MICROSTAR LABORATORIES™

2265 116th Avenue N.E., Bellevue, WA 98004
Sales & Customer Support: (425) 453-2345
Finance & Administration: (425) 453-9489
Fax: (425) 453-3199
World Wide Web: http://www.mstarlabs.com/

Technical Note TN-243 **Version 1.10**

Timing channels on the iDSC 1816

The iDSC 1816 has two timing channels, External Timing Channel 0 and External Timing Channel 1, which can be utilized for triggering and correlating the clock signal from an external device to the sample clock onboard. They are TTL pulses, which are rising-edge triggered and the falling edges are not detected. They record the number of clock ticks of the clock onboard of the iDSC, or counts, elapsed between consecutive rising edges. The internal clock of the iDSC runs at 19.66 MHz, whereas the analog-to-digital converters run at half the clock speed (i.e. 9.8 MHz) because that is about the fastest they can run. The two timing channels run at the full clock rate of 19.6 MHz because they do not need to be digitized by the analog-to-digital converters. One count of the timing channel is equivalent to the period of the clock onboard, which is approximately 50.8 ns.

With no rising edge detected the value of the timing channel is at its maximum, which is equivalent to the ratio between the input clock and the sample rate specified by the user. This maximum value is called `TcMaximum` and it can be determined by invoking the function `DscTcMaximum` from the DSCIO Dynamically Linked Library (i.e. DLL). When a rising edge is detected, the timing channel reports a value that indicates the time at which the event occurs in terms of counts. This value is always less than `TcMaximum` for the specific sample rate. The table below shows the values of `TcMaximum` for some of the valid sample rates.

Sample Rate (kS/s)	`TcMaximum`
12.8	1536
76.8	256
153.6	128

You can see that `TcMaximum` is inversely proportional to the sample rate. Their relationship can be expressed in the following equation.

`TcMaximum` = (19.66 MHz) / sample rate

The valid sample rates on the iDSC range from 8 to 153,600 Hz, which leads to a range between 2,457,600 and 128 for `TcMaximum`.

The timing channels need to be enabled in software before they can be used. The `DscTcEnabledSet` function sets the enabled timing channels on the iDSC board. One or both of the timing channels can be enabled with one call to this function, with the option to enable `TcWidth32` to force the timing channel width to be 4 bytes (32-bits) regardless of the sample rate. If `TcWidth32` is not enabled, the timing channel width can be either 2 bytes or 4 bytes, as it ranges from 0 to 2,457,600 depending on the sample rate. The actual timing channel width can be determine by using the `DscTcWidth` function.

Once the timing channels are enabled, they are interleaved in the data stream of the iDSC with the eight inputs on the board. The `DscBufferGet` or `DscBufferGetEx` function is used to read the data stream in the PC program. If one timing channel is enabled, the stream contains one word for each of the active inputs, followed by one word or long value for the timing channel. In the case where the width of the timing channel is 4-byte, `DscBufferGetEx` returns the lower 2-byte and then the higher 2-byte.

The relative time of the rising edges in the timing channel can be determined by calculating the number of counts elapsed between the time at which the iDSC starts (time = 0 s) and the arrival of the event. The relative time of the nth sample in the timing channel can be calculated by the following equation:

$$\textit{relative time} = [((n\text{-}1) \times \textit{TcMaximum}) + (\textit{count at nth sample})] \times (\textit{50 ns/count})$$

For example, the iDSC is sampling at 12.8 kHz per channel and also reading one of its timing channels. The value of `TcMaximum` for this sample rate is 1536, as indicated in the Table above. The first five samples from the active timing channel read the following:

1460 1536 1536 1536 **193**

The data stream indicates that rising edges were detected at the 1st and 5th sample and their relative times are 74.2 and 321.9 ms from the time at which the iDSC was started.

Unlike the eight analog inputs, the timing channels do not have group delay because they do not go through the low-pass filters. It leads to a skew in the relationship between the sample count in the timing channel and the data value in any of the input channels; the data value that corresponds to each sample count in the timing channel comes in at one group delay after the sample count. So in order to match up the data value and the sample count, the group delay needs to be added to the same count after it is converted to the time domain. The group delay can be determined by the `DscGroupDelay` function.

The timing channels can be connected to the terminals on the iDSC termination board (i.e. MSXB042) or directly to pin 12 and 13 of the input/output connector of the iDSC board. The two timing channels are independent of each other; they can be used for two different inputs at the same time. Refer to the pinout of the connector in the iDSC Reference Manual.

The following example was implemented using Visual C++ 6.0. It reads from External Timing Channel 0 and reports the relative time of a rising edge if it is detected.

```
#include <stdio.h>
#include <stdlib.h>
#include <windows.h>
#include "Dscio.h"

#define MAXSIZE 8192

const char DSC_ADDRESS[] = "\\\\PC83\\Dap1";

void main()
{
  HDSC hDsc;
  TBufferGetEx bge;
  long TcMax, Tc, CountAccum, SampleRate;
  double TimeEvent;
  short Data[MAXSIZE], i, j, BlockPerCh, blocksize;

  hDsc = DscHandleOpen(DSC_ADDRESS);  // Open handle to iDSC.

  // Enable Tc0 and force the width of the channel to be 4-byte
  if ( !DscTcEnabledSet(hDsc, DscTc_Tc0|DscTc_TcWidth32) )
       printf("Error enabling Timing Channel A.\n");

  //DscSampleRateSet(hDsc, 800);        // Set sample rate to 800 S/s.
  DscFilterDialogShow(hDsc);          // Design filters and configure system.

  // Determine TcMaximum
  TcMax = DscTcMaximum(hDsc);

  // Read 5 values per channel. One timing channel equals to two word values.
  BlockPerCh = 5;
  blocksize = (DscPinEnabledCount(hDsc) + 2 ) * BlockPerCh;

  DscStructPrepare(&bge, sizeof(bge));
  bge.iMinBytes = blocksize*2;
  bge.iMaxBytes = blocksize*2;
  bge.iTimeWait = 1000;
```

```
  bge.iTimeOut = 1000;

  // Number of counts that should have been accumulated from the timing channel
  CountAccum = 0;

  SampleRate = DscSampleRateGet(hDsc);

  if (DscStartAcquiring(hDsc))
  {

       // Read a data block from the iDSC. The pattern of the stream is one short integer for
       // each active input, followed by two short integers making up the long timing  //
channel.

       DscBufferGetEx(hDsc, &bge, Data);

       i = 0;
       while(i<blocksize)
       {
              // Read and display the active inputs
              for(j=0; j<DscPinEnabledCount(hDsc); j++)
              {
                     printf("%d\t", Data[i]);
                     i++;
              }

              // Read and display the timing channel. Shift the second word value, which
              // represents the higher order of the timing channel, to the left by 16 bits.
              // Then add it to the lower order.
              Tc = ((unsigned short)Data[i]) | (Data[i+1] << 16);

              printf("Tc = %d\n", Tc);
              i = i + 2;

              // An event is detected if Tc is less than TcMax. One count = 50.8 ns
              if ( Tc < TcMax )
              {
                     TimeEvent = (double)(CountAccum + Tc) * 0.0000000508;
                     printf("An event has been detected at %f second.\n", TimeEvent);
              }

              // Update the amount of time elapsed.
              CountAccum = CountAccum + TcMax;
       }

       DscHandleClose(hDsc);               // Close handle to iDSC.
  }
}
```