```
/*
* 23-Feb-2010
* Version 1.1
*
* www.tctec.net
*
* Top16 C# example
*
* Console applicaion.
* Uses top16.dll and FTD2XX.dll, these dlls must be in the execution directory (release and debug).
* Alternatively, they can be located in the system32 folder
*
*
*23-Feb-2011 Version 1.1    Better use of ListTop16Boards function.
*
*/
using System;
using System.Collections.Generic;
using System.Collections;
using System.Text;
using System.Runtime.InteropServices;
// Import functions from Top16 DLL
//
class top16
{
    [DllImport("top16.dll")]
    public static unsafe extern Int32 dllversion();
    [DllImport("top16.dll")]
    public static extern Int32 ListTop16Boards(byte[] sBoardNames);
    [DllImport("top16.dll")]
    public static unsafe extern Int32 SetOutputs(UInt32 Handle, Byte mask, Byte set);
    [DllImport("top16.dll")]
    public static unsafe extern Int32 GetInputs(UInt32 Handle);
    [DllImport("top16.dll")]
    public static unsafe extern Int32 readAnalogInput(UInt32 Handle, Int16 input, byte gain);
    [DllImport("top16.dll")]
    public static unsafe extern UInt32 OpenBoard(StringBuilder boardName);
    [DllImport("top16.dll")]
    public static unsafe extern Int32 CloseBoard(UInt32 Handle);
}
namespace ConsoleApplication1
{
    class Program
    {
        static void Main(string[] args)
        {
            try
```

```
{
    const int MAX_NO_BOARDS = 124;
    const int MAX_NAME_LEN = 20;

    // print the Dll version
    //
    Console.WriteLine("DLL version = {0}\n", top16.dllversion());

    // Read the number of boards and names
    //
    byte[] byBoardNames = new byte[MAX_NO_BOARDS * MAX_NAME_LEN];
    Int32 boardsFound = top16.ListTop16Boards(byBoardNames);
    String sListOfBoards = Encoding.ASCII.GetString(byBoardNames, 0, MAX_NO_BOARDS * MAX_NAME_LEN);

    // Make a list of all the board names
    //
    ArrayList sBoardNames = new ArrayList();

    for (int i = 0; i < boardsFound; i++)
    {
        // List the names of all Top16 boards found, and add them to an ArrayList
        //
        sBoardNames.Add(sListOfBoards.Substring(i * MAX_NAME_LEN, MAX_NAME_LEN));

        Console.WriteLine("Found : {0}", sBoardNames[i]);

    }

    // loop forever
    //
    while (true)
    {

        // for each board, open it, set digital outputs, read some analog inputs
        // ( no exception handling, or return value checking (!))
        //

        for (int i = 0; i < sBoardNames.Count; i++)
        {

            UInt32 handle = top16.OpenBoard(new StringBuilder((String)sBoardNames[i]));

            if (handle > 0)
            {
                // dummy read to get current state of outputs and increment
                //
                byte byOutput = (byte)top16.SetOutputs(handle, 0, 0);
```

```
                    byOutput += 1;

                    Console.WriteLine("------------\nSetting output to {0}", byOutput);

                    // turn off unset bits
                    //
                    top16.SetOutputs(handle, (byte)(byOutput ^ 0xFF), (byte)0);


                    // turn ON set bits
                    //
                    top16.SetOutputs(handle, byOutput, (byte)1);

                    // read an analog input
                    //
                    int aInput = top16.readAnalogInput(handle, 1, (byte)'Z');

                    Console.Write("\nAnalog Input 1, Gain 1 = {0}\n", aInput);

                    // read another analog input at different gain
                    //
                    aInput = top16.readAnalogInput(handle, 2, (byte)'X');
                    Console.Write("\nAnalog Input 2, Gain 4 = {0}\n", aInput);

                    // close the board
                    //
                    top16.CloseBoard(handle);
                }
            }
            Console.Write("Press a key ('q' to exit) \n");

            ConsoleKeyInfo key = Console.ReadKey();

            if (key.KeyChar == 'q')
            {
                return;
            }
        }
    }
    catch (Exception ex)
    {

        // Display an error if caught

        Console.WriteLine("ERROR : {0}", ex.Message);
```

```
                Console.ReadLine();

            }
        }
    }

}
```