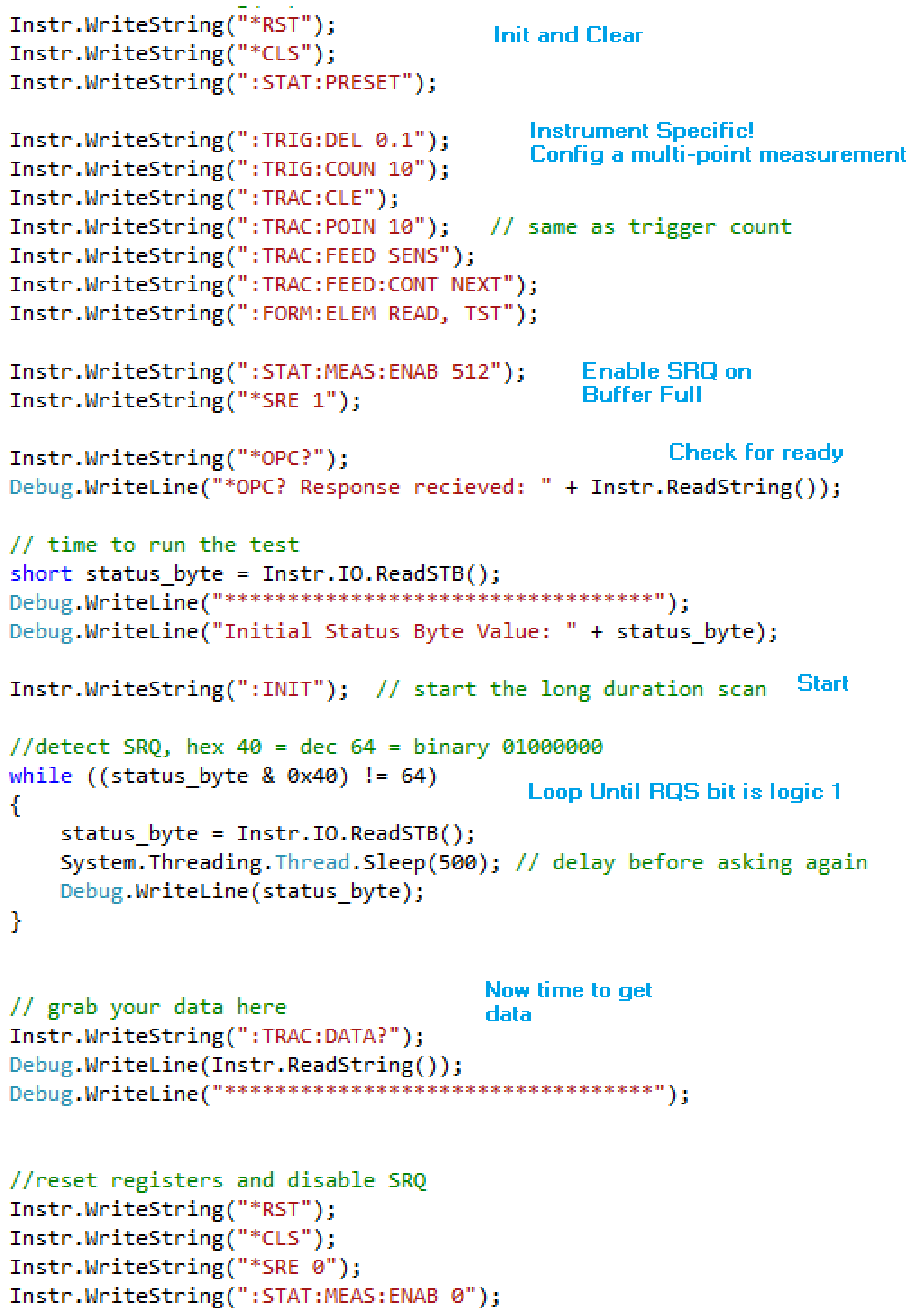
# Using Status System to know when Measurements are Done

Suppose you have a long running measurement task:

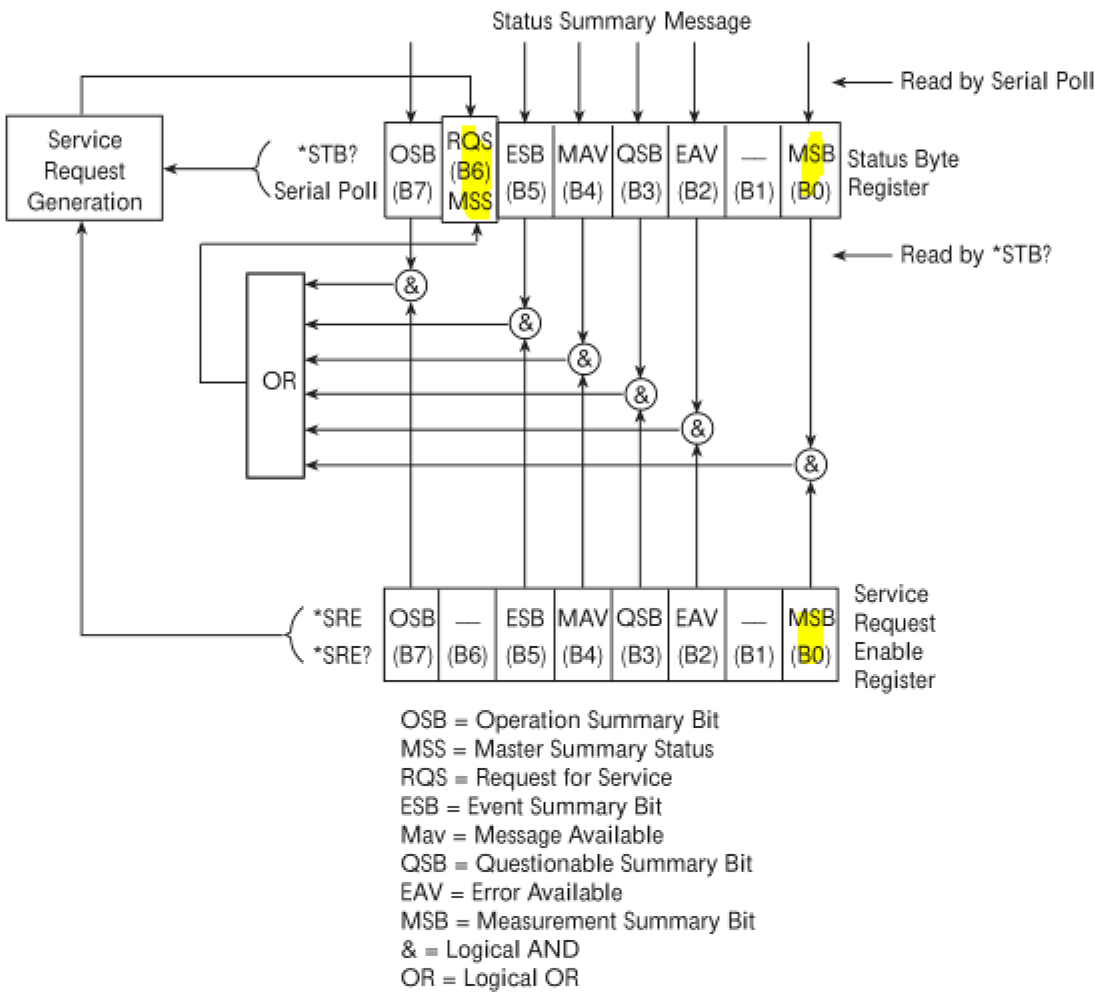
In this case, a model 2700 is used to obtain 10 readings with a 100msec delay between each reading.



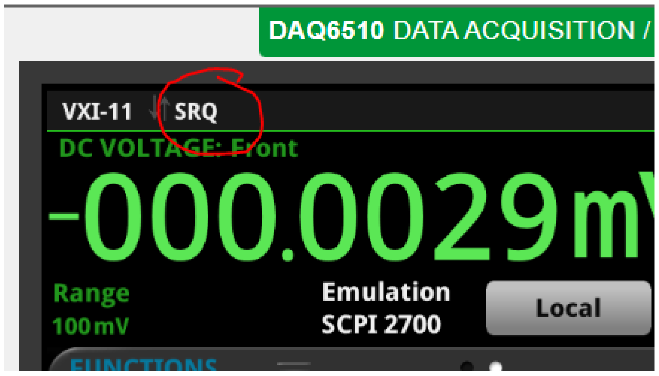
```
Instr.WriteString("*RST");
Instr.WriteString("*CLS");
Instr.WriteString(":STAT:PRESET");

Instr.WriteString(":TRIG:DEL 0.1");
Instr.WriteString(":TRIG:COUN 10");
Instr.WriteString(":TRAC:CLE");
Instr.WriteString(":TRAC:POIN 10");   // same as trigger count
Instr.WriteString(":TRAC:FEED SENS");
Instr.WriteString(":TRAC:FEED:CONT NEXT");
Instr.WriteString(":FORM:ELEM READ, TST");

Instr.WriteString(":STAT:MEAS:ENAB 512");
Instr.WriteString("*SRE 1");

Instr.WriteString("*OPC?");
Debug.WriteLine("*OPC? Response recieved: " + Instr.ReadString());

// time to run the test
short status_byte = Instr.IO.ReadSTB();
Debug.WriteLine("***********************************");
Debug.WriteLine("Initial Status Byte Value: " + status_byte);

Instr.WriteString(":INIT");  // start the long duration scan

//detect SRQ, hex 40 = dec 64 = binary 01000000
while ((status_byte & 0x40) != 64)
{
    status_byte = Instr.IO.ReadSTB();
    System.Threading.Thread.Sleep(500); // delay before asking again
    Debug.WriteLine(status_byte);
}


// grab your data here
Instr.WriteString(":TRAC:DATA?");
Debug.WriteLine(Instr.ReadString());
Debug.WriteLine("***********************************");


//reset registers and disable SRQ
Instr.WriteString("*RST");
Instr.WriteString("*CLS");
Instr.WriteString("*SRE 0");
Instr.WriteString(":STAT:MEAS:ENAB 0");
```

The ":STAT:MEAS:ENAB 512" and "*SRE 1" commands are configuring the instrument to give us an SRQ indication when the buffer is full.

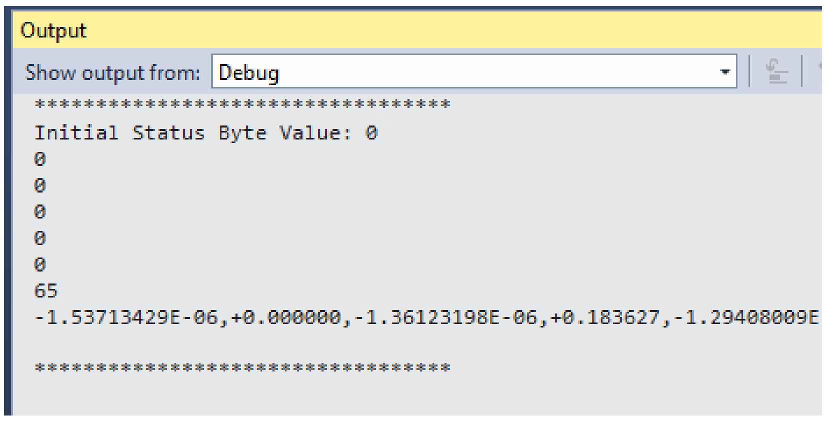
The while loop on ReadSTB will repeat until the RQS bit goes to logic 1.  RQS is bit 6 which has value of 64.



When this occurs, the front panel of instrument will also show the SRQ annunciator:



Here is the debug print output from this program:

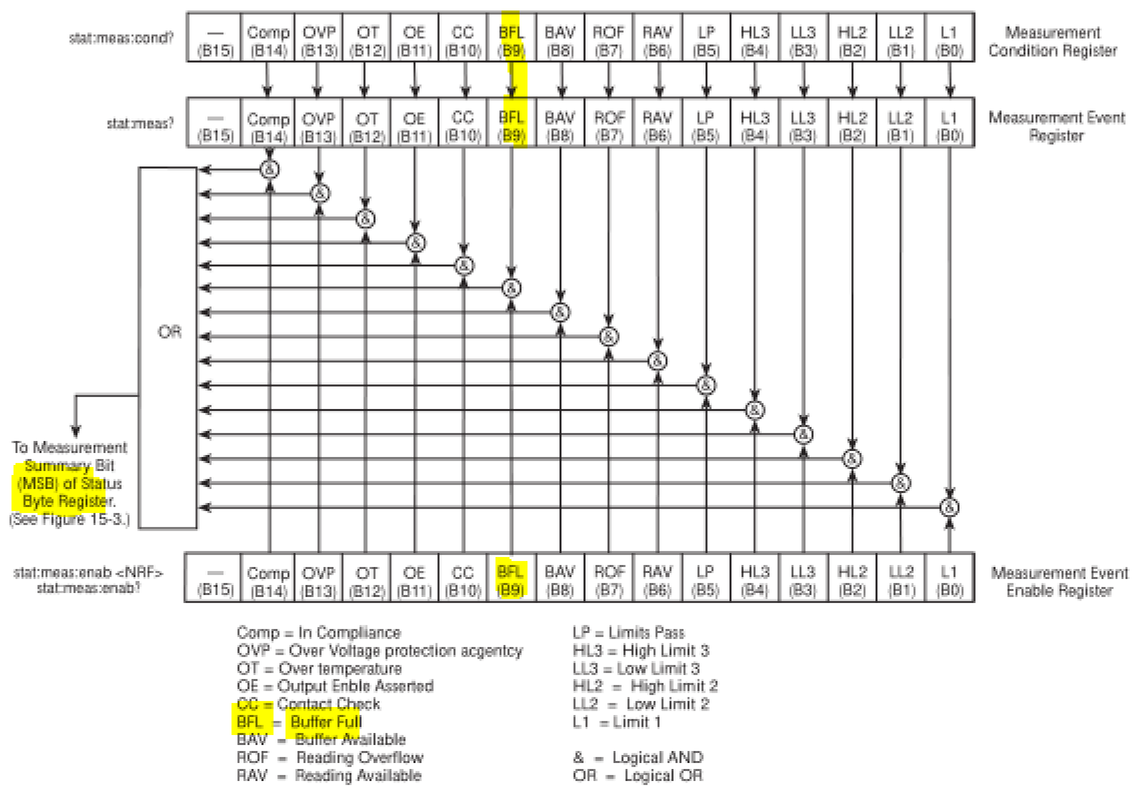


```
**********************************
Initial Status Byte Value: 0
0
0
0
0
0
65
-1.53713429E-06,+0.000000,-1.36123198E-06,+0.183627,-1.29408009E

**********************************
```

The last status byte value is 65 = $2^6 + 2^0$

The BFL (buffer full) event was enabled so that it signals to the MSB bit in the higher-level status byte register.

**Measurement event status**



NOTE: multiple conditions can be configured to cause SRQ assertion. Status polling on value 64 tells you that SRQ did occur. Values of other (enabled) bits tell you which enabled condition caused the SRQ.