USB-CTR04 USB-CTR08 Measuring Frequency using CIn32

The following example demonstrates how to read frequency using a counter input and the MCCDAQ dot net API.

To access the API, add a reference to the MccDaq object. Adding the reference is usually accomplished by right clicking the Project [under the Project Explorer] and selecting Add Reference. The complete project is attached to the article and was created with Visual Studio 2008 using CSharp. Compiling and execution is quick as there is minimal Windows overhead and associated code.

To make the code more readable a few convenience functions were added such as IsError, GetBoardNum, and WaitForKey. The IsError function checks the error number in the ErrorInfo object and if not zero displays the error message. The GetBoardNum function begins reading the Device strings from each number location and as soon as it finds one that contains the identifying string it exits. The WaitForKey does just that - waits for someone to press a key. To view the complete program project download the attachment below.

Disclaimer:
The attached Code or Example is provided As Is. It has not been tested or validated as a product, for use in a deployed application or system, or for use in hazardous environments. You assume all risks for use of the Code or Example.

```
using System;
using MccDaq; // also, add reference to MccDaq to the project

namespace PeriodMode
{
   class Program
   {
      public const int                CHANCOUNT = 1;
      public const string              DEVICE = "USB-CTR"; //for USB-CTR04 or USB-CTR08
      public const int                CounterNum = 0;     //used to measure the signal
generator
      public const int                OutTimerNum = 0;    //used as signal generator
      public const CounterMode           Mode = CounterMode.Period;
      public const CounterDebounceTime    DebounceTime =
CounterDebounceTime.Debounce500ns;
      public const CounterDebounceMode    DebounceMode =
CounterDebounceMode.TriggerAfterStable;
      public const CounterEdgeDetection   EdgeDetection =
CounterEdgeDetection.RisingEdge;
      public const CounterTickSize        TickSize = CounterTickSize.Tick20pt83ns;    // 0
```

```
equates to 20.833 nS ticksize
        public const double              tickTime = 0.000000020833f;
        public const int                 MapCounter = 0;

        /*////////////////////////////////////////////////////////////////////////////////////*/

        static void Main(string[] args)
        {
            int BoardNum    = 0;
            double freq     = 1000.0f;
            double duty     = 0.50f;
            double intDelay = 0;

            Console.WriteLine("Locating Device...Please wait\n");

            BoardNum = GetBoardNum(DEVICE);

            if (BoardNum == -1)
            {
                Console.WriteLine("No {0} detected!", DEVICE);
                WaitForKey();
                return;
            }
            else
            {
                MccBoard daq = new MccDaq.MccBoard(BoardNum);

                IsError(daq.CConfigScan(CounterNum,
                                        Mode,
                                        DebounceTime,
                                        DebounceMode,
                                        EdgeDetection,
                                        TickSize,
                                        MapCounter));

                //for test purposes,jumper a wire from timer zero over to counter in zero.
                //use timer as test signal

                IsError(daq.PulseOutStart(OutTimerNum,
                                          ref freq,
                                          ref duty, 0,
                                          ref intDelay,
                                          IdleState.Low,
                                          PulseOutOptions.Default));
```

```
            System.ConsoleKeyInfo cki = new System.ConsoleKeyInfo();

            Console.WriteLine("Retrieving Data\n");
            Console.WriteLine("Press any key to Exit...\n");

            int     tickCounts;
            double  temp = 0;

            do
            {
               IsError(daq.CIn32(CounterNum, out tickCounts));

               temp = 1 / (tickCounts * tickTime);

               Console.Write("{0}Hz\r\n", temp.ToString("0000.00"));

               System.Threading.Thread.Sleep(100);

            } while (!Console.KeyAvailable);

            cki = Console.ReadKey();

            IsError(daq.PulseOutStop(OutTimerNum));

            WaitForKey();
         }

      }
      /*////////////////////////////////////////////////////////////////////////////////*/

      public static int GetBoardNum(string dev)
      {
         for (int BoardNum = 0; BoardNum < 99; BoardNum++)
         {
            MccDaq.MccBoard daq = new MccDaq.MccBoard(BoardNum);
            if (daq.BoardName.Contains(dev))
            {
               Console.WriteLine("{0} board number = {1}\n", daq.BoardName,
BoardNum.ToString());
               return BoardNum;
            }
         }
         return -1;
      }
      /*////////////////////////////////////////////////////////////////////////////////*/
```

```
        public static void WaitForKey()
        {
            Console.WriteLine("\nPress any key to continue...");
            do
            {   //idle loop
                System.Threading.Thread.Sleep(10);
            } while (!Console.KeyAvailable);
        }

        /*//////////////////////////////////////////////////////////////////////////////////*/

        public static int IsError(ErrorInfo e)
        {
            if (e.Value != 0)
            {
                Console.WriteLine(e.Message);
                WaitForKey();
                return 1;
            }
            return 0;
        }

        /*//////////////////////////////////////////////////////////////////////////////////*/

    }
}
```

Measurement Computing Data Acquisition Knowledgebase
https://kb.mccdaq.com/KnowledgebaseArticle50576.aspx