Application Note

Anritsu
envision : ensure

Externally Synchronized VNA Measurements

Summary

Typically in an antenna measurement setup the VNA is externally triggered. This is required to synchronize the VNA's data collection with other hardware such as signal sources and antenna positioners. External and remote trigger sources are used in many other scenarios such as parts handling and testing, on-wafer testing, and multi-instrument setups to control and synchronize data acquisition. The use of external or remote trigger sources requires a mechanism to monitor the status of the End of Sweep event so that in a remote control scenario we know when a full sweep of new data is available to transfer to external software. This application note will demonstrate the use of the End of Sweep monitoring via programming examples using the National Instruments VISA driver.

1. Trigger Source Commands

When controlling the VectorStar locally from the Front Panel there are 4 possible Trigger Sources, but from remote operation there are 5. The descriptions of 5 sources are listed here along with example code.

command parameters: AUTO | MANual | EXTTogpib | EXTernal | REMote

Internal/Auto - VectorStar supplies continuous trigger signals

```
//AUTO Trigger is the same as internal trigger
mbSession.Write(":TRIG:SOUR AUTO");
```

Manual - User supplies a trigger signal by pushing a front panel button

```
//Manual Trigger is initiated with a front panel button push
mbSession.Write(":TRIG:SOUR MAN");
```

EXTT (Remote Only) - The triggering source is from the VNA Rear Panel BNC connector to the GPIB parser which handles it as a GPIB Group Execute Trigger (via *DDT buffer). An external trigger signal initiates the commands stored in the *DDT buffer.

```
//EXTT Trigger while we set up Segment
mbSession.Write(":TRIG:SOUR EXTT");
```

External - The triggering source is from the rear panel external trigger port (BNC)

```
//External Trigger
mbSession.Write(":TRIG:SOUR EXT");
```

GPIB – the trigger is from a *TRG or :TRIG command

```
//Remote Trigger
mbSession.Write(":TRIG:SOUR REM");
```

This document focuses on the external (EXT) trigger only.

2. External Trigger Setup Commands

Trigger can be by point, by sweep, by channel or by all channels.

2.1 By Point

Each trigger pulse initiates the measurement of 1 point in one direction. For example, if we are displaying uncorrected S11 and S22 and we have 100 points, it takes 200 triggers to measure all the data in both directions. We measure all S11 (forward data) and then all S22 (reverse data). EOS is set when either the S11 or the S22 data is completed.

```
mbSession.Write(“:TRIG:EXT:TYP POIN”);
```

2.2 By Sweep

Each external trigger initiates a single sweep. For example, if we are displaying uncorrected S11 and S22 and we have 100 points, it takes 2 triggers to measure all the data in both directions. We measure all S11 (forward data) and then all S22 (reverse data). EOS is set when either the S11 or the S22 data is completed.

```
mbSession.Write(“:TRIG:EXT:TYP SWE”);
```

2.3 By Channel

Each external trigger initiates all sweeps for the active channel. For example, if we are displaying uncorrected S11 and S22 and we have 100 points, it takes 1 trigger to measure all the data in both directions. We measure all S11 (forward data) and then all S22 (reverse data). EOS is set when S22 data is completed for the current channel. If multiple channels are setup then the external trigger initiates all sweeps for the current channel. The next external trigger initiates all sweeps for the next channel.

```
mbSession.Write(“:TRIG:EXT:TYP CHAN”);
```

2.4 By All Channels

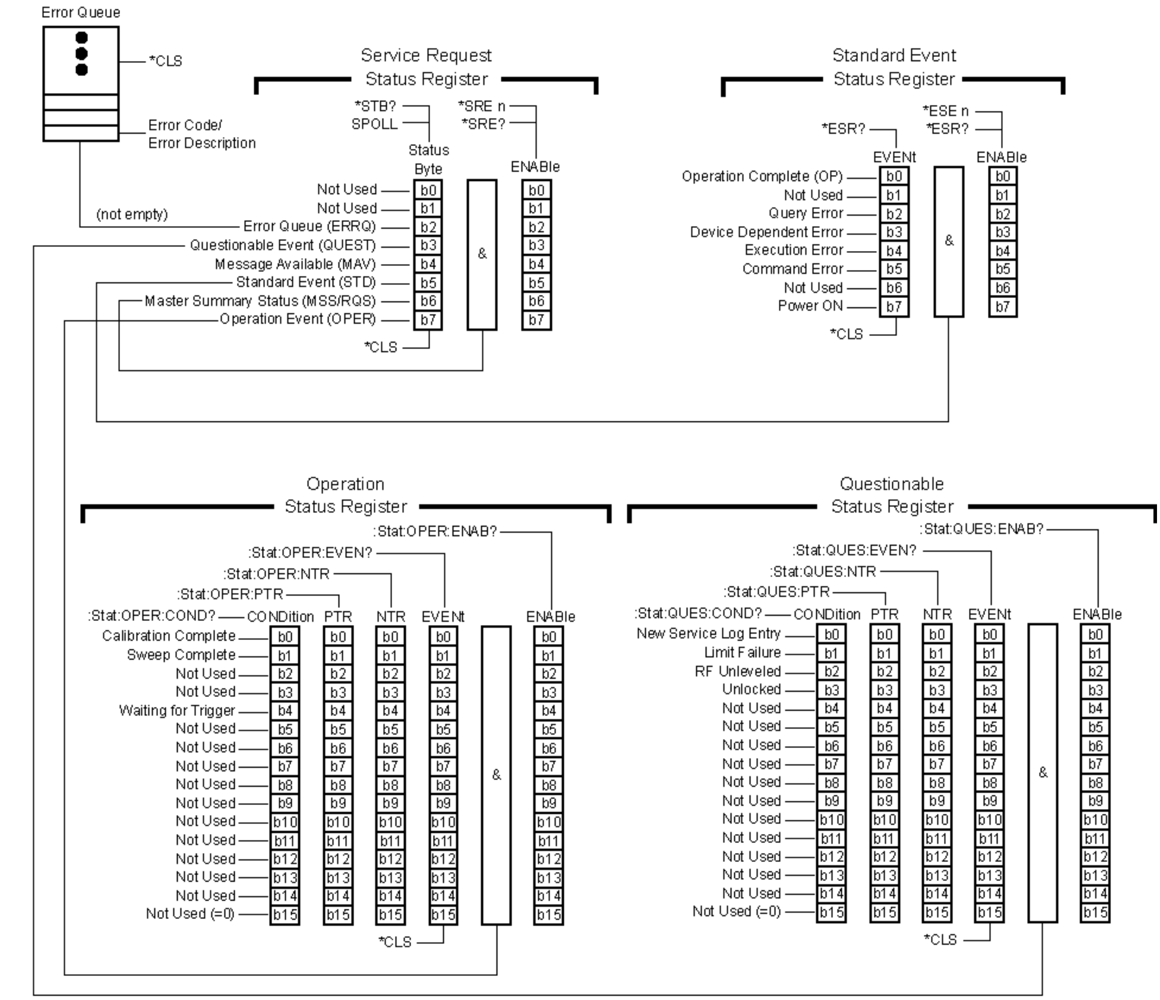
Each external trigger initiates all sweeps for the all channels. For example if we are displaying uncorrected S11 and S22 and we have 100 points and this is the case for 8 channels, it takes 1 trigger to measure all the data in both directions on all 8 channels. We measure all S11 (forward data) and then all S22 (reverse data) on channel 1, then channel 2, etc.. EOS is set when all data has been swept on all channels.

```
mbSession.Write(“:TRIG:EXT:TYP ALL”);
```

3. Monitoring for End of Sweep

We want to set up the Operation Status Register to monitor for Sweep Complete. This is b1 of that register. We’ll look for a positive transition, and ignore the negative transition.

```
//set up to query the sweep complete bit
mbSession.Write(“:STAT:OPER:ENAB 2”);
mbSession.Write(“:STAT:OPER:PTR 2”);
mbSession.Write(“:STAT:OPER:NTR 0”);
```



Bit Weight

b0	1	b8	256
b1	2	b9	512
b2	4	b10	1024
b3	8	b11	2048
b4	16	b12	4096
b5	32	b13	8192
b6	64	b14	16384
b7	128	b15	32768

Note: Not Used bits are always cleared to 0.

This feeds into the Operation Event bit (OPER) which is b7 of the Service Request Status Register. We query this register using the VISA function "ReadStausByte" (at the VISA function level this is viReadSTB)

```
//Read the Status Byte of Service Request Status Register
StatusByteFlags sb = mbSession.ReadStatusByte();
```

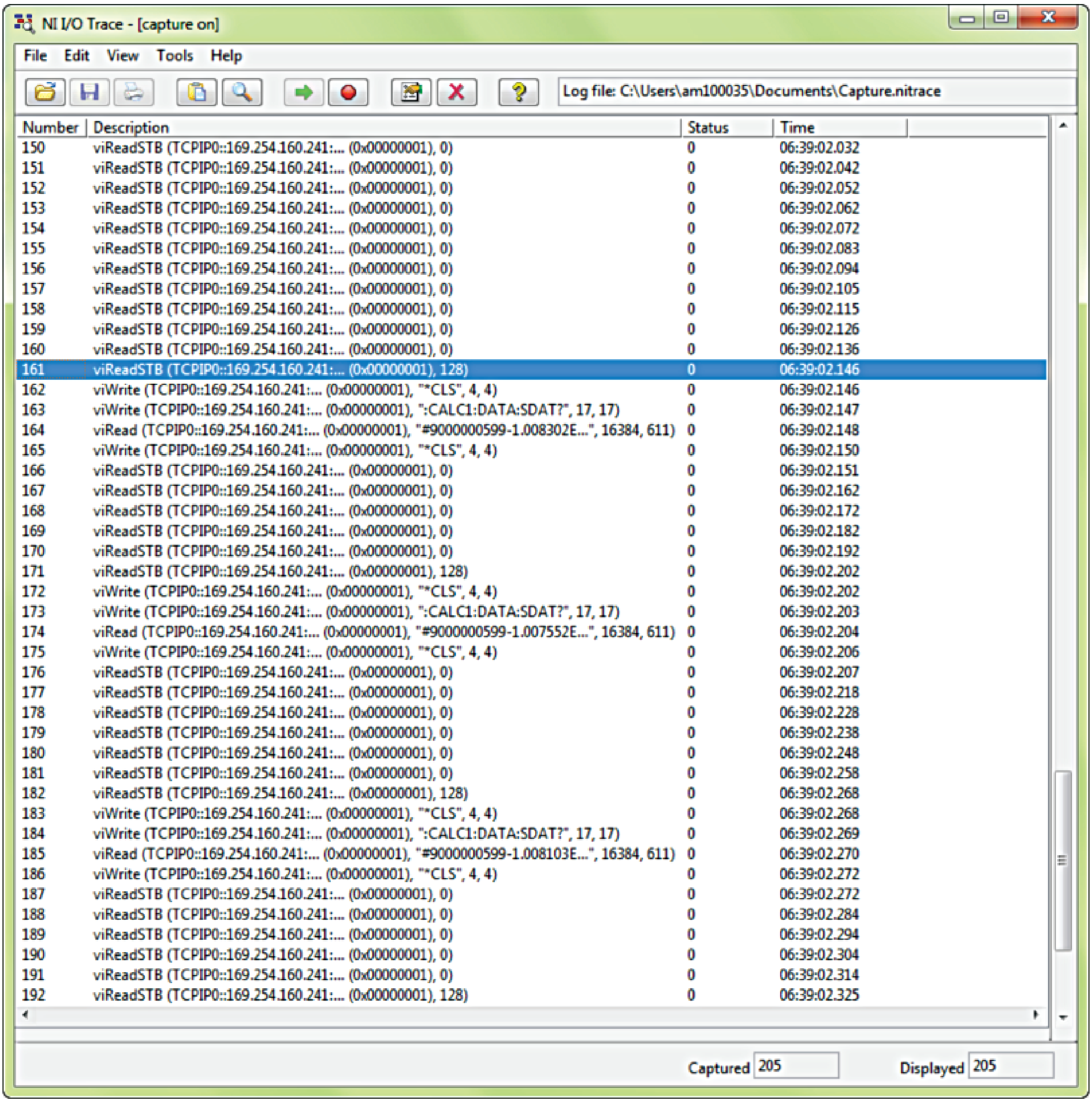
We can continuously monitor the status byte if we're waiting for an event like the Sweep Complete by placing this in a loop. In the following code we check for b2, b4 or b7.

```
//These are the bits to check
 int b2 = 4,      //Error Queue is not empty
     b4 = 16,     //MAV = Message Available
     b7 = 128;    //End of Sweep detected

//Read the Status Byte of Service Request Status Register
StatusByteFlags sb = mbSession.ReadStatusByte();

while (((int)sb & (b2 + b4 + b7)) == 0)
{
     if (stopwatch.ElapsedMilliseconds > 10000) //timeout mechanism, 10 seconds
         break;
     Thread.Sleep(10);
     sb = mbSession.ReadStatusByte();
}
```

If the status byte is zero we put the thread to sleep for 10ms and then continue. We added a stopwatch based timeout mechanism (using System.Diagnostics.Stopwatch) to make sure we don't get stuck in an infinite loop. The stopwatch is reset prior to entering the while loop (not shown above). If one of the bits is detected we fall out of the while loop and process as desired. Note that reading the status byte with the ReadStatusByte function does not reset the status byte. This reset can be done with the "*CLS" command.

4. Example with NI I/O Trace

Here we see the display of NI I/O Trace which has captured the continuous monitoring of the Sweep Complete bit via the ReadStatusByte command (which is shown here as the viReadSTB function).



Number	Description	Status	Time
150	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.032
151	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.042
152	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.052
153	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.062
154	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.072
155	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.083
156	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.094
157	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.105
158	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.115
159	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.126
160	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.136
161	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 128)	0	06:39:02.146
162	viWrite (TCPIP0::169.254.160.241:... (0x00000001), "*CLS", 4, 4)	0	06:39:02.146
163	viWrite (TCPIP0::169.254.160.241:... (0x00000001), ":CALC1:DATA:SDAT?", 17, 17)	0	06:39:02.147
164	viRead (TCPIP0::169.254.160.241:... (0x00000001), "#9000000599-1.008302E...", 16384, 611)	0	06:39:02.148
165	viWrite (TCPIP0::169.254.160.241:... (0x00000001), "*CLS", 4, 4)	0	06:39:02.150
166	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.151
167	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.162
168	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.172
169	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.182
170	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.192
171	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 128)	0	06:39:02.202
172	viWrite (TCPIP0::169.254.160.241:... (0x00000001), "*CLS", 4, 4)	0	06:39:02.202
173	viWrite (TCPIP0::169.254.160.241:... (0x00000001), ":CALC1:DATA:SDAT?", 17, 17)	0	06:39:02.203
174	viRead (TCPIP0::169.254.160.241:... (0x00000001), "#9000000599-1.007552E...", 16384, 611)	0	06:39:02.204
175	viWrite (TCPIP0::169.254.160.241:... (0x00000001), "*CLS", 4, 4)	0	06:39:02.206
176	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.207
177	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.218
178	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.228
179	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.238
180	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.248
181	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.258
182	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 128)	0	06:39:02.268
183	viWrite (TCPIP0::169.254.160.241:... (0x00000001), "*CLS", 4, 4)	0	06:39:02.268
184	viWrite (TCPIP0::169.254.160.241:... (0x00000001), ":CALC1:DATA:SDAT?", 17, 17)	0	06:39:02.269
185	viRead (TCPIP0::169.254.160.241:... (0x00000001), "#9000000599-1.008103E...", 16384, 611)	0	06:39:02.270
186	viWrite (TCPIP0::169.254.160.241:... (0x00000001), "*CLS", 4, 4)	0	06:39:02.272
187	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.272
188	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.284
189	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.294
190	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.304
191	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 0)	0	06:39:02.314
192	viReadSTB (TCPIP0::169.254.160.241:... (0x00000001), 128)	0	06:39:02.325

Captured 205 Displayed 205

At lines 150 to 160 we see the while loop which reads the status byte, gets a zero response, then waits 10ms. Finally at line 161 we receive a non-zero result of decimal 128 (xF0) which corresponds to a value of 1 at the b7 bit (note that the bits start at b0 so b7 is the MSB of the 8 bit register). We immediately clear the status byte with a *CLS, and then retrieve the active trace data with:

```
mbSession.Write(":CALC1:DATA:SDAT?");
responseString = mbSession.ReadString();
```

Within a few milliseconds of transferring the data we're again monitoring for the Sweep Complete bit at line 166 and again b7 is set at line 171. Here the sweep time is about 56 ms (the difference in time between line 171 and 161.

5. More information on VISA driver

There is a lengthy discussion of the usage of the VISA driver with C# in Appendix C of the VectorStar Programming Manual. The VISA driver is the preferred way to monitor for Sweep Complete but the LabWindows/CVI driver should also work well.

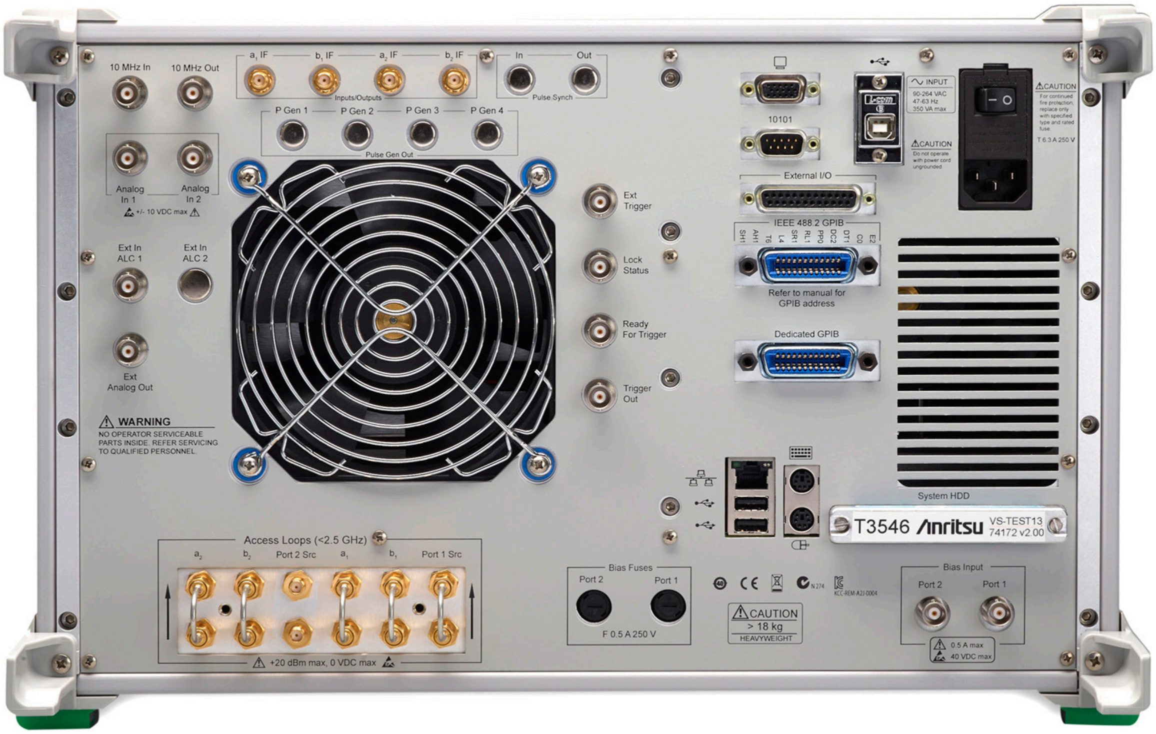
Here is how the session to the VNA is initiated using VISA:

```
//VXI-11 Connection string
string sAddress = "TCPIP0::192.168.1.6::INSTR"; //local network
//string sAddress = "TCPIP0::169.254.160.241::INSTR"; //auto-ip address
//string sAddress = "TCPIP0::127.0.0.1::INSTR"; //local address

//The VNA uses a message based session
 MessageBasedSession mbSession = null;
//But we'll just open a generic Session first
Session mySession = null;
try
{
     //open a Session to the VNA
     mySession = ResourceManager.GetLocalManager().Open(sAddress);
     //cast this to a message based session
     mbSession = (MessageBasedSession)mySession;

     OpenSession(mbSession);

Etc…
  }
```

6. More on Point Triggering

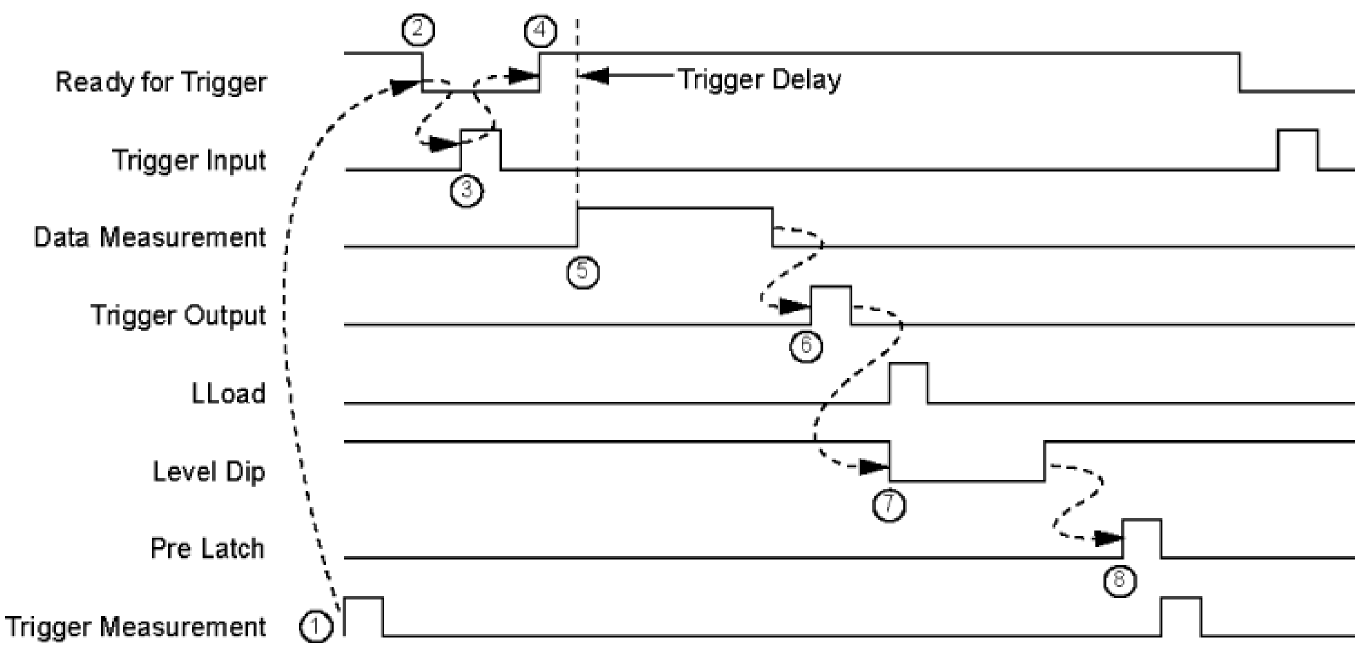
There are many measurement scenarios, but one common example is to use a Point Trigger so that each VNA measurement is synchronized with an external control signal. We don't want to transfer data after each measurement so we'll monitor the Sweep Complete bit. The set of measurements – for example the complete sweep for each S-parameter -- is to be transferred to the remote program at the completion of each sweep or set of sweeps. We need to know when each measurement has completed (each Point) and also when the each sweep is complete so that the measured data can be made available to the external program.

The three basic VNA synchronization hardware signals -- Ready for Trigger, External Trigger In, and Trigger Out – are all accessed via BNC connectors found on the rear panel of the VNA. It's also possible to use Ext Analog Out to monitor VNA events (see Utilities, System, Rear Panel Output, Edit Output Mode for more information).



7. Trigger Timing for per-point triggering

The timing diagram below illustrates the general measurement sequence of per-point triggering using a positive trigger edge. Trigger Delay, Ready for Trigger, and Trigger Output are only available with an External Trigger Input.



The trigger sequence with trigger handshake turned on is as follows (Trigger Handshake enables “Ready for Trigger” and “Trigger Output”:

1. A trigger measurement is received from the State machine.
2. Ready for Trigger (rear panel BNC output) is set to low to indicate that the instrument is ready for trigger.
3. External trigger (rear panel BNC input) is received.
4. After the external trigger is received, the Ready for Trigger (rear panel BNC output) is set to high to indicate that the system is not ready for trigger and the trigger delay is added.
5. The data measurement is started.
6. At the completion of the data measurement, the Trigger Output (rear panel BNC output) is pulsed to indicate that the measurement is completed.
7. The Level Dip and/or the LLoad pulse are executed.
8. The next frequency is preloaded and the State machine is triggered.

An efficient configuration to maximize the VNA measurement speed in a synchronized environment would be to monitor the Ready for Trigger BNC connector. When a falling edge is detected, we then should send the External Trigger.

Specifications for BNC Ports

Ext Trigger	BNC (female) 0-3.3 V input (5 V tolerant) High impedance (> 100 kΩ) 100 ns minimum input pulse width
Ready for Trigger	BNC (female) 0-3.3 V latched output Low impedance (~ 50 Ω) Voh = 2 V min @ -12 mA Vol = 0.8 V max @ +12 mA
Trigger Out	BNC (female) 0-3.3 V pulse output 1 usec positive pulse Low impedance (~ 50 Ω) Voh = 2 V min @ -12 mA Vol = 0.8 V max @ +12 mA
Ext Analog Out	BNC (female), for external attenuator control, external switch control, analog triggering assistance, measurement system integration and other purposes. Normal operating modes: Sawtooth sync sweep, TTL indication of driving port, open loop level controller.
	Range: -10 V to +10 V; low impedance drive Accuracy: 20 mV + 2% Load: > 5 kΩ

Anritsu envision : ensure

• United States
Anritsu Company
1155 East Collins Boulevard, Suite 100,
Richardson, TX, 75081 U.S.A.
Toll Free: 1-800-267-4878
Phone: +1-972-644-1777
Fax: +1-972-671-1877

• Canada
Anritsu Electronics Ltd.
700 Silver Seven Road, Suite 120,
Kanata, Ontario K2V 1C3, Canada
Phone: +1-613-591-2003
Fax: +1-613-591-1006

• Brazil
Anritsu Electrônica Ltda.
Praça Amadeu Amaral, 27 - 1 Andar
01327-010 - Bela Vista - São Paulo - SP - Brazil
Phone: +55-11-3283-2511
Fax: +55-11-3288-6940

• Mexico
Anritsu Company, S.A. de C.V.
Av. Ejército Nacional No. 579 Piso 9, Col. Granada
11520 México, D.F., México
Phone: +52-55-1101-2370
Fax: +52-55-5254-3147

• United Kingdom
Anritsu EMEA Ltd.
200 Capability Green, Luton, Bedfordshire LU1 3LU, U.K.
Phone: +44-1582-433280
Fax: +44-1582-731303

• France
Anritsu S.A.
12 avenue du Québec, Batiment Iris 1-Silic 612,
91140 Villebon-sur-Yvette, France
Phone: +33-1-60-92-15-50
Fax: +33-1-64-46-10-65

• Germany
Anritsu GmbH
Nemetschek Haus, Konrad-Zuse-Platz 1
81829 München, Germany
Phone: +49-89-442308-0
Fax: +49-89-442308-55

• Italy
Anritsu S.r.l.
Via Elio Vittorini 129, 00144 Roma Italy
Phone: +39-06-509-9711
Fax: +39-06-502-2425

• Sweden
Anritsu AB
Kistagången 20B, 164 40 KISTA, Sweden
Phone: +46-8-534-707-00
Fax: +46-8-534-707-30

• Finland
Anritsu AB
Teknobulevardi 3-5, FI-01530 Vantaa, Finland
Phone: +358-20-741-8100
Fax: +358-20-741-8111

• Denmark
Anritsu A/S
Kay Fiskers Plads 9, 2300 Copenhagen S, Denmark
Phone: +45-7211-2200
Fax: +45-7211-2210

• Russia
Anritsu EMEA Ltd.
Representation Office in Russia
Tverskaya str. 16/2, bld. 1, 7th floor.
Russia, 125009, Moscow
Phone: +7-495-363-1694
Fax: +7-495-935-8962

• United Arab Emirates
Anritsu EMEA Ltd.
Dubai Liaison Office
P O Box 500413 - Dubai Internet City
Al Thuraya Building, Tower 1, Suite 701, 7th Floor
Dubai, United Arab Emirates
Phone: +971-4-3670352
Fax: +971-4-3688460

• Singapore
Anritsu Pte. Ltd.
11 Chang Charn Road, #04-01, Shriro House
Singapore 159640
Phone: +65-6282-2400
Fax: +65-6282-2533

• India
Anritsu India Pvt Ltd
2nd & 3rd Floor, #837/1, Binnamangla 1st Stage,
Indiranagar, 100ft Road, Bangalore - 560038, India
Phone: +91-80-4058-1300
Fax: +91-80-4058-1301

• P. R. China (Shanghai)
Anritsu (China) Co., Ltd.
27th Floor, Tower A,
New Caohejing International Business Center
No. 391 Gui Ping Road Shanghai, Xu Hui Di District,
Shanghai 200233, P.R. China
Phone: +86-21-6237-0898
Fax: +86-21-6237-0899

• P. R. China (Hong Kong)
Anritsu Company Ltd.
Unit 1006-7, 10/F., Greenfield Tower, Concordia Plaza,
No. 1 Science Museum Road, Tsim Sha Tsui East,
Kowloon, Hong Kong, P. R. China
Phone: +852-2301-4980
Fax: +852-2301-3545

• Japan
Anritsu Corporation
8-5, Tamura-cho, Atsugi-shi,
Kanagawa, 243-0016 Japan
Phone: +81-46-296-1221
Fax: +81-46-296-1238

• Korea
Anritsu Corporation, Ltd.
5FL, 235 Pangyoyeok-ro, Bundang-gu, Seongnam-si,
Gyeonggi-do, 463-400 Korea
Phone: +82-31-696-7750
Fax: +82-31-696-7751

• Australia
Anritsu Pty Ltd.
Unit 21/270 Ferntree Gully Road,
Notting Hill, Victoria 3168, Australia
Phone: +61-3-9558-8177
Fax: +61-3-9558-8255

• Taiwan
Anritsu Company Inc.
7F, No. 316, Sec. 1, Neihu Rd., Taipei 114, Taiwan
Phone: +886-2-8751-1816
Fax: +886-2-8751-1817

Application Note No. 11410-00836, Rev. A Printed in United States 2014-12