

```

using System;
using System.Collections.Generic;
using System.Text;
using System.Runtime.InteropServices;
using System.IO;

namespace WindowsApplication1
{
    public class rp2005interface
    {
        [DllImport("rp2005.dll")]
        public static extern UInt32 RP_ListDIO(ref UInt32 NumBrds, StringBuilder SN, StringBuilder Dsc);
        [DllImport("rp2005.dll")]
        public static extern UInt32 RP_OpenDIO(string SN, ref UInt32 hDIO);
        [DllImport("rp2005.dll")]
        public static extern UInt32 RP_CloseDIO(UInt32 hDIO);
        [DllImport("rp2005.dll")]
        public static extern UInt32 RP_GetVer(UInt32 hDIO, StringBuilder SwVer, StringBuilder ErrMsg);
        [DllImport("rp2005.dll")]
        public static extern UInt32 RP_ReadPort(UInt32 hDIO, byte ucPort, ref byte PVal, StringBuilder ErrMsg);
        [DllImport("rp2005.dll")]
        public static extern UInt32 RP_WritePort(UInt32 hDIO, byte ucPort, byte PVal, StringBuilder ErrMsg);
    }

    public class rp2005
    {
        public UInt32 numDevs;
        public string[] SN; // Serial number(s) of the I/O board(s).
        public string[] Desc; // I/O board description(s)
        public UInt32[] hDIO; // handles to the I/O board(s)
        public UInt32 listDevErrCode; // error code returned for listing the devices
        public UInt32[] errCode; // last error code received. One error code per device.
        public string[] errMsg; // last error message for the corresponding board.
        StringBuilder tmpErrMsg = new StringBuilder(512);

        //
        // constructor
        public rp2005()
        {
            // need to create temporary buffers to hold return string values for
            // board names and serial numbers.
            StringBuilder sn = new StringBuilder(512);
            StringBuilder desc = new StringBuilder(512);

            try
            {
                numDevs = 0;
                listDevErrCode = rp2005interface.RP_ListDIO(ref numDevs, sn, desc);
            }
            catch
            {
                if (!System.IO.File.Exists(Environment.SystemDirectory + "\\rp2005.dll") &&
                    !System.IO.File.Exists(Directory.GetCurrentDirectory() + "\\rp2005.dll"))
                {
                    listDevErrCode = 500; // rp2005 dll doesn't exist.
                }
                else
                {
                    listDevErrCode = 1000; // some unknown error occurred.
                }
            }
        }
    }
}

```

```

if (listDevErrCode == 0)
{
    SN = sn.ToString().Split(',');
    Desc = desc.ToString().Split(',');
    hDIO = new uint[numDevs];
    errCode = new uint[numDevs];
    errMsg = new string[numDevs];

    for (int i = 0; i < numDevs; i++)
    { // open the boards
        hDIO[i] = 0;
        try
        {
            errCode[i] = rp2005interface.RP_OpenDIO(SN[i], ref hDIO[i]);
        }
        catch
        {
            errCode[i] = 1000;
        }
    }
}

//
// ReadPort - read a port.
// Parameters:
// brd - 0 based board number to read.
// port - port number to read. 0 is input port 0, 1 is input port 1, 2 is input port 2.
//       port 16 is output port 0, 17 is output port 1, 18 is output port 2.
//       Note: reading outputs will not return actual state of the bit. It will return
//       the last output value.
// value - If the port read was successful, the "value" parameter will contain the
//         value read from the port.
//
// Return Value:
// Returns 0 if successful, otherwise it returns false. You can read the error string
// for the operation by accessing the errMsg array using the brd number as the index
// into the array.
public UInt32 ReadPort(int brd, byte port, ref byte value)
{
    if (brd < numDevs)
    {
        errCode[brd] = rp2005interface.RP_ReadPort(hDIO[brd], port, ref value, tmpErrMsg);
        if (errCode[brd] == 0)
            return (0);
        else
        {
            errMsg[brd] = tmpErrMsg.ToString();
            return errCode[brd];
        }
    }
    else
    {
        return 1001; // invalid number of boards.
    }
}

public UInt32 WritePort(int brd, byte port, byte value)
{
    if (brd < numDevs)

```

```

    {
        errCode[brd] = rp2005interface.RP_WritePort(hDIO[brd], port, value, tmpErrMsg);
        if (errCode[brd] == 0)
            return (0);
        else
        {
            errMsg[brd] = tmpErrMsg.ToString();
            return errCode[brd];
        }
    }
    else
    {
        return 1001; // invalid board number. Must be between 0 and the number of devices.
    }
}
//
// destructor
~rp2005()
{
    for (int i = 0; i < numDevs; i++)
    { // close the boards
        errCode[i] = rp2005interface.RP_CloseDIO(hDIO[i]);
    }
}
}
}

```