2:17] ODTU13(PT21): PSI[2:33] ODTU23(PT20): PSI[2:17] ODTU23(PT20): PSI[2:33] ODTU31: PSI[2:33] ODTU3E28: PSI[2:33] ODTU2E3E1: PSI[2:17] ODTU3TS: PSI[2:33] ODTU41: PSI[2:81] ODTU42: PSI[2:81] ODTU48: PSI[2:81] ODTU431: PSI[2:81] ODTU4TS: PSI[2:81]
Example	`MEAS:SET:OTN:RX1:MSIM1:MSIM?` → `0`
Note	

15.3.6 MEASurement:SETup:PERFormance:OTN:RX<Pt>:PARameter

Syntax	MEASurement:SETup:PERFormance:OTN:RX<Pt>:PARameter <param>
Description	This command sets the performance parameter for OTN. Enables/Disables the G.8201 or M.2401 performance evaluation.
Parameters	<Pt> = Port number <param> = <CHARACTER PROGRAM DATA> G8201: G.8201 M2401: M.2401/M.2110 *DEFault = M2401*
Response	None
Example	`MEAS:SET:PERF:OTN:RX1:PAR G8201`
Note	

Syntax	MEASurement:SETup:PERFormance:OTN:RX<Pt>:PARameter?
Description	This query returns the performance parameter for OTN.
Response	<param> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:PERF:OTN:RX1:PAR?` → `G8201`
Note	

15.3.7 MEASurement:SETup:PERFormance:OTN:RX<Pt>:ALLocation

Syntax	MEASurement:SETup:PERFormance:OTN:RX<Pt>:ALLocation <path>,<percentage>
Description	This command sets the allocation percentage for M.2401/M.2110.
Parameters	<Pt> = Port number
	<path> = <CHARACTER PROGRAM DATA> OTU: ODU: TCMi: i = 1 - 6 *DEFault = OTU*
	<number> = <NUMERIC PROGRAM DATA> *MINimum=0.00, MAXimum=100.00, DEFault=100.00*
Response	None
Example	`MEAS:SET:PERF:OTN:RX1:ALL OTU,0`
Note	

Syntax	MEASurement:SETup:PERFormance:OTN:RX<Pt>:ALLocation? <path>
Description	This query returns the allocation percentage for M.2401/M.2110.
Parameters	<Pt> = Port number
	<path> = <CHARACTER RESPONSE DATA>
Response	<percentage> = <NR2 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:PERF:OTN:RX1:ALL? OTU` → `0.00`
Note	

15.3.8 MEASurement:SETup:PERFormance:OTN:TPERiod

Syntax	MEASurement:SETup:PERFormance:OTN:TPERiod <period>
Description	This command sets the evaluation period for OTN. For M.2401/M.2110.
Parameter	<period> = <CHARACTER PROGRAM DATA> 15M: 15 minutes 1H: 1 hour 2H: 2 hours 24H: 24 hours 7D: 7 days *DEFault = 15M*
Response	None
Example	`MEAS:SET:PERF:OTN:TPER 15M`
Note	

Syntax	MEASurement:SETup:PERFormance:OTN:TPERiod?
Description	This query returns the evaluation item for OTN. For M.2401/M.2110.
Response	<period> = <CHARACTER RESPONSE DATA> 15M: 15 minutes 1H: 1 hour 2H: 2 hours 24H: 24 hours 7D: 7 days
Example	`MEAS:SET:PERF:OTN:TPER?` → `15M`
Note	

15.3.9 MEASurement:SETup:PERFormance:OTN:RX<Pt>:OBJectives

Syntax	MEASurement:SETup:PERFormance:OTN:RX<Pt>:OBJectives <sesr>,<bber>
Description	This command sets the Performance Objectives.
Parameters	<Pt> = Port number
	<sesr> = <NUMERIC PROGRAM DATA> 0.01E-8 to 9.99E-1, default is 1.00E-1
	<bber> = <NUMERIC PROGRAM DATA> 0.01E-8 to 9.99E-1, default is 1.00E-1
Response	None
Example	`MEAS:SET:PERF:OTN:RX1:OBJ 1.00E-1,1.00E-1`
Note	

Syntax	MEASurement:SETup:PERFormance:OTN:RX<Pt>:OBJectives?
Description	This query returns the Performance Objectives.
Parameter	<Pt> = Port number
Responses	<sser> = <NR3 NUMERIC RESPONSE DATA>
	<bber> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:PERF:OTN:RX1:OBJ?` → `1.00E-1,1.00E-1`
Note	

15.4 Result

15.4.1 OTN:RX<Pt>:IFETch?

Syntax	OTN:RX<Pt>:IFETch? <parameter>
Description	This query returns the results of item specified by the parameter.
Parameters	<Pt> = Port number
	({<parameter>} + {,}*) = <EXPRESSION PROGRAM DATA> The response format is listed for each parameter. **LLD**[1] LOFLLD: LOF Lane OOFLLD: OOF FASLLD: FAS LLD MFASLLD: MFAS LLD LLMLLD: LLM LLD LOR: LOR OOR: OOR OLA: OLA MMAP: Marker Map RSKEW: Relative Skew **OTU** LOS: LOS LOF: LOF LOM: LOM OOF: OOF FAS: FAS OOM: OOM MFAS: MFAS FECC: FEC-Correctable FECU: FEC-Uncorrectable SMTIM: SM-TIM SMBIAE: SM-BIAE SMBDI: SM-BDI SMIAE: SM-IAE SMBIP8: SM-BIP8 SMBEI: SM-BEI AIS: AIS **Stage n ODU (n = 0-3)** SnLOFLOM: Stage n LOFLOM [2] SnOOF: Stage n OOF SnFAS: Stage n FAS SnOOM: Stage n OOM SnMFAS: Stage n MFAS SnAIS: Stage n AIS SnOCI: Stage n OCI SnLCK: Stage n LCK SnPMTIM: Stage n PM-TIM SnPMBDI: Stage n PM-BDI SnPMBIP8: Stage n PM-BIP8 SnPMBEI: Stage n PM-BEI SnPLM: Stage n PLM SnFSF: Stage n FSF SnFSD: Stage n FSD SnBSF: Stage n BSF SnBSD: Stage n BSD

continued on next page...

. . . continued from previous page

	SnMSIM: Stage n MSIM [2] **Client** AISC: Client-AIS CSF: CSF LSS: LSS PRBSBIT: PRBS Bit Error CFREQ: Client frequency COFFS: Client frequency offset **High-order ODU TCM (i = 1-6)** TCMiTIM: TCMi-TIM TCMiBIAE: TCMi-BIAE TCMiBDI: TCMi-BDI TCMiIAE: TCMi-IAE TCMiLTC: TCMi-LTC TCMiBIP8: TCMi-BIP8 TCMiBEI: TCMi-BEI **Stage n Justification (n = 0-2)** SnFREQ: Stage n Frequency deviation SnINC: Stage n Inc SnINC1: Stage n Inc >1 SnINC2: Stage n Inc >2 SnDEC: Stage n Dec SnDEC1: Stage n Dec >1 SnDEC2: Stage n Dec >2 SnINCO: Stage n Inc Over SnDECO: Stage n Dec Over SnCRC8: Stage n CRC8 Error SnCRC5: Stage n CRC5 Error SnCMMIN: Stage n Cm(t) Minimum SnCMMAX: Stage n Cm(t) Maximum SnPJC: Stage n PJC SnPJC2: Stage n 2PJC SnPJC3: Stage n 3PJC SnNJC: Stage n NJC SnNJC2: Stage n 2NJC **GFP** GTFR: GFP-F Total Frame GIFR: GFP-F Idle Frame GCFR: GFP-F Client Frame GSSF: GFP-F SSF GCMFSIG: GFP-F CMF Loss of Signal GCMFSYNC: GFP-F CMF Loss of Sync GFCS: GFP-F FCS Error GCHECC: GFP-F/T cHEC correctable errors GCHECU: GFP-F/T cHEC uncorrectable errors GTHECC: GFP-F/T tHEC correctable errors GTHECU: GFP-F/T tHEC uncorrectable errors GEHECC: GFP-F eHEC correctable errors GEHECU: GFP-F eHEC uncorrectable errors GIGFR: GFP-T Invalid GFP Frame GSCRC: GFP-T Superblock CRC Error GCSFSIG: GFP-T CSF Signal GCSFSYNC: GFP-T CSF Sync

continued on next page. . .

. . . continued from previous page

	GPTI: GFP-F PTI Mismatch GUPI: GFP-F UPI Mismatch GTTFR: GFP-F Tx Total Frame GTIFR: GFP-F Tx Idle Frame GTCFR: GFP-F Tx Client Frame GTCMF: GFP-F Tx CMF **OTU Performance** NSTAT, FSTAT: Near-end/Far-end Status. Response: <STRING> NSES, FSES: Near-end/Far-end SES. Response: <Count>,<Ratio%> NBBE, FBBE: Near-end/Far-end BBE. Response: <Count>,<Ratio%> NUAS, FUAS: Near-end/Far-end UAS. Response: <Count> **Stage n ODU Performance (n = 0-3)** SnNSTAT, SnFSTAT, SnBSTAT: Near-end/Far-end/Bidirectional Status. Response: <STRING> SnNSES, SnFSES, SnBSES: Near-end/Far-end/Bidirectional SES. Response: <Count>,<Ratio%> SnNBBE, SnFBBE, SnBBBE: Near-end/Far-end/Bidirectional BBE. Response: <Count>,<Ratio%> SnNUAS, SnFUAS, SnBUAS: Near-end/Far-end/Bidirectional UAS. Response: <Count> **High-order ODU TCMi Performance (i = 1-6)** TCMiNSTAT, TCMiFSTAT, TCMiBSTAT: Near-end/Far-end/Bidirectional Status. Response: <STRING> TCMiNSES, TCMiFSES, TCMiBSES: Near-end/Far-end/Bidirectional SES. Response: <Count>,<Ratio%> TCMiNBBE, TCMiFBBE, TCMiBBBE: Near-end/Far-end/Bidirectional BBE. Response: <Count>,<Ratio%> TCMiNUAS, TCMiFUAS, TCMiBUAS: Near-end/Far-end/Bidirectional UAS. Response: <Count>
Response	{(<result>),}* = <EXPRESSION RESPONSE DATA> Format: Numeric List Each result is formatted according to the specification in the parameter field. Values that are not relevant or applicable for the current measurement return NaN (section 1.6.1).
Example	`OTN:RX1:IFET? (LOS,LOF,OOF)` → `(3,0.00532),(4,0.00709),(5,0.00887)`
Notes	This command fetches the results from the interval selected using the `MEASurement:SETup:SELect` command (see section 18.2.2). [1] Results are only available at OTU3/4. [2] This parameter is not available at stage 0. If the requested result is not available, NaN (section 1.6.1) is returned. If there is one or more results, the last ”,” is always removed.

15.5 Status

15.5.1 OTN:STATus:RX<Pt>:AESummary[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:AESummary[:EVENt]?
Description	This query returns the alarms and errors summary event register. The content of this event register is summarized in DB8 of the STATus:INTerface:PORT<Pt>:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Alarm section 1 summary DB2 (2) Alarm section 2 summary DB3 (4) Alarm section 3 summary DB4 (8) Alarm section 4 summary DB5 (16) Alarm section 5 summary DB6 (32) Alarm section 6 summary DB7 (64) Alarm section 7 summary DB8 (128) Alarm section 8 summary DB9 (256) Alarm section 9 summary DB10 (512) Alarm section 10 summary DB11 (1024) Error section 1 summary DB12 (2048) Error section 2 summary DB13 (4096) Error section 3 summary DB14 (8192) Error section 4 summary DB15 (16384) Error section 5 summary DB16 (32768) Error section 6 summary
Example	`OTN:STAT:RX1:AES?` → `65`
Note	This command is for compatibility of V2.xx or V1.xx. Following commands are endorsed on V3.00 or later. OTN:STATus:RX<Pt>:MSTage:AESummary[:EVENt]?

15.5.2 OTN:STATus:RX<Pt>:AESummary:CONDition?

Syntax	OTN:STATus:RX<Pt>:AESummary:CONDition?
Description	This query returns alarms and errors summary condition register. DB1-DB10 is summary of OTN:STATus:RX<Pt>:ALARm<section>[:EVENt]? DB11-DB16 is summary of OTN:STATus:RX<Pt>:ERRor<section>[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Alarm section 1 summary DB2 (2) Alarm section 2 summary DB3 (4) Alarm section 3 summary DB4 (8) Alarm section 4 summary DB5 (16) Alarm section 5 summary DB6 (32) Alarm section 6 summary DB7 (64) Alarm section 7 summary DB8 (128) Alarm section 8 summary DB9 (256) Alarm section 9 summary DB10 (512) Alarm section 10 summary DB11 (1024) Error section 1 summary DB12 (2048) Error section 2 summary DB13 (4096) Error section 3 summary DB14 (8192) Error section 4 summary DB15 (16384) Error section 5 summary DB16 (32768) Error section 6 summary
Example	`OTN:STAT:RX1:AES:COND?` → `1024`
Note	

continued on next page...

. . . continued from previous page

Note	This command is for compatibility of V2.xx or V1.xx. Following commands are endorsed on V3.00 or later. OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition?

15.5.3 OTN:STATus:RX<Pt>:ALARm<section>[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:ALARm<section>[:EVENt]?
Description	This query returns the alarms event register. The content of this register is summarized in DB1 through DB10 of the OTN:STATus:RX<Pt>:AESummary:CONDition register.
Parameters	<Pt> = Port number
	<section> = Section number (1-10)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> See Table 15.18
Example	`OTN:STAT:RX1:ALAR5?` → `92`
Note	This command is for compatibility of V2.xx or V1.xx. Following commands are endorsed on V3.00 or later. OTN:STATus:RX<Pt>:MSTage:AESummary[:EVENt]? OTN:STATus:RX<Pt>:OTU:ASUMmary[:EVENt]? OTN:STATus:RX<Pt>:SONE:ASUMmary[:EVENt]? OTN:STATus:RX<Pt>:STWO:ASUMmary[:EVENt]? OTN:STATus:RX<Pt>:STHRee:ASUMmary[:EVENt]? OTN:STATus:RX<Pt>:CLIent:ASUMmary[:EVENt]? OTN:STATus:RX<Pt>:OTU:ALARm<section>[:EVENt]? OTN:STATus:RX<Pt>:SONE:ALARm<section>[:EVENt]? OTN:STATus:RX<Pt>:STWO:ALARm<section>[:EVENt]? OTN:STATus:RX<Pt>:STHRee:ALARm<section>[:EVENt]? OTN:STATus:RX<Pt>:CLIent:ALARm<section>[:EVENt]?

15.5.4 OTN:STATus:RX<Pt>:ALARm<section>:CONDition?

Syntax	OTN:STATus:RX<Pt>:ALARm<section>:CONDition?
Description	This query returns the alarms condition register query.
Parameters	<Pt> = Port number
	<section> = Section number (1-10)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> See Table 15.18
Example	`OTN:STAT:RX1:ALAR5:COND?` → `0`
Note	This command is for compatibility of V2.xx or V1.xx. Following commands are endorsed on V3.00 or later. OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition? OTN:STATus:RX<Pt>:OTU:ASUMmary:CONDition? OTN:STATus:RX<Pt>:SONE:ASUMmary:CONDition? OTN:STATus:RX<Pt>:STWO:ASUMmary:CONDition? OTN:STATus:RX<Pt>:STHRee:ASUMmary:CONDition? OTN:STATus:RX<Pt>:CLIent:ASUMmary:CONDition? OTN:STATus:RX<Pt>:OTU:ALARm<section>:CONDition? OTN:STATus:RX<Pt>:SONE:ALARm<section>:CONDition? OTN:STATus:RX<Pt>:STWO:ALARm<section>:CONDition? OTN:STATus:RX<Pt>:STHRee:ALARm<section>:CONDition? OTN:STATus:RX<Pt>:CLIent:ALARm<section>:CONDition?

Table 15.18: Alarm Items

OTU/ODU **Section**	1	2	3	4
DB1 (1)	LOS	TCM1-TIM	TCM5-BDI	Client-AIS
DB2 (2)	OTU-AIS	TCM2-TIM	TCM6-BDI	CSF
DB3 (4)	LOF	TCM3-TIM	TCM1-IAE	LSS
DB4 (8)	OOF	TCM4-TIM	TCM2-IAE	LOF-LLD
DB5 (16)	LOM	TCM5-TIM	TCM3-IAE	OOF-LLD
DB6 (32)	OOM	TCM6-TIM	TCM4-IAE	LOR-LLD
DB7 (64)	SM-TIM	TCM1-BIAE	TCM5-IAE	OOR-LLD
DB8 (128)	SM-BIAE	TCM2-BIAE	TCM6-IAE	Not used
DB9 (256)	SM-BDI	TCM3-BIAE	TCM1-LTC	Not used
DB10 (512)	SM-IAE	TCM4-BIAE	TCM2-LTC	Not used
DB11 (1024)	ODU-AIS	TCM5-BIAE	TCM3-LTC	Not used
DB12 (2048)	ODU-LCK	TCM6-BIAE	TCM4-LTC	Not used
DB13 (4096)	ODU-OCI	TCM1-BDI	TCM5-LTC	Not used
DB14 (8192)	PM-TIM	TCM2-BDI	TCM6-LTC	Not used
DB15 (16384)	PM-BDI	TCM3-BDI	PLM	Not used
DB16 (32768)		TCM4-BDI	MSIM	Not used
Stage 1 **Section**	5	6	7	
DB1 (1)	LOFLOM	TCM3-BIAE	TCM1-LTC	
DB2 (2)	OOF	TCM4-BIAE	TCM2-LTC	
DB3 (4)	OOM	TCM5-BIAE	TCM3-LTC	
DB4 (8)	ODU-AIS	TCM6-BIAE	TCM4-LTC	
DB5 (16)	ODU-LCK	TCM1-BDI	TCM5-LTC	
DB6 (32)	ODU-OCI	TCM2-BDI	TCM6-LTC	
DB7 (64)	PM-TIM	TCM3-BDI	PLM	
DB8 (128)	PM-BDI	TCM4-BDI	Client-AIS	
DB9 (256)	TCM1-TIM	TCM5-BDI	CSF	
DB10 (512)	TCM2-TIM	TCM6-BDI	MSIM	
DB11 (1024)	TCM3-TIM	TCM1-IAE	Not used	
DB12 (2048)	TCM4-TIM	TCM2-IAE	Not used	
DB13 (4096)	TCM5-TIM	TCM3-IAE	Not used	
DB14 (8192)	TCM6-TIM	TCM4-IAE	Not used	
DB15 (16384)	TCM1-BIAE	TCM5-IAE	Not used	
DB16 (32768)	TCM2-BIAE	TCM6-IAE	Not used	
Stage 2 **Section**	8	9	10	
DB1 (1)	LOFLOM	TCM3-BIAE	TCM1-LTC	
DB2 (2)	OOF	TCM4-BIAE	TCM2-LTC	
DB3 (4)	OOM	TCM5-BIAE	TCM3-LTC	
DB4 (8)	ODU-AIS	TCM6-BIAE	TCM4-LTC	
DB5 (16)	ODU-LCK	TCM1-BDI	TCM5-LTC	
DB6 (32)	ODU-OCI	TCM2-BDI	TCM6-LTC	
DB7 (64)	PM-TIM	TCM3-BDI	PLM	
DB8 (128)	PM-BDI	TCM4-BDI	Client-AIS	
DB9 (256)	TCM1-TIM	TCM5-BDI	CSF	
DB10 (512)	TCM2-TIM	TCM6-BDI	Not used	
DB11 (1024)	TCM3-TIM	TCM1-IAE	Not used	
DB12 (2048)	TCM4-TIM	TCM2-IAE	Not used	
DB13 (4096)	TCM5-TIM	TCM3-IAE	Not used	
DB14 (8192)	TCM6-TIM	TCM4-IAE	Not used	
DB15 (16384)	TCM1-BIAE	TCM5-IAE	Not used	
DB16 (32768)	TCM2-BIAE	TCM6-IAE	Not used	

15.5.5 OTN:STATus:RX<Pt>:ERRor<section>[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:ERRor<section>[:EVENt]?
Description	This query returns the errors event register. The content of this register is summarized in DB11 through DB16 of the OTN:STATus:RX<Pt>:AESummary:CONDition register.
Parameters	<Pt> = Port number
	<section> = Section number (1-6)
Response	<register>=<NR1 NUMERIC RESPONSE DATA> See Table 15.19
Response	
Example	`OTN:STAT:RX1:ERR1?` → `0`
Note	This command is for compatibility of V2.xx or V1.xx. Following commands are endorsed on V3.00 or later. OTN:STATus:RX<Pt>:MSTage:AESummary[:EVENt]? OTN:STATus:RX<Pt>:OTU:ESUMmary[:EVENt]? OTN:STATus:RX<Pt>:SONE:ESUMmary[:EVENt]? OTN:STATus:RX<Pt>:STWO:ESUMmary[:EVENt]? OTN:STATus:RX<Pt>:STHRee:ESUMmary[:EVENt]? OTN:STATus:RX<Pt>:CLIent:ESUMmary[:EVENt]? OTN:STATus:RX<Pt>:OTU:ERRor<section>[:EVENt]? OTN:STATus:RX<Pt>:SONE:ERRor<section>[:EVENt]? OTN:STATus:RX<Pt>:STWO:ERRor<section>[:EVENt]? OTN:STATus:RX<Pt>:STHRee:ERRor<section>[:EVENt]? OTN:STATus:RX<Pt>:CLIent:ERRor<section>[:EVENt]?

15.5.6 OTN:STATus:RX<Pt>:ERRor<section>:CONDition?

Syntax	OTN:STATus:RX<Pt>:ERRor<section>:CONDition?
Description	This query returns the errors condition register query.
Parameters	<Pt> = Port number
	<section> = Section number (1-6)
Response	<register>=<NR1 NUMERIC RESPONSE DATA> See Table 15.19
Example	`OTN:STAT:RX1:ERR1:COND?` → `0`
Note	This command is for compatibility of V2.xx or V1.xx. Following commands are endorsed on V3.00 or later. OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition? OTN:STATus:RX<Pt>:OTU:ESUMmary:CONDition? OTN:STATus:RX<Pt>:SONE:ESUMmary:CONDition? OTN:STATus:RX<Pt>:STWO:ESUMmary:CONDition? OTN:STATus:RX<Pt>:STHRee:ESUMmary:CONDition? OTN:STATus:RX<Pt>:CLIent:ESUMmary:CONDition? OTN:STATus:RX<Pt>:OTU:ERRor<section>:CONDition? OTN:STATus:RX<Pt>:SONE:ERRor<section>:CONDition? OTN:STATus:RX<Pt>:STWO:ERRor<section>:CONDition? OTN:STATus:RX<Pt>:STHRee:ERRor<section>:CONDition? OTN:STATus:RX<Pt>:CLIent:ERRor<section>:CONDition?

Table 15.19: Error Items

OTU/ODU **Section**	1	2
DB1 (1)	FAS	TCM3-BEI
DB2 (2)	MFAS	TCM4-BEI
DB3 (4)	SM-BIP8	TCM5-BEI
DB4 (8)	SM-BEI	TCM6-BEI
DB5 (16)	FCE	CRC8 Error
DB6 (32)	FUEB	CRC5 Error
DB7 (64)	PM-BIP8	FAS-LLD
DB8 (128)	PM-BEI	ILA/OLA
DB9 (256)	TCM1-BIP8	MFAS LLD
DB10 (512)	TCM2-BIP8	LLM LLD
DB11 (1024)	TCM3-BIP8	Not used
DB12 (2048)	TCM4-BIP8	Not used
DB13 (4096)	TCM5-BIP8	Not used
DB14 (8192)	TCM6-BIP8	Not used
DB15 (16384)	TCM1-BEI	Not used
DB16 (32768)	TCM2-BEI	Not used
Stage 1 **Section**	3	4
DB1 (1)	FAS	CRC5 Error
DB2 (2)	PM-BIP8	Not used
DB3 (4)	PM-BEI	Not used
DB4 (8)	TCM1-BIP8	Not used
DB5 (16)	TCM2-BIP8	Not used
DB6 (32)	TCM3-BIP8	Not used
DB7 (64)	TCM4-BIP8	Not used
DB8 (128)	TCM5-BIP8	Not used
DB9 (256)	TCM6-BIP8	Not used
DB10 (512)	TCM1-BEI	Not used
DB11 (1024)	TCM2-BEI	Not used
DB12 (2048)	TCM3-BEI	Not used
DB13 (4096)	TCM4-BEI	Not used
DB14 (8192)	TCM5-BEI	Not used
DB15 (16384)	TCM6-BEI	Not used
DB16 (32768)	CRC8 Error	Not used
Stage 2 **Section**	5	6
DB1 (1)	FAS	cHEC correctable
DB2 (2)	PM-BIP8	cHEC uncorrectable
DB3 (4)	PM-BEI	tHEC correctable
DB4 (8)	TCM1-BIP8	tHEC uncorrectable
DB5 (16)	TCM2-BIP8	Invalid GFP Frame
DB6 (32)	TCM3-BIP8	Superblock CRC
DB7 (64)	TCM4-BIP8	CSF Signal
DB8 (128)	TCM5-BIP8	CSF Sync
DB9 (256)	TCM6-BIP8	Not used
DB10 (512)	TCM1-BEI	Not used
DB11 (1024)	TCM2-BEI	Not used
DB12 (2048)	TCM3-BEI	Not used
DB13 (4096)	TCM4-BEI	Not used
DB14 (8192)	TCM5-BEI	Not used
DB15 (16384)	TCM6-BEI	Not used
DB16 (32768)	PRBS Bit Error	Not used

15.5.7 OTN:STATus:RX<Pt>:MSTage:AESummary[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:MSTage:AESummary[:EVENt]?
Description	This query returns the alarms and errors summary event register. The content of this event register is summarized in DB8 of the STATus:INTerface:PORT<Pt>:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) OTU Alarm summary DB2 (2) Stage 1 Alarm summary DB3 (4) Stage 2 Alarm summary DB4 (8) Stage 3 Alarm summary DB5 - DB7 NOT USED DB8 (128) Client Alarm summary DB9 (256) OTU Error summary DB10 (512) Stage 1 Error summary DB11 (1024) Stage 2 Error summary DB12 (2048) Stage 3 Error summary DB13 - DB15 NOT USED DB16 (32768) Client Error summary
Example	`OTN:STAT:RX1:MST:AES?` → `65`
Note	

15.5.8 OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition?

Syntax	OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition?
Description	This query returns alarms and errors summary condition register. DB1 is summary of OTN:STATus:RX<Pt>:OTU:ASUMmary[:EVENt]? DB2 is summary of OTN:STATus:RX<Pt>:SONE:ASUMmary[:EVENt]? DB3 is summary of OTN:STATus:RX<Pt>:STWO:ASUMmary[:EVENt]? DB4 is summary of OTN:STATus:RX<Pt>:STHRee:ASUMmary[:EVENt]? DB8 is summary of OTN:STATus:RX<Pt>:CLIent:ASUMmary[:EVENt]? DB9 is summary of OTN:STATus:RX<Pt>:OTU:ESUMmary[:EVENt]? DB10 is summary of OTN:STATus:RX<Pt>:SONE:ESUMmary[:EVENt]? DB11 is summary of OTN:STATus:RX<Pt>:STWO:ESUMmary[:EVENt]? DB12 is summary of OTN:STATus:RX<Pt>:STHRee:ESUMmary[:EVENt]? DB16 is summary of OTN:STATus:RX<Pt>:CLIent:ESUMmary[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) OTU Alarm summary DB2 (2) Stage 1 Alarm summary DB3 (4) Stage 2 Alarm summary DB4 (8) Stage 3 Alarm summary DB5 - DB7 NOT USED DB8 (128) Client Alarm summary DB9 (256) OTU Error summary DB10 (512) Stage 1 Error summary DB11 (1024) Stage 2 Error summary DB12 (2048) Stage 3 Error summary DB13 - DB15 NOT USED DB16 (32768) Client Error summary
Example	`OTN:STAT:RX1:MST:AES:COND?` → `1024`
Note	

15.5.9 OTN:STATus:RX<Pt>:OTU:ASUMmary[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:OTU:ASUMmary[:EVENt]?
Description	This query returns the OTU alarms summary event register. The content of this register is summarized in DB1 of the OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition? register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Alarm section 1 summary DB2 (2) Alarm section 2 summary DB3 (4) Alarm section 3 summary DB4 (8) Alarm section 4 summary DB5 - DB8 NOT USED DB9 (256) Alarm section 9 summary DB10 - DB16 NOT USED
Example	`OTN:STAT:RX1:OTU:ASUM?` → `14`
Note	

15.5.10 OTN:STATus:RX<Pt>:OTU:ASUMmary:CONDition?

Syntax	OTN:STATus:RX<Pt>:OTU:ASUMmary:CONDition?
Description	This query returns OTU alarms summary condition register. DB1-DB4/DB9 is summary of OTN:STATus:RX<Pt>:OTU:ALARm<section>:CONDition?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Alarm section 1 summary DB2 (2) Alarm section 2 summary DB3 (4) Alarm section 3 summary DB4 (8) Alarm section 4 summary DB5 - DB8 NOT USED DB9 (256) Alarm section 9 summary DB10 - DB16 NOT USED
Example	`OTN:STAT:RX1:OTU:ASUM:COND?` → `12`
Note	

15.5.11 OTN:STATus:RX<Pt>:OTU:ESUMmary[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:OTU:ESUMmary[:EVENt]?
Description	This query returns the OTU errors summary event register. The content of this event register is summarized in DB9 of the OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition? register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Error section 1 summary DB2 (2) Error section 2 summary DB3 - DB8 NOT USED DB9 (256) Error section 9 summary DB10 - DB16 NOT USED
Example	`OTN:STAT:RX1:OTU:ESUM?` → `14`
Note	

15.5.12 OTN:STATus:RX<Pt>:OTU:ESUMmary:CONDition?

Syntax	OTN:STATus:RX<Pt>:OTU:ESUMmary:CONDition?
Description	This query returns OTU errors summary condition register. DB1/DB2/DB9 is summary of OTN:STATus:RX<Pt>:OTU:ERRor<section>:CONDition?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Error section 1 summary DB2 (2) Error section 2 summary DB3 - DB8 NOT USED DB9 (256) Error section 9 summary DB10 - DB16 NOT USED
Example	`OTN:STAT:RX1:OTU:ESUM:COND?` → `12`
Note	

15.5.13 OTN:STATus:RX<Pt>:OTU:ALARm<section>[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:OTU:ALARm<section>[:EVENt]?
Description	This query returns the OTU alarms event register. The content of this register is summarized in DB1-DB4/DB9 of the OTN:STATus:RX<Pt>:OTU:ASUMmary:CONDition? register.
Parameters	<Pt> = Port number
	<section> = Section number (1-4,9)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> See Table 15.20
Example	`OTN:STAT:RX1:OTU:ALAR5?` → `92`
Note	

15.5.14 OTN:STATus:RX<Pt>:OTU:ALARm<section>:CONDition?

Syntax	OTN:STATus:RX<Pt>:OTU:ALARm<section>:CONDition?
Description	This query returns the OTU alarms condition register query.
Parameters	<Pt> = Port number
	<section> = Section number (1-4,9)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> See Table 15.20
Response	<register>=<NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STAT:RX1:OTU:ALAR5:COND?` → `0`
Note	

Table 15.20: Alarm Items

OTU/ODU **Section**	1	2	3	4
DB1 (1)	LOS	TCM1-TIM	TCM5-BDI	FSF
DB2 (2)	OTU-AIS	TCM2-TIM	TCM6-BDI	FSD
DB3 (4)	LOF	TCM3-TIM	TCM1-IAE	BSF
DB4 (8)	OOF	TCM4-TIM	TCM2-IAE	BSD
DB5 (16)	LOM	TCM5-TIM	TCM3-IAE	Not used
DB6 (32)	OOM	TCM6-TIM	TCM4-IAE	Not used
DB7 (64)	SM-TIM	TCM1-BIAE	TCM5-IAE	Not used
DB8 (128)	SM-BIAE	TCM2-BIAE	TCM6-IAE	Not used
DB9 (256)	SM-BDI	TCM3-BIAE	TCM1-LTC	Not used
DB10 (512)	SM-IAE	TCM4-BIAE	TCM2-LTC	Not used
DB11 (1024)	ODU-AIS	TCM5-BIAE	TCM3-LTC	Not used
DB12 (2048)	ODU-LCK	TCM6-BIAE	TCM4-LTC	Not used
DB13 (4096)	ODU-OCI	TCM1-BDI	TCM5-LTC	Not used
DB14 (8192)	PM-TIM	TCM2-BDI	TCM6-LTC	Not used
DB15 (16384)	PM-BDI	TCM3-BDI	PLM	Not used
DB16 (32768)	Not used	TCM4-BDI	MSIM	Not used
OTL **Section**	5-8	9		
DB1 (1)	Not used	ILA/OLA		
DB2 (2)	Not used	LOF-OTL		
DB3 (4)	Not used	OOF-OTL		
DB4 (8)	Not used	LOR-OTL		
DB5 (16)	Not used	OOR-OTL		
DB6 (32)	Not used	Not used		
DB7 (64)	Not used	Not used		
DB8 (128)	Not used	Not used		
DB9 (256)	Not used	Not used		
DB10 (512)	Not used	Not used		
DB11 (1024)	Not used	Not used		
DB12 (2048)	Not used	Not used		
DB13 (4096)	Not used	Not used		
DB14 (8192)	Not used	Not used		
DB15 (16384)	Not used	Not used		
DB16 (32768)	Not used	Not used		

15.5.15 OTN:STATus:RX<Pt>:OTU:ERRor<section>[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:OTU:ERRor<section>[:EVENt]?
Description	This query returns the OTU errors event register. The content of this register is summarized in DB1/DB2/DB9 of the OTN:STATus:RX<Pt>:OTU:ESUMmary:CONDition? register.
Parameters	<Pt> = Port number
	<section> = Section number (1-2,9)
Response	<register>=<NR1 NUMERIC RESPONSE DATA> See Table 15.21
Response	
Example	`OTN:STAT:RX1:OTU:ERR1?` → `0`
Note	

Table 15.21: Error Items

OTU/ODU **Section**	1	2
DB1 (1)	FAS	TCM3-BEI
DB2 (2)	MFAS	TCM4-BEI
DB3 (4)	SM-BIP8	TCM5-BEI
DB4 (8)	SM-BEI	TCM6-BEI
DB5 (16)	FEC Correctable	CRC8 Error
DB6 (32)	FEC Uncorrectable	CRC5 Error
DB7 (64)	PM-BIP8	Not used
DB8 (128)	PM-BEI	Not used
DB9 (256)	TCM1-BIP8	Not used
DB10 (512)	TCM2-BIP8	Not used
DB11 (1024)	TCM3-BIP8	Not used
DB12 (2048)	TCM4-BIP8	Not used
DB13 (4096)	TCM5-BIP8	Not used
DB14 (8192)	TCM6-BIP8	Not used
DB15 (16384)	TCM1-BEI	Not used
DB16 (32768)	TCM2-BEI	Not used
OTL **Section**	3-8	9
DB1 (1)	Not used	FAS-OTL
DB2 (2)	Not used	MFAS-OTL
DB3 (4)	Not used	LLM-OTL
DB4 (8)	Not used	Not used
DB5 (16)	Not used	Not used
DB6 (32)	Not used	Not used
DB7 (64)	Not used	Not used
DB8 (128)	Not used	Not used
DB9 (256)	Not used	Not used
DB10 (512)	Not used	Not used
DB11 (1024)	Not used	Not used
DB12 (2048)	Not used	Not used
DB13 (4096)	Not used	Not used
DB14 (8192)	Not used	Not used
DB15 (16384)	Not used	Not used
DB16 (32768)	Not used	Not used

15.5.16 OTN:STATus:RX<Pt>:OTU:ERRor<section>:CONDition?

Syntax	OTN:STATus:RX<Pt>:OTU:ERRor<section>:CONDition?
Description	This query returns the OTU errors condition register query.
Parameters	<Pt> = Port number
	<section> = Section number (1-2,9)
Response	<register>=<NR1 NUMERIC RESPONSE DATA> See Table 15.21
Example	`OTN:STAT:RX1:OTU:ERR1:COND?` → `0`
Note	

15.5.17 OTN:STATus:RX<Pt>:SONE:ASUMmary[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:SONE:ASUMmary[:EVENt]?
Description	This query returns the stage1 ODU alarms summary event register. The content of this event register is summarized in DB2 of the OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Alarm section 1 summary DB2 (2) Alarm section 2 summary DB3 (4) Alarm section 3 summary DB4 (8) Alarm section 4 summary DB5 - DB16 NOT USED
Example	`OTN:STAT:RX1:SONE:ASUM?` → `6`
Note	

15.5.18 OTN:STATus:RX<Pt>:SONE:ASUMmary:CONDition?

Syntax	OTN:STATus:RX<Pt>:SONE:ASUMmary:CONDition?
Description	This query returns stage1 ODU alarms summary condition register. DB1-DB4 is summary of OTN:STATus:RX<Pt>:SONE:ALARm<section>[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Alarm section 1 summary DB2 (2) Alarm section 2 summary DB3 (4) Alarm section 3 summary DB4 (8) Alarm section 4 summary DB5 - DB16 NOT USED
Example	`OTN:STAT:RX1:SONE:ASUM:COND?` → `3`
Note	

15.5.19 OTN:STATus:RX<Pt>:SONE:ESUMmary[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:SONE:ESUMmary[:EVENt]?
Description	This query returns the stage1 ODU errors summary event register. The content of this event register is summarized in DB10 of the OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Error section 1 summary DB2 (2) Error section 2 summary DB3 - DB16 NOT USED
Example	`OTN:STAT:RX1:SONE:ESUM?` → `6`
Note	

15.5.20 OTN:STATus:RX<Pt>:SONE:ESUMmary:CONDition?

Syntax	OTN:STATus:RX<Pt>:SONE:ESUMmary:CONDition?
Description	This query returns stage1 ODU errors summary condition register. DB1/DB2 is summary of OTN:STATus:RX<Pt>:SONE:ERRor<section>:CONDition?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Error section 1 summary DB2 (2) Error section 2 summary DB3 - DB16 NOT USED
Example	`OTN:STAT:RX1:SONE:ESUM:COND?` → `3`
Note	

Table 15.22: Alarm Items

ODU **Section**	1	2	3	4
DB1 (1)	Not used	Not used	Not used	FSF
DB2 (2)	Not used	Not used	Not used	FSD
DB3 (4)	LOFLOM	Not used	Not used	BSF
DB4 (8)	OOF	Not used	Not used	BSD
DB5 (16)	Not used	Not used	Not used	Not used
DB6 (32)	OOM	Not used	Not used	Not used
DB7 (64)	Not used	Not used	Not used	Not used
DB8 (128)	Not used	Not used	Not used	Not used
DB9 (256)	Not used	Not used	Not used	Not used
DB10 (512)	Not used	Not used	Not used	Not used
DB11 (1024)	ODU-AIS	Not used	Not used	Not used
DB12 (2048)	ODU-LCK	Not used	Not used	Not used
DB13 (4096)	ODU-OCI	Not used	Not used	Not used
DB14 (8192)	PM-TIM	Not used	Not used	Not used
DB15 (16384)	PM-BDI	Not used	PLM	Not used
DB16 (32768)	Not used	Not used	MSIM	Not used

15.5.21 OTN:STATus:RX<Pt>:SONE:ALARm<section>[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:SONE:ALARm<section>[:EVENt]?
Description	This query returns the stage1 ODU alarms event register. The content of this register is summarized in DB1-DB4 of the OTN:STATus:RX<Pt>:SONE:ASUMmary:CONDition? register.
Parameters	<Pt> = Port number
	<section> = Section number (1-4)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> See Table 15.22
Example	`OTN:STAT:RX1:SONE:ALAR4?` → `1`
Note	

15.5.22 OTN:STATus:RX<Pt>:SONE:ALARm<section>:CONDition?

Syntax	OTN:STATus:RX<Pt>:SONE:ALARm<section>:CONDition?
Description	This query returns the stage1 ODU alarms condition register query.
Parameters	<Pt> = Port number
	<section> = Section number (1-4)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> See Table 15.22
Response	<register>=<NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STAT:RX1:SONE:ALAR5:COND?` → `0`
Note	

Table 15.23: Error Items

ODU **Section**	1	2
DB1 (1)	FAS	Not used
DB2 (2)	Not used	Not used
DB3 (4)	Not used	Not used
DB4 (8)	Not used	Not used
DB5 (16)	Not used	CRC8 Error
DB6 (32)	Not used	CRC5 Error
DB7 (64)	PM-BIP8	Not used
DB8 (128)	PM-BEI	Not used
DB9 (256)	Not used	Not used
DB10 (512)	Not used	Not used
DB11 (1024)	Not used	Not used
DB12 (2048)	Not used	Not used
DB13 (4096)	Not used	Not used
DB14 (8192)	Not used	Not used
DB15 (16384)	Not used	Not used
DB16 (32768)	Not used	Not used

15.5.23 OTN:STATus:RX<Pt>:SONE:ERRor<section>[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:SONE:ERRor<section>[:EVENt]?
Description	This query returns the stage1 ODU errors event register. The content of this register is summarized in DB1/DB2 of the OTN:STATus:RX<Pt>:SONE:ESUMmary:CONDition? register.
Parameters	<Pt> = Port number
	<section> = Section number (1-2)
Response	<register>=<NR1 NUMERIC RESPONSE DATA> See Table 15.23
Response	
Example	`OTN:STAT:RX1:SONE:ERR1?` → `0`
Note	

15.5.24 OTN:STATus:RX<Pt>:SONE:ERRor<section>:CONDition?

Syntax	OTN:STATus:RX<Pt>:SONE:ERRor<section>:CONDition?
Description	This query returns the stage1 ODU errors condition register query.
Parameters	<Pt> = Port number
	<section> = Section number (1-2)
Response	<register>=<NR1 NUMERIC RESPONSE DATA> See Table 15.23
Example	`OTN:STAT:RX1:SONE:ERR1:COND?` → `0`
Note	

15.5.25 OTN:STATus:RX<Pt>:STWO:ASUMmary[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:STWO:ASUMmary[:EVENt]?
Description	This query returns the stage2 ODU alarms summary event register. The content of this event register is summarized in DB3 of the OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition? register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Alarm section 1 summary DB2 (2) Alarm section 2 summary DB3 (4) Alarm section 3 summary DB4 (8) Alarm section 4 summary DB5 - DB16 NOT USED
Example	`OTN:STAT:RX1:STWO:ASUM?` → `6`
Note	

15.5.26 OTN:STATus:RX<Pt>:STWO:ASUMmary:CONDition?

Syntax	OTN:STATus:RX<Pt>:STWO:ASUMmary:CONDition?
Description	This query returns stage2 ODU errors summary condition register. DB1-DB4 is summary of OTN:STATus:RX<Pt>:STWO:ALARm<section>[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Alarm section 1 summary DB2 (2) Alarm section 2 summary DB3 (4) Alarm section 3 summary DB4 (8) Alarm section 4 summary DB5 - DB16 NOT USED
Example	`OTN:STAT:RX1:STWO:ASUM:COND?` → `3`
Note	

15.5.27 OTN:STATus:RX<Pt>:STWO:ESUMmary[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:STWO:ESUMmary[:EVENt]?
Description	This query returns the stage2 ODU errors summary event register. The content of this event register is summarized in DB11 of the OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Error section 1 summary DB2 (2) Error section 2 summary DB3 - DB16 NOT USED
Example	`OTN:STAT:RX1:STWO:ESUM?` → `6`
Note	

15.5.28 OTN:STATus:RX<Pt>:STWO:ESUMmary:CONDition?

Syntax	OTN:STATus:RX<Pt>:STWO:ESUMmary:CONDition?
Description	This query returns stage2 ODU errors summary condition register. DB1/DB2 is summary of OTN:STATus:RX<Pt>:STWO:ERRor<section>[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Error section 1 summary DB2 (2) Error section 2 summary DB3 - DB16 NOT USED
Example	`OTN:STAT:RX1:STWO:ESUM:COND?` → `3`
Note	

15.5.29 OTN:STATus:RX<Pt>:STWO:ALARm<section>[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:STWO:ALARm<section>[:EVENt]?
Description	This query returns the stage2 ODU alarms event register. The content of this register is summarized in DB1-DB4 OTN:STATus:RX<Pt>:STWO:ASUMmary:CONDition? register.
Parameters	<Pt> = Port number
	<section> = Section number (1-4)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> See Table 15.22
Example	`OTN:STAT:RX1:STWO:ALAR5?` → `92`
Note	

15.5.30 OTN:STATus:RX<Pt>:STWO:ALARm<section>:CONDition?

Syntax	OTN:STATus:RX<Pt>:STWO:ALARm<section>:CONDition?
Description	This query returns the stage2 ODU alarms condition register query.
Parameters	<Pt> = Port number
	<section> = Section number (1-4)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> See Table 15.22
Response	<register>=<NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STAT:RX1:STWO:ALAR5:COND?` → `0`
Note	

15.5.31 OTN:STATus:RX<Pt>:STWO:ERRor<section>[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:STWO:ERRor<section>[:EVENt]?
Description	This query returns the stage2 ODU errors event register. The content of this register is summarized in DB1/DB2 of the OTN:STATus:RX<Pt>:STWO:ESUMmary:CONDition? register.
Parameters	<Pt> = Port number
	<section> = Section number (1-2)
Response	<register>=<NR1 NUMERIC RESPONSE DATA> See Table 15.23
Response	
Example	`OTN:STAT:RX1:STWO:ERR1?` → `0`
Note	

15.5.32 OTN:STATus:RX<Pt>:STWO:ERRor<section>:CONDition?

Syntax	OTN:STATus:RX<Pt>:STWO:ERRor<section>:CONDition?
Description	This query returns the stage2 ODU errors condition register query.
Parameters	<Pt> = Port number
	<section> = Section number (1-2)
Response	<register>=<NR1 NUMERIC RESPONSE DATA> See Table 15.23
Example	`OTN:STAT:RX1:STWO:ERR1:COND?` → `0`
Note	

15.5.33 OTN:STATus:RX<Pt>:STHRee:ASUMmary[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:STHRee:ASUMmary[:EVENt]?
Description	This query returns the stage3 ODU alarms summary event register. The content of this event register is summarized in DB4 of the OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Alarm section 1 summary DB2 (2) Alarm section 2 summary DB3 (4) Alarm section 3 summary DB4 (8) Alarm section 4 summary DB5 - DB16 NOT USED
Example	`OTN:STAT:RX1:STHR:ASUM?` → `6`
Note	

15.5.34 OTN:STATus:RX<Pt>:STHRee:ASUMmary:CONDition?

Syntax	OTN:STATus:RX<Pt>:STHRee:ASUMmary:CONDition?
Description	This query returns stage3 ODU alarms summary condition register. DB1-DB4 is summary of OTN:STATus:RX<Pt>:STHRee:ALARm<section>[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Alarm section 1 summary DB2 (2) Alarm section 2 summary DB3 (4) Alarm section 3 summary DB4 (8) Alarm section 4 summary DB5 - DB16 NOT USED
Example	`OTN:STAT:RX1:STHR:ASUM:COND?` → `3`
Note	

15.5.35 OTN:STATus:RX<Pt>:STHRee:ESUMmary[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:STHRee:ESUMmary[:EVENt]?
Description	This query returns the stage3 ODU errors summary event register. The content of this event register is summarized in DB12 of the OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition? register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Error section 1 summary DB2 (2) Error section 2 summary DB3 - DB16 NOT USED
Example	`OTN:STAT:RX1:STHR:ESUM?` → `6`
Note	

15.5.36 OTN:STATus:RX<Pt>:STHRee:ESUMmary:CONDition?

Syntax	OTN:STATus:RX<Pt>:STHRee:ESUMmary:CONDition?
Description	This query returns stage3 ODU errors summary condition register. DB1/DB2 is summary of OTN:STATus:RX<Pt>:STHRee:ERRor<section>[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Error section 1 summary DB2 (2) Error section 2 summary DB3 - DB16 NOT USED
Example	`OTN:STAT:RX1:STHR:ESUM:COND?` → `3`
Note	

15.5.37 OTN:STATus:RX<Pt>:STHRee:ALARm<section>[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:STHRee:ALARm<section>[:EVENt]?
Description	This query returns the stage3 ODU alarms event register. The content of this register is summarized in DB1-DB4 of the OTN:STATus:RX<Pt>:STHRee:ASUMmary:CONDition? register.
Parameters	<Pt> = Port number
	<section> = Section number (1-4)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> See Table 15.22
Example	`OTN:STAT:RX1:STHR:ALAR5?` → `92`
Note	

15.5.38 OTN:STATus:RX<Pt>:STHRee:ALARm<section>:CONDition?

Syntax	OTN:STATus:RX<Pt>:STHRee:ALARm<section>:CONDition?
Description	This query returns the stage3 ODU alarms condition register query.
Parameters	<Pt> = Port number
	<section> = Section number (1-4)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> See Table 15.22
Response	<register>=<NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STAT:RX1:STHR:ALAR5:COND?` → `0`
Note	

15.5.39 OTN:STATus:RX<Pt>:STHRee:ERRor<section>[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:STHRee:ERRor<section>[:EVENt]?
Description	This query returns the stage3 ODU errors event register. The content of this register is summarized in DB1/DB2 of the OTN:STATus:RX<Pt>:STHRee:ESUMmary:CONDition? register.
Parameters	<Pt> = Port number
	<section> = Section number (1-2)
Response	<register>=<NR1 NUMERIC RESPONSE DATA> See Table 15.23
Response	<register>=<NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STAT:RX1:STHR:ERR1?` → `0`
Note	

15.5.40 OTN:STATus:RX<Pt>:STHRee:ERRor<section>:CONDition?

Syntax	OTN:STATus:RX<Pt>:STHRee:ERRor<section>:CONDition?
Description	This query returns the stage3 ODU errors condition register query.
Parameters	<Pt> = Port number
	<section> = Section number (1-2)
Response	<register>=<NR1 NUMERIC RESPONSE DATA> See Table 15.23
Example	`OTN:STAT:RX1:STHR:ERR1:COND?` → `0`
Note	

15.5.41 OTN:STATus:RX<Pt>:CLIent:ASUMmary[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:CLIent:ASUMmary[:EVENt]?
Description	This query returns the client alarms summary event register. The content of this event register is summarized in DB8 of the OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition? register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Alarm section 1 summary DB2 - DB16 NOT USED
Example	`OTN:STAT:RX1:CLI:ASUM?` → `1`
Note	

15.5.42 OTN:STATus:RX<Pt>:CLIent:ASUMmary:CONDition?

Syntax	OTN:STATus:RX<Pt>:CLIent:ASUMmary:CONDition?
Description	This query returns client alarms summary condition register. DB1 is summary of OTN:STATus:RX<Pt>:CLIent:ALARm<section>[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Alarm section 1 summary DB2 - DB16 NOT USED
Example	`OTN:STAT:RX1:CLI:ASUM:COND?` → `1`
Note	

15.5.43 OTN:STATus:RX<Pt>:CLIent:ESUMmary[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:CLIent:ESUMmary[:EVENt]?
Description	This query returns the client errors summary event register. The content of this event register is summarized in DB16 of the OTN:STATus:RX<Pt>:MSTage:AESummary:CONDition? register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Error section 1 summary DB2 (2) Error section 2 summary DB3 - DB16 NOT USED
Example	`OTN:STAT:RX1:CLI:ESUM?` → `2`
Note	

15.5.44 OTN:STATus:RX<Pt>:CLIent:ESUMmary:CONDition?

Syntax	OTN:STATus:RX<Pt>:CLIent:ESUMmary:CONDition?
Description	This query returns client errors summary condition register. DB1/DB2 is summary of OTN:STATus:RX<Pt>:CLIent:ERRor<section>[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) Alarm section 1 summary DB2 (2) Alarm section 2 summary DB3 - DB16 NOT USED
Example	`OTN:STAT:RX1:CLI:ESUM:COND?` → `3`
Note	

Table 15.24: Alarm Items

Client **Section**	 1
DB1 (1)	Client-AIS
DB2 (2)	CSF
DB3 (4)	LSS
DB4 (8)	Not used
DB5 (16)	Not used
DB6 (32)	Not used
DB7 (64)	Not used
DB8 (128)	Not used
DB9 (256)	Not used
DB10 (512)	Not used
DB11 (1024)	Not used
DB12 (2048)	Not used
DB13 (4096)	Not used
DB14 (8192)	Not used
DB15 (16384)	Not used
DB16 (32768)	Not used

15.5.45 OTN:STATus:RX<Pt>:CLIent:ALARm<section>[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:CLIent:ALARm<section>[:EVENt]?
Description	This query returns the client alarms event register. The content of this register is summarized in DB1 of the OTN:STATus:RX<Pt>:CLIent:ASUMmary:CONDition? register.
Parameters	<Pt> = Port number
	<section> = Section number (1)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> See Table 15.24
Example	`OTN:STAT:RX1:CLI:ALAR1?` → `4`
Note	

15.5.46 OTN:STATus:RX<Pt>:CLIent:ALARm<section>:CONDition?

Syntax	OTN:STATus:RX<Pt>:CLIent:ALARm<section>:CONDition?
Description	This query returns the client alarms condition register query.
Parameters	<Pt> = Port number
	<section> = Section number (1)
Response	<register> = <NR1 NUMERIC RESPONSE DATA> See Table 15.24
Response	<register>=<NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STAT:RX1:CLI:ALAR5:COND?` → `6`
Note	

Table 15.25: Error Items

Client **Section**	1	2
DB1 (1)	PRBS Bit Error	cHEC Correctable
DB2 (2)	Not used	cHEC Uncorrectable
DB3 (4)	Not used	tHEC Correctable
DB4 (8)	Not used	tHEC Uncorrectable
DB5 (16)	Not used	Invalid GFP Frame
DB6 (32)	Not used	Superblock CRC
DB7 (64)	Not used	CSF Signal
DB8 (128)	Not used	CSF Sync
DB9 (256)	Not used	FCS
DB10 (512)	Not used	eHEC Correctable
DB11 (1024)	Not used	eHEC Uncorrectable
DB12 (2048)	Not used	CMF Loss of Sync.
DB13 (4096)	Not used	CMF Loss of Signal
DB14 (8192)	Not used	SSF
DB15 (16384)	Not used	PTI Mismatch
DB16 (32768)	Not used	UPI Mismatch

15.5.47 OTN:STATus:RX<Pt>:CLIent:ERRor<section>[:EVENt]?

Syntax	OTN:STATus:RX<Pt>:CLIent:ERRor<section>[:EVENt]?
Description	This query returns the client errors event register. The content of this register is summarized in DB1/DB2 of the OTN:STATus:RX<Pt>:CLIent:ESUMmary:CONDition? register.
Parameters	<Pt> = Port number
	<section> = Section number (1-2)
Response	<register>=<NR1 NUMERIC RESPONSE DATA> See Table 15.25
Response	<register>=<NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STAT:RX1:CLI:ERR1?` → `0`
Note	

15.5.48 OTN:STATus:RX<Pt>:CLIent:ERRor<section>:CONDition?

Syntax	OTN:STATus:RX<Pt>:CLIent:ERRor<section>:CONDition?
Description	This query returns the client errors condition register query.
Parameters	<Pt> = Port number
	<section> = Section number (1-2)
Response	<register>=<NR1 NUMERIC RESPONSE DATA> See Table 15.25
Example	`OTN:STAT:RX1:CLI:ERR1:COND?` → `0`
Note	

15.5.49 OTN:STATus:RX<Pt>:PSLevel?

Syntax	OTN:STATus:RX<Pt>:PSLevel?
Description	This query returns the physical signal input level. Unit: dBm.
Parameter	<Pt> = Port number
Response	<signallevel> = <STRING RESPONSE DATA> "<power> dBm" : Min: "< -27 dBm" , Max: "Exceeds Level" "Module not ready" "Unknown module" "No module"
Example	`OTN:STAT:RX1:PSL?` → `"-12 dBm"`
Note	

15.5.50 OTN:STATus:TX<Pt>:PSLevel?

Syntax	OTN:STATus:TX<Pt>:PSLevel?
Description	This query returns the physical signal output level. Unit: dBm.
Parameter	<Pt> = Port number
Response	<signallevel> = <STRING RESPONSE DATA> "<power> dBm" : Min: "< -27 dBm" , Max: "Exceeds Level" "Module not ready" "Unknown module" "No module"
Example	`OTN:STAT:TX1:PSL? → "-3 dBm"`
Note	

15.5.51 OTN:STATus:RX<Pt>:FREQuency?

Syntax	OTN:STATus:RX<Pt>:FREQuency? <unit>
Description	This query returns the physical frequency error.
Parameters	<Pt> = Port number
	<unit> = <CHARACTER PROGRAM DATA> PPM = Parts per million HZ = Hz *DEFault = PPM*
Response	<signalfreq> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STAT:RX1:FREQ? PPM → 3`
Note	

15.5.52 OTN:STATus:TX<Pt>:FREQuency?

Syntax	OTN:STATus:TX<Pt>:FREQuency? <unit>
Description	This query returns the physical frequency error.
Parameters	<Pt> = Port number
	<unit> = <CHARACTER PROGRAM DATA> PPM = Parts per million HZ = Hz *DEFault = PPM*
Response	<signalfreq> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STAT:TX1:FREQ? PPM → 3`
Note	

15.5.53 OTN:STATus:RX<Pt>:CAPTure:OH?

Syntax	OTN:STATus:RX<Pt>:CAPTure:OH? <odutype>,<row>,<column>
Description	This query returns the multi frame sequence data of selected position of OH. Data length depends on the selected position of OH.
Parameters	<Pt> = Port number
	<odutype> = <CHARACTER PROGRAM DATA> ODU2: ODU2 ODU1: ODU1 ODU0: ODU0 ODU2E: ODU2e ODU1E: ODU1e ODU2F: ODU2f ODU1F: ODU1f ODUFLEX: ODUflex ODU3: ODU3 ODU3E1: ODU3e1 ODU3E2: ODU3e2 ODU4: ODU4 ODUC: ODUC
	<row> = OH row number(1-4)
	<column> = OH column number (1-16)
Response	<data> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`OTN:STAT:RX1:CAPT:OH? ODU2,2,16` → `#H08`
Note	When querying the row 4, columns 15 with this command, it always returns PSI[0].

15.6 Stimuli

15.6.1 OTN:STIMuli:TX<Pt>:TYPE

Syntax	OTN:STIMuli:TX<Pt>:TYPE <type>
Description	Choose the type of alarm/error that is inserted by the command SYSTem:STIMuli:INSert.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> AEINsert: Normal alarm/error GMP: GMP error GFP: GFP error FEC: FEC test error LLD: LLD error/alarm [1] *DEFault = AEINsert*
Response	None
Example	`OTN:STIM:TX1:TYPE AEIN`
Note	[1] LLD error/alarm are only available at OTU3/4.

Syntax	OTN:STIMuli:TX<Pt>:TYPE?
Description	
Parameter	<Pt> = Port number
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`OTN:STIM:TX1:TYPE? → AEIN`
Note	

15.6.2 OTN:STIMuli:TX<Pt>:AEINsert:LEVel

Syntax	OTN:STIMuli:TX<Pt>:AEINsert:LEVel <odutype>
Description	This command sets the ODU type to insert the error or alarm.
Parameters	<Pt> = Port number
	<odutype> = <CHARACTER PROGRAM DATA> ODU2: ODU2 ODU1: ODU1 ODU0: ODU0 ODU2E: ODU2e ODU1E: ODU1e ODU2F: ODU2f ODU1F: ODU1f ODUFLEX: ODUflex ODU3: ODU3 ODU3E1: ODU3e1 ODU3E2: ODU3e2 ODU4: ODU4 ODUC: ODUC *DEFault = ODU2*
Response	None
Example	`OTN:STIM:TX1:AEIN:LEV ODU2`
Note	

Syntax	OTN:STIMuli:TX<Pt>:AEINsert:LEVel?
Description	This query returns the ODU type to insert the error or alarm.
Parameter	<Pt> = Port number
Response	<odutype> = <CHARACTER RESPONSE DATA>
Example	`OTN:STIM:TX1:AEIN:LEV? → ODU2`
Note	

15.6.3 OTN:STIMuli:TX<Pt>:AEINsert:TYPE

Syntax	OTN:STIMuli:TX<Pt>:AEINsert:TYPE <aerrortype>
Description	This command sets the type to insert alarm/error.
Parameters	<Pt> = Port number
	<aerrortype> = <CHARACTER PROGRAM DATA> NONE: (None) BITALL: Bit All FAS: OTU-FAS MFAS: MFAS SMBIP8: SM-BIP8 SMBEI: SM-BEI PMBIP8: PM-BIP8 PMBEI: PM-BEI TCMiBIP8: TCMi-BIP8 (i = 1-6) TCMiBEI: TCMi-BEI (i = 1-6) FASODU: ODU-FAS PRBSBIT: PRBS Bit Error OOF: OTU-OOF/LOF OOM: OOM/LOM AISOTU: OTU-AIS SMTIM: SM-TIM SMBIAE: SM-BIAE SMBDI: SM-BDI SMIAE: SM-IAE AISODU: ODU-AIS OCI: ODU-OCI LCK: ODU-LCK PMTIM: PM-TIM PMBDI: PM-BDI TCMiTIM: TCMi-TIM (i = 1-6) TCMiBIAE: TCMi-BIAE (i = 1-6) TCMiIAE: TCMi-IAE (i = 1-6) TCMiBDI: TCMi-BDI (i = 1-6) TCMiLTC: TCMi-LTC (i = 1-6) OOFODU: ODU-OOF/LOF OOMODU: ODU-OOM/LOM AISC: Client-AIS CSF: CSF FSF: FSF FSD: FSD BSF: BSF BSD: BSD LOS: LOS *DEFault = NONE*
Response	None
Example	`OTN:STIM:TX1:AEIN:TYPE OOF`
Note	

Syntax	OTN:STIMuli:TX<Pt>:AEINsert:TYPE?
Description	This query returns the alarm/error insertion type.
Parameter	<Pt> = Port number
Response	<aerrortype> = <CHARACTER RESPONSE DATA>
Example	`OTN:STIM:TX1:AEIN:TYPE?` → `OOF`
Note	

15.6.4 OTN:STIMuli:TX<Pt>:AEINsert:INSert

Syntax	OTN:STIMuli:TX<Pt>:AEINsert:INSert <insertion>
Description	This command sets the method to insert alarm/error.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> SINGle BURSt ALTernate RATE ALL *DEFault = SINGle*
Response	None
Example	`OTN:STIM:TX1:AEIN:INS SING`
Note	If insertion is set to SINGle, errors are inserted with SYST:STIM:INS. See section 2.3.23.

Syntax	OTN:STIMuli:TX<Pt>:AEINsert:INSert?
Description	This query returns the alarm/error insertion method.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`OTN:STIM:TX1:AEIN:INS?` → `SING`
Note	

15.6.5 OTN:STIMuli:TX<Pt>:AEINsert:BURSt

Syntax	OTN:STIMuli:TX<Pt>:AEINsert:BURSt <frames>
Description	This command sets the alarm/error burst length to generate.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=4300000, DEFault=1*
Response	None
Example	`OTN:STIM:TX1:AEIN:BURS 1`
Note	

Syntax	OTN:STIMuli:TX<Pt>:AEINsert:BURSt?
Description	This query returns the alarm/error burst length to generate.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STIM:TX1:AEIN:BURS?` → `1`
Note	

15.6.6 OTN:STIMuli:TX<Pt>:AEINsert:AERRor

Syntax	OTN:STIMuli:TX<Pt>:AEINsert:AERRor <frames>
Description	This command sets the alternate alarm/error length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=4300000, DEFault=0*
Response	None
Example	`OTN:STIM:TX1:AEIN:AERR 0`
Note	

Syntax	OTN:STIMuli:TX<Pt>:AEINsert:AERRor?
Description	This query returns the alternate alarm/error length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STIM:TX1:AEIN:AERR?` → `0`
Note	

15.6.7 OTN:STIMuli:TX<Pt>:AEINsert:NORMal

Syntax	OTN:STIMuli:TX<Pt>:AEINsert:NORMal <frames>
Description	This command sets the alternate normal length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=4300000, DEFault=1* *DEFault = 1*
Response	None
Example	`OTN:STIM:TX1:AEIN:NORM 1`
Note	

Syntax	OTN:STIMuli:TX<Pt>:AEINsert:NORMal?
Description	This query returns the alternate normal length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STIM:TX1:AEIN:NORM?` → `1`
Note	

15.6.8 OTN:STIMuli:TX<Pt>:AEINsert:RATE

Syntax	OTN:STIMuli:TX<Pt>:AEINsert:RATE <rate>
Description	This command sets the rate to insert errors.
Parameters	<Pt> = Port number
	<rate> = <CHARACTER PROGRAM DATA> R1E2: 1.0E-2 R1E3: 1.0E-3 R1E4: 1.0E-4 R1E5: 1.0E-5 R1E6: 1.0E-6 R1E7: 1.0E-7 R1E8: 1.0E-8 R1E9: 1.0E-9 R1E10: 1.0E-10 *DEFault = R1E9*
Response	None
Example	`OTN:STIM:TX1:AEIN:RATE R1E9`
Note	Available when the insertion mode is Rate. Not all error rates are available for all error types.

Syntax	OTN:STIMuli:TX<Pt>:AEINsert:RATE?
Description	This command returns the rate to insert errors.
Parameter	<Pt> = Port number
Response	<rate> = <CHARACTER RESPONSE DATA>
Example	`OTN:STIM:TX1:AEIN:RATE?` → `R1E9`
Note	

15.6.9 OTN:STIMuli:TX<Pt>:JUSTification:LEVel

Syntax	OTN:STIMuli:TX<Pt>:JUSTification:LEVel <odutype>
Description	This command sets the ODU type to adjust the pointer. Only OPUk those are mapped into AMP or BMP are available.
Parameters	<Pt> = Port number
	<odutype> = <CHARACTER PROGRAM DATA> ODU2: ODU2 ODU1: ODU1 ODU0: ODU0 ODU2E: ODU2e ODU3: ODU3 ODU3E1: ODU3e1 *DEFault = ODU2*
Response	None
Example	`OTN:STIM:TX1:JUST:LEV ODU2`
Note	

Syntax	OTN:STIMuli:TX<Pt>:JUSTification:LEVel?
Description	This query returns the ODU type to adjust the pointer.
Parameter	<Pt> = Port number
Response	<odutype> = <CHARACTER RESPONSE DATA>
Example	`OTN:STIM:TX1:JUST:LEV?` → `ODU2`
Note	

15.6.10 OTN:STIMuli:TX<Pt>:JUSTification:BURSt

Syntax	OTN:STIMuli:TX<Pt>:JUSTification:BURSt <frames>
Description	This command specifies the number of count to move the pointer in the direction over a time.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=100, DEFault=1*
Response	None
Example	`OTN:STIM:TX1:JUST:BURS 1`
Note	

Syntax	OTN:STIMuli:TX<Pt>:JUSTification:BURSt?
Description	This query returns the number of count to move the pointer in the direction over a time.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STIM:TX1:JUST:BURS?` → `1`
Note	

15.6.11 OTN:STIMuli:TX<Pt>:JUSTification:MOVement

Syntax	OTN:STIMuli:TX<Pt>:JUSTification:MOVement <count>
Description	This command sets the OTN pointer movement sequence.
Parameters	<Pt> = Port number
	<count> = <CHARACTER PROGRAM DATA> POS1: Positive (+1) POS2: Positive (+2) POS3: Positive (+3) NEG1: Negative (-1) NEG2: Negative (-2) *DEFault = POS1*
Response	None
Example	`OTN:STIM:TX1:JUST:MOV POS1`
Note	

Syntax	OTN:STIMuli:TX<Pt>:JUSTification:MOVement?
Description	This query returns the OTN pointer movement sequence.
Parameter	<Pt> = Port number
Response	<count> = <CHARACTER RESPONSE DATA>
Example	`OTN:STIM:TX1:JUST:MOV?` → `POS1`
Note	

15.6.12 OTN:STIMuli:TX<Pt>:JMOVe

Syntax	OTN:STIMuli:TX<Pt>:JMOVe
Description	This command moves OTN pointer to instruct pointer.
Parameter	<Pt> = Port number
Response	None
Example	`OTN:STIM:TX1:JMOV`
Note	

15.6.13 OTN:STIMuli:TX<Pt>:FREQuency:OFFSet

Syntax	OTN:STIMuli:TX<Pt>:FREQuency:OFFSet <offset>
Description	This command sets the frequency offset for the clock source. Unit: ppm.
Parameters	<Pt> = Port number
	<offset> = <NUMERIC PROGRAM DATA> MT1000A: *MINimum=-50, MAXimum=50, DEFault=0* MT1100A: *MINimum=-200.0, MAXimum=200.0, DEFault = 0.0*
Response	None
Example	`OTN:STIM:TX1:FREQ:OFFS 0`
Note	The offset is applied to the internal clock source only.

Syntax	OTN:STIMuli:TX<Pt>:FREQuency:OFFSet?
Description	This query returns the frequency offset for the clock source. Unit: ppm.
Parameter	<Pt> = Port number
Response	MT1000A: <offset> = <NR1 NUMERIC RESPONSE DATA> MT1100A: <offset> = <NR2 NUMERIC RESPONSE DATA>
Example	MT1000A: `OTN:STIM:TX1:FREQ:OFFS?` → `0` MT1100A: `OTN:STIM:TX1:FREQ:OFFS?` → `0.0`
Note	

15.6.14 OTN:STIMuli:TX<Pt>:PAYLoad:OFFSet<stage>

Syntax	OTN:STIMuli:TX<Pt>:PAYLoad:OFFSet<stage> <offset>
Description	This command sets the payload offset.
Parameters	<Pt> = Port number
	<stage> = Multiplexing stage (1-4)
	<offset> = <NUMERIC PROGRAM DATA> *MINimum=-150, MAXimum=150, DEFault=0* ODU2/ODU1: -65.6 to 65.6 step 0.1ppm ODTU12: -113.6 to 83.3 step 0.1ppm ODTU01: -131.3 to 65.0 step 0.1ppm ODTUk.xx (GMP): -150.0 to 150.0 step 0.1ppm ODU3: -65.6 to 65.6 step 0.1ppm ODTU23: -95.8 to 101.1 step 0.1ppm ODTU13: -96.3 to 101.3 step 0.1ppm *DEFault = 0*
Response	None
Example	`OTN:STIM:TX1:PAYL:OFFS1 0`
Note	This parameter is defined at each multiplexing stage where GMP or AMP is used.

Syntax	OTN:STIMuli:TX<Pt>:PAYLoad:OFFSet<stage>?
Description	This query returns the payload offset.
Parameter	<Pt> = Port number
	<stage> = Multiplexing stage (1-4)
Response	<offset> = <NR2 NUMERIC RESPONSE DATA>
Example	`OTN:STIM:TX1:PAYL:OFFS1?` → `0`
Note	

15.6.15 OTN:STIMuli:TX<Pt>:GMP:AEINsert:LEVel

Syntax	OTN:STIMuli:TX<Pt>:GMP:AEINsert:LEVel <stage>
Description	This command sets the multiplexing stage to insert the error. It can be select those have GMP on the current mapping path.
Parameters	<Pt> = Port number
	<stage> = <CHARACTER PROGRAM DATA> ODU2: ODU2 ODU1: ODU1 ODU0: ODU0 ODU3: ODU3 ODU3E2: ODU3e2 ODU4: ODU4 *DEFault = ODU2*
Response	None
Example	`OTN:STIM:TX1:GMP:AEIN:LEV ODU2`
Note	

Syntax	OTN:STIMuli:TX<Pt>:GMP:AEINsert:LEVel?
Description	This query returns the multiplexing stage to insert the error.
Parameter	<Pt> = Port number
Response	<stage> = <CHARACTER RESPONSE DATA>
Example	`OTN:STIM:TX1:GMP:AEIN:LEV?` → `ODU2`
Note	

15.6.16 OTN:STIMuli:TX<Pt>:GMP:AEINsert:TYPE

Syntax	OTN:STIMuli:TX<Pt>:GMP:AEINsert:TYPE <aerrortype>
Description	This command sets the method to insert errors.
Parameters	<Pt> = Port number
	<aerrortype> = <CHARACTER PROGRAM DATA> CRC8: CRC8 Error CRC5: CRC5 Error JC1: Invalid JC1 JC2: Invalid JC2 JC1JC2: Invalid JC1&JC2 *DEFault = CRC8*
Response	None
Example	`OTN:STIM:TX1:GMP:AEIN:TYPE CRC8`
Note	

Syntax	OTN:STIMuli:TX<Pt>:GMP:AEINsert:TYPE?
Description	This query returns the method to insert errors.
Parameter	<Pt> = Port number
Response	<aerrortype> = <CHARACTER RESPONSE DATA>
Example	`OTN:STIM:TX1:GMP:AEIN:TYPE?` → `CRC8`
Note	

15.6.17 OTN:STIMuli:TX<Pt>:GMP:AEINsert:INSert

Syntax	OTN:STIMuli:TX<Pt>:GMP:AEINsert:INSert <insertion>
Description	This command sets the stimuli error insertion mode.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> SINGle
Response	None
Example	`OTN:STIM:TX1:GMP:AEIN:INS SING`
Note	Currently only one insertion mode is available.

Syntax	OTN:STIMuli:TX<Pt>:GMP:AEINsert:INSert?
Description	This query returns the stimuli error insertion mode.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`OTN:STIM:TX1:GMP:AEIN:INS? → SING`
Note	

15.6.18 OTN:STIMuli:TX<Pt>:GMP:AEINsert:EBITs

Syntax	OTN:STIMuli:TX<Pt>:GMP:AEINsert:EBITs <insertion>
Description	This command sets 16-bit value to specify which bits are affected. For some error items, only the lower bits are effective (for example, CRC5 will use lower 5 bits.) UPPer bits of the specified value will be ignored in such a case.
Parameters	<Pt> = Port number
	<insertion> = <NUMERIC PROGRAM DATA> *MINimum=#B0000000000000000, MAXimum=#B1111111111111111, DEFault=#B1111111111111111*
Response	None
Example	`OTN:STIM:TX1:GMP:AEIN:EBIT #B1001000001` This command add error into bit 0, 3 and 9.
Note	

Syntax	OTN:STIMuli:TX<Pt>:GMP:AEINsert:EBITs?
Description	This query returns the 16-bit value to specify which bits are affected.
Parameter	<Pt> = Port number
Response	<insertion> = <BINARY NUMERIC RESPONSE DATA>
Example	`OTN:STIM:TX1:GMP:AEIN:EBIT? → #B1001000001000000`
Note	

15.6.19 OTN:STIMuli:TX<Pt>:GFP:AEINsert:TYPE

Syntax	OTN:STIMuli:TX<Pt>:GFP:AEINsert:TYPE <aerrortype>
Description	This command sets the error type of GFP-T and GFP-F.
Parameters	<Pt> = Port number
	<aerrortype> = <CHARACTER PROGRAM DATA> OFF: Off CHEC: cHEC THEC: tHEC SCRC: Super block CRC (only GFP-T) EHEC: eHEC (only GFP-F) FCSE: FCS error (only GFP-F) CMFSIGNAL: CMF Signal (only GFP-F) CMFSYNC: CMF Sync (only GFP-F) *DEFault = OFF*
Response	None
Example	`OTN:STIM:TX1:GFP:AEIN:TYPE OFF`
Note	

Syntax	OTN:STIMuli:TX<Pt>:GFP:AEINsert:TYPE?
Description	This query returns the error type of GFP.
Parameter	<Pt> = Port number
Response	<aerrortype> = <CHARACTER RESPONSE DATA>
Example	`OTN:STIM:TX1:GFP:AEIN:TYPE? → OFF`
Note	

15.6.20 OTN:STIMuli:TX<Pt>:GFP:AEINsert:INSert

Syntax	OTN:STIMuli:TX<Pt>:GFP:AEINsert:INSert <insertion>
Description	This command sets the insertion mode of GFP Error.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> SINGle BURSt ALL *DEFault = SINGle*
Response	None
Example	`OTN:STIM:TX1:GFP:AEIN:INS SING`
Note	

Syntax	OTN:STIMuli:TX<Pt>:GFP:AEINsert:INSert?
Description	This query returns the insertion mode of GFP Error.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`OTN:STIM:TX1:GFP:AEIN:INS? → SING`
Note	

15.6.21 OTN:STIMuli:TX<Pt>:GFP:AEINsert:EBITs

Syntax	OTN:STIMuli:TX<Pt>:GFP:AEINsert:EBITs <insertion>
Description	This command sets the 16-bit value to specify which bits are affected. For some error items, only the lower bits are effective (for example, CRC5 will use lower 5 bits.) UPPer bits of the specified value will be ignored in such a case.
Parameters	<Pt> = Port number
	<insertion> = <NUMERIC PROGRAM DATA> *MINimum=#B0000000000000000, MAXimum=#B1111111111111111, DEFault=#B1111111111111111*
Response	None
Example	`OTN:STIM:TX1:GFP:AEIN:EBIT #B1001000001`
Note	

Syntax	OTN:STIMuli:TX<Pt>:GFP:AEINsert:EBITs?
Description	This query returns the 16-bit value to specify which bits are affected.
Parameter	<Pt> = Port number
Response	<insertion> = <BINARY NUMERIC RESPONSE DATA>
Example	`OTN:STIM:TX1:GFP:AEIN:EBIT? → #B1001000001000000`
Note	

15.6.22 OTN:STIMuli:TX<Pt>:GFP:AEINsert:BURSt

Syntax	OTN:STIMuli:TX<Pt>:GFP:AEINsert:BURSt <frames>
Description	This command sets the alarm/error burst length to generate.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=4300000, DEFault=1*
Response	None
Example	`OTN:STIM:TX1:GFP:AEIN:BURS 1`
Note	

Syntax	OTN:STIMuli:TX<Pt>:GFP:AEINsert:BURSt?
Description	This query returns the alarm/error burst length to generate.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STIM:TX1:GFP:AEIN:BURS? → 1`
Note	

15.6.23 OTN:STIMuli:TX<Pt>:GFP:AEINsert:CMF:INTerval

Syntax	OTN:STIMuli:TX<Pt>:GFP:AEINsert:CMF:INTerval <interval>
Description	This command sets the interval time of CMF insertion. Unit: ms.
Parameters	<Pt> = Port number
	<interval> = <NUMERIC PROGRAM DATA> *MINimum=10, MAXimum=2560, DEFault=10*
Response	None
Example	`OTN:STIM:TX1:GFP:AEIN:CMF:INT 10`
Note	

Syntax	OTN:STIMuli:TX<Pt>:GFP:AEINsert:CMF:INTerval?
Description	This query returns the interval time of CMF insertion. Unit: ms.
Parameter	<Pt> = Port number
Response	<interval> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STIM:TX1:GFP:AEIN:CMF:INT? → 10`
Note	

15.6.24 OTN:STIMuli:TX<Pt>:FEC:AEINsert:TYPE

Syntax	OTN:STIMuli:TX<Pt>:FEC:AEINsert:TYPE <aerrortype>
Description	This command sets the error type inserted to FEC.
Parameters	<Pt> = Port number
	<aerrortype> = <CHARACTER PROGRAM DATA> OFF: Off O182: O.182 Poisson UNCORRECT: Uncorretcable Error CORRECT: Correctable Error *DEFault = OFF*
Response	None
Example	`OTN:STIM:TX1:FEC:AEIN:TYPE O182`
Note	

Syntax	OTN:STIMuli:TX<Pt>:FEC:AEINsert:TYPE?
Description	This query returns the error type inserted to FEC.
Parameter	<Pt> = Port number
Response	<aerrortype> = <CHARACTER RESPONSE DATA>
Example	`OTN:STIM:TX1:FEC:AEIN:TYPE?` → `O182`
Note	

15.6.25 OTN:STIMuli:TX<Pt>:FEC:AEINsert:INSert

Syntax	OTN:STIMuli:TX<Pt>:FEC:AEINsert:INSert?
Description	This query returns the insertion method of FEC error.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA> SINGle RATE
Example	`OTN:STIM:TX1:FEC:AEIN:INS?` → `RATE`
Note	

15.6.26 OTN:STIMuli:TX<Pt>:FEC:AEINsert:RMANtissa

Syntax	OTN:STIMuli:TX<Pt>:FEC:AEINsert:RMANtissa <mantissa>
Description	This command sets the magnitude of the rate to insert errors.
Parameters	<Pt> = Port number
	<mantissa> = <NUMERIC PROGRAM DATA> 0.1-9.9 step 0.1 *MINimum=0.1, MAXimum=9.9, DEFault=1.0*
Response	None
Example	`OTN:STIM:TX1:FEC:AEIN:RMAN 1.0`
Note	

Syntax	OTN:STIMuli:TX<Pt>:FEC:AEINsert:RMANtissa?
Description	This query returns the magnitude of the rate to insert errors.
Parameter	<Pt> = Port number
Response	<mantissa> = <NR2 NUMERIC RESPONSE DATA>
Example	`OTN:STIM:TX1:FEC:AEIN:RMAN?` → `1.0`
Note	

15.6.27 OTN:STIMuli:TX<Pt>:FEC:AEINsert:REXPnent

Syntax	OTN:STIMuli:TX<Pt>:FEC:AEINsert:REXPnent <exponent>
Description	This command sets the exponential value of the rate to insert errors.
Parameters	<Pt> = Port number
	<exponent> = <CHARACTER PROGRAM DATA> R1E2: 1.0E-2 R1E3: 1.0E-3 R1E4: 1.0E-4 R1E5: 1.0E-5 R1E6: 1.0E-6 R1E7: 1.0E-7 R1E8: 1.0E-8 R1E9: 1.0E-9 R1E10: 1.0E-10 *DEFault = R1E9*
Response	None
Example	`OTN:STIM:TX1:FEC:AEIN:REXP R1E9`
Note	

Syntax	OTN:STIMuli:TX<Pt>:FEC:AEINsert:REXPnent?
Description	This query returns the exponential value of the rate to insert errors.
Parameter	<Pt> = Port number
Response	<exponent> = <CHARACTER RESPONSE DATA>
Example	`OTN:STIM:TX1:FEC:AEIN:REXP?` → `R1E9`
Note	

15.6.28 OTN:STIMuli:TX<Pt>:FEC:AEINsert:EFAS

Syntax	OTN:STIMuli:TX<Pt>:FEC:AEINsert:EFAS <insertion>
Description	This command specifies whether to include the error over the whole frame or exclude FAS fields.
Parameters	<Pt> = Port number
	<insertion> = <BOOLEAN PROGRAM DATA> ON: Includes FAS OFF: Excludes FAS *DEFault = OFF*
Response	None
Example	`OTN:STIM:TX1:FEC:AEIN:EFAS OFF`
Note	Available only when the error item is set to O.182 and the insertion mode is Rate. It takes effect when the rate is greater than or equal to 2.0E-3.

Syntax	OTN:STIMuli:TX<Pt>:FEC:AEINsert:EFAS?
Description	This query returns status of FAS fields (to include the error over the whole frame or exclude FAS fields)
Parameter	<Pt> = Port number
Response	<insertion> = <BOOLEAN RESPONSE DATA>
Example	`OTN:STIM:TX1:FEC:AEIN:EFAS?` → `0`
Note	

15.6.29 OTN:STIMuli:TX<Pt>:FEC:AEINsert:SUBRow

Syntax	OTN:STIMuli:TX<Pt>:FEC:AEINsert:SUBRow <row>
Description	This command specifies the subrow inserting the alarm or the error.
Parameters	<Pt> = Port number
	<row> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=16, DEFault=1*
Response	None
Example	`OTN:STIM:TX1:FEC:AEIN:SUBR 1`
Note	

Syntax	OTN:STIMuli:TX<Pt>:FEC:AEINsert:SUBRow?
Description	This query returns the subrow inserting the alarm or the error.
Parameter	<Pt> = Port number
Response	<row> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STIM:TX1:FEC:AEIN:SUBR?` → `1`
Note	

15.6.30 OTN:STIMuli:TX<Pt>:LLD:AEINsert:TYPE

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:TYPE <type>
Description	This command sets the type of LLD alarm/error insertion.
Parameter	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> NONE: No alarm or error ALARM: Insert alarm ERROR: Insert error *DEFault = NONE*
Response	None
Example	`OTN:STIM:TX1:LLD:AEIN:TYPE ALARM`
Note	This command can be used on 40/100G

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:TYPE?
Description	This query returns the type of LLD alarm/error insertion.
Parameter	<Pt> = Port number
Response	<row> = <CHARACTER PROGRAM DATA>
Example	`OTN:STIM:TX1:LLD:AEIN:TYPE?` → `ALARM`
Note	This command can be used on 40/100G

15.6.31 OTN:STIMuli:TX<Pt>:LLD:AEINsert:ALARm

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:ALARm <type>
Description	This command sets the type of the Alarm.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> OOFLLD: OOF/LOF OOR: OOR/LOR *DEFault = OOFLLD*
Response	None
Example	`OTN:STIM:TX1:LLD:AEIN:ALAR OOFLLD`
Note	This command can be used on 40/100G

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:ALARm?
Description	This query returns the type of the Alarm.
Parameter	<Pt> = Port number
Response	<row> = <CHARACTER PROGRAM DATA>
Example	`OTN:STIM:TX1:LLD:AEIN:ALAR?` → `OOFLLD`
Note	This command can be used on 40/100G

15.6.32 OTN:STIMuli:TX<Pt>:LLD:AEINsert:ERRor

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:ERRor <type>
Description	This command sets the type of the Error.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> FASLLD: FAS-LLD MFASLLD: MFAS-LLD LLMLLD: LLM-LLD *DEFault = FASLLD*
Response	None
Example	`OTN:STIM:TX1:LLD:AEIN:ERR FASLLD`
Note	This command can be used on 40/100G

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:ERRor?
Description	This query returns the type of the Error.
Parameter	<Pt> = Port number
Response	<CHARACTER PROGRAM DATA>
Example	`OTN:STIM:TX1:LLD:AEIN:ERR?` → `FASLLD`
Note	This command can be used on 40/100G

15.6.33 OTN:STIMuli:TX<Pt>:LLD:AEINsert:INSert

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:INSert <insertion>
Description	This command sets the mode of the insertion.
Parameters	<Pt> = Port number
	<insertion> = <CHARACTER PROGRAM DATA> BURSt ALTernate ALL *DEFault = ALL*
Response	None
Example	`OTN:STIM:TX1:LLD:AEIN:INS ALL`
Note	This command can be used on 40/100G

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:INSert?
Description	This query returns the mode of the insertion.
Parameter	<Pt> = Port number
Response	<insertion> = <CHARACTER RESPONSE DATA>
Example	`OTN:STIM:TX1:LLD:AEIN:INS?` → `ALL`
Note	This command can be used on 40/100G

15.6.34 OTN:STIMuli:TX<Pt>:LLD:AEINsert:BURSt

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:BURSt <frames>
Description	This command sets the alarm/error burst length to generate.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> 1 to 215000 step 1
Response	None
Example	`OTN:STIM:TX1:LLD:AEIN:BURS 1`
Note	This command can be used on 40/100G

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:BURSt?
Description	This query returns the alarm/error burst length to generate.
Parameter	<Pt> = Port number
Response	<NUMERIC PROGRAM DATA>
Example	`OTN:STIM:TX1:LLD:AEIN:BURS?` → `1`
Note	This command can be used on 40/100G

15.6.35 OTN:STIMuli:TX<Pt>:LLD:AEINsert:AERRor

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:AERRor <frames>
Description	This command sets the alternate alarm/error length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> 0 to 215000 step 1
Response	None
Example	`OTN:STIM:TX1:LLD:AEIN:AERR 0`
Note	This command can be used on 40/100G

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:AERRor?
Description	This query returns the alternate alarm/error length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STIM:TX1:LLD:AEIN:AERR?` → `0`
Note	This command can be used on 40/100G

15.6.36 OTN:STIMuli:TX<Pt>:LLD:AEINsert:NORMal

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:NORMal <frames>
Description	This command sets the alternate normal length.
Parameters	<Pt> = Port number
	<frames> = <NUMERIC PROGRAM DATA> 1 to 215000 step 1
Response	None
Example	`OTN:STIM:TX1:LLD:AEIN:NORM 1`
Note	This command can be used on 40/100G

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:NORMal?
Description	This command returns the alternate normal length.
Parameter	<Pt> = Port number
Response	<frames> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:STIM:TX1:LLD:AEIN:NORM?` → `1`
Note	This command can be used on 40/100G

15.6.37 OTN:STIMuli:TX<Pt>:LLD:AEINsert:LANE

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:LANE <lane>
Description	This command sets the lane On/Off that bit corresponds lane.
Parameters	<Pt> = Port number
	<lane> = <NUMERIC PROGRAM DATA> *MINimum=#B00000000000000000000, MAXimum=#B11111111111111111111, DEFault=#B10000000000000000000*
Response	None
Example	`OTN:STIM:TX1:LLD:AEIN:LANE #B1001000001` This command add error into lane 0, 3 and 9.
Note	This command can be used on 40/100G

Syntax	OTN:STIMuli:TX<Pt>:LLD:AEINsert:LANE?
Description	This command returns the lane On/Off that bit corresponds lane.
Parameter	<Pt> = Port number
Response	<lane> = <BINARY NUMERIC RESPONSE DATA>
Example	`OTN:STIM:TX1:LLD:AEIN:LANE?` → `#B10010000010000000000`
Note	This command can be used on 40/100G

15.6.38 OTN:STIMuli:TX<Pt>:LLD:SKEW:LANE

Syntax	OTN:STIMuli:TX<Pt>:LLD:SKEW:LANE <lane>
Description	This command sets lane On/Off of the Tx Lane.
Parameters	<Pt> = Port number
	<lane> = <NUMERIC PROGRAM DATA> *MINimum=#B00000000000000000000, MAXimum=#B11111111111111111111, DEFault=#B10000000000000000000*
Response	None
Example	`OTN:STIM:TX1:LLD:SKEW:LANE #B1001000001` This command add skew into lane 0, 3 and 9.
Note	This command can be used on 40/100G

Syntax	OTN:STIMuli:TX<Pt>:LLD:SKEW:LANE?
Description	This command returns lane On/Off of the Tx Lane.
Parameter	<Pt> = Port number
Response	<lane> = <BINARY NUMERIC RESPONSE DATA>
Example	`OTN:STIM:TX1:LLD:SKEW:LANE?` → `#B10010000010000000000`
Note	This command can be used on 40/100G

15.6.39 OTN:STIMuli:TX<Pt>:LLD:SKEW:BIT

Syntax	OTN:STIMuli:TX<Pt>:LLD:SKEW:BIT <bit>
Description	This command sets the number of skew bit.
Parameters	<Pt> = Port number
	<bit> = <NUMERIC PROGRAM DATA> 0 to 32000 step 1
Response	None
Example	`OTN:STIM:TX1:LLD:SKEW:BIT 0`
Note	This command can be used on 40/100G

Syntax	OTN:STIMuli:TX<Pt>:LLD:SKEW:BIT?
Description	This command returns the number of skew bit.
Parameter	<Pt> = Port number
Response	<NUMERIC PROGRAM DATA>
Example	`OTN:STIM:TX1:LLD:SKEW:BIT?` → `0`
Note	This command can be used on 40/100G

15.6.40 OTN:STIMuli:TX<Pt>:LLD:SKEW:TYPE

Syntax	OTN:STIMuli:TX<Pt>:LLD:SKEW:TYPE <type>
Description	This command sets type of the insert lane.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> TXLANE: Tx Lane PHYSICALLANE: Physical Lane *DEFault = TXLANE*
Response	None
Example	`OTN:STIM:TX1:LLD:SKEW:TYPE TXLANE`
Note	This command can be used on 40/100G

Syntax	OTN:STIMuli:TX<Pt>:LLD:SKEW:TYPE?
Description	This command returns type of the insert lane.
Parameter	<Pt> = Port number
Response	<CHARACTER PROGRAM DATA>
Example	`OTN:STIM:TX1:LLD:SKEW:TYPE?` → `TXLANE`
Note	This command can be used on 40/100G

15.7 Tributary Scan

15.7.1 OTN:TSCan:STARt

Syntax	OTN:TSCan:STARt
Description	This command starts the tributary scan test.
Parameter	None.
Response	None.
Example	`OTN:TSC:STAR`
Note	

15.7.2 OTN:TSCan:STOP

Syntax	OTN:TSCan:STOP
Description	This command stops the tributary scan test.
Parameter	None.
Response	None.
Example	`OTN:TSC:STOP`
Note	

15.7.3 OTN:TSCan:RX<Pt>:ODU2:NUMBer?

Syntax	OTN:TSCan:RX<Pt>:ODU2:NUMBer?
Description	This query returns the number of ODU2/ODU2e.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:TSC:RX1:ODU2:NUMB?` → `1`
Note	

15.7.4 OTN:TSCan:RX<Pt>:ODU2:GET?

Syntax	OTN:TSCan:RX<Pt>:ODU2:GET?
Description	This query returns the detailed alarm and error information from a ODU2/ODU2e.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA> The values are presented in ascending order, meaning that ODU2 #1 is the rst on the list. 0: No alarms or errors. 1: Alarms or errors present.
Example	`OTN:TSC:RX1:ODU2:GET?` → `0`
Note	

15.7.5 OTN:TSCan:RX<Pt>:ODU2:DETailed?

Syntax	OTN:TSCan:RX<Pt>:ODU2:DETailed? <number>
Description	
Parameter	<Pt> = Port number
	<number> = ODU2 number MINimum=1, MAXimum=1
Response	<selected> = <STRING RESPONSE DAT> RXng {ODU2 Order}, separated by one space character. RXn = RX1 or RX2 ODU2-order = ODU2#1
	(<alarmserrors>) = <STRING RESPONSE DATA> List of alarms and errors.
Example	`OTN:TSC:RX1:ODU2:DET? 1` → `"RX1 OTU2#1",("PM-BDI","ODU-AIS")`
Note	

15.7.6 OTN:TSCan:RX<Pt>:ODU1:NUMBer?

Syntax	OTN:TSCan:RX<Pt>:ODU1:NUMBer?
Description	Description This query returns the number of ODU1/ODU1e.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:TSC:RX1:ODU1:NUMB?` → `4`
Note	

15.7.7 OTN:TSCan:RX<Pt>:ODU1:GET?

Syntax	OTN:TSCan:RX<Pt>:ODU1:GET?
Description	This query returns the state of ODU1/ODU1e.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA> The values are presented in ascending order, meaning that ODU1 #1 is the rst on the list. 0: No alarms or errors. 1: Alarms or errors present.
Example	`OTN:TSC:RX1:ODU1:GET?` → `1`
Note	

15.7.8 OTN:TSCan:RX<Pt>:ODU1:SCANning

Syntax	OTN:TSCan:RX<Pt>:ODU1:SCANning <number>
Description	This command sets the ODU1/ODU1e for scanning.
Parameters	<Pt> = Port number
	<number> = High Order number MINimum=1, MAXimum=4 *MINimum=1, MAXimum=4, DEFault=1*
Response	None
Example	`OTN:TSC:RX1:ODU1:SCAN 2`
Note	

Syntax	OTN:TSCan:RX<Pt>:ODU1:SCANning?
Description	This query returns the ODU1/ODU1e for scanning.
Parameter	<Pt> = Port number
Response	<number>= High Order number (1-4)
Example	`OTN:TSC:RX1:ODU1:SCAN?` → `2`
Note	

15.7.9 OTN:TSCan:RX<Pt>:ODU1:DETailed?

Syntax	OTN:TSCan:RX<Pt>:ODU1:DETailed? <number>
Description	This query returns the detailed alarm and error information from a ODU1/ODU1e.
Parameter	<Pt> = Port number
	<number> = ODU1 number MINimum=1, MAXimum=1
Response	<selected> = <STRING RESPONSE DATA> {RXng {ODU2 Order}, separated by one space character. RXn = RX1 or RX2 ODU2-order = ODU2#1
	(<alarmserrors>) = <STRING RESPONSE DATA> List of alarms and errors.
Example	`OTN:TSC:RX1:ODU1:DET? 1` → `"RX1 OTU1#1",("PM-BDI","ODU-AIS")`
Note	

15.7.10 OTN:TSCan:RX<Pt>:ODU0:NUMBer?

Syntax	OTN:TSCan:RX<Pt>:ODU0:NUMBer?
Description	This query returns the number of ODU0.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:TSC:RX1:ODU0:NUMB?` → `2`
Note	

15.7.11 OTN:TSCan:RX<Pt>:ODU0:GET?

Syntax	OTN:TSCan:RX<Pt>:ODU0:GET?
Description	This query returns the state of ODU0.
Parameter	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA> The values are presented in ascending order, meaning that ODU0 #1 is the rst on the list. 0: No alarms or errors. 1: Alarms or errors present.
Example	`OTN:TSC:RX1:ODU0:GET?` → `(0,1,1,0)`
Note	

15.7.12 OTN:TSCan:RX<Pt>:ODU0:DETailed?

Syntax	OTN:TSCan:RX<Pt>:ODU0:DETailed? <number>
Description	This query returns the detailed alarm and error information from a ODU0.
Parameter	<Pt> = Port number
	<number> = ODU0 number MINimum=1, MAXimum=1
Response	<selected> = <STRING RESPONSE DATA> {RXng {ODU0 Order}, separated by one space character. RXn = RX1 or RX2 ODU0-order = ODU0#1
	(<alarmserrors>) = <STRING RESPONSE DATA> List of alarms and errors.
Example	`OTN:TSC:RX1:ODU0:DET? 1` → `"RX1 OTU2#:ODU1#1:ODU0#1",("PM-BDI","ODU-AIS")`
Note	

15.8 APS

15.8.1 OTN:APS:RX<Pt>:BEVent

Syntax	OTN:APS:RX<Pt>:BEVent <event>
Description	This command sets the time reference begin (start) event.
Parameters	<Pt> = Port number
	<event> = <CHARACTER PROGRAM DATA> ANYERROR: Any errors LOS: Loss of Signal OTUAIS: OTU-AIS LOF: Loss of Frame OOF: Out of Frame LOM: LOM OOM: OOM SMBIAE: SM-BIAE SMBDI: SM-BDI SMIAE: SM-IAE ODUAIS: ODU-AIS ODUOCI: ODU-OCI ODULCK: ODU-LCK PMBDI: PM-BDI FAS: FAS MFAS: MFAS SMBIP8: SM-BIP8 SMBEI: SM-BEI PMBIP8: PM-BIP8 PMBEI: PM-BEI PERRor: Pattern error LOFLLD: LOF-OTL OOFLLD: OOF-OTL LORLLD: LOR-OTL OORLLD: OOR-OTL FASLLD: FAS-OTL MFASLLD: MFAS-OTL LLMLLD: LLM-OTL *DEFault = LOF*
Response	None
Example	`OTN:APS:RX1:BEV LOF`
Note	

Syntax	OTN:APS:RX<Pt>:BEVent?
Description	This query returns the time reference begin (start) event.
Parameter	<Pt> = Port number
Response	<event> = <CHARACTER RESPONSE DATA>
Example	`OTN:APS:RX1:BEV?` → `LOF`
Note	

15.8.2 OTN:APS:RX<Pt>:EEVent

Syntax	OTN:APS:RX<Pt>:EEVent <event>
Description	This command sets the time reference end (stop) event.
Parameters	<Pt> = Port number
	<event> = <CHARACTER PROGRAM DATA> ANYERROR: Any errors LOS: Loss of Signal OTUAIS: OTU-AIS LOF: Loss of Frame OOF: Out of Frame LOM: LOM OOM: OOM SMBIAE: SM-BIAE SMBDI: SM-BDI SMIAE: SM-IAE ODUAIS: ODU-AIS ODUOCI: ODU-OCI ODULCK: ODU-LCK PMBDI: PM-BDI FAS: FAS MFAS: MFAS SMBIP8: SM-BIP8 SMBEI: SM-BEI PMBIP8: PM-BIP8 PMBEI: PM-BEI PERRor: Pattern error LOFLLD: LOF-OTL OOFLLD: OOF-OTL LORLLD: LOR-OTL OORLLD: OOR-OTL FASLLD: FAS-OTL MFASLLD: MFAS-OTL LLMLLD: LLM-OTL *DEFault = LOF*
Response	None
Example	`OTN:APS:RX1:EEV LOF`
Note	

Syntax	OTN:APS:RX<Pt>:EEVent?
Description	This query returns the time reference end (stop) event.
Parameter	<Pt> = Port number
Response	<event> = <CHARACTER RESPONSE DATA>
Example	`OTN:APS:RX1:EEV?` → `LOF`
Note	

15.8.3 OTN:APS:RX<Pt>:PERiod

Syntax	OTN:APS:RX<Pt>:PERiod <period>
Description	This command sets the period that no error/alarm specified by the end event should not be detected.
Parameters	<Pt> = Port number
	<period> = <NUMERIC PROGRAM DATA> 1,10,100,200,300,400,500,600,700,800,900,1000 Unit ms *DEFault = 1*
Response	None
Example	`OTN:APS:RX1:PER 1`
Note	

Syntax	OTN:APS:RX<Pt>:PERiod?
Description	This query returns the period that no error/alarm specified by the end event should not be detected.
Parameter	<Pt> = Port number
Response	<period> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:APS:RX1:PER?` → `1`
Note	

15.8.4 OTN:APS:RX<Pt>:MLIMit

Syntax	OTN:APS:RX<Pt>:MLIMit <max>
Description	This command sets the time reference maximum limit. Unit: ms.
Parameters	<Pt> = Port number
	<max> = <NUMERIC PROGRAM DATA> *MINimum = 0.000, MAXimum = 10000.000, DEFault = 50.000*
Response	None
Example	`OTN:APS:RX1:MLIM 50.000`
Note	

Syntax	OTN:APS:RX<Pt>:MLIMit?
Description	This query returns the time reference maximum limit. Unit: ms.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
Example	`OTN:APS:RX1:MLIM?` → `50.000`
Note	

15.8.5 OTN:APS:STARt

Syntax	OTN:APS:STARt
Description	This command starts the APS (Automatic Protection Switching) measurement.
Parameter	None
Response	None
Example	`OTN:APS:STAR`
Note	

15.8.6 OTN:APS:STOP

Syntax	OTN:APS:STOP
Description	This command stops the APS (Automatic Protection Switching) measurement.
Parameter	None
Response	None
Example	`OTN:APS:STOP`
Note	

15.8.7 OTN:APS:RX<Pt>:NUMBer?

Syntax	OTN:APS:RX<Pt>:NUMBer?
Description	This query returns the number of times has measured.
Parameter	<Pt> = Port number
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:APS:RX1:NUMB?` → `17`
Note	

15.8.8 OTN:APS:RX<Pt>:MTIMe?

Syntax	OTN:APS:RX<Pt>:MTIMe?
Description	This query returns the maximum time has measured. Unit: ms.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
	<limit exceeded> = <NR1 NUMERIC RESPONSE DATA> Returns 1, if the reference maximum limit has been exceeded.
Example	`OTN:APS:RX1:MTIM?` → `4.000,0`
Note	The maximum measurable time is 10000 ms. The maximum measurable time will be responded if the result exceeds 10000 ms.

15.8.9 OTN:APS:RX<Pt>:LTIMe?

Syntax	OTN:APS:RX<Pt>:LTIMe?
Description	This query returns the least (minimum) time has measured. Unit: ms.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`OTN:APS:RX1:LTIM?` → `4.000`
Note	The maximum measurable time is 10000 ms. The maximum measurable time will be responded if the result exceeds 10000 ms.

15.8.10 OTN:APS:RX<Pt>:ATIMe?

Syntax	OTN:APS:RX<Pt>:ATIMe?
Description	This query returns the average time has measured. Unit: ms.
Parameter	<Pt> = Port number
Response	<avg> = <NR2 NUMERIC RESPONSE DATA>
Example	`OTN:APS:RX1:ATIM?` → `4.000`
Note	The maximum measurable time is 10000 ms. The maximum measurable time will be responded if the result exceeds 10000 ms.

15.8.11 OTN:APS:RX<Pt>:CTIMe?

Syntax	OTN:APS:RX<Pt>:CTIMe?
Description	This query returns the current time has measured. Unit: ms.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`OTN:APS:RX1:CTIM?` → `4.000`
Note	The maximum measurable time is 10000 ms. The maximum measurable time will be responded if the result exceeds 99999.999 ms.

15.9 RTD

This section document commands for the Round Trip Delay application.
Commands for general RTD settings are described in section 17.1 on page 953.

15.9.1 OTN:RTD:RX<Pt>:MLIMit

Syntax	OTN:RTD:RX<Pt>:MLIMit <max>
Description	This command sets the time reference maximum limit. Unit: us.
Parameters	<Pt> = Port number
	<max> = <NUMERIC PROGRAM DATA> *MINimum = 0.0, MAXimum = 10000000.0, DEFault = MAXimum*
Response	None.
Example	`OTN:RTD:RX1:MLIM 0.0`
Note	

Syntax	OTN:RTD:RX<Pt>:MLIMit?
Description	This query returns the time reference maximum limit. Unit: us.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
Example	`OTN:RTD:RX1:MLIM?` → `0.0`
Note	

15.9.2 OTN:RTD:RX<Pt>:NUMBer?

Syntax	OTN:RTD:RX<Pt>:NUMBer?
Description	This query returns the number of the RTD data.
Parameter	<Pt> = Port number
Response	<number> = <NR1 NUMERIC RESPONSE DATA>
Example	`OTN:RTD:RX1:NUMB?` → `2`
Note	

15.9.3 OTN:RTD:RX<Pt>:ATIMe?

Syntax	OTN:RTD:RX<Pt>:ATIMe?
Description	This query returns the average time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<avg> = <NR2 NUMERIC RESPONSE DATA>
Example	`OTN:RTD:RX1:ATIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

15.9.4 OTN:RTD:RX<Pt>:MTIMe?

Syntax	OTN:RTD:RX<Pt>:MTIMe?
Description	This query returns the maximum time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
	<limit exceeded> = <NR1 NUMERIC RESPONSE DATA> Returns 1, if the reference maximum limit has been exceeded.
Example	`OTN:RTD:RX1:MTIM?` → `1.0,0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

15.9.5 OTN:RTD:RX<Pt>:LTIMe?

Syntax	OTN:RTD:RX<Pt>:LTIMe?
Description	This query returns the least (minimum) time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`OTN:RTD:RX1:LTIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds the maximum measurable time.

15.9.6 OTN:RTD:RX<Pt>:CTIMe?

Syntax	OTN:RTD:RX<Pt>:CTIMe?
Description	This query returns the current time of RTD. Unit: us.
Parameter	<Pt> = Port number
Response	<min> = <NR2 NUMERIC RESPONSE DATA>
Example	`OTN:RTD:RX1:CTIM?` → `1.0`
Note	The maximum measurable time will be responded if the result exceeds 99999999.9 us.

Chapter 16

Pluggable Module

16.1 Status

16.1.1 PMODule:STATus:PORT<Pt>:PRESent?

Syntax	PMODule:STATus:PORT<Pt>:PRESent?
Description	This query returns if a pluggable module is present and ready.
Parameter	<Pt> = Port number
Response	<present> = <STRING RESPONSE DATA> "OK": Module is present and ready. "N/A": Port off, module not present or not ready.
Example	`PMOD:STAT:PORT1:PRES?` → `"OK"`
Note	

16.1.2 PMODule:STATus:PORT<Pt>:VENDor?

Syntax	PMODule:STATus:PORT<Pt>:VENDor?
Description	This query returns the pluggable module vendor name.
Parameter	<Pt> = Port number
Response	<vendor> = <STRING RESPONSE DATA>
Example	`PMOD:STAT:PORT1:VEND?` → `"COMPANY"`
Note	

16.1.3 PMODule:STATus:PORT<Pt>:WAVelength?

Syntax	PMODule:STATus:PORT<Pt>:WAVelength?
Description	This query returns the pluggable module laser wavelength.
Parameter	<Pt> = Port number
Response	<wavelength> = <STRING RESPONSE DATA> <wavelength max>,<wavelength min> = <STRING RESPONSE DATA>
Example	`PMOD:STAT:PORT1:WAV?` → `"1310 nm"` `PMOD:STAT:PORT1:WAV?` → `"1310 nm,1295 nm"`
Note	Response format is depende on pluggable module. CFP, CFP2, CFP4: <wavelength max>,<wavelength min> Others: <wavelength>

16.1.4 PMODule:STATus:PORT<Pt>:BRATe?

Syntax	PMODule:STATus:PORT<Pt>:BRATe?
Description	This query returns the pluggable module bit rate.
Parameter	<Pt> = Port number
Response	<bitrate> = <STRING RESPONSE DATA>
Example	`PMOD:STAT:PORT1:BRAT?` → `"2500 Mbps"`
Note	

16.1.5 PMODule:STATus:PORT<Pt>:COMPliance?

Syntax	PMODule:STATus:PORT<Pt>:COMPliance?
Description	This query returns the pluggable module specification compliance.
Parameter	<Pt> = Port number
Response	<compliance> ,<compliance>* = <STRING RESPONSE DATA>
Example	`PMOD:STAT:PORT1:COMP?` → `"10G Base-LR","VSR2000-3R2","VSR2000-3R2F"`
Note	

16.1.6 PMODule:STATus:PORT<Pt>:PNUMber?

Syntax	PMODule:STATus:PORT<Pt>:PNUMber?
Description	This query returns the vendor part number.
Parameter	<Pt> = Port number
Response	<partnumber> = <STRING RESPONSE DATA>
Example	`PMOD:STAT:PORT1:PNUM?` → `"ABC123"`
Note	

16.1.7 PMODule:STATus:PORT<Pt>:REVision?

Syntax	PMODule:STATus:PORT<Pt>:REVision?
Description	This query returns the vendor revision level for part number.
Parameter	<Pt> = Port number
Response	<revision> = <STRING RESPONSE DATA> <MSA version>,<HW version>,<SW version> = <STRING RESPONSE DATA>
Example	`PMOD:STAT:PORT1:REV?` → `"1A"` `PMOD:STAT:PORT1:REV?` → `"2.0,1.0,1.0"`
Note	Response format is depende on pluggable module. CFP, CFP2, CFP4: <MSA version>,<HW version>,<SW version> Others: <revision>

16.1.8 PMODule:STATus:PORT<Pt>:SNUMber?

Syntax	PMODule:STATus:PORT<Pt>:SNUMber?
Description	This query returns the vendor serial number.
Parameter	<Pt> = Port number
Response	<serialnumber> = <STRING RESPONSE DATA>
Example	`PMOD:STAT:PORT1:SNUM?` → `"ABC123"`
Note	

16.1.9 PMODule:STATus:PORT<Pt>:DATE?

Syntax	PMODule:STATus:PORT<Pt>:DATE?
Description	This query returns the vendor date code.
Parameter	<Pt> = Port number
Response	<date> = <STRING RESPONSE DATA>
Example	`PMOD:STAT:PORT1:DATE?` → `"2000-01-01"`
Note	

16.1.10 PMODule:STATus:PORT<Pt>:LOT?

Syntax	PMODule:STATus:PORT<Pt>:LOT?
Description	This query returns the vendor specific lot code.
Parameter	<Pt> = Port number
Response	<lot> = <STRING RESPONSE DATA>
Example	`PMOD:STAT:PORT1:LOT?` → `""`
Note	

16.1.11 PMODule:STATus:PORT<Pt>:TPOWer?

Syntax	PMODule:STATus:PORT<Pt>:TPOWer?
Description	This query returns the transmitter power.
Parameter	<Pt> = Port number
Response	<total_power>{,<lane_n_power>}* = <STRING RESPONSE DATA> <lane_n_power> will be responded when the optical module has multiple lanes.
Example	`PMOD:STAT:PORT1:TPOW?` → `"-1.00dBm"`
Note	

16.1.12 PMODule:STATus:PORT<Pt>:RPOWer?

Syntax	PMODule:STATus:PORT<Pt>:RPOWer?
Description	This query returns the receiver power.
Parameter	<Pt> = Port number
Response	<total_power>{,<lane_n_power>}* = <STRING RESPONSE DATA> <lane_n_power> will be responded when the optical module has multiple lanes.
Example	`PMOD:STAT:PORT1:RPOW?` → `"-40.00dBm"`
Note	

16.1.13 PMODule:STATus:PORT<Pt>:STATus?

Syntax	PMODule:STATus:PORT<Pt>:STATus?
Description	This query returns the Status of module.
Parameter	<Pt> = Port number
Response	<status> = <STRING RESPONSE DATA>
Example	`PMOD:STAT:PORT1:STAT?` → `"Data Ready"`
Note	

16.1.14 PMODule:STATus:PORT<Pt>:GALarm?

Syntax	PMODule:STATus:PORT<Pt>:GALarm?
Description	This query returns the Global Alarm of selected module.
Parameter	<Pt> = Port number
Response	<galarm> = <STRING RESPONSE DATA>
Example	`PMOD:STAT:PORT1:GAL?` → `"0"`
Note	This command can be used on CFP, CFP2, and CFP4.

16.1.15 PMODule:STATus:PORT<Pt>:PALarm?

Syntax	PMODule:STATus:PORT<Pt>:PALarm?
Description	This query returns the programmable alarm of selected module.
Parameter	<Pt> = Port number
Response	<palarm> = <STRING RESPONSE DATA>
Example	`PMOD:STAT:PORT1:PAL?` → `"0,0,0"`
Note	This command can be used on CFP, CFP2, and CFP4.

16.1.16 PMODule:STATus:PORT<Pt>:AESummary[:EVENt]?

Syntax	PMODule:STATus:PORT<Pt>:AESummary[:EVENt]?
Description	This query returns the pluggable module alarms and errors summary event register. The content of this event register is summarized in DB10 of the STATus:INTerface:PORT<Pt>:CONDition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 - DB16 = NOT USED
Example	`PMOD:STAT:PORT1:AES?` → `1`
Note	This command can be used on CFP, CFP2, and CFP4.

16.1.17 PMODule:STATus:PORT<Pt>:AESummary:CONDition?

Syntax	PMODule:STATus:PORT<Pt>:AESummary:CONDition?
Description	This query returns the pluggable module alarms and errors summary condition register. DB1 is summary of PMODule:STATus:PORT<Pt>:ALARm[:EVENt]?
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Alarm summary DB2 - DB16 = NOT USED
Example	`PMOD:STAT:PORT1:AES:COND?`
Note	This command can be used on CFP, CFP2, and CFP4.

16.1.18 PMODule:STATus:PORT<Pt>:ALARm[:EVENt]?

Syntax	PMODule:STATus:PORT<Pt>:ALARm[:EVENt]?
Description	This query returns the alarms event register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Module is not present DB2 (2) = NOT USED DB3 (4) = Global Alarm DB4 (8) = Programmable alarm 1 DB5 (16) = Programmable alarm 2 DB6 (32) = Programmable alarm 3 DB7 - DB16 = NOT USED
Example	`PMOD:STAT:PORT1:ALAR?`
Note	This command can be used on CFP, CFP2, and CFP4.

16.1.19 PMODule:STATus:PORT<Pt>:ALARm:CONDition?

Syntax	PMODule:STATus:PORT<Pt>:ALARm:CONDition?
Description	This query returns the alarms condition register.
Parameter	<Pt> = Port number
Response	<register> = <NR1 NUMERIC RESPONSE DATA> DB1 (1) = Module is not present DB2 (2) = NOT USED DB3 (4) = Global Alarm DB4 (8) = Programmable alarm 1 DB5 (16) = Programmable alarm 2 DB6 (32) = Programmable alarm 3 DB7 - DB16 = NOT USED
Example	`PMOD:STAT:PORT1:ALAR:COND?`
Note	This command can be used on CFP, CFP2, and CFP4.

16.2 MDIO

16.2.1 PMODule:PORT<Pt>:MDIO:WRITe

Syntax	PMODule:PORT<Pt>:MDIO:WRITe <addr>,<data>[,<trans>]
Description	MDIO write access.
Parameters	<Pt> = Port number
	<addr> = <HEXADECIMAL NUMERIC PROGRAM DATA> *MINimum=#H0000, MAXimum=#HFFFF, DEFault=#H0000*
	<data> = <HEXADECIMAL NUMERIC PROGRAM DATA> *MINimum=#H0000, MAXimum=#HFFFF, DEFault=#H0000*
	<trans> = <NUMERIC PROGRAM DATA> 0 : TX 1 : RX *DEFault = 0*
Example	`PMOD:PORT1:MDIO:WRIT #H1234,#HFFFF`
Note	This command can be used on V3.00 or later. This command support CFP, CFP2, and CFP4.

16.2.2 PMODule:PORT<Pt>:MDIO:READ?

Syntax	PMODule:PORT<Pt>:MDIO:READ? <addr>[,<trans>]
Description	MDIO read access.
Parameter	<Pt> = Port number
	<addr> = <HEXADECIMAL NUMERIC PROGRAM DATA> *MINimum=#H0000, MAXimum=#HFFFF, DEFault=#H0000*
	<trans> = <NUMERIC PROGRAM DATA> 0 : TX 1 : RX *DEFault = 0*
Response	<register> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`PMOD:PORT1:MDIO:READ? #H1234,0` → `#HFFFF`
Note	This command can be used on V3.00 or later. This command support CFP, CFP2, and CFP4.

16.3 I2C

16.3.1 PMODule:PORT<Pt>:I2C:WRITe

Syntax	PMODule:PORT<Pt>:I2C:WRITe <page>,<addr>,<data>[,<memory>]
Description	I2C write access.
Parameters	<Pt> = Port number
	<page> = <HEXADECIMAL NUMERIC PROGRAM DATA> *MINimum=#H00, MAXimum=#HFF, DEFault=#H00*
	<addr> = <HEXADECIMAL NUMERIC PROGRAM DATA> *MINimum=#H0000, MAXimum=#HFFFF, DEFault=#H0000*
	<data> = <HEXADECIMAL NUMERIC PROGRAM DATA> *MINimum=#H0000, MAXimum=#HFFFF, DEFault=#H0000* *DEFault = 0*
	<memory> = <NUMERIC PROGRAM DATA> #HA0 or #HA2 : SFP Memory Address. *DEFault = #A0*
Example	`PMOD:PORT1:I2C:WRIT #H00,12,#HFF`
Note	This command can be used on V7.02 or later. This command support SFP, SFP+, SFP28, QSFP+, QSFP28 and QSFP28 Adpt.

16.3.2 PMODule:PORT<Pt>:I2C:READ?

Syntax	PMODule:PORT<Pt>:I2C:READ? <page>,<addr>[,<memory>]
Description	I2C read access.
Parameter	<Pt> = Port number
	<addr> = <HEXADECIMAL NUMERIC PROGRAM DATA> *MINimum=#H0000, MAXimum=#HFFFF, DEFault=#H0000*
	<memory> = <NUMERIC PROGRAM DATA> #HA0 or #HA2 : SFP Memory Address. *DEFault = #HA0*
Response	<register> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`PMOD:PORT1:I2C:READ? #H00,12` → `#HFFFF`
Note	This command can be used on V7.02 or later. This command support SFP, SFP+, SFP28, QSFP+, QSFP28 and QSFP28 Adpt.

16.4 CFP2-QSFP28 Adaptor

16.4.1 PMODule:PORT<Pt>:MDIO:ADPT:WRITe

Syntax	PMODule:PORT<Pt>:MDIO:ADPT:WRITe <addr>,<data>
Description	MDIO write access in Adapter.
Parameters	<Pt> = Port number
	<addr> = <HEXADECIMAL NUMERIC PROGRAM DATA> *MINimum=#H0000, MAXimum=#HFFFF, DEFault=#H0000*
	<data> = <HEXADECIMAL NUMERIC PROGRAM DATA> *MINimum=#H0000, MAXimum=#HFFFF, DEFault=#H0000*
Example	`PMOD:PORT1:MDIO:ADPT:WRIT #H8000,#HFFFF`
Note	This command can be used with J1756A CFP2-QSFP28 Adaptor.

16.4.2 PMODule:PORT<Pt>:MDIO:ADPT:READ?

Syntax	PMODule:PORT<Pt>:MDIO:ADPT:READ? <addr>
Description	MDIO read access in Adapter.
Parameter	<Pt> = Port number
	<addr> = <HEXADECIMAL NUMERIC PROGRAM DATA> *MINimum=#H0000, MAXimum=#HFFFF, DEFault=#H0000*
Response	<register> = <HEXADECIMAL NUMERIC RESPONSE DATA>
Example	`PMOD:PORT1:MDIO:ADPT:READ? #H8000` → `#HFFFF`
Note	This command can be used with J1756A CFP2-QSFP28 Adaptor.

16.5 Tx reference clock output

16.5.1 PMODule:TOUTput:PORT<Pt>[:ENABle]

Syntax	PMODule:TOUTput:PORT<Pt>[:ENABle] <enable>
Description	This command enables/disables Tx reference clock output.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`PMOD:TOUT:PORT1 ON`
Note	Tx reference clock output can be enabled only one port on the same module. This command can be used on MT1100A.

Syntax	PMODule:TOUTput:PORT<Pt>[:ENABle]?
Description	This query returns the state of the Tx reference clock output.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`PMOD:TOUT:PORT1?` → `1`
Note	This command can be used on MT1100A.

16.5.2 PMODule:TOUTput:PORT<Pt>:FREQuency

Syntax	PMODule:TOUTput:PORT<Pt>:FREQuency <frequency>
Description	This command sets the Tx reference clock output frequency on this module.
Parameters	<Pt> = Port number
	<frequency> = <CHARACTER PROGRAM DATA> DIV16: 1/16 (Tx clock frequency divided by 16) DIV64: 1/64 (Tx clock frequency divided by 64) *DEFault = DIV16*
Response	None.
Example	`PMOD:TOUT:PORT1:FREQ DIV16`
Note	This setting applies to all ports on the same module. This command can be used on MT1100A.

Syntax	PMODule:TOUTput:PORT<Pt>:FREQuency?
Description	This query returns the Tx reference clock output frequency on this module.
Parameter	<Pt> = Port number
Response	<frequency> = <CHARACTER RESPONSE DATA>
Example	`PMOD:TOUT:PORT1:FREQ?` → `DIV16`
Note	This command can be used on MT1100A.

16.5.3 PMODule:TOUTput:PORT<Pt>:SYNC:FREQuency

Syntax	PMODule:TOUTput:PORT<Pt>:SYNC:FREQuency <frequency>
Description	This command sets the sync clock output frequency on this module.
Parameters	<Pt> = Port number
	<frequency> = <CHARACTER PROGRAM DATA> Off DIV8: 1/8 (Sync clock frequency divided by 8) DIV16: 1/16 (Sync clock frequency divided by 64) *DEFault = DIV16*
Response	None.
Example	`PMOD:TOUT:PORT1:SYNC:FREQ DIV8`
Note	This command can be used on MU110013A. This command can be used when interface type is CFP2 or QSFP28 Adpt..

Syntax	PMODule:TOUTput:PORT<Pt>:SYNC:FREQuency?
Description	This query returns the sync clock output frequency on this module.
Parameter	<Pt> = Port number
Response	<frequency> = <CHARACTER RESPONSE DATA>
Example	`PMOD:TOUT:PORT1:SYNC:FREQ?` → `DIV8`
Note	This command can be used on MU110013A.

16.6 tunable SFP

16.6.1 PMODule:PORT<Pt>:TSFP:CHANnel

Syntax	PMODule:PORT<Pt>:TSFP:CHANnel <channel>
Description	This command sets the channel about tunable SFP.
Parameters	<Pt> = Port number
	<channel> = <NR1 NUMERIC RESPONSE DATA> *MINimum=1, MAXimum=65536, DEFault=1*
Example	`PMOD:PORT1:TSFP:CHAN 10`
Note	This command can be used on V7.03 or later.

16.6.2 PMODule:PORT<Pt>:TSFP:CHANnel?

Syntax	PMODule:PORT<Pt>:TSFP:CHANnel?
Description	This query returns the channel about the tunabel SFP.
Parameter	<Pt> = Port number
Response	<channel> = <NR1 NUMERIC RESPONSE DATA>
Example	`PMOD:PORT1:TSFP:CHAN? → 10`
Note	This command can be used on V7.03 or later.

16.6.3 PMODule:PORT<Pt>:TSFP:WLENgth

Syntax	PMODule:PORT<Pt>:TSFP:WLENgth <channel>
Description	This command sets the wavelength about the tunable SFP.
Parameters	<Pt> = Port number
	<channel> = <NR3 NUMERIC RESPONSE DATA> *MINimum=1, MAXimum=65536, DEFault=1*
Example	`PMOD:PORT1:TSFP:WLEN 1556.55`
Note	This command can be used on V7.03 or later.

16.6.4 PMODule:PORT<Pt>:TSFP:WLENgth?

Syntax	PMODule:PORT<Pt>:TSFP:WLENgth?
Description	This query returns the wavelength about the tunabel SFP.
Parameter	<Pt> = Port number
Response	<channel> = <NR3 NUMERIC RESPONSE DATA>
Example	`PMOD:PORT1:TSFP:WLEN? → 1556.55`
Note	This command can be used on V7.03 or later.

16.7 Transceiver settings - CFP2-QSFP28 Adaptor

16.7.1 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:ATTN:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:ATTN:TRACking <enable>
Description	This command enables or disables the tracking for Tx attenuator.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:Q28A:TX:ATTN:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:ATTN:TRACking?
Description	This query returns enables or disables the tracking for Tx attenuator.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:Q28A:TX:ATTN:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.7.2 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:ATTN

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:ATTN <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx attenuator .
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:Q28A:TX:ATTN (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:ATTN?
Description	This query returns the parameter for Tx attenuator.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:Q28A:TX:ATTN?` → `(0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.7.3 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:PRE:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:PRE:TRACking <enable>
Description	This command enables or disables the tracking for Tx Pre .
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:Q28A:TX:PRE:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:PRE:TRACking?
Description	This query returns enables or disables the tracking for Tx Pre.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:Q28A:TX:PRE:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.7.4 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:PRE

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:PRE <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx PRE.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:Q28A:TX:PRE (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:PRE?
Description	This query returns the parameter for Tx PRE.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:Q28A:TX:PRE? → (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.7.5 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:POST:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:POST:TRACking <enable>
Description	This command enables or disables the tracking for Tx POST.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:Q28A:TX:POST:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:POST:TRACking?
Description	This query returns enables or disables the tracking for Tx POST.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:Q28A:TX:POST:TRAC? → OFF`
Note	This command can be used on V7.05 or later.

16.7.6 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:POST

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:POST <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx POST.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:Q28A:TX:POST (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:TX:POST?
Description	This query returns the parameter for Tx POST.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:Q28A:TX:POST? → (0)`
Note	This command can be used on V7.05 or later If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.7.7 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:EAUTo

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:EAUTo <enable>
Description	This command enables or disables the Auto RX equalizer.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:Q28A:RX:EAUT OFF`
Note	This command can be used on V7.05 or later and valid on MU110013A only.

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:EAUTo?
Description	This query returns enables or disables the Auto RX equalizer.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:Q28A:RX:EAUT?` → `OFF`
Note	This command can be used on V7.05 or later and valid on MU110013A only.

16.7.8 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:EQUalizer:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:EQUalizer:TRACking <enable>
Description	This command enables or disables the tracking for Rx Equalizer.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:Q28A:RX:EQU:TRAC OFF`
Note	This command can be used on V7.05 or later and valid on MU110013A only.

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:EQUalizer:TRACking?
Description	This query returns enables or disables tracking for Rx Equalizer.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:Q28A:RX:EQU:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later and valid on MU110013A only.

16.7.9 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:EQUalizer

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:EQUalizer <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the Rx equalizer value.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:Q28A:RX:EQU (0)`
Note	This command can be used on V7.05 or later and valid on MU110013A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:EQUalizer?
Description	This query returns the Rx equalizer value.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:Q28A:RX:EQU?` → `(0)`
Note	This command can be used on V7.05 or later and valid on MU110013A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.7.10 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:RESTore

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:RESTore
Description	This command Restore RX Equalizer.
Parameters	<Pt> = Port number
Example	`PMOD:PORT1:TRAN:Q28A:RX:REST`
Note	This command can be used on V7.05 or later.

16.7.11 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:AUTotune

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:RX:AUTotune
Description	This command execute autotune for RX Equalizer.
Parameters	<Pt> = Port number
Example	`PMOD:PORT1:TRAN:Q28A:RX:AUT`
Note	This command can be used on V7.05 or later.

16.7.12 PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:DEFault

Syntax	PMODule:PORT<Pt>:TRANsceiver:Q28Adpt:DEFault
Description	This command Restore default of Transceiver settings.
Parameters	<Pt> = Port number
Example	`PMOD:PORT1:TRAN:Q28A:DEF`
Note	This command can be used on V7.05 or later.

16.8 Transceiver settings - CFP2

16.8.1 PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:ATTN:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:ATTN:TRACking <enable>
Description	This command enables or disables the tracking for Tx attenuator.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CFP2:TX:ATTN:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:ATTN:TRACking?
Description	This query returns enables or disables the tracking for Tx attenuator.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP2:TX:ATTN:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.8.2 PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:ATTN

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:ATTN <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx attenuator.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CFP2:TX:ATTN (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:ATTN?
Description	This query returns the parameter for Tx attenuator.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP2:TX:ATTN?` → `(0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.8.3 PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:PRE:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:PRE:TRACking <enable>
Description	This command enables or disables the tracking for Tx Pre.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CFP2:TX:PRE:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:PRE:TRACking?
Description	This query returns enables or disables the tracking for Tx Pre.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP2:TX:PRE:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.8.4 PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:PRE

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:PRE <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx PRE.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CFP2:TX:PRE (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:PRE?
Description	This query returns the parameter for Tx PRE.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP2:TX:PRE?` → `(0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.8.5 PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:POST:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:POST:TRACking <enable>
Description	This command enables or disables the tracking for Tx POST.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CFP2:TX:POST:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:POST:TRACking?
Description	This query returns enables or disables the tracking for Tx POST.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP2:TX:POST:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.8.6 PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:POST

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:POST <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx POST.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CFP2:TX:POST (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:TX:POST?
Description	This query returns the parameter for Tx POST.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP2:TX:POST?` → `(0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.8.7 PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:EAUTo

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:EAUTo <enable>
Description	This command enables or disables the Auto RX equalizer.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CFP2:RX:EAUT OFF`
Note	This command can be used on V7.05 or later and valid on MU110013A only.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:EAUTo?
Description	This query returns enables or disables the Auto RX equalizer.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP2:RX:EAUT?` → `OFF`
Note	This command can be used on V7.05 or later and valid on MU110013A only.

16.8.8 PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:EQUalizer:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:EQUalizer:TRACking <enable>
Description	This command enables or disables the tracking for Rx Equalizer.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CFP2:RX:EQU:TRAC OFF`
Note	This command can be used on V7.05 or later and valid on MU110013A only.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:EQUalizer:TRACking?
Description	This query returns enables or disables tracking for Rx Equalizer.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP2:RX:EQU:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later and valid on MU110013A only.

16.8.9 PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:EQUalizer

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:EQUalizer <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the Rx equalizer value.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CFP2:RX:EQU (0)`
Note	This command can be used on V7.05 or later and valid on MU110013A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:EQUalizer?
Description	This query returns the Rx equalizer value.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP2:RX:EQU?` → `(0)`
Note	This command can be used on V7.05 or later and valid on MU110013A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.8.10 PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:RESTore

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:RESTore
Description	This command Restore RX Equalizer.
Parameters	<Pt> = Port number
Example	`PMOD:PORT1:TRAN:CFP2:RX:REST`
Note	This command can be used on V7.05 or later.

16.8.11 PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:AUTotune

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:RX:AUTotune
Description	This command execute autotune for RX Equalizer.
Parameters	<Pt> = Port number
Example	`PMOD:PORT1:TRAN:CFP2:RX:AUT`
Note	This command can be used on V7.05 or later.

16.8.12 PMODule:PORT<Pt>:TRANsceiver:CFP2:DEFault

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP2:DEFault
Description	This command Restore default of Transceiver settings.
Parameters	<Pt> = Port number
Example	`PMOD:PORT1:TRAN:CFP2:DEF`
Note	This command can be used on V7.05 or later.

16.9 Transceiver settings - CXP

16.9.1 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:VOD:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:VOD:TRACking <enable>
Description	This command enables or disables the tracking for Tx VOD.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CXP:TX:VOD:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:VOD:TRACking?
Description	This query returns enables or disables the tracking for Tx attenuator.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CXP:TX:VOD:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.9.2 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:VOD

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:VOD <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>,<Lane4 Data>,<Lane5 Data>,<Lane6 Data>,<Lane7 Data>,<Lane8 Data>,<Lane9 Data>]
Description	This command set the parameter for Tx VOD .
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CXP:TX:VOD (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:VOD?
Description	This query returns the parameter for Tx VOD.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CXP:TX:VOD?` → `(0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.9.3 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:FIRSt:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:FIRSt:TRACking <enable>
Description	This command enables or disables the tracking for Tx First.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CXP:TX:FIRS:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:FIRSt:TRACking?
Description	This query returns enables or disables the tracking for Tx First.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CXP:TX:FIRS:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.9.4 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:FIRSt

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:FIRSt <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>,<Lane4 Data>,<Lane5 Data>,<Lane6 Data>,<Lane7 Data>,<Lane8 Data>,<Lane9 Data>]
Description	This command set the parameter for Tx VOD .
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CXP:TX:FIRS (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:FIRSt?
Description	This query returns the parameter for Tx First.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CXP:TX:FIRS?` → `(0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.9.5 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:PRE:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:PRE:TRACking <enable>
Description	This command enables or disables the tracking for Tx PRE.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CXP:TX:PRE:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:PRE:TRACking?
Description	This query returns enables or disables the tracking for Tx PRE.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CXP:TX:PRE:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.9.6 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:PRE

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:PRE <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>,<Lane4 Data>,<Lane5 Data>,<Lane6 Data>,<Lane7 Data>,<Lane8 Data>,<Lane9 Data>]
Description	This command set the parameter for Tx PRE .
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CXP:TX:PRE (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:PRE?
Description	This query returns the parameter for Tx PRE.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CXP:TX:PRE?` → `(0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.9.7 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:SECond:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:SECond:TRACking <enable>
Description	This command enables or disables the tracking for Tx Second.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CXP:TX:SEC:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:SECond:TRACking?
Description	This query returns enables or disables the tracking for Tx Second.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CXP:TX:SEC:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.9.8 PMODule:PORT<Pt>:TRANsceiver:CXP:TX:SECond

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:SECond <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>,<Lane4 Data>,<Lane5 Data>,<Lane6 Data>,<Lane7 Data>,<Lane8 Data>,<Lane9 Data>]
Description	This command set the parameter for Tx Second .
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CXP:TX:SEC (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:TX:SECond?
Description	This query returns the parameter for Tx Second.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CXP:TX:SEC?` → `(0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.9.9 PMODule:PORT<Pt>:TRANsceiver:CXP:RX:CONTrol:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:RX:CONTrol:TRACking <enable>
Description	This command enables or disables the tracking for RX Control.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CXP:RX:CONT:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:RX:CONTrol:TRACking?
Description	This query returns enables or disables the tracking for RX Control.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CXP:RX:CONT:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.9.10 PMODule:PORT<Pt>:TRANsceiver:CXP:RX:CONTrol

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:RX:CONTrol <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>,<Lane4 Data>,<Lane5 Data>,<Lane6 Data>,<Lane7 Data>,<Lane8 Data>,<Lane9 Data>]
Description	This command set the parameter for RX Control .
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CXP:RX:CONT (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:RX:CONTrol?
Description	This query returns the parameter for RX Control.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CXP:RX:CONT?` → `(0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.9.11 PMODule:PORT<Pt>:TRANsceiver:CXP:RX:DCGain:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:RX:DCGain:TRACking <enable>
Description	This command enables or disables the tracking for RX DC Gain.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CXP:RX:DCG:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:RX:DCGain:TRACking?
Description	This query returns enables or disables the tracking for RX DC Gain.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CXP:RX:DCG:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.9.12 PMODule:PORT<Pt>:TRANsceiver:CXP:RX:DCGain

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:RX:DCGain <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>,<Lane4 Data>,<Lane5 Data>,<Lane6 Data>,<Lane7 Data>,<Lane8 Data>,<Lane9 Data>]
Description	This command set the parameter for RX DC Gain.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CXP:RX:DCG (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:RX:DCGain?
Description	This query returns the parameter for RX DC Gain.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CXP:RX:DCG? → (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.9.13 PMODule:PORT<Pt>:TRANsceiver:CXP:DEFault

Syntax	PMODule:PORT<Pt>:TRANsceiver:CXP:DEFault
Description	This command Restore default of Transceiver settings.
Parameters	<Pt> = Port number
Example	`PMOD:PORT1:TRAN:CXP:DEF`
Note	This command can be used on V7.05 or later.

16.10 Transceiver settings - CFP

16.10.1 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:VOD:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:VOD:TRACking <enable>
Description	This command enables or disables the tracking for Tx VOD.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CFP:TX:VOD:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:VOD:TRACking?
Description	This query returns enables or disables the tracking for Tx VOD.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP:TX:VOD:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.10.2 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:VOD

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:VOD <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data> [,<Lane4 Data>,<Lane5 Data>,<Lane6 Data>,<Lane7 Data>,<Lane8 Data>,<Lane9 Data>]]
Description	This command set the parameter for Tx VOD.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CFP:TX:VOD (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input 4 or 10 lane. The number of lane depends on the traffic rate.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:VOD?
Description	This query returns the parameter for Tx VOD.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP:TX:VOD?` → `(0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, 4 or 10 lane value will return. The number of lane depends on the traffic rate.

16.10.3 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:FIRSt:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:FIRSt:TRACking <enable>
Description	This command enables or disables the tracking for Tx First.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CFP:TX:FIRS:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:FIRSt:TRACking?
Description	This query returns enables or disables the tracking for Tx First.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP:TX:FIRS:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.10.4 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:FIRSt

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:FIRSt <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data> [,<Lane4 Data>,<Lane5 Data>,<Lane6 Data>,<Lane7 Data>,<Lane8 Data>,<Lane9 Data>]]
Description	This command set the parameter for Tx VOD.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CFP:TX:FIRS (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input 4 or 10 lane. The number of lane depends on the traffic rate.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:FIRSt?
Description	This query returns the parameter for Tx First.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP:TX:FIRS? → (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, 4 or 10 lane value will return. The number of lane depends on the traffic rate.

16.10.5 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:PRE:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:PRE:TRACking <enable>
Description	This command enables or disables the tracking for Tx PRE.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CFP:TX:PRE:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:PRE:TRACking?
Description	This query returns enables or disables the tracking for Tx PRE.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP:TX:PRE:TRAC? → OFF`
Note	This command can be used on V7.05 or later.

16.10.6 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:PRE

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:PRE <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data> [,<Lane4 Data>,<Lane5 Data>,<Lane6 Data>,<Lane7 Data>,<Lane8 Data>,<Lane9 Data>]]
Description	This command set the parameter for Tx PRE.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CFP:TX:PRE (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input 4 or 10 lane. The number of lane depends on the traffic rate.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:PRE?
Description	This query returns the parameter for Tx PRE.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP:TX:PRE?` → `(0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, 4 or 10 lane value will return. The number of lane depends on the traffic rate.

16.10.7 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:SECond:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:SECond:TRACking <enable>
Description	This command enables or disables the tracking for Tx Second.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CFP:TX:SEC:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:SECond:TRACking?
Description	This query returns enables or disables the tracking for Tx Second.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP:TX:SEC:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.10.8 PMODule:PORT<Pt>:TRANsceiver:CFP:TX:SECond

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:SECond <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data> [,<Lane4 Data>,<Lane5 Data>,<Lane6 Data>,<Lane7 Data>,<Lane8 Data>,<Lane9 Data>]]
Description	This command set the parameter for Tx Second.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CFP:TX:SEC (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input 4 or 10 lane. The number of lane depends on the traffic rate.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:TX:SECond?
Description	This query returns the parameter for Tx Second.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP:TX:SEC?` → `(0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, 4 or 10 lane value will return. The number of lane depends on the traffic rate.

16.10.9 PMODule:PORT<Pt>:TRANsceiver:CFP:RX:CONTrol:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:RX:CONTrol:TRACking <enable>
Description	This command enables or disables the tracking for RX Control.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CFP:RX:CONT:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:RX:CONTrol:TRACking?
Description	This query returns enables or disables the tracking for RX Control.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP:RX:CONT:TRAC? → OFF`
Note	This command can be used on V7.05 or later.

16.10.10 PMODule:PORT<Pt>:TRANsceiver:CFP:RX:CONTrol

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:RX:CONTrol <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data> [,<Lane4 Data>,<Lane5 Data>,<Lane6 Data>,<Lane7 Data>,<Lane8 Data>,<Lane9 Data>]]
Description	This command set the parameter for RX Control.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CFP:RX:CONT (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input 4 or 10 lane. The number of lane depends on the traffic rate.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:RX:CONTrol?
Description	This query returns the parameter for RX Control.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP:RX:CONT? → (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, 4 or 10 lane value will return. The number of lane depends on the traffic rate.

16.10.11 PMODule:PORT<Pt>:TRANsceiver:CFP:RX:DCGain:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:RX:DCGain:TRACking <enable>
Description	This command enables or disables the tracking for RX DC Gain.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CFP:RX:DCG:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:RX:DCGain:TRACking?
Description	This query returns enables or disables the tracking for RX DC Gain.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP:RX:DCG:TRAC? → OFF`
Note	This command can be used on V7.05 or later.

16.10.12 PMODule:PORT<Pt>:TRANsceiver:CFP:RX:DCGain

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:RX:DCGain <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data> [,<Lane4 Data>,<Lane5 Data>,<Lane6 Data>,<Lane7 Data>,<Lane8 Data>,<Lane9 Data>]]
Description	This command set the parameter for RX DC Gain.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CFP:RX:DCG (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input 4 or 10 lane. The number of lane depends on the traffic rate.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:RX:DCGain?
Description	This query returns the parameter for RX DC Gain.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP:RX:DCG? → (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, 4 or 10 lane value will return. The number of lane depends on the traffic rate.

16.10.13 PMODule:PORT<Pt>:TRANsceiver:CFP:DEFault

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP:DEFault
Description	This command Restore default of Transceiver settings.
Parameters	<Pt> = Port number
Example	`PMOD:PORT1:TRAN:CFP:DEF`
Note	This command can be used on V7.05 or later.

16.11 Transceiver settings - QSFP

16.11.1 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:VOD:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:VOD:TRACking <enable>
Description	This command enables or disables the tracking for Tx VOD.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:QSFP:TX:VOD:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:VOD:TRACking?
Description	This query returns enables or disables the tracking for Tx VOD.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:TX:VOD:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.11.2 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:VOD

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:VOD <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx VOD.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:QSFP:TX:VOD (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:VOD?
Description	This query returns the parameter for Tx VOD.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:TX:VOD?` → `(0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.11.3 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:FIRSt:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:FIRSt:TRACking <enable>
Description	This command enables or disables the tracking for Tx First.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:QSFP:TX:FIRS:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:FIRSt:TRACking?
Description	This query returns enables or disables the tracking for Tx First.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:TX:FIRS:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.11.4 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:FIRSt

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:FIRSt <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx VOD.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:QSFP:TX:FIRS (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:FIRSt?
Description	This query returns the parameter for Tx First.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:TX:FIRS? → (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.11.5 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:PRE:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:PRE:TRACking <enable>
Description	This command enables or disables the tracking for Tx PRE.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:QSFP:TX:PRE:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:PRE:TRACking?
Description	This query returns enables or disables the tracking for Tx PRE.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:TX:PRE:TRAC? → OFF`
Note	This command can be used on V7.05 or later.

16.11.6 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:PRE

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:PRE <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx PRE.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:QSFP:TX:PRE (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:PRE?
Description	This query returns the parameter for Tx PRE.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:TX:PRE? → (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.11.7 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:SECond:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:SECond:TRACking <enable>
Description	This command enables or disables the tracking for Tx Second.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:QSFP:TX:SEC:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:SECond:TRACking?
Description	This query returns enables or disables the tracking for Tx Second.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:TX:SEC:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.11.8 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:SECond

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:SECond <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx Second.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:QSFP:TX:SEC (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:SECond?
Description	This query returns the parameter for Tx Second.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:TX:SEC?` → `(0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.11.9 PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:CONTrol:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:CONTrol:TRACking <enable>
Description	This command enables or disables the tracking for RX Control.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:QSFP:RX:CONT:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:CONTrol:TRACking?
Description	This query returns enables or disables the tracking for RX Control.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:RX:CONT:TRAC?` → `OFF`
Note	This command can be used on V7.05 or later.

16.11.10 PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:CONTrol

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:CONTrol <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for RX Control.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:QSFP:RX:CONT (0)`
Note	This command can be used on V7.05 or later. If you set tracking is OFF, you should input all lane. if tracking is ON, you should input only first lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:CONTrol?
Description	This query returns the parameter for RX Control.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:RX:CONT? → (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.11.11 PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DCGain:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DCGain:TRACking <enable>
Description	This command enables or disables the tracking for RX DC Gain.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:QSFP:RX:DCG:TRAC OFF`
Note	This command can be used on V7.05 or later.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DCGain:TRACking?
Description	This query returns enables or disables the tracking for RX DC Gain.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:RX:DCG:TRAC? → OFF`
Note	This command can be used on V7.05 or later.

16.11.12 PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DCGain

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DCGain <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for RX DC Gain.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:QSFP:RX:DCG (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DCGain?
Description	This query returns the parameter for RX DC Gain.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:RX:DCG? → (0)`
Note	This command can be used on V7.05 or later. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.11.13 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:POST:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:POST:TRACking <enable>
Description	This command enables or disables the tracking for Tx POST.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:QSFP:TX:POST:TRAC OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:POST:TRACking?
Description	This query returns enables or disables the tracking for Tx POST.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:TX:POST:TRAC?` → `OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

16.11.14 PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:POST

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:POST <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx POST.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:QSFP:TX:POST (0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:TX:POST?
Description	This query returns the parameter for Tx POST.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:TX:POST?` → `(0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.11.15 PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DFE:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DFE:TRACking <enable>
Description	This command enables or disables the tracking for Rx Equalizer.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:QSFP:RX:DFE:TRAC OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DFE:TRACking?
Description	This query returns enables or disables tracking for Rx DFE.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:RX:DFE:TRAC?` → `OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

16.11.16 PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DFE

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DFE <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the Rx DFE value.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:QSFP:RX:DFE (0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:RX:DFE?
Description	This query returns the Rx DFE value.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP:RX:DFE?` → `(0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.11.17 PMODule:PORT<Pt>:TRANsceiver:QSFP:DEFault

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP:DEFault
Description	This command Restore default of Transceiver settings.
Parameters	<Pt> = Port number
Example	`PMOD:PORT1:TRAN:QSFP:DEF`
Note	This command can be used on V7.05 or later.

16.12 Transceiver settings - SFP28

16.12.1 PMODule:PORT<Pt>:TRANsceiver:SFP28:TX:VOD

Syntax	PMODule:PORT<Pt>:TRANsceiver:SFP28:TX:VOD <Lane0 Data>
Description	This command set the parameter for Tx VOD.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:SFP28:TX:VOD (0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:SFP28:TX:VOD?
Description	This query returns the parameter for Tx VOD.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:SFP28:TX:VOD?` → `(0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.12.2 PMODule:PORT<Pt>:TRANsceiver:SFP28:TX:PRE

Syntax	PMODule:PORT<Pt>:TRANsceiver:SFP28:TX:PRE <Lane0 Data>
Description	This command set the parameter for Tx PRE.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:SFP28:TX:PRE (0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:SFP28:TX:PRE?
Description	This query returns the parameter for Tx PRE.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:SFP28:TX:PRE?` → `(0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.12.3 PMODule:PORT<Pt>:TRANsceiver:SFP28:TX:POST

Syntax	PMODule:PORT<Pt>:TRANsceiver:SFP28:TX:POST <Lane0 Data>
Description	This command set the parameter for Tx POST.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:SFP28:TX:POST (0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:SFP28:TX:POST?
Description	This query returns the parameter for Tx POST.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:SFP28:TX:POST?` → `(0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.12.4 PMODule:PORT<Pt>:TRANsceiver:SFP28:RX:DFE

Syntax	PMODule:PORT<Pt>:TRANsceiver:SFP28:RX:DFE <Lane0 Data>
Description	This command set the Rx DFE value.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:SFP28:RX:DFE (0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:SFP28:RX:DFE?
Description	This query returns the Rx DFE value.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:SFP28:RX:DFE?` → `(0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.12.5 PMODule:PORT<Pt>:TRANsceiver:SFP28:DEFault

Syntax	PMODule:PORT<Pt>:TRANsceiver:SFP28:DEFault
Description	This command Restore default of Transceiver settings.
Parameters	<Pt> = Port number
Example	`PMOD:PORT1:TRAN:SFP28:DEF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

16.13 Transceiver settings - CFP4

16.13.1 PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:VOD:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:VOD:TRACking <enable>
Description	This command enables or disables the tracking for Tx VOD.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CFP4:TX:VOD:TRAC OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:VOD:TRACking?
Description	This query returns enables or disables the tracking for Tx VOD.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP4:TX:VOD:TRAC?` → `OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

16.13.2 PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:VOD

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:VOD <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx VOD.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CFP4:TX:VOD (0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:VOD?
Description	This query returns the parameter for Tx VOD.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP4:TX:VOD?` → `(0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.13.3 PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:PRE:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:PRE:TRACking <enable>
Description	This command enables or disables the tracking for Tx Pre.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CFP4:TX:PRE:TRAC OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:PRE:TRACking?
Description	This query returns enables or disables the tracking for Tx Pre.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP4:TX:PRE:TRAC?` → `OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

16.13.4 PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:PRE

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:PRE <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx PRE.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CFP4:TX:PRE (0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:PRE?
Description	This query returns the parameter for Tx PRE.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP4:TX:PRE?` → `(0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.13.5 PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:POST:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:POST:TRACking <enable>
Description	This command enables or disables the tracking for Tx POST.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CFP4:TX:POST:TRAC OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:POST:TRACking?
Description	This query returns enables or disables the tracking for Tx POST.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP4:TX:POST:TRAC?` → `OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

16.13.6 PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:POST

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:POST <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx POST.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CFP4:TX:POST (0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:TX:POST?
Description	This query returns the parameter for Tx POST.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP4:TX:POST?` → `(0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.13.7 PMODule:PORT<Pt>:TRANsceiver:CFP4:RX:DFE:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:RX:DFE:TRACking <enable>
Description	This command enables or disables the tracking for Rx DFE.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:CFP4:RX:DFE:TRAC OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:RX:DFE:TRACking?
Description	This query returns enables or disables tracking for Rx DFE.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP4:RX:DFE:TRAC?` → `OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

16.13.8 PMODule:PORT<Pt>:TRANsceiver:CFP4:RX:DFE

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:RX:DFE <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the Rx DFE value.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:CFP4:RX:DFE (0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:RX:DFE
Description	This query returns the Rx DFE value.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:CFP4:RX:DFE?` → `(0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.13.9 PMODule:PORT<Pt>:TRANsceiver:CFP4:DEFault

Syntax	PMODule:PORT<Pt>:TRANsceiver:CFP4:DEFault
Description	This command Restore default of Transceiver settings.
Parameters	<Pt> = Port number
Example	`PMOD:PORT1:TRAN:CFP4:DEF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

16.14 Transceiver settings - QSFP28

16.14.1 PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:VOD:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:VOD:TRACking <enable>
Description	This command enables or disables the tracking for Tx VOD.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:QSFP28:TX:VOD:TRAC OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:VOD:TRACking?
Description	This query returns enables or disables the tracking for Tx VOD.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP28:TX:VOD:TRAC?` → `OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

16.14.2 PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:VOD

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:VOD <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx VOD.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:QSFP28:TX:VOD (0)`
Note	This command can be used on VV8.00 or later and valid on MU100011A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:VOD?
Description	This query returns the parameter for Tx VOD.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP28:TX:VOD?` → `(0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.14.3 PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:PRE:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:PRE:TRACking <enable>
Description	This command enables or disables the tracking for Tx Pre.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:QSFP28:TX:PRE:TRAC OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:PRE:TRACking?
Description	This query returns enables or disables the tracking for Tx Pre.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP28:TX:PRE:TRAC?` → `OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

16.14.4 PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:PRE

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:PRE <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx PRE.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:QSFP28:TX:PRE (0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:PRE?
Description	This query returns the parameter for Tx PRE.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP28:TX:PRE?` → `(0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.14.5 PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:POST:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:POST:TRACking <enable>
Description	This command enables or disables the tracking for Tx POST.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:QSFP28:TX:POST:TRAC OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:POST:TRACking?
Description	This query returns enables or disables the tracking for Tx POST.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP28:TX:POST:TRAC?` → `OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

16.14.6 PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:POST

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:POST <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the parameter for Tx POST.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:QSFP28:TX:POST (0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:TX:POST?
Description	This query returns the parameter for Tx POST.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP28:TX:POST? → (0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.14.7 PMODule:PORT<Pt>:TRANsceiver:QSFP28:RX:DFE:TRACking

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:RX:DFE:TRACking <enable>
Description	This command enables or disables the tracking for Rx DFE.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Example	`PMOD:PORT1:TRAN:QSFP28:RX:DFE:TRAC OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:RX:DFE:TRACking?
Description	This query returns enables or disables tracking for Rx DFE.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP28:RX:DFE:TRAC? → OFF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

16.14.8 PMODule:PORT<Pt>:TRANsceiver:QSFP28:RX:DFE

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:RX:DFE <Lane0 Data> [,<Lane1 Data>,<Lane2 Data>,<Lane3 Data>]
Description	This command set the Rx DFE value.
Parameters	<Pt> = Port number
	{(<value>), }* = <EXPRESSION PROGRAM DATA> Format: Numeric List
Example	`PMOD:PORT1:TRAN:QSFP28:RX:DFE (0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, you should input only first lane. If you set tracking is OFF, you should input all lane.

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:RX:DFE?
Description	This query returns the Rx DFE value.
Parameter	<Pt> = Port number
Response	{(<value>), }* = <EXPRESSION PROGRAM DATA>
Example	`PMOD:PORT1:TRAN:QSFP28:RX:DFE?` → `(0)`
Note	This command can be used on V8.00 or later and valid on MU100011A only. If tracking is ON, the first lane value will return. If you set tracking is OFF, all lane value will return.

16.14.9 PMODule:PORT<Pt>:TRANsceiver:QSFP28:DEFault

Syntax	PMODule:PORT<Pt>:TRANsceiver:QSFP28:DEFault
Description	This command Restore default of Transceiver settings.
Parameters	<Pt> = Port number
Example	`PMOD:PORT1:TRAN:QSFP28:DEF`
Note	This command can be used on V8.00 or later and valid on MU100011A only.

Chapter 17

Common Application Settings

17.1 RTD

The following commands are valid for all Round Trip Delay applications unless otherwise noted.

17.1.1 RTD:MODE

Syntax	RTD:MODE <mode>
Description	This command sets the measurement mode.
Parameters	<mode> = <CHARACTER PROGRAM DATA> SINGle REPeat *DEFault = REPeat*
Response	None.
Example	`RTD:MODE REP`
Note	

Syntax	RTD:MODE?
Description	This query returns the measurement mode.
Parameter	None.
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`RTD:MODE?` → `REP`
Note	

17.1.2 RTD:PERiod

Syntax	RTD:PERiod <period>
Description	This command sets the measurement period.
Parameters	<period> = <CHARACTER PROGRAM DATA> MS500: 500 milli seconds SEC1: 1 second SEC2: 2 seconds SEC5: 5 seconds SEC10: 10 seconds *DEFault = SEC1*
Response	None.
Example	`RTD:PER SEC1`
Note	

Syntax	RTD:PERiod?
Description	This query returns the measurement period.
Parameter	None.
Response	<period> = <CHARACTER RESPONSE DATA>
Example	`RTD:PER?` → `SEC1`
Note	

17.1.3 RTD:IFMData

Syntax	RTD:IFMData <ignore>
Description	
Parameters	<ignore> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`RTD:IFMD OFF`
Note	

Syntax	RTD:IFMData?
Description	
Parameter	None.
Response	<ignore> = <BOOLEAN RESPONSE DATA>
Example	`RTD:IFMD?` → `0`
Note	

17.1.4 RTD:PDH:RX<Pt>:MLIMit

Syntax	RTD:PDH:RX<Pt>:MLIMit <max>
Description	This command sets the time reference maximum limit for all PDH interfaces. Unit: us.
Parameters	<Pt> = Port number
	<max> = <NUMERIC PROGRAM DATA> *MINimum = 0.0, MAXimum = 10000000.0, DEFault = MAXimum*
Response	None.
Example	`RTD:PDH:RX1:MLIM 0.0`
Note	This command is not valid for a RTD_OTN application.

Syntax	RTD:PDH:RX<Pt>:MLIMit?
Description	This query returns the time reference maximum limit for all PDH interfaces. Unit: us.
Parameter	<Pt> = Port number
Response	<max> = <NR2 NUMERIC RESPONSE DATA>
Example	`RTD:PDH:RX1:MLIM?` → `0.0`
Note	This command is not valid for a RTD_OTN application.

17.2 Hierarchy

17.2.1 HIERarchy:PORT<Pt>:OTN:TX

Syntax	HIERarchy:PORT<Pt>:OTN:TX <enable>
Description	This command enables or disables the OTN Tx hierarchy.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`HIER:PORT1:OTN:TX ON`
Note	

Syntax	HIERarchy:PORT<Pt>:OTN:TX?
Description	This query returns whether or not OTN Tx hierarchy is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`HIER:PORT1:OTN:TX? → 1`
Note	

17.2.2 HIERarchy:PORT<Pt>:OTN:RX

Syntax	HIERarchy:PORT<Pt>:OTN:RX <enable>
Description	This command enables or disables the OTN Rx hierarchy.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`HIER:PORT1:OTN:RX ON`
Note	

Syntax	HIERarchy:PORT<Pt>:OTN:RX?
Description	This query returns whether or not OTN Rx hierarchy is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`HIER:PORT1:OTN:RX? → 1`
Note	

17.2.3 HIERarchy:PORT<Pt>:SDH:TX

Syntax	HIERarchy:PORT<Pt>:SDH:TX <enable>
Description	This command enables or disables the SDH Tx hierarchy.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`HIER:PORT1:SDH:TX ON`
Note	

Syntax	HIERarchy:PORT<Pt>:SDH:TX?
Description	This query returns whether or not SDH Tx hierarchy is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`HIER:PORT1:SDH:TX? → 1`
Note	

17.2.4 HIERarchy:PORT<Pt>:SDH:RX

Syntax	HIERarchy:PORT<Pt>:SDH:RX <enable>
Description	This command enables or disables the SDH Rx hierarchy.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`HIER:PORT1:SDH:RX ON`
Note	

Syntax	HIERarchy:PORT<Pt>:SDH:RX?
Description	This query returns whether or not SDH Rx hierarchy is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`HIER:PORT1:SDH:RX? → 1`
Note	

17.2.5 HIERarchy:PORT<Pt>:PDH:TX

Syntax	HIERarchy:PORT<Pt>:PDH:TX <enable>
Description	This command enables or disables the PDH Tx hierarchy.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`HIER:PORT1:PDH:TX ON`
Note	

Syntax	HIERarchy:PORT<Pt>:PDH:TX?
Description	This query returns whether or not PDH Tx hierarchy is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`HIER:PORT1:PDH:TX? → 1`
Note	

17.2.6 HIERarchy:PORT<Pt>:PDH:RX

Syntax	HIERarchy:PORT<Pt>:PDH:RX <enable>
Description	This command enables or disables the PDH Rx hierarchy.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`HIER:PORT1:PDH:RX ON`
Note	

Syntax	HIERarchy:PORT<Pt>:PDH:RX?
Description	This query returns whether or not PDH Rx hierarchy is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Example	`HIER:PORT1:PDH:RX? → 1`
Note	

Chapter 18

Measurement

18.1 Application, Start and Stop

18.1.1 MEASurement:APPLication?

Syntax	MEASurement:APPLication?
Description	This query returns the application server type.
Parameter	None.
Response	<application> = <CHARACTER RESPONSE DATA> TP-APS-OTN: OTN Automatic Protection Switching application. TP-APS-SDHPDH: SDH/PDH Automatic Protection Switching application. TP-APS-SDHPDH-OTN: SDH/PDH over OTN Automatic Protection Switching application. TP-BERT-CPRI: CPRI Bit Error Rate Test application. TP-BERT-CPRI-OTN: CPRI over OTN Bit Error Rate Test application. TP-BERT-ETH: Ethernet Bit Error Rate Test application. TP-BERT-ETH-OTN: Ethernet over OTN Bit Error Rate Test application. TP-BERT-FC: Fibre Channel Bit Error Rate Test application. TP-BERT-FC-OTN: Fibre Channel over OTN Bit Error Rate Test application. TP-BERT-OTN: OTN Bit Error Rate Test application. TP-BERT-SDHPDH: PDH/SDH Bit Error Rate Test application. TP-BERT-SDHPDH-OTN: PDH/SDH over OTN Bit Error Rate Test application. TP-BERT-ROE: eCPRI/RoE Bit Error Rate Test application. TP-CABLE-ETH: Ethernet cable test application. TP-CHSTAT-ETH: Ethernet channel statistics application. TP-DISC-ETH: Ethernet discovery application. TP-MONGEN-ETH: Ethernet monitor/generate application. TP-MONGEN-ETH-OTN: Ethernet over OTN monitor/generate application. TP-NOFRAME-DEVICE: No frame device test (Unframed Bit Error Rate Test) application. TP-PASS-CPRI: CPRI pass-through application. TP-PASS-ETH: Ethernet pass-through application. TP-PERF-FC: Fibre Channel performance test application. TP-PERF-FC-OTN: Fibre Channel over OTN performance test application. TP-PING-ETH: Ethernet ICMP ping application. TP-REFL-ETH: Ethernet reflector application. TP-REFL-ETH-OTN: Ethernet over OTN reflector application. TP-REFL-FC: Fibre Channel reflector application. TP-REFL-FC-OTN: Fibre Channel over OTN reflector application. TP-RFC-ETH: Ethernet RFC-2544 test application. TP-RFC-ETH-OTN: Ethernet over OTN RFC-2544 test application. TP-RFC6349-ETH: Ethernet RFC-6349 test application. TP-RTD-OTN: OTN Round Trip Delay test application. TP-RTD-SDHPDH: SDH/PDH Round Trip Delay test application. TP-RTD-SDHPDH-OTN: SDH/PDH Round Trip Delay test application. TP-SAT-ETH: Ethernet Service Activation Test application. TP-SAT-ETH-OTN: Ethernet over OTN Service Activation Test application. TP-SYNCTEST-ETH: Ethernet sync test application. TP-TRACE-ETH: Ethernet trace-route application. OTDR-OTDR: Standard OTDR application. OSA-WDM: OSA WDM application.
Example	`MEAS:APPL?` → `TP-BERT-ETH`
Note	

18.1.2 MEASurement:STARt

Syntax	MEASurement:STARt
Description	This command starts a measurement. Same as pressing the START button on the GUI.
Parameter	None.
Response	None.
Example	`MEAS:STAR`
Note	

18.1.3 MEASurement:STOP

Syntax	MEASurement:STOP
Description	This command stops an ongoing measurement. Same as pressing the STOP button on the GUI.
Parameter	None.
Response	None.
Example	`MEAS:STOP`
Note	

18.2 Setup

18.2.1 MEASurement:SETup:PORT<Pt>:TERMinology

Syntax	MEASurement:SETup:PORT<Pt>:TERMinology <mode>
Description	This command sets the SDH/SONET terminology.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> SONet SDH *DEFault = SDH*
Response	None.
Example	`MEAS:SET:PORT1:TERM SDH`
Note	

Syntax	MEASurement:SETup:PORT<Pt>:TERMinology?
Description	This query returns the SDH/SONET terminology.
Parameter	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:PORT1:TERM?` → `SDH`
Note	

18.2.2 MEASurement:SETup:SELect

Syntax	MEASurement:SETup:SELect <interval>
Description	This command selects which interval to fetch. After setting this command, the :IFETch on the respective interfaces should be performed.
Parameter	<interval> = <CHARACTER PROGRAM DATA> TOTal: Total CURRent: Current Or the index number of the interval. With the first index number being 0. *DEFault = TOTal*
Response	None.
Example	`MEAS:SET:SEL 5`
Note	Use MEAS:INFO:IFIR? and MEAS:INFO:ILAS? to get the minimum and maximum interval number. The MEAS:INFO:IMIN? and MEAS:INFO:IMAX? queries are kept for maintenance purpose. Recommend using MEAS:INFO:IFIR? and MEAS:INFO:ILAS? queries instead of MEAS:INFO:IMIN? and MEAS:INFO:IMAX?, because they return +1 larger value than recommended queries. This setting is not stored as part of a settings- or result file.

Syntax	MEASurement:SETup:SELect?
Description	This query returns the selected interval.
Parameter	None.
Response	<interval> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:SEL?` → `5`
Note	

18.2.3 MEASurement:SETup:ILENgth

Syntax	MEASurement:SETup:ILENgth <length>
Description	This command sets the interval length.
Parameter	<length> = <CHARACTER PROGRAM DATA> 1S: 1 second 2S: 2 seconds 5S: 5 seconds 10S: 10 seconds 15S: 15 seconds 30S: 30 seconds 1M: 1 minute 5M: 5 minutes 10M: 10 minutes 15M: 15 minutes 30M: 30 minutes 1H: 1 hour 2H: 2 hours 4H: 4 hours 6H: 6 hours 12H: 12 hours NONE: No intervals *DEFault = 5S*
Response	None.
Example	`MEAS:SET:ILEN 1S`
Note	

Syntax	MEASurement:SETup:ILENgth?
Description	This query returns the interval length.
Parameter	None.
Response	<action> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:ILEN?` → `1S`
Note	

18.2.4 MEASurement:SETup:STARt

Syntax	MEASurement:SETup:STARt <action>
Description	This command sets how the measurement should start.
Parameter	<action> = <CHARACTER PROGRAM DATA> IMMediate: Start measurement immediate. SAT: Start measurement at a specific time. *DEFault = IMMediate*
Response	None.
Example	`MEAS:SET:STAR SAT`
Note	Use the MEASurement:SETup:STAT command to set the specific start time.

Syntax	MEASurement:SETup:STARt?
Description	This query returns how the measurement starts.
Parameter	None.
Response	<action> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:STAR?` → `SAT`
Note	

18.2.5 MEASurement:SETup:STAT

Syntax	MEASurement:SETup:STAT <year>,<month>,<day>,<hour>,<minute>,<second>
Description	This command sets the start time for the MEASurement:SETup:STARt SAT command.
Parameters	<year> = <NUMERIC PROGRAM DATA> *MINimum = 1997, MAXimum = 2036*
	<month> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 12*
	<day> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 31*
	<hour> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 23*
	<minute> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 59*
	<second> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 59*
Response	None.
Example	`MEAS:SET:STAT 2015,7,16,10,11,39`
Note	

Syntax	MEASurement:SETup:STAT?
Description	This query returns the start time for the MEASurement:SETup:STARt SAT command
Parameter	None.
Response	<year> = <NR1 NUMERIC RESPONSE DATA>
	<month> = <NR1 NUMERIC RESPONSE DATA>
	<day> = <NR1 NUMERIC RESPONSE DATA>
	<hour> = <NR1 NUMERIC RESPONSE DATA>
	<minute> = <NR1 NUMERIC RESPONSE DATA>
	<second> = <NR1 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:STAT?` → `2015,7,16,10,11,39`
Note	If the SAT parameter is not set in the MEASurement:SETup:STARt command, this query returns NaN (section 1.6.1).

18.2.6 MEASurement:SETup:STOP

Syntax	MEASurement:SETup:STOP <action>
Description	This command sets how the measurement should stop.
Parameter	<action> = <CHARACTER PROGRAM DATA> MANual: Manual stop the measurement. SAT: Stop measurement at a specific time. DURation: Stop measurement after a specific duration. *DEFault = MANual*
Response	None.
Example	`MEAS:SET:STOP DUR`
Note	To set the specific stop time, use the MEASurement:SETup:SPAT command.

Syntax	MEASurement:SETup:STOP?
Description	This query returns how the measurement stops.
Parameter	None.
Response	<action> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:STOP?` → `DUR`
Note	

18.2.7 MEASurement:SETup:SPAT

Syntax	MEASurement:SETup:SPAT <year>,<month>,<day>,<hour>,<minute>,<second>
Description	This command sets the stop time for the MEASurement:SETup:STOP SAT command
Parameters	<year> = <NUMERIC PROGRAM DATA> *MINimum = 1997, MAXimum = 2036*
	<month> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 12*
	<day> = <NUMERIC PROGRAM DATA> *MINimum = 1, MAXimum = 31*
	<hour> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 23*
	<minute> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 59*
	<second> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 59*
Response	None.
Example	`MEAS:SET:SPAT 2015,7,16,10,12,39`
Note	

Syntax	MEASurement:SETup:SPAT?
Description	This query returns the stop time for the MEASurement:SETup:STOP SAT command
Parameter	None.
Response	<year> = <NR1 NUMERIC RESPONSE DATA>
	<month> = <NR1 NUMERIC RESPONSE DATA>
	<day> = <NR1 NUMERIC RESPONSE DATA>
	<hour> = <NR1 NUMERIC RESPONSE DATA>
	<minute> = <NR1 NUMERIC RESPONSE DATA>
	<second> = <NR1 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:SPAT?` → `2015,7,16,10,12,39`
Note	If the SAT parameter is not set in the MEASurement:SETup:STOP command, this query returns NaN (section 1.6.1).

18.2.8 MEASurement:SETup:SDURation

Syntax	MEASurement:SETup:SDURation <days>,<hours>,<minutes>,<seconds>
Description	This command sets the duration time for the MEASurement:SETup:STOP DUR command.
Parameters	<days> = <NUMERIC PROGRAM DATA>
	<hours> = <NUMERIC PROGRAM DATA>
	<minutes> = <NUMERIC PROGRAM DATA>
	<seconds> = <NUMERIC PROGRAM DATA>
Response	None.
Example	`MEAS:SET:SDUR 16,0,0,0`
Note	

Syntax	MEASurement:SETup:SDURation?
Description	This query return the duration time for the MEASurement:SETup:STOP DUR command
Parameter	None.
Response	<days> = <NR1 NUMERIC RESPONSE DATA>
	<hours> = <NR1 NUMERIC RESPONSE DATA>
	<minutes> = <NR1 NUMERIC RESPONSE DATA>
	<seconds> = <NR1 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:SDUR?` → `16,0,0,0`
Note	If not defined or the DUR parameter is not set in the MEASurement:SETup:STOP command, this query returns NaN (section 1.6.1).

18.2.9 MEASurement:SETup:MALLocation

Syntax	MEASurement:SETup:MALLocation <mode>
Description	This command sets the memory allocation.
Parameter	<mode> = <CHARACTER PROGRAM DATA> CONTinous: When storage space runs out old intervals will be overwritten. ALL: Uses all available storage space and then stops the measurement. *DEFault = CONTinous*
Response	None.
Example	`MEAS:SET:MALL CONT`
Note	

Syntax	MEASurement:SETup:MALLocation?
Description	This query returns the type of memory allocation.
Parameter	None.
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:MALL?` → `CONT`
Note	

18.2.10 MEASurement:SETup:PERFormance:TMBPs:PARameter

Syntax	MEASurement:SETup:PERFormance:TMBPs:PARameter <param>
Description	This command sets the performance parameter for 2 Mbps and V-Series.
Parameter	<param> = <CHARACTER PROGRAM DATA> G821: G.821 G826: G.826 M2100: M.2100 G821E: G.821 (expired revision) *DEFault = M2100*
Response	None.
Example	`MEAS:SET:PERF:TMBP:PAR G826`
Note	

Syntax	MEASurement:SETup:PERFormance:TMBPs:PARameter?
Description	This query returns the performance parameter for 2 Mbps and V-Series.
Parameter	None.
Response	<param> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:PERF:TMBP:PAR?` → `G826`
Note	

18.2.11 MEASurement:SETup:PERFormance:T1:PARameter

Syntax	MEASurement:SETup:PERFormance:T1:PARameter <param>
Description	This command sets the performance parameter for 2 Mbps and V-Series.
Parameter	<param> = <CHARACTER PROGRAM DATA> G821: G.821 G826: G.826 M2100: M.2100 G821E: G.821 (expired revision) *DEFault = M2100*
Response	None.
Example	`MEAS:SET:PERF:T1:PAR G826`
Note	

Syntax	MEASurement:SETup:PERFormance:T1:PARameter?
Description	This query returns the performance parameter for 2 Mbps and V-Series.
Parameter	None.
Response	<param> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:PERF:T1:PAR?` → `G826`
Note	

18.2.12 MEASurement:SETup:PERFormance:E3:PARameter

Syntax	MEASurement:SETup:PERFormance:E3:PARameter <param>
Description	This command sets the performance parameter for E3.
Parameter	<param> = <CHARACTER PROGRAM DATA> G826: G.826 M2100: M.2100 *DEFault = M2100*
Response	None.
Example	`MEAS:SET:PERF:E3:PAR M2100`
Note	

Syntax	MEASurement:SETup:PERFormance:E3:PARameter?
Description	This query returns the performance parameter for E3.
Parameter	None.
Response	<param> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:PERF:E3:PAR?` → `M2100`
Note	

18.2.13 MEASurement:SETup:PERFormance:T3:PARameter

Syntax	MEASurement:SETup:PERFormance:T3:PARameter <param>
Description	This command sets the performance parameter for E3.
Parameter	<param> = <CHARACTER PROGRAM DATA> G826: G.826 M2100: M.2100 *DEFault = M2100*
Response	None.
Example	`MEAS:SET:PERF:T3:PAR M2100`
Note	

Syntax	MEASurement:SETup:PERFormance:T3:PARameter?
Description	This query returns the performance parameter for E3.
Parameter	None.
Response	<param> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:PERF:T3:PAR?` → `M2100`
Note	

18.2.14 MEASurement:SETup:PERFormance:E4:PARameter

Syntax	MEASurement:SETup:PERFormance:E4:PARameter <param>
Description	This command sets the performance parameter for E4.
Parameter	<param> = <CHARACTER PROGRAM DATA> G826: G.826 M2100: M.2100 *DEFault = M2100*
Response	None.
Example	`MEAS:SET:PERF:E4:PAR M2100`
Note	

Syntax	MEASurement:SETup:PERFormance:E4:PARameter?
Description	This query returns the performance parameter for E4.
Parameter	None.
Response	<param> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:PERF:E4:PAR?` → `M2100`
Note	

18.2.15 MEASurement:SETup:PERFormance:SDH:PARameter

Syntax	MEASurement:SETup:PERFormance:SDH:PARameter <param>
Description	This command sets the performance parameter for SDH.
Parameter	<param> = <CHARACTER PROGRAM DATA> G826: G.826 G828: G.828+G.829 M2101: M.2101.1(M.2100) *DEFault = M2101*
Response	None.
Example	`MEAS:SET:PERF:SDH:PAR M2101`
Note	

Syntax	MEASurement:SETup:PERFormance:SDH:PARameter?
Description	This query returns the performance parameter for SDH.
Parameter	None.
Response	<param> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:PERF:SDH:PAR?` → `M2101`
Note	

18.2.16 MEASurement:SETup:PERFormance:SDH:MUX

Syntax	MEASurement:SETup:PERFormance:SDH:MUX <value>
Description	This command sets the MUX allocation in percentage.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0.00, MAXimum=100.00, DEFault=0*
Response	None.
Example	`MEAS:SET:PERF:SDH:MUX 10.00`
Note	

Syntax	MEASurement:SETup:PERFormance:SDH:MUX?
Description	This query returns the MUX allocation in percentage.
Parameter	None.
Response	<value> = <NR2 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:PERF:SDH:MUX?` → `10.00`
Note	

18.2.17 MEASurement:SETup:PERFormance:SDH:VC4

Syntax	MEASurement:SETup:PERFormance:SDH:VC4 <value>
Description	This command sets the VC4 allocation in percentage.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0.00, MAXimum=100.00, DEFault=0*
Response	None.
Example	`MEAS:SET:PERF:SDH:VC4 10.00`
Note	

Syntax	MEASurement:SETup:PERFormance:SDH:VC4?
Description	This query returns the VC4 Allocation in percentage.
Parameter	None.
Response	<value> = <NR2 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:PERF:SDH:VC4?` → `10.00`
Note	

18.2.18 MEASurement:SETup:PERFormance:SDH:VC3

Syntax	MEASurement:SETup:PERFormance:SDH:VC3 <value>
Description	This command sets the VC3 allocation in percentage.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0.00, MAXimum=100.00, DEFault=0*
Response	None.
Example	`MEAS:SET:PERF:SDH:VC3 10.00`
Note	

Syntax	MEASurement:SETup:PERFormance:SDH:VC3?
Description	This query returns the VC3 allocation in percentage.
Parameter	None.
Response	<value> = <NR2 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:PERF:SDH:VC3?` → `10.00`
Note	

18.2.19 MEASurement:SETup:PERFormance:SDH:VC12

Syntax	MEASurement:SETup:PERFormance:SDH:VC12 <value>
Description	This command sets the VC12 allocation in percentage.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0.00, MAXimum=100.00, DEFault=0*
Response	None.
Example	`MEAS:SET:PERF:SDH:VC12 10.00`
Note	

Syntax	MEASurement:SETup:PERFormance:SDH:VC12?
Description	This query returns the VC12 allocation in percentage.
Parameter	None.
Response	<value> = <NR2 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:PERF:SDH:VC12?` → `10.00`
Note	

18.2.20 MEASurement:SETup:PERFormance:SDH:TPERiod

Syntax	MEASurement:SETup:PERFormance:SDH:TPERiod <period>
Description	This command sets the evaluation item for SDH.
Parameter	<period> = <CHARACTER PROGRAM DATA> 15M: 15 minutes 1H: 1 hour 2H: 2 hours 24H: 24 hours 7D: 7 days *DEFault = 15M*
Response	None.
Example	`MEAS:SET:PERF:SDH:TPER 1H`
Note	Only valid when the SDH performance parameter is M.2101.1.

Syntax	MEASurement:SETup:PERFormance:SDH:TPERiod?
Description	This query returns the evaluation item for SDH.
Parameter	None.
Response	<period> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:PERF:SDH:TPER?` → `1H`
Note	

18.2.21 MEASurement:SETup:PERFormance:SONet:PARameter

Syntax	MEASurement:SETup:PERFormance:SONet:PARameter <param>
Description	This command sets the performance parameter for SDH.
Parameter	<param> = <CHARACTER PROGRAM DATA> G826: G.826 G828: G.828+G.829 M2101: M.2101.1(M.2100) *DEFault = M2101*
Response	None.
Example	`MEAS:SET:PERF:SON:PAR M2101`
Note	

Syntax	MEASurement:SETup:PERFormance:SONet:PARameter?
Description	This query returns the performance parameter for SONET.
Parameter	None.
Response	<param> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:PERF:SON:PAR?` → `M2101`
Note	

18.2.22 MEASurement:SETup:PERFormance:SONet:MUX

Syntax	MEASurement:SETup:PERFormance:SONet:MUX <value>
Description	This command sets the MUX allocation in percentage.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0.00, MAXimum=100.00, DEFault=0*
Response	None.
Example	`MEAS:SET:PERF:SON:MUX 10.00`
Note	

Syntax	MEASurement:SETup:PERFormance:SONet:MUX?
Description	This query returns the MUX allocation in percentage.
Parameter	None.
Response	<value> = <NR2 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:PERF:SON:MUX?` → `10.00`
Note	

18.2.23 MEASurement:SETup:PERFormance:SONet:STS3

Syntax	MEASurement:SETup:PERFormance:SONet:STS3 <value>
Description	This command sets the STS3 allocation in percentage.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0.00, MAXimum=100.00, DEFault=0*
Response	None.
Example	`MEAS:SET:PERF:SON:STS3 10.00`
Note	

Syntax	MEASurement:SETup:PERFormance:SONet:STS3?
Description	This query returns the STS3 Allocation in percentage.
Parameter	None.
Response	<value> = <NR2 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:PERF:SON:STS3?` → `10.00`
Note	

18.2.24 MEASurement:SETup:PERFormance:SONet:STS1

Syntax	MEASurement:SETup:PERFormance:SONet:STS1 <value>
Description	This command sets the STS1 allocation in percentage.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0.00, MAXimum=100.00, DEFault=0*
Response	None.
Example	`MEAS:SET:PERF:SON:STS1 10.00`
Note	

Syntax	MEASurement:SETup:PERFormance:SONet:STS1?
Description	This query returns the STS1 allocation in percentage.
Parameter	None.
Response	<value> = <NR2 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:PERF:SON:STS1?` → `10.00`
Note	

18.2.25 MEASurement:SETup:PERFormance:SONet:VT2

Syntax	MEASurement:SETup:PERFormance:SONet:VT2 <value>
Description	This command sets the VC12 allocation in percentage.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0.00, MAXimum=100.00, DEFault=0*
Response	None.
Example	`MEAS:SET:PERF:SON:VT2 10.00`
Note	

Syntax	MEASurement:SETup:PERFormance:SONet:VT2?
Description	This query returns the VT2 allocation in percentage.
Parameter	None.
Response	<value> = <NR2 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:PERF:SON:VT2?` → `10.00`
Note	

18.2.26 MEASurement:SETup:PERFormance:SONet:TPERiod

Syntax	MEASurement:SETup:PERFormance:SONet:TPERiod <period>
Description	This command sets the evaluation item for SONET.
Parameter	<period> = <CHARACTER PROGRAM DATA> 15M: 15 minutes 1H: 1 hour 2H: 2 hours 24H: 24 hours 7D: 7 days *DEFault = 15M*
Response	None.
Example	`MEAS:SET:PERF:SON:TPER 1H`
Note	Only valid when the SDH performance parameter is M.2101.1.

Syntax	MEASurement:SETup:PERFormance:SONet:TPERiod?
Description	This query returns the evaluation item for SONET.
Parameter	None.
Response	<period> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:PERF:SON:TPER?` → `1H`
Note	

18.2.27 MEASurement:SETup:PERFormance:WAN:PARameter

Syntax	MEASurement:SETup:PERFormance:WAN:PARameter <param>
Description	This command sets the performance parameter for 10G WAN.
Parameter	<param> = <CHARACTER PROGRAM DATA> G826: G.826 G828: G.828+G.829 M2101: M.2101.1(M.2100) *DEFault = G826*
Response	None.
Example	`MEAS:SET:PERF:WAN:PAR M2101`
Note	This command can be used on V2.00 or later.

Syntax	MEASurement:SETup:PERFormance:WAN:PARameter?
Description	This query returns the performance parameter for 10G WAN.
Parameter	None.
Response	<param> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:PERF:WAN:PAR? → M2101`
Note	This command can be used on V2.00 or later.

18.2.28 MEASurement:SETup:EVALuation:RX<Pt>[:ENABle]

Syntax	MEASurement:SETup:EVALuation:RX<Pt>[:ENABle] <enable>
Description	This command enables/disables the transport evaluation.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Examples	`MEAS:SET:EVAL:RX ON`
Note	

Syntax	MEASurement:SETup:EVALuation:RX<Pt>[:ENABle]?
Description	This query returns whether or not the transport evaluation is enabled.
Parameter	<Pt> = Port number
Response	<enable> = <NR1 NUMERIC RESPONSE DATA>
Examples	`MEAS:SET:EVAL:RX? → 1`
Note	

18.2.29 MEASurement:SETup:EVALuation:RX<Pt>:INTerface

Syntax	MEASurement:SETup:EVALuation:RX<Pt>:INTerface <interface>
Description	This command sets the transport evaluation interface type.
Parameters	<Pt> = Port number
	<interface> = <CHARACTER PROGRAM DATA> E1 E3 DS1 DS3 E4 SDH OTN CPRI The selectable interface type varies depending on the interface configuration.
Response	None.
Example	`MEAS:SET:EVAL:RX:INT SDH`
Note	

Syntax	MEASurement:SETup:EVALuation:RX<Pt>:INTerface?
Description	This query returns the transport evaluation interface type.
Parameter	<Pt> = Port number
Response	<interface> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:RX:INT?` → `SDH`
Note	

18.2.30 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:ITEM

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:ITEM <item>
Description	This commands sets the evaluation item for CPRI.
Parameters	<Pt> = Port number
	<item> = <CHARACTER PROGRAM DATA> ANY: Any Alarm or Error ALARm: Alarm ERRor: Error *DEFault = ANY*
Response	None.
Example	`MEAS:SET:EVAL:CPRI:PORT1:ITEM ANY`
Note	None.

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:ITEM?
Description	This query returns the evaluation item for CPRI.
Parameter	<Pt> = Port number
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:CPRI:PORT1:ITEM?` → `ANY`
Note	

18.2.31 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:ITEM:ALARm

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:ITEM:ALARm <item>
Description	This commands sets the evaluation alarm item for CPRI.
Parameters	<Pt> = Port number
	<item> = <CHARACTER PROGRAM DATA> SLOS: Optical Loss of signal OPL: SLOS LOS: Loss of Signal LOF: Loss of Frame PSL: PSL L1LOS: Remote LOS L1LOF: Remote LOF RAI: RAI SDI: SDI *DEFault = SLOS*
Response	None.
Example	`MEAS:SET:EVAL:CPRI:PORT1:ITEM:ALAR SLOS`
Note	None.

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:ITEM:ALARm?
Description	This query returns the evaluation alarm item for CPRI.
Parameter	<Pt> = Port number
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:CPRI:PORT1:ITEM:ALAR?` → `LOS`
Note	

18.2.32 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:ITEM:ERRor

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:ITEM:ERRor <item>
Description	This commands sets the evaluation errors item for CPRI.
Parameters	<Pt> = Port number
	<item> = <CHARACTER PROGRAM DATA> LCV: LCV SHV: SHV K307: K30.7 PATT: Pattern Error *DEFault = LCV*
Response	None.
Example	`MEAS:SET:EVAL:CPRI:PORT1:ITEM:ERR LCV`
Note	None.

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:ITEM:ERRor?
Description	This query returns the evaluation errors item for CPRI.
Parameter	<Pt> = Port number
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:CPRI:PORT1:ITEM:ERR?` → `LCV`
Note	

18.2.33 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:TYPE

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:TYPE <type>
Description	This commands sets the evaluation type for CPRI.
Parameters	<Pt> = Port number
	<item> = <CHARACTER PROGRAM DATA> CNT: Count RATio: Ratio *DEFault = CNT*
Response	None.
Example	`MEAS:SET:EVAL:CPRI:PORT1:TYPE CNT`
Note	None.

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:TYPE?
Description	This query returns the evaluation type for CPRI.
Parameter	<Pt> = Port number
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:CPRI:PORT1:TYPE?` → `CNT`
Note	

18.2.34 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:CNT:PASS

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:CNT:PASS <value>
Description	This command sets the count for the CPRI PASS limit. <Pt> = Port number
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100000, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:CPRI:PORT1:CNT:PASS 10`
Note	This setting applies when :TYPE is CNT, ESCount or SESCount. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:CNT:PASS?
Description	This query returns the count for the CPRI PASS limit.
Parameter	<Pt> = Port number
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:CPRI:PORT1:CNT:PASS? → 10`
Note	

18.2.35 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:CNT:FAIL

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:CNT:FAIL <value>
Description	This command sets the count for the CPRI fail limit.
Parameter	<Pt> = Port number <value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100000, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:CPRI:PORT1:CNT:FAIL 10`
Note	This setting applies when :TYPE is CNT, ESCount or SESCount. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:CNT:FAIL?
Description	This query returns the count for the CPRI fail limit.
Parameter	<Pt> = Port number
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:CPRI:PORT1:CNT:FAIL? → 10`
Note	

18.2.36 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:RATio:PASS

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:RATio:PASS <value>
Description	This command sets the ratio for the CPRI pass limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1.00, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:CPRI:PORT1:RAT:PASS 0.00`
Note	This setting applies when :TYPE is RATio. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:RATio:PASS?
Description	This query returns the ratio for the CPRI pass limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:CPRI:PORT1:RAT:PASS? → 1.00E-01`
Note	

18.2.37 MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:RATio:FAIL

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:RATio:FAIL <value>
Description	This command sets the ratio for the CPRI fail limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1.00, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:CPRI:PORT1:RAT:FAIL 0.00`
Note	This setting applies when :TYPE is RATio. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:CPRI:PORT<Pt>:RATio:FAIL?
Description	This query returns the ratio for the CPRI fail limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:CPRI:PORT1:RAT:FAIL?` → `1.00E-01`
Note	

18.2.38 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:ITEM

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:ITEM <item>
Description	This command sets the evaluation item for 2 Mbps.
Parameter	<Pt> = Port number <item> = <CHARACTER PROGRAM DATA> ANY: Any error or alarm PATTern: Payload Pattern FAS: FAS CRC: CRC-4 EBIT: E-bit *DEFault = ANY*
Response	None.
Example	`MEAS:SET:EVAL:TMBP:RX1:ITEM PATT`
Note	Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:ITEM?
Description	This query the evaluation item for 2 Mbps.
Parameter	None.
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:TMBP:RX1:ITEM?` → `PATT`
Note	

18.2.39 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:TYPE

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:TYPE <type>
Description	This command sets the evaluation type for 2 Mbps.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> CNT: Error Count RATio: Error Ratio EPCT: Error percentage ESCount: ES error count ESRatio: ES error ratio ESPCt: ES error percentage SESCount: SES error count SESPct: SES error percentage HRPCt: HR% HRPBis: HR% BIS *DEFault = CNT*
Response	None.
Example	`MEAS:SET:EVAL:TMBP:RX1:TYPE ERAT`
Note	Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:TYPE?
Description	This query returns the evaluation type for 2 Mbps.
Parameter	None.
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:TMBP:RX1:TYPE?` → `ERAT`
Note	

18.2.40 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:CNT:PASS

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:CNT:PASS <value>
Description	This command sets the error count for the 2Mbps pass limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100000, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:TMBP:RX1:CNT:PASS 10`
Note	This setting applies when :TYPE is CNT, ESCount or SESCount. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:CNT:PASS?
Description	This query returns the error count for the 2Mbps pass limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:TMBP:RX1:CNT:PASS?` → `10`
Note	

18.2.41 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:CNT:FAIL

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:CNT:FAIL <value>
Description	This command sets the error count for the 2Mbps fail limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100000, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:TMBP:RX1:CNT:FAIL 10`
Note	This setting applies when :TYPE is CNT, ESCount or SESCount. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:CNT:FAIL?
Description	This query returns the error count for the 2Mbps fail limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:TMBP:RX1:CNT:FAIL?` → `10`
Note	

18.2.42 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:RATio:PASS

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:RATio:PASS <value>
Description	This command sets the ratio for the 2Mbps pass limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1.00, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:TMBP:RX1:RAT:PASS 10.00`
Note	This setting applies when :TYPE is RATio or ESRatio. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:RATio:PASS?
Description	This query returns the ratio for the 2Mbps pass limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:TMBP:RX1:RAT:PASS?` → `1.00E-01`
Note	

18.2.43 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:RATio:FAIL

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:RATio:FAIL <value>
Description	This command sets the ratio for the 2Mbps fail limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1.00, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:TMBP:RX1:RAT:FAIL 10.00`
Note	This setting applies when :TYPE is RATio or ESRatio. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:RATio:FAIL?
Description	This query returns the ratio for the 2Mbps fail limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:TMBP:RX1:RAT:FAIL?` → `1.00E-01`
Note	

18.2.44 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:PCT:PASS

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:PCT:PASS <value>
Description	This command sets the percentage for the 2 Mbps pass limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:TMBP:RX1:PCT:PASS 10`
Note	This setting applies when :TYPE is EPCT, ESPCt or SESPct. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:PCT:PASS?
Description	This query returns the percentage for the 2 Mbps pass limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:TMBP:RX1:PCT:PASS?` → `10`
Note	

18.2.45 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:PCT:FAIL

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:PCT:FAIL <value>
Description	This command sets the percentage for the 2 Mbps fail limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:TMBP:RX1:PCT:FAIL 10`
Note	This setting applies when :TYPE is EPCT, ESPCt or SESPct. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:PCT:FAIL?
Description	This query returns the percentage for the 2 Mbps fail limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:TMBP:RX1:PCT:FAIL?` → `10`
Note	

18.2.46 MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:HREFerence

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:HREFerence <percent>
Description	This command sets the HR%.
Parameter	<Pt> = Port number <percent> = <NUMERIC PROGRAM DATA> *MINimum=0.00, MAXimum=100.00, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:TMBP:RX1:HREF 10.00`
Note	This setting applies when :TYPE is HRPCt or HRPBis. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:TMBPs:RX<Pt>:HREFerence?
Description	This query return the HR%.
Parameter	None.
Response	<percent> = <NR2 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:TMBP:RX1:HREF?` → `10.00`
Note	

18.2.47 MEASurement:SETup:EVALuation:E3:RX<Pt>:ITEM

Syntax	MEASurement:SETup:EVALuation:E3:RX<Pt>:ITEM <item>
Description	This command sets the evaluation item for E3.
Parameters	<Pt> = Port number
	<item> = <CHARACTER PROGRAM DATA> ANY: Any error or alarm NSIGnal: No signal AIS: Alarm Indication Signal NFRame: No frame DISTant: Distant NSYNc: No sync FASWords: FAS words FASBits: FAS bits CODE: Code PATTern: Pattern PSLips: Pattern slips BLOCk: Block *DEFault = ANY*
Response	None.
Example	`MEAS:SET:EVAL:E3:RX1:ITEM NSIG`
Note	Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:E3:RX<Pt>:ITEM?
Description	This query returns the evaluation item for E3.
Parameter	<Pt> = Port number
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:E3:RX1:ITEM?` → `NSIG`
Note	

18.2.48 MEASurement:SETup:EVALuation:E3:RX<Pt>:TYPE

Syntax	MEASurement:SETup:EVALuation:E3:RX<Pt>:TYPE <type>
Description	This command sets the evaluation type for E3.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> CNT: Count RATio: Ratio *DEFault = CNT*
Response	None.
Example	`MEAS:SET:EVAL:E3:RX1:TYPE RAT`
Note	Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:E3:RX<Pt>:TYPE?
Description	This query returns the evaluation type for E3.
Parameter	None.
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:E3:RX1:TYPE?` → `RAT`
Note	

18.2.49 MEASurement:SETup:EVALuation:E3:RX<Pt>:CNT:PASS

Syntax	MEASurement:SETup:EVALuation:E3:RX<Pt>:CNT:PASS <value>
Description	This command sets the error count for the E3 pass limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100000, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:E3:RX1:CNT:PASS 10`
Note	This setting applies when :TYPE is CNT. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:E3:RX<Pt>:CNT:PASS?
Description	This query returns the error count for the E3 pass limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:E3:RX1:CNT:PASS? → 10`
Note	

18.2.50 MEASurement:SETup:EVALuation:E3:RX<Pt>:CNT:FAIL

Syntax	MEASurement:SETup:EVALuation:E3:RX<Pt>:CNT:FAIL <value>
Description	This command sets the error count for the E3 fail limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100000, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:E3:RX1:CNT:FAIL 10`
Note	This setting applies when :TYPE is CNT. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:E3:RX<Pt>:CNT:FAIL?
Description	This query returns the error count for the E3 fail limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:E3:RX1:CNT:FAIL? → 10`
Note	

18.2.51 MEASurement:SETup:EVALuation:E3:RX<Pt>:RATio:PASS

Syntax	MEASurement:SETup:EVALuation:E3:RX<Pt>:RATio:PASS <value>
Description	This command sets the ratio for the E3 pass limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1.00, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:E3:RX1:RAT:PASS 10.00`
Note	This setting applies when :TYPE is RATio. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:E3:RX<Pt>:RATio:PASS?
Description	This query returns the ratio for the E3 pass limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:E3:RX1:RAT:PASS? → 1.00E-01`
Note	

18.2.52 MEASurement:SETup:EVALuation:E3:RX<Pt>:RATio:FAIL

Syntax	MEASurement:SETup:EVALuation:E3:RX<Pt>:RATio:FAIL <value>
Description	This command sets the ratio for the E3 fail limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1.00, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:E3:RX1:RAT:FAIL 10.00`
Note	This setting applies when :TYPE is RATio. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:E3:RX<Pt>:RATio:FAIL?
Description	This query returns the ratio for the E3 fail limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:E3:RX1:RAT:FAIL?` → `1.00E-01`
Note	

18.2.53 MEASurement:SETup:EVALuation:E4:RX<Pt>:ITEM

Syntax	MEASurement:SETup:EVALuation:E4:RX<Pt>:ITEM <item>
Description	This command sets the evaluation item for E4.
Parameters	<Pt> = Port number
	<item> = <CHARACTER PROGRAM DATA> ANY: Any error or alarm NSIGnal: No signal AIS: Alarm Indication Signal NFRame: No frame DISTant: Distant NSYNc: No sync FASWords: FAS words FASBits: FAS bits CODE: Code PATTern: Pattern PSLips: Pattern slips PBLock: Pattern block FBLock: Frame block *DEFault = ANY*
Response	None.
Example	`MEAS:SET:EVAL:E4:RX1:ITEM NSIG`
Note	Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:E4:RX<Pt>:ITEM?
Description	This query returns the evaluation item for E4.
Parameter	<Pt> = Port number
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:E4:RX1:ITEM?` → `NSIG`
Note	

18.2.54 MEASurement:SETup:EVALuation:E4:RX<Pt>:TYPE

Syntax	MEASurement:SETup:EVALuation:E4:RX<Pt>:TYPE <type>
Description	This command sets the evaluation type for E4.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> CNT: Count RATio: Ratio *DEFault = CNT*
Response	None.
Example	`MEAS:SET:EVAL:E4:RX1:TYPE RAT`
Note	Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:E4:RX<Pt>:TYPE?
Description	This query returns the evaluation type for E4.
Parameter	None.
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:E4:RX1:TYPE? → RAT`
Note	

18.2.55 MEASurement:SETup:EVALuation:E4:RX<Pt>:CNT:PASS

Syntax	MEASurement:SETup:EVALuation:E4:RX<Pt>:CNT:PASS <value>
Description	This command sets the error count for the E4 pass limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100000, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:E4:RX1:CNT:PASS 10`
Note	This setting applies when :TYPE is CNT. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:E4:RX<Pt>:CNT:PASS?
Description	This query returns the error count for the E4 pass limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:E4:RX1:CNT:PASS? → 10`
Note	

18.2.56 MEASurement:SETup:EVALuation:E4:RX<Pt>:CNT:FAIL

Syntax	MEASurement:SETup:EVALuation:E4:RX<Pt>:CNT:FAIL <value>
Description	This command sets the error count for the E4 fail limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100000, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:E4:RX1:CNT:FAIL 10`
Note	This setting applies when :TYPE is CNT. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:E4:RX<Pt>:CNT:FAIL?
Description	This query returns the error count for the E4 fail limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:E4:RX1:CNT:FAIL? → 10`
Note	

18.2.57 MEASurement:SETup:EVALuation:E4:RX<Pt>:RATio:PASS

Syntax	MEASurement:SETup:EVALuation:E4:RX<Pt>:RATio:PASS <value>
Description	This command sets the ratio for the E4 pass limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1.00, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:E4:RX1:RAT:PASS 10.00`
Note	This setting applies when :TYPE is RATio. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:E4:RX<Pt>:RATio:PASS?
Description	This query returns the ratio for the E4 pass limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:E4:RX1:RAT:PASS?` → `1.00E-01`
Note	

18.2.58 MEASurement:SETup:EVALuation:E4:RX<Pt>:RATio:FAIL

Syntax	MEASurement:SETup:EVALuation:E4:RX<Pt>:RATio:FAIL <value>
Description	This command sets the ratio for the E4 fail limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1.00, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:E4:RX1:RAT:FAIL 10.00`
Note	This setting applies when :TYPE is RATio. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:E4:RX<Pt>:RATio:FAIL?
Description	This query returns the ratio for the E4 fail limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:E4:RX1:RAT:FAIL?` → `1.00E-01`
Note	

18.2.59 MEASurement:SETup:EVALuation:SDH:RX<Pt>:ITEM

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:ITEM <item>
Description	This command sets the evaluation item for SDH.
Parameter	<Pt> = Port number <item> = <CHARACTER PROGRAM DATA> ANY: Any error or alarm ALARm: An alarm defined by `MEAS:SET:EVAL:SDH:RX<Pt>:ITEM:ALAR` ERRor: An error defined by `MEAS:SET:EVAL:SDH:RX<Pt>:ITEM:ERR` *DEFault = ANY*
Response	None.
Example	`MEAS:SET:EVAL:SDH:RX1:ITEM ALAR`
Note	Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:ITEM?
Description	This query the evaluation item for 2 Mbps.
Parameter	None.
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:SDH:RX1:ITEM?` → `PATT`
Note	

18.2.60 MEASurement:SETup:EVALuation:SDH:RX<Pt>:ITEM:ALARm

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:ITEM:ALARm <item>
Description	This commands sets the evaluation item for SDH.
Parameters	<Pt> = Port number
	<item> = <CHARACTER PROGRAM DATA> LOS: LOS LOF: LOF OOF: OOF MSAIS: MS-AIS MSRDI: MS-RDI AUAIS: AU-AIS AULOP: AU-LOP HPTIM: HP-TIM HPPLM: HP-PLM HPUNEQ: HP-UNEQ HPRDI: HP-RDI TUAIS: TU-AIS TULOP: TU-LOP TULOM: TU-LOM LPTIM: LP-TIM LPUNEQ: LP-UNEQ LPRDI: LP-RDI LSS: LSS LPPLM: LP-PLM TCUNEQ: TC-UNEQ TCLTC: TC-LTC TCTIM: TC-TIM TCAIS: TC-AIS TCRDI: TC-RDI TCODI: TC-ODI GAIS: G-AIS *DEFault = LOS*
Response	None.
Example	`MEAS:SET:EVAL:SDH:RX1:ITEM:ALAR LOS`
Note	This setting applies when :ITEM is ALARm. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:ITEM:ALARm?
Description	This query returns the evaluation item for SDH.
Parameter	<Pt> = Port number
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:SDH:RX1:ITEM:ALAR?` → `LOS`
Note	

18.2.61 MEASurement:SETup:EVALuation:SDH:RX<Pt>:ITEM:ERRor

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:ITEM:ERRor <item>
Description	This commands sets the evaluation item for SDH.
Parameters	<Pt> = Port number
	<item> = <CHARACTER PROGRAM DATA> A1A2: A1A2 B1: B1-BIP B2: B2-BIP MSREI: MS-REI B3: B3-BIP HPREI: HP-REI V5LP: V5/LP-B3 LPREI: LP-REI TUNDF: TU-NDF AUNDF: AU-NDF APS: Switch APS TUMN: TU-MVT-Negative TUMP: TU-MVT-Positive AUMN: AU-MVT-Negative AUMP: AU-MVT-Positive TCIEC: TC-IEC TCBIP: TC-BIP-2 TCREI: TC-REI TCOEI: TC-OEI *DEFault = A1A2*
Response	None.
Example	`MEAS:SET:EVAL:SDH:RX1:ITEM:ERR A1A2`
Note	This setting applies when :ITEM is ERRor. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:ITEM:ERRor?
Description	This query returns the evaluation item for SDH.
Parameter	<Pt> = Port number
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:SDH:RX1:ITEM:ERR?` → `A1A2`
Note	

18.2.62 MEASurement:SETup:EVALuation:SDH:RX<Pt>:TYPE

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:TYPE <type>
Description	This command sets the evaluation type for SDH.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> CNT: Count RATio: Ratio *DEFault = CNT*
Response	None.
Example	`MEAS:SET:EVAL:SDH:RX1:TYPE RAT`
Note	Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:TYPE?
Description	This query returns the evaluation type for SDH.
Parameter	None.
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:SDH:RX1:TYPE?` → `RAT`
Note	

18.2.63 MEASurement:SETup:EVALuation:SDH:RX<Pt>:CNT:PASS

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:CNT:PASS <value>
Description	This command sets the error count for the SDH pass limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100000, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:SDH:RX1:CNT:PASS 10`
Note	This setting applies when :TYPE is CNT. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:CNT:PASS?
Description	This query returns the error count for the SDH pass limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:SDH:RX1:CNT:PASS? → 10`
Note	

18.2.64 MEASurement:SETup:EVALuation:SDH:RX<Pt>:CNT:FAIL

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:CNT:FAIL <value>
Description	This command sets the error count for the SDH fail limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100000, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:SDH:RX1:CNT:FAIL 10`
Note	This setting applies when :TYPE is CNT. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:CNT:FAIL?
Description	This query returns the error count for the SDH fail limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:SDH:RX1:CNT:FAIL? → 10`
Note	

18.2.65 MEASurement:SETup:EVALuation:SDH:RX<Pt>:RATio:PASS

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:RATio:PASS <value>
Description	This command sets the ratio for the SDH pass limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1.00, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:SDH:RX1:RAT:PASS 10.00`
Note	This setting applies when :TYPE is RATio. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:RATio:PASS?
Description	This query returns the ratio for the SDH pass limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:SDH:RX1:RAT:PASS? → 1.00E-01`
Note	

18.2.66 MEASurement:SETup:EVALuation:SDH:RX<Pt>:RATio:FAIL

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:RATio:FAIL <value>
Description	This command sets the ratio for the SDH fail limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1.00, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:SDH:RX1:RAT:FAIL 10.00`
Note	This setting applies when :TYPE is RATio. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SDH:RX<Pt>:RATio:FAIL?
Description	This query returns the ratio for the SDH fail limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:SDH:RX1:RAT:FAIL?` → `1.00E-01`
Note	

18.2.67 MEASurement:SETup:EVALuation:SONet:RX<Pt>:ITEM

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:ITEM <item>
Description	This command sets the evaluation item for SDH.
Parameter	<Pt> = Port number <item> = <CHARACTER PROGRAM DATA> ANY: Any error or alarm ALARm: An alarm defined by `MEAS:SET:EVAL:SON:RX<Pt>:ITEM:ALAR` ERRor: An error defined by `MEAS:SET:EVAL:SON:RX<Pt>:ITEM:ERR` *DEFault = ANY*
Response	None.
Example	`MEAS:SET:EVAL:SON:RX1:ITEM ALAR`
Note	Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:ITEM?
Description	This query the evaluation item for 2 Mbps.
Parameter	None.
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:SON:RX1:ITEM?` → `PATT`
Note	

18.2.68 MEASurement:SETup:EVALuation:SONet:RX<Pt>:ITEM:ALARm

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:ITEM:ALARm <item>
Description	This commands sets the evaluation item for SDH.
Parameters	<Pt> = Port number
	<item> = <CHARACTER PROGRAM DATA> LOS: LOS LOF: LOF OOF: OOF AISL: MS-AIS RDIL: MS-RDI AISP: AU-AIS LOPP: AU-LOP TIMP: HP-TIM PLMP: HP-PLM UNEQP: HP-UNEQ RDIP: HP-RDI AISV: TU-AIS LOPV: TU-LOP LOMV: TU-LOM TIMV: LP-TIM UNEQV: LP-UNEQ RDIV: LP-RDI LSS: LSS PLMV: LP-PLM TCUNEQ: TC-UNEQ TCLTC: TC-LTC TCTIM: TC-TIM TCAIS: TC-AIS TCRDI: TC-RDI TCODI: TC-ODI GAIS: G-AIS *DEFault = LOS*
Response	None.
Example	`MEAS:SET:EVAL:SON:RX1:ITEM:ALAR LOS`
Note	This setting applies when :ITEM is ALARm. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:ITEM:ALARm?
Description	This query returns the evaluation item for SDH.
Parameter	<Pt> = Port number
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:SON:RX1:ITEM:ALAR?` → `LOS`
Note	

18.2.69 MEASurement:SETup:EVALuation:SONet:RX<Pt>:ITEM:ERRor

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:ITEM:ERRor <item>
Description	This commands sets the evaluation item for SDH.
Parameters	<Pt> = Port number
	<item> = <CHARACTER PROGRAM DATA> A1A2: A1A2 B1: B1-BIP B2: B2-BIP REIL: MS-REI B3: B3-BIP REIP: HP-REI V5LP: V5/LP-B3 REIV: LP-REI VTNDF: TU-NDF STSNDF:AU-NDF APS: Switch APS TUMN: TU-MVT-Negative TUMP: TU-MVT-Positive AUMN: AU-MVT-Negative AUMP: AU-MVT-Positive TCIEC: TC-IEC TCBIP: TC-BIP-2 TCREI: TC-REI TCOEI: TC-OEI *DEFault = A1A2*
Response	None.
Example	`MEAS:SET:EVAL:SON:RX1:ITEM:ERR A1A2`
Note	This setting applies when :ITEM is ERRor. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:ITEM:ERRor?
Description	This query returns the evaluation item for SDH.
Parameter	<Pt> = Port number
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:SON:RX1:ITEM:ERR?` → `A1A2`
Note	

18.2.70 MEASurement:SETup:EVALuation:SONet:RX<Pt>:TYPE

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:TYPE <type>
Description	This command sets the evaluation type for SDH.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> CNT: Count RATio: Ratio *DEFault = CNT*
Response	None.
Example	`MEAS:SET:EVAL:SON:RX1:TYPE RAT`
Note	Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:TYPE?
Description	This query returns the evaluation type for SDH.
Parameter	None.
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:SON:RX1:TYPE?` → `RAT`
Note	

18.2.71 MEASurement:SETup:EVALuation:SONet:RX<Pt>:CNT:PASS

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:CNT:PASS <value>
Description	This command sets the error count for the SDH pass limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100000, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:SON:RX1:CNT:PASS 10`
Note	This setting applies when :TYPE is CNT. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:CNT:PASS?
Description	This query returns the error count for the SDH pass limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:SON:RX1:CNT:PASS?` → `10`
Note	

18.2.72 MEASurement:SETup:EVALuation:SONet:RX<Pt>:CNT:FAIL

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:CNT:FAIL <value>
Description	This command sets the error count for the SDH fail limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100000, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:SON:RX1:CNT:FAIL 10`
Note	This setting applies when :TYPE is CNT. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:CNT:FAIL?
Description	This query returns the error count for the SDH fail limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:SON:RX1:CNT:FAIL?` → `10`
Note	

18.2.73 MEASurement:SETup:EVALuation:SONet:RX<Pt>:RATio:PASS

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:RATio:PASS <value>
Description	This command sets the ratio for the SDH pass limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1.00, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:SON:RX1:RAT:PASS 10.00`
Note	This setting applies when :TYPE is RATio. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:RATio:PASS?
Description	This query returns the ratio for the SDH pass limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:SON:RX1:RAT:PASS?` → `1.00E-01`
Note	

18.2.74 MEASurement:SETup:EVALuation:SONet:RX<Pt>:RATio:FAIL

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:RATio:FAIL <value>
Description	This command sets the ratio for the SDH fail limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1.00, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:SON:RX1:RAT:FAIL 10.00`
Note	This setting applies when :TYPE is RATio. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:SONet:RX<Pt>:RATio:FAIL?
Description	This query returns the ratio for the SDH fail limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:SON:RX1:RAT:FAIL?` → `1.00E-01`
Note	

18.2.75 MEASurement:SETup:EVALuation:OTN:RX<Pt>:ITEM

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:ITEM <item>
Description	This command sets the evaluation item for OTN.
Parameter	<Pt> = Port number <item> = <CHARACTER PROGRAM DATA> ANY: Any error or alarm AL1: An alarm at level 1 defined by `MEAS:SET:EVAL:OTN:RX<Pt>:ITEM:ALAR` AL2: An alarm at level 1 defined by `MEAS:SET:EVAL:OTN:RX<Pt>:ITEM:ALAR` AL3: An alarm at level 1 defined by `MEAS:SET:EVAL:OTN:RX<Pt>:ITEM:ALAR` EL1: An error at level 1 defined by `MEAS:SET:EVAL:OTN:RX<Pt>:ITEM:ERR` EL2: An error at level 1 defined by `MEAS:SET:EVAL:OTN:RX<Pt>:ITEM:ERR` EL3: An error at level 1 defined by `MEAS:SET:EVAL:OTN:RX<Pt>:ITEM:ERR` *DEFault = ANY*
Response	None.
Example	`MEAS:SET:EVAL:OTN:RX1:ITEM PATT`
Note	Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:ITEM?
Description	This query the evaluation item for 2 Mbps.
Parameter	None.
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:OTN:RX1:ITEM?` → `PATT`
Note	

18.2.76 MEASurement:SETup:EVALuation:OTN:RX<Pt>:ITEM:ALARm

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:ITEM:ALARm <item>
Description	This commands sets the evaluation item for OTN.
Parameters	<Pt> = Port number
	<item> = <CHARACTER PROGRAM DATA> LOS: LOS OTUAIS: OTU-AIS LOF: LOF LOFLOM: LOFLOM OOF: OOF LOM: LOM OOM: OOM

continued on next page...

. . . continued from previous page

	SMTIM: SM-TIM SMBIAE: SM-BIAE SMBDI: SM-BDI SMIAE: SM-IAE ODUAIS: ODU-AIS ODULCK: ODU-LCK ODUOCI: ODU-OCI PMTIM: PM-TIM PMBDI: PM-BDI TCM1TIM: TCM1-TIM TCM1BIAE: TCM1-BIAE TCM1BDI: TCM1-BDI TCM1IAE: TCM1-IAE TCM1LTC: TCM1-LTC TCM2TIM: TCM2-TIM TCM2BIAE: TCM2-BIAE TCM2BDI: TCM2-BDI TCM2IAE: TCM2-IAE TCM2LTC: TCM2-LTC TCM3TIM: TCM3-TIM TCM3BIAE: TCM3-BIAE TCM3BDI: TCM3-BDI TCM3IAE: TCM3-IAE TCM3LTC: TCM3-LTC TCM4TIM: TCM4-TIM TCM4BIAE: TCM4-BIAE TCM4BDI: TCM4-BDI TCM4IAE: TCM4-IAE TCM4LTC: TCM4-LTC TCM5TIM: TCM5-TIM TCM5BIAE: TCM5-BIAE TCM5BDI: TCM5-BDI TCM5IAE: TCM5-IAE TCM5LTC: TCM5-LTC TCM6TIM: TCM6-TIM TCM6BIAE: TCM6-BIAE TCM6BDI: TCM6-BDI TCM6IAE: TCM6-IAE TCM6LTC: TCM6-LTC PLM: PLM MSIM: MSIM AISC: Cl-AIS CSF: CSF LSS: LSS *DEFault = LOS*
Response	None.
Example	`MEAS:SET:EVAL:OTN:RX1:ITEM:ALAR LOS`
Note	This setting applies when :ITEM is AL1, AL2 or AL3. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:ITEM:ALARm?
Description	This query returns the evaluation item for OTN.
Parameter	<Pt> = Port number
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:OTN:RX1:ITEM:ALAR?` → `LOS`
Note	

18.2.77 MEASurement:SETup:EVALuation:OTN:RX<Pt>:ITEM:ERRor

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:ITEM:ERRor <item>
Description	This commands sets the evaluation item for OTN.
Parameters	<Pt> = Port number
	<item> = <CHARACTER PROGRAM DATA> FAS: FAS MFAS: MFAS SMBIP: SM-BIP8 SMBEI: SM-BEI FCE: FCE FUEB: FUEB PMBIP: PM-BIP8 PMBEI: PM-BEI TCM1BIP: TCM1-BIP8 TCM1BEI: TCM1-BEI TCM2BIP: TCM2-BIP8 TCM2BEI: TCM2-BEI TCM3BIP: TCM3-BIP8 TCM3BEI: TCM3-BEI TCM4BIP: TCM4-BIP8 TCM4BEI: TCM4-BEI TCM5BIP: TCM5-BIP8 TCM5BEI: TCM5-BEI TCM6BIP: TCM6-BIP8 TCM6BEI: TCM6-BEI CHECCO: ECTABLE CHECUNCO: cHEC-UNCORRECTABLE THECCOR: tHEC-CORRECTABLE THECUNCO: tHEC-UNCORRECTABLE IGFPFRAME: INVALID-GFP-FRAME SBLOCKCRC: SUPERBLOCK-CRC CSFSIGNAL: CSF-SIGNAL CSFSYNC: CSF-SYNC CRC8: CRC8 CRC5: CRC5 FCS: FCS *DEFault = FAS*
Response	None.
Example	`MEAS:SET:EVAL:OTN:RX1:ITEM:ERR FAS`
Note	This setting applies when :ITEM is EL1, EL2 or EL3. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:ITEM:ERRor?
Description	This query returns the evaluation item for OTN.
Parameter	<Pt> = Port number
Response	<item> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:OTN:RX1:ITEM:ERR?` → `LOS`
Note	

18.2.78 MEASurement:SETup:EVALuation:OTN:RX<Pt>:TYPE

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:TYPE <type>
Description	This command sets the evaluation type for OTN.
Parameters	<Pt> = Port number
	<type> = <CHARACTER PROGRAM DATA> CNT: Count RATio: Ratio *DEFault = CNT*
Response	None.
Example	`MEAS:SET:EVAL:OTN:RX1:TYPE RAT`
Note	Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:TYPE?
Description	This query returns the evaluation type for OTN.
Parameter	None.
Response	<type> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:OTN:RX1:TYPE? → RAT`
Note	

18.2.79 MEASurement:SETup:EVALuation:OTN:RX<Pt>:CNT:PASS

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:CNT:PASS <value>
Description	This command sets the error count for the OTN pass limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100000, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:OTN:RX1:CNT:PASS 10`
Note	This setting applies when :TYPE is CNT. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:CNT:PASS?
Description	This query returns the error count for the OTN pass limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:OTN:RX1:CNT:PASS? → 10`
Note	

18.2.80 MEASurement:SETup:EVALuation:OTN:RX<Pt>:CNT:FAIL

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:CNT:FAIL <value>
Description	This command sets the error count for the OTN fail limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=100000, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:OTN:RX1:CNT:FAIL 10`
Note	This setting applies when :TYPE is CNT. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:CNT:FAIL?
Description	This query returns the error count for the OTN fail limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:OTN:RX1:CNT:FAIL? → 10`
Note	

18.2.81 MEASurement:SETup:EVALuation:OTN:RX<Pt>:RATio:PASS

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:RATio:PASS <value>
Description	This command sets the ratio for the OTN pass limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1.00, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:OTN:RX1:RAT:PASS 10.00`
Note	This setting applies when :TYPE is RATio. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:RATio:PASS?
Description	This query returns the ratio for the OTN pass limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:OTN:RX1:RAT:PASS?` → `1.00E-01`
Note	

18.2.82 MEASurement:SETup:EVALuation:OTN:RX<Pt>:RATio:FAIL

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:RATio:FAIL <value>
Description	This command sets the ratio for the OTN fail limit.
Parameter	<value> = <NUMERIC PROGRAM DATA> *MINimum=0, MAXimum=1.00, DEFault=0*
Response	None.
Example	`MEAS:SET:EVAL:OTN:RX1:RAT:FAIL 10.00`
Note	This setting applies when :TYPE is RATio. Command is valid for BERT applications only.

Syntax	MEASurement:SETup:EVALuation:OTN:RX<Pt>:RATio:FAIL?
Description	This query returns the ratio for the OTN fail limit.
Parameter	None.
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:OTN:RX1:RAT:FAIL?` → `1.00E-01`
Note	

18.2.83 MEASurement:SETup:EVALuation:BER:OBAMeasuring

Syntax	MEASurement:SETup:EVALuation:BER:OBAMeasuring <enable>
Description	This command enables/disables only show BER alarms when measuring.
Parameters	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`MEAS:SET:EVAL:BER:OBAM ON`
Note	This setting applies to all ports. This only applies to Ethernet and Fibre Channel.

Syntax	MEASurement:SETup:EVALuation:BER:OBAMeasuring?
Description	This query returns if BER alarms should only be shown when measuring.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`MEAS:SET:EVAL:BER:OBAM?` → `1`
Note	This setting applies to all ports. This only applies to Ethernet and Fibre Channel.

18.2.84 MEASurement:SETup:EVALuation:BER:IAFFilter

Syntax	MEASurement:SETup:EVALuation:BER:IAFFilter <enable>
Description	This command enables/disables include addresses in frame filter on receiver.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = ON*
Response	None.
Example	`MEAS:SET:EVAL:BER:IAFF OFF`
Note	This setting applies to all ports. This only applies to Ethernet and Fibre Channel.

Syntax	MEASurement:SETup:EVALuation:BER:IAFFilter?
Description	This query returns if include addresses in frame filter on receiver is enabled/disabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`MEAS:SET:EVAL:BER:IAFF?` → `0`
Note	This setting applies to all ports. This only applies to Ethernet and Fibre Channel.

18.2.85 MEASurement:SETup:EVALuation:BER:CLFrames

Syntax	MEASurement:SETup:EVALuation:BER:CLFrames <enable>
Description	This command enables/disables count lost frames as pattern errors.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`MEAS:SET:EVAL:BER:CLF ON`
Note	This setting applies to all ports. This only applies to Ethernet and Fibre Channel.

Syntax	MEASurement:SETup:EVALuation:BER:CLFrames?
Description	This query return if count lost frames as pattern errors is enabled/disabled.
Parameter	None.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`MEAS:SET:EVAL:BER:CLF?` → `1`
Note	This setting applies to all ports. This only applies to Ethernet and Fibre Channel.

18.2.86 MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds [:ENABle]

Syntax	MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds[:ENABle] <enable>
Description	This command enables/disables BER Pattern error thresholds.
Parameters	<Pt> = Port number
	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Examples	`MEAS:SET:EVAL:BER:PORT1:PTHR ON`
Note	

Syntax	MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds[:ENABle]?
Description	This query returns whether or not Pattern error thresholds are enabled.
Parameter	<Pt> = Port number
Response	<enable> = <BOOLEAN RESPONSE DATA>
Examples	`MEAS:SET:EVAL:BER:PORT1:PTHR?` → `1`
Note	

18.2.87 MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds:MODE

Syntax	MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds:MODE <mode>
Description	This command sets the Pattern errors threshold mode.
Parameters	<Pt> = Port number
	<mode> = <CHARACTER PROGRAM DATA> COUNT: Count RATE: Rate PERCENT: Rate Percent *DEFault = COUNT*
Response	None.
Example	`MEAS:SET:EVAL:BER:PORT1:PTHR:MODE COUNT`
Note	

Syntax	MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds:MODE?
Description	This query return the Pattern errors threshold mode.
Parameters	<Pt> = Port number
Response	<mode> = <CHARACTER RESPONSE DATA>
Example	`MEAS:SET:EVAL:BER:PORT1:PTHR:MODE? COUNT`
Note	

18.2.88 MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds:COUNt

Syntax	MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds:COUNt <value>
Description	This command sets the Pattern errors threshold count value.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0, MAXimum = 4294967295, DEFault = 0*
Response	None.
Example	`MEAS:SET:EVAL:BER:PORT1:PTHR:COUN 100`
Note	

Syntax	MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds:COUNt?
Description	This query return the Pattern errors threshold count value.
Parameters	<Pt> = Port number
Response	<value> = <NR1 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:BER:PORT1:PTHR:COUN? → 100`
Note	

18.2.89 MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds:RATio

Syntax	MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds:RATio <ratio>
Description	This command sets the Pattern errors threshold ratio value.
Parameters	<Pt> = Port number
	<ratio> = <NUMERIC PROGRAM DATA> *MINimum = 0.00, MAXimum = 1.00, DEFault = 0.00*
Response	None.
Example	`MEAS:SET:EVAL:BER:PORT1:PTHR:RAT 1.50E-01`
Note	

Syntax	MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds:RATio?
Description	This query return the Pattern errors threshold ratio value.
Parameters	<Pt> = Port number
Response	<value> = <NR3 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:BER:PORT1:PTHR:RAT?` → `1.50E-01`
Note	

18.2.90 MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds :PERCent

Syntax	MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds:PERCent <value>
Description	This command sets the Pattern errors threshold percentage value.
Parameters	<Pt> = Port number
	<value> = <NUMERIC PROGRAM DATA> *MINimum = 0.00000, MAXimum = 100.00000, DEFault = 0.00000*
Response	None.
Example	`MEAS:SET:EVAL:BER:PORT1:PTHR:PERC 10.00000`
Note	

Syntax	MEASurement:SETup:EVALuation:BER:PORT<Pt>:PTHResholds:PERCent?
Description	This query return the Pattern errors threshold percentage value.
Parameters	<Pt> = Port number
Response	<value> = <NR2 NUMERIC RESPONSE DATA>
Example	`MEAS:SET:EVAL:BER:PORT1:PTHR:PERC?` → `10.00000`
Note	

18.2.91 MEASurement:SETup:POINtermovement:SDH:MODE

Syntax	MEASurement:SETup:POINtermovement:SDH:MODE <enable>
Description	This command enables/disables pointer movement measurement.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`MEAS:SET:POIN:SDH:MODE ON`
Note	This setting applies to all ports.

Syntax	MEASurement:SETup:POINtermovement:SDH:MODE?
Description	This query return if pointer movement is enabled/disabled.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`MEAS:SET:POIN:SDH:MODE?` → `1`
Note	This setting applies to all ports.

18.2.92 MEASurement:SETup:POINtermovement:SONet:MODE

Syntax	MEASurement:SETup:POINtermovement:SONet:MODE <enable>
Description	This command enables/disables pointer movement measurement.
Parameter	<enable> = <BOOLEAN PROGRAM DATA> *DEFault = OFF*
Response	None.
Example	`MEAS:SET:POIN:SON:MODE ON`
Note	This setting applies to all ports.

Syntax	MEASurement:SETup:POINtermovement:SONet:MODE?
Description	This query return if pointer movement is enabled/disabled.
Response	<enable> = <BOOLEAN RESPONSE DATA>
Example	`MEAS:SET:POIN:SON:MODE?` → `1`
Note	This setting applies to all ports.

18.3 Information

18.3.1 MEASurement:INFO:TBEGin?

Syntax	MEASurement:INFO:TBEGin?
Description	This query returns the start time of the measurement.
Parameter	None.
Response	<datetime> = <STRING RESPONSE DATA> Format: YYYY-MM-DD HH:MM:SS
Example	`MEAS:INFO:TBEG?` → `"2014-07-17 11:26:11"`
Note	Only possible if a measurement is in the memory.

18.3.2 MEASurement:INFO:TEND?

Syntax	MEASurement:INFO:TEND?
Description	This query returns the stop time of the measurement.
Parameter	None.
Response	<datetime> = <STRING RESPONSE DATA> Format: YYYY-MM-DD HH:MM:SS
Example	`MEAS:INFO:TEND?` → `"2014-07-17 11:26:21"`
Note	Only possible if a measurement is in the memory and a measurement is not running.

18.3.3 MEASurement:INFO:MDURation?

Syntax	MEASurement:INFO:MDURation?
Description	This query returns the measurement duration.
Parameter	None.
Response	<duration> = <STRING RESPONSE DATA> Format: DD-HH:MM:SS
Example	`MEAS:INFO:MDUR?` → `"01-00:03:23"`
Note	Only possible if a measurement is in the memory.

18.3.4 MEASurement:INFO:STATus<Pt>?

Syntax	MEASurement:INFO:STATus<Pt>?
Description	This query returns the alarm/error status of all active interfaces and the state of the evaluation limits.
Parameter	<Pt> = Port number
Response	<status> = <STRING RESPONSE DATA> "No Trouble" "Alarms" "Errors" "Alarms and Errors" "N/A": Data not available
	<state> = <STRING RESPONSE DATA> "Disabled / Not Supported" "Below Limits" "Within Limits" "Exceed Limits" "N/A": Data not available
Example	`MEAS:INFO:STAT1?` → `"Errors","Within Limits"`
Note	Only possible if a measurement is in the memory.

18.3.5 MEASurement:INFO:IMIN?

Syntax	MEASurement:INFO:IMIN?
Description statistics intervals.	This query returns the value by adding +1 to minimum index of
Parameter	None
Response	<intervalindex> = <NR1 NUMERIC RESPONSE DATA>
Example	`MEAS:INFO:IMIN?` → `1`
Note	If no intervals are available the NaN (section 1.6.1) is returned. This query is kept for maintenance purpose. Please see MEAS:SET:SEL for details.

18.3.6 MEASurement:INFO:IMAX?

Syntax	MEASurement:INFO:IMAX?
Description statistics intervals.	This query returns the value by adding +1 to maximum index of
Parameter	None
Response	<intervalindex> = <NR1 NUMERIC RESPONSE DATA>
Example	`MEAS:INFO:IMAX?` → `16`
Note	If no intervals are available the NaN (section 1.6.1) is returned. This query is kept for maintenance purpose. Please see MEAS:SET:SEL for details.

18.3.7 MEASurement:INFO:IFIRst?

Syntax	MEASurement:INFO:IFIRst?
Description	This query returns the minimum index of statistics intervals.
Parameter	None
Response	<intervalindex> = <NR1 NUMERIC RESPONSE DATA>
Example	`MEAS:INFO:IFIR?` → `0`
Note	If no intervals are available the NaN (section 1.6.1) is returned.

18.3.8 MEASurement:INFO:ILASt?

Syntax	MEASurement:INFO:ILASt?
Description	This query returns the maximum index of statistics intervals.
Parameter	None
Response	<intervalindex> = <NR1 NUMERIC RESPONSE DATA>
Example	`MEAS:INFO:ILAS?` → `15`
Note	If no intervals are available the NaN (section 1.6.1) is returned.

18.4 Event Log

18.4.1 MEASurement:ELOG:MINimum?

Syntax	MEASurement:ELOG:MINimum?
Description	This query returns the first index present in the event log.
Parameter	None
Response	<index> = <NR1 NUMERIC RESPONSE DATA>
Example	`MEAS:ELOG:MIN?` → `1`
Note	

18.4.2 MEASurement:ELOG:MAXimum?

Syntax	MEASurement:ELOG:MAXimum?
Description	This query returns the last or the latest index present in the event log.
Parameter	None
Response	<index> = <NR1 NUMERIC RESPONSE DATA>
Example	`MEAS:ELOG:MAX?` → `321`
Note	

18.4.3 MEASurement:ELOG:FETCh?

Syntax	MEASurement:ELOG:FETCh? [<index>[,<count>]]
Description	This query returns a number of log entries starting from the specified index.
Parameters	<index> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=64000, DEFault=1*
	<count> = <NUMERIC PROGRAM DATA> *MINimum=1, MAXimum=400, DEFault=20*
Response	The result is returned as a list of result items. A result item has the format: `(<index>,<dateTime>,<portNo>,<type>,<source>,<description>,<countDuration>)` The `<portNo>` field is '0' when the event is not related to a specific port. The `<countDuration>` is an error counter for errors and a duration in seconds for alarms.

Example	`MEAS:ELOG:FETC?` →
	`(1,"2015-01-09 16:20:45",0,"Test","SYST","Started",0),`
	`(2,"2015-01-09 16:20:47",1,"Error","ETH","Pattern errors",5),`
	`(3,"2015-01-09 16:20:50",0,"Test","SYST","Stopped",0)`

Note	The Cable Test and the No Frame (MT1100A) applications have no event log feature.

18.4.4 MEASurement:ELOG:FETCh:NEXT?

Syntax	MEASurement:ELOG:FETCh:NEXT?
Description	This query returns a number of log entries starting from where the previous `:FETCh` or `:FETCh:NEXT` stopped.
Parameters	None
Response	The response is similar to the response of the MEASurement:ELOG:FETCh? command.
Example	`MEAS:ELOG:FETC:NEXT? 1`
Note	

18.4.5 MEASurement:ELOG:EXPort

Syntax	MEASurement:ELOG:EXPort <file> [<delimiter>]
Description	This command writes the complete event log to the specified file in a CSV format. The file can be written to the internal disk or to a connected USB memory device. The specified delimiter is used as field delimiter. Lines are terminated with CR-LF (0x0d,0x10).
Parameters	<file> = <STRING PROGRAM DATA> The path and name of the file to store the data.
	<delimiter> = <CHARACTER PROGRAM DATA> COMMa: Use ',' as field separator. SEMicolon: Use ';' as field separator. TABulator: Use a tabulator character (0x09) as field separator. *DEFault = COMMa*
Response	None.
Example	`MEAS:ELOG:EXP "Internal/my-log-data.csv"`
Note	Files must be saved to the `Internal/` directory or a sub-directory hereof.
	When a USB storage device is mounted, files are stored via the Usb/ directory.

18.5 Result

18.5.1 MEASurement:RESult:SUMMary?

Syntax	MEASurement:RESult:SUMMary?
Description	This query returns the measurement result.
Response	<result> = <CHARACTER RESPONSE DATA> Pass Trouble Warning N/A
Example	`MEAS:RES:SUMM?` → `Pass`
Note	

18.5.2 MEASurement:RESult:SUMMary:STATistics:PORT<Pt>?

Syntax	MEASurement:RESult:SUMMary:STATistics:PORT<Pt>?
Description	This query returns the measurement result of statistics.
Response	<result> = <CHARACTER RESPONSE DATA> Pass Trouble Warning N/A
Example	`MEAS:RES:SUMM:STAT:PORT1?` → `PASS`
Note	

Appendix A

Example Scripts

This chapter shows various example scripts for all interfaces which are remote controllable.

A.1 Hints

To ensure that the instrument always start from a well defined state, it is in general a good idea to begin all scripts with the following command. It will terminate all application servers (virtual instruments).

```
*RST
```

A.2 2 Mbps BERT

This example runs a BER test. It requires an unbalanced cable to be connected from TX1 to RX1. It configures TX1 and sets RX1 to follow TX1. TX2 and RX2 are switched off. Stimuli is set to generate pattern errors with a rate of $1 \cdot 10^{-4}$. A Statistics measurement is set to run for 10 seconds and it finally reads the measured pattern errors from the total interval.
The expected execution of the last query is `TMBP:RX1:IFET? (PATT)` → `(1024,0.0001)`

```
*RST
INST:STAR TP-BERT-SDHPDH,1-PORT1

TMBP:TX1 ON
TMBP:TX1:CONN UNB; DINS OFF; CSO INT; FOFF 0; CODE HDB3; PCMF ON; CRC4 ON
TMBP:TX1:SAB #B11111,#B11111,#B11111,#B11111
TMBP:TX1:PATT PRBS11; PINV OFF; PTSL (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16)
TMBP:TX1:UTSL #B01010101
TMBP:TX1:CAS OFF

TMBP:RX1:ENAB ON; FOLL TX

TMBP:STIM:TX1:ERR PATT; EINS B04; EBL 1
SYST:WAIT:DUR 3

MEAS:SET:ILEN 5S; STAR IMM; STOP DUR; SDUR 0,0,0,10
MEAS:STAR
SYST:WAIT:IDLE

MEAS:SET:SEL TOT
TMBP:RX1:IFET? (PATT)
MEAS:SET:SEL 0
TMBP:RX1:IFET? (PATT)
MEAS:SET:SEL 1
TMBP:RX1:IFET? (PATT)

SYST:ERR?
INST:TERM
```

A.3 2 Mbps Status & Stimuli

This example demonstrates status and stimuli. It requires 2mbps interface and a loopback cable from TX1 to RX1. The functionality is as following:

- reset to factory default, enable device dependent status and clear status
- setup interface and read status
- clear status, set AIS alarm and read status
- clear status, insert one pattern error and read status

Input script

```
*RST
INST:STAR TP-BERT-SDHPDH,1-PORT1

STAT:PRES
STAT:PORT:ENAB 1-PORT1,1; PTR  1-PORT1,1; NTR 1-PORT1,0

TMBP:TX1 ON; RX1 ON
TMBP:RX1:FOLL TX
SYST:WAIT:DUR 3
*CLS

*STB?; STAT:PORT:EVEN? 1-PORT1; COND? 1-PORT1
STAT:INT:PORT1:EVEN?; COND?
TMBP:STAT:RX1:AES:EVEN?; COND?
TMBP:STAT:RX1:ALAR:EVEN?; COND?

TMBP:STIM:TX1:ALAR AIS
SYST:WAIT:DUR 2

*STB?; STAT:PORT:EVEN? 1-PORT1; COND? 1-PORT1
STAT:INT:PORT1:EVEN?; COND?
TMBP:STAT:RX1:AES:EVEN?; COND?
TMBP:STAT:RX1:ALAR:EVEN?; COND?
TMBP:STIM:TX1:ALAR NAL
SYST:WAIT:DUR 2
*CLS

TMBP:STIM:TX1:EINS MAN; EBL 1; ERR PATT
SYST:STIM:INS
SYST:WAIT:DUR 3

STAT:INT:PORT1:EVEN?; COND?
TMBP:STAT:RX1:AES:EVEN?; COND?
TMBP:STAT:RX1:ERR:EVEN?; COND?

SYST:ERR?
INST:TERM
```

→

Output

```
0;0;0
0;0
0;0
0;0

1;1;1
1;1
1;1
64;64

1;1
2;2
2;0

0,"No error"
```

A.4 E3 BERT

This example runs a BER test. It requires a cable to be connected from TX1 to RX1. It configures TX1 and sets RX1 to follow TX1. TX2 and RX2 are switched off. Stimuli is set to generate pattern errors with a rate of $1 \cdot 10^{-3}$. A Statistics measurement is set to run for 10 seconds and it finally reads the measured pattern errors from the total interval.
The expected execution of the last query is `TMBP:RX1:IFET? (PATT)` → `(343680,0.001)`

```
*RST
INST:STAR TP-BERT-SDHPDH,1-PORT1

*CLS
STAT:PRES

E3:TX1:ENAB ON
E3:TX1:CSO INT
E3:TX1:FOFF 0
E3:TX1:PCMF OFF
E3:TX1:PATT PRBS11
E3:TX1:PINV ON

E3:RX1:ENAB ON
E3:RX1:FOLL TX

E3:STIM:TX1:ERR PATT
E3:STIM:TX1:EINS B03
E3:STIM:TX1:EBL 1

MEAS:SET:ILEN 5S
MEAS:SET:STAR IMM
MEAS:SET:STOP DUR
MEAS:SET:SDUR 0, 0, 0, 10
MEAS:STAR

SYST:WAIT:IDLE

MEAS:SET:SEL TOT
E3:RX1:IFET? (PATT)
```

A.5 E4 BERT

This example runs a BER test. It requires a cable to be connected from TX1 to RX1. It configures TX1 and sets RX1 to follow TX. Stimuli is set to generate pattern errors with a rate of $1 \cdot 10^{-3}$. A Statistics measurement is set to run for 11 seconds and it finally reads the measured pattern errors and pattern slips from the total interval.
The expected execution of the last two queries are `E4:RX<Rx>:IFET? (PATT)` → `(1392640,0.001)`

```
*RST
INST:STAR TP-BERT-SDHPDH,1-PORT1,1-PORT2

*CLS
STAT:PRES

E4:TX:ENAB ON
E4:TX:CSO INT
E4:TX:FOFF 0
E4:TX:PCMF OFF
E4:TX:PATT PRBS11
E4:TX:PINV ON

E4:RX1:ENAB ON
E4:RX1:FOLL TX

E4:RX2:ENAB ON
E4:RX2:FOLL TX

E4:STIM:TX:ERR PATT
E4:STIM:TX:EINS B03
E4:STIM:TX:EBL 1

MEAS:SET:ILEN 5S
MEAS:SET:STAR IMM
MEAS:SET:STOP DUR
MEAS:SET:SDUR 0, 0, 0, 11
MEAS:STAR

SYST:WAIT:IDLE

MEAS:SET:SEL TOT
E4:RX1:IFET? (PATT,PSL)
E4:RX2:IFET? (PATT,PSL)

SYST:ERR?
INST:TERM
```

A.6 SDH BERT

This example runs a BER test. It requires a cable to be connected from TX1 to RX1. It configures TX1 and sets RX1 to follow TX. Stimuli is set to generate pattern errors with a rate of $1 \cdot 10^{-7}$. A Statistics measurement is set to run for 10 seconds and it finally reads the measured pattern errors from the total interval.
The expected execution of the last two queries are `SDH:RX<Rx>:IFET? (ERRPRBS)` → `(150,1E-07)`

```
*RST
INST:STAR TP-BERT-SDHPDH,1-PORT1,1-PORT2

SDH:TX:ELEC NORM

SDH:RX1:INT ELEC
SDH:RX1:FOLL TX

SDH:RX2:INT ELEC
SDH:RX2:FOLL TX

SDH:STIM:TX:ERR PRBS
SDH:STIM:TX:EINS B07
SDH:STIM:TX:EBL 1

MEAS:SET:ILEN 5S
MEAS:SET:STAR IMM
MEAS:SET:STOP DUR
MEAS:SET:SDUR 0, 0, 0, 10
MEAS:STAR

SYST:WAIT:IDLE

MEAS:SET:SEL TOT
SDH:RX1:IFET? (ERRPRBS)
SDH:RX2:IFET? (ERRPRBS)

SYST:ERR?
INST:TERM
```

A.7 Ethernet BERT

This example runs a BER test. It requires a cable from Port 1 to Port 2 (PORT1 to PORT2). It configures PORT1 and set PORT2 to follow PORT1. Stimuli is set to generate pattern errors with a rate of $1 \cdot 10^{-7}$. A Statistics measurement is set to run for 10 seconds and it finally reads the measured pattern errors from the total interval.
The expected execution of the last two queries are approximately `ETH:PORT<Pt>:IFET? (BPE)` → `(20,1E-07)`.

```
*RST
INST:STAR TP-BERT-ETH,1-PORT1,1-PORT2

ETH:PORT1:MODE ANEG
ETH:PORT2:MODE ANEG
ETH:PORT1:ANEG (100MF)
ETH:PORT2:ANEG (100MF)
SYST:WAIT:DUR 4

ETH:FOLL:STR ON
ETH:FOLL:TRAF ON

ETH:PORT1:STR1:PAYL PRBS11

ETH:PORT1:TRAF:DMOD CONT
ETH:PORT1:TRAF:STR1:LL:PROF CONS
ETH:PORT1:TRAF:STR1:LL 100 PCT

ETH:PORT1:STIM:ERR PRBS
ETH:PORT2:STIM:ERR PRBS
ETH:PORT1:STIM:EINS BE7
ETH:PORT2:STIM:EINS BE7

MEAS:SET:ILEN 5S
MEAS:SET:STAR IMM
MEAS:SET:STOP DUR
MEAS:SET:SDUR 0, 0, 0, 10

ETH:PORT1:TRAF:GEN:STAR
ETH:PORT2:TRAF:GEN:STAR
SYST:WAIT:DUR 2

MEAS:STAR
SYST:WAIT:IDLE

ETH:PORT1:TRAF:GEN:STOP
ETH:PORT2:TRAF:GEN:STOP

MEAS:SET:SEL TOT
ETH:PORT1:IFET? (BPE)
ETH:PORT2:IFET? (BPE)

SYST:ERR?
INST:TERM
```

A.8 Ethernet Discovery

This section describes how to create Discovery Test programs. To discover the Network Masters, it is necessary to enable the Allow In-band control function of the application running on the target Network Master(s).

1. Specify the search type and perform a measurement.

```
ETH:DISC:SET:TYPE MULTI
MEAS:STAR
SYST:WAIT:IDLE
```

2. Query the number of the Network Masters discovered over a network.

```
ETH:DISC:RES:NREQ?
```

3. Query the detailed information about the Network Master where you want to switch the application. For example, the following command queries the name of the Network Master No.1.

```
ETH:DISC:RES:SREQ:DET? 1,(NICN)
```

4. Specify the application to be switched and its settings on the Application Change function. For example, the following commands specify RFC2544 as the application to be switched, and specify the each setting required for the measurement.

```
ETH:DISC:CAPP:SAPP RFC
ETH:DISC:CAPP:TARG:MAC:SOUR "00-50-C2-35-D2-EF"
ETH:DISC:CAPP:TARG:IPV4:SOUR "172.29.2.36"
ETH:DISC:CAPP:TARG:IPV4:DEST "172.29.2.37"
```

5. Apply the specified application to the target Network Master. For example, the following command applies the specified application to the Network Master No.1.

```
ETH:DISC:CAPP 1
```

6. Update the specified settings to the target Network Master, and query the result of updating repeatedly until SUCCESS is returned. For example, the following commands update information about the Network Master No.1 and query the result of updating.

```
ETH:DISC:RES:UPD 1
ETH:DISC:RES:UPD:RES?
```

7. Confirm whether the application has switched to the specified one. For example, the following command queries the application name of the Network Master No.1.

```
ETH:DISC:RES:SREQ:DET? 1,(CAPP)
```

A.9 Fibre Channel BERT

This example runs a BER test. It requires a cable from PORT1 to PORT2. It configures PORT1 and set PORT2 to follow PORT1. Stimuli is set to generate pattern errors with a rate of $1 \cdot 10^{-7}$. A Statistics measurement is set to run for 10 seconds and it finally reads the measured pattern errors from the total interval.
The expected execution of the last two queries are approximately `FCH:PORT<Pt>:IFET? (PERR)` → `(567,1E-07)`.

```
*RST
INST:STAR TP-BERT-FC,1-PORT1,1-PORT2

FCH:PORT1:MODE FC400
FCH:PORT2:MODE FC400

FCH:PORT1:TOP PTP; :FCH:PORT1:PTP:LOG ON
FCH:PORT2:TOP PTP; :FCH:PORT2:PTP:LOG ON
FCH:PORT1:DEST:LOG
SYST:WAIT:DUR 2

FCH:PORT2:FOLL:FRAM 1
FCH:PORT1:FRAM:FRAM FT1
FCH:PORT1:FRAM:CONT PRBS23

FCH:PORT2:FOLL:GEN 1; STR 1
FCH:PORT1:TRAF:STR:LL 50.0

FCH:PORT1:STIM:ERR BIT; EINS BE7; EBL 1
FCH:PORT2:STIM:ERR BIT; EINS BE7; EBL 1

FCH:TRAF:GEN:STAR
SYST:WAIT:DUR 2

MEAS:SET:ILEN 5S
MEAS:SET:STAR IMM
MEAS:SET:STOP DUR
MEAS:SET:SDUR 0, 0, 0, 10
MEAS:STAR

SYST:WAIT:IDLE

MEAS:SET:SEL TOT
FCH:PORT1:IFET? (BPE)
FCH:PORT2:IFET? (BPE)

SYST:ERR?

INST:TERM
```

A.10 OTN BERT

This example runs a BER test. It requires a cable to be connected from TX1 to RX1. It configures TX1 and sets RX1 to follow TX. Stimuli is set to generate pattern errors with a rate of $1 \cdot 10^{-7}$. A Statistics measurement is set to run for 10 seconds and it finally reads the measured pattern errors from the total interval.
The expected execution of the last two queries are `OTN:RX<Rx>:IFET? (PRBSBIT)` → `(9995,1E-07)`

```
*RST
INST:STAR TP-BERT-OTN,1-PORT1,1-PORT2

OTN:TX1:ENAB NORM
OTN:TX1:MAPP:OUTP OTU2
OTN:TX1:MAPP:CSIG PRBS

OTN:TX2:FOLL TX1

OTN:RX1:INT SFP
OTN:RX1:FOLL TX

OTN:RX2:INT SFP
OTN:RX2:FOLL TX

OTN:STIM:TX1:TYPE AEIN
OTN:STIM:TX1:AEIN:LEV ODU2
OTN:STIM:TX1:AEIN:TYPE PRBSBIT
OTN:STIM:TX1:AEIN:INS RATE
OTN:STIM:TX1:AEIN:RATE R1E7

OTN:STIM:TX2:TYPE AEIN
OTN:STIM:TX2:AEIN:LEV ODU2
OTN:STIM:TX2:AEIN:TYPE PRBSBIT
OTN:STIM:TX2:AEIN:INS RATE
OTN:STIM:TX2:AEIN:RATE R1E7

MEAS:SET:ILEN 5S
MEAS:SET:STAR IMM
MEAS:SET:STOP DUR
MEAS:SET:SDUR 0, 0, 0, 10
MEAS:STAR

SYST:WAIT:IDLE

MEAS:SET:SEL TOT
OTN:RX1:IFET? (PRBSBIT)
OTN:RX2:IFET? (PRBSBIT)

SYST:ERR?
INST:TERM
```

A.11 CPRI BERT

This example runs a BER test. It requires a cable from PORT1 to PORT2. It configures PORT1 and set PORT2 to follow PORT1. Stimuli is set to generate pattern errors with a rate of $1 \cdot 10^{-4}$. A Statistics measurement is set to run for 10 seconds and it finally reads the measured pattern errors from the total interval.
The expected execution of the last two queries are approximately `CPRI:PORT<Pt>:IFET? (PE)` → `(7372800,0.0001)`.

```
*RST
INST:STAR TP-BERT-CPRI,1-PORT1,1-PORT2

CPRI:PORT1:MODE NORM
CPRI:PORT2:MODE NORM

CPRI:PORT1:LRAT 9830M
CPRI:PORT2:LRAT 9830M

CPRI:PORT1:CONT LINK
CPRI:PORT2:CONT LINK
SYST:WAIT:DUR 2

CPRI:PORT2:SET:FOLL ON
CPRI:PORT1:PATT PRBS15

CPRI:PORT1:STIM:ERR PE; EINS B04
CPRI:PORT2:STIM:ERR PE; EINS B04


MEAS:SET:ILEN 5S
MEAS:SET:STAR IMM
MEAS:SET:STOP DUR
MEAS:SET:SDUR 0, 0, 0, 10
MEAS:STAR

SYST:WAIT:IDLE

MEAS:SET:SEL TOT
CPRI:PORT1:IFET? (PE)
CPRI:PORT2:IFET? (PE)

SYST:ERR?

INST:TERM
```

Abbreviations

AISL	Line Alarm Indication Signal
AISP	STS Path Alarm Indication Signal
AISV	Virtual Tributary Alarm Indication Signal
ALS	Alarm Seconds
AMI	Alternate Mark Inversion
APS	Automatic Protection Switching
ARP	Address Resolution Protocol
AU	Administrative Unit
AU-AIS	Administrative Unit Alarm Indication Signal
AU-LOP	Administrative Unit Loss Of Pointer
AU-NDF	Administrative Unit New Data Flag
AVT	Available Time
B-TAG	Backbone VLAN tag
B8ZS	Bipolar 8-zero substitution *(code with a maximum of 7 consecutive zeroes)*
BBE	Background Block Error
BDP	Bandwidth Delay Product
BER	Bit Error Ratio
BERT	Bit Error Rate Test
BIS	Bringing Into Service
BPV	Bipolar Violation
C-bits	Bit stuffing control bits
CAS	Channel Associated Signaling
CCM	Continuity Check Message
CRC	Cyclic Redundancy Check
DEI	Drop Eligible Indicator
DM1	One-way delay measurement
DMM	Delay Measurement Message
DNS	Domain Name System
DSCP	Differentiated Services Code Point
EFS	Error Free Seconds

EoMPLS	Ethernet over MPLS
ES	Errored Second
ESF	Extended superframe
ESMC	Ethernet Synchronization Messaging Channel
EXM	Experimental OAM Message
EXZ	Excessive Zeroes
F-bits	Framing bits
FDL	Facility Data Link
fps	frames per second
fps	frames per second
HDB3	High Density Bipolar 3 *(code with a maximum of 3 consecutive zeroes)*
HP	High-order Path
HP-PLM	High-order Path Payload Label Mismatch
HP-RDI	High-order Path Remote Defect Indicator
HP-REI	High-order Path Remote Error Indication
HP-TIM	High-order Path Trace Identifier Mismatch
HP-UNEQ	High-order Path Unequipped
HR	Hypothetical Reference allocation
IEEE	Institute of Electrical and Electronic Engineers
IFG	InterFrame Gap
ILA	In-Lane Alignment
IP	Internet Protocol
ISDN	Integrated Services Digital Network
LAN	Local Area Network
LAN	Local Area Network
LBM	Loop-back Message
LBR	Loop-back Response
LFS	Link Fault Signaling
LMM	Loss Measurement Message
LMP	Link Management Protocol
LOF	Loss Of Frame
LOMV	Virtual Tributary Loss Of Multiframe
LOPP	STS Path Loss Of Pointer
LOPV	Virtual Tributary Loss Of Pointer
LOR	Loss Of Recovery
LOS	Loss Of Signal

LP	Low-order Path
LP-PLM	Low-order Path Payload Label Mismatch
LP-RDI	Low-order Path Remote Defect Indicator
LP-REI	Low-order Path Remote Error Indication
LP-TIM	Low-order Path Trace Identifier Mismatch
LP-UNEQ	Low-order Path Unequipped
LSS	Loss of Signal Synchronization
LTM	Link Trace Message
MAC	Media Access Control
Mbps	Mega bit per second
MCC	Maintenance Communications Channel
ME	Maintenance Entity
MEG	Maintenance Entity Group
MEP	Management End Point
MF	Multi Frame
MFAS	Multi Frame Alignment Signal
MiM	MAC-in-MAC
MPLS	Multiprotocol Label Switching
MS	Multiplex Section
MS-AIS	Multiplex Section Alarm Indication Signal
MS-RDI	Multiplex Section Remote Defect Indication
MS-REI	Multiplex Section Remote Error Indication
MSS	Maximum Segment Size
MTU	Maximum Transmission Unit
MUX	Multiplex
NDF	New Data Flag
NFAS	Non Frame Alignment Signal
NTP	Network Time Protocol
OAM	Operation, Administration and Maintenance
OC-N	Optical Carrier at level N (N = 1, 3, 12, 24, 48, 192 or 768)
OLA	Out of Lane Alignment
OOF	Out Of Frame
OOR	Out Of Recovery
OUI	Organizationally Unique Identifier
P-bits	Parity bits
PBB	Provider Backbone Bridges, known as MAC-in-MAC (MiM)

PCP	Priority Code Point
PDH	Plesiochronous Digital Hierarchy
PLMP	High-order Path Payload Label Mismatch
PLMV	Low-order Path Payload Label Mismatch
POH	Path Overhead
ppb	parts per billion
PRBS	Pseudo Random Binary Sequence
QL	Quality Level
QRSS	Quasi Random Signal Source
RAI	Remote Alarm Indication
RDI	Remote Defect Indication
RDIL	Line Remote Defect Indication
RDIP	High-order Path Remote Defect Indicator
RDIV	Low-order Path Remote Defect Indicator
REI	Remote Error Indication
REIL	Line Remote Error Indication
REIP	STS Path Remote Error Indication
REIV	Virtual Tributary Remote Error Indication
RPM	Real-time Performance Monitoring
RTT	Round-Trip Time
Rx	Receiver
SDH	Synchronous Digital Hierarchy
SES	Severely Errored Second
SF	Superframe
SID	Backbone Service Instance Identifier
SLM	Synthetic (frame-) Loss Measurement
SOH	Section Overhead
SONET	Synchronous Optical NETwork
SSF	Server Signal Fail
STAT	Status
STL	Synchronous Transport Lane
STM	Synchronous Transport Module
STS	Synchronous Transport Signal
TAI	Time, Atomic International
TC	Tandem Connection
TC-AIS	Tandem Connection Alarm Indication Signal

TC-BIP2	Tandem Connection Bit Interleaved Parity-2
TC-IEC	Tandem Connection Incoming Error Count
TC-LTC	Tandem Connection Loss of Tandem Connection
TC-ODI	Tandem Connection Outgoing Defect Indicator
TC-OEI	Tandem Connection Outgoing Error Indication
TC-RDI	Tandem Connection Remote Defect Indicator
TC-REI	Tandem Connection Remote Error Indication
TC-TIM	Tandem Connection Trace Identifier Mismatch
TC-UNEQ	Tandem Connection Unequipped
TCM	Tandem Connection Monitoring
TCP	Transmission Control Protocol
TIM	Trace Identifier Mismatch
TIMP	High-order Path Trace Identifier Mismatch
TIMV	Low-order Path Trace Identifier Mismatch
TLV	Type, Length, Value encoding
TOH	Transport Overhead
TP	Tributary Port
TRAU	Transcoder and Rate Adaptation Unit
TS	Tributary Slot
TTL	Time To Live (hop limit)
TU	Tributary Unit
TU-AIS	Tributary Unit Alarm Indication Signal
TU-LOM	Tributary Unit Loss Of Multiframe
TU-LOP	Tributary Unit Loss Of Pointer
TU-NDF	Tributary Unit New Data Flag
TUG	Tributary Unit Group
Tx	Transmitter
UAT	Unavailable Time
UCA	Use Customer Address
UDP	User Datagram Protocol
UNAV	Unavailable
UNEQP	High-order Path Unequipped
UNEQV	Low-order Path Unequipped
UTC	Universal Time Coordinated
VC	Virtual Container
VLAN	Virtual Local Area Network

VSM	Vendor Specific OAM Message
VT	Virtual Tributary
WAN	Wide Area Network
WWN	Worldwide Name