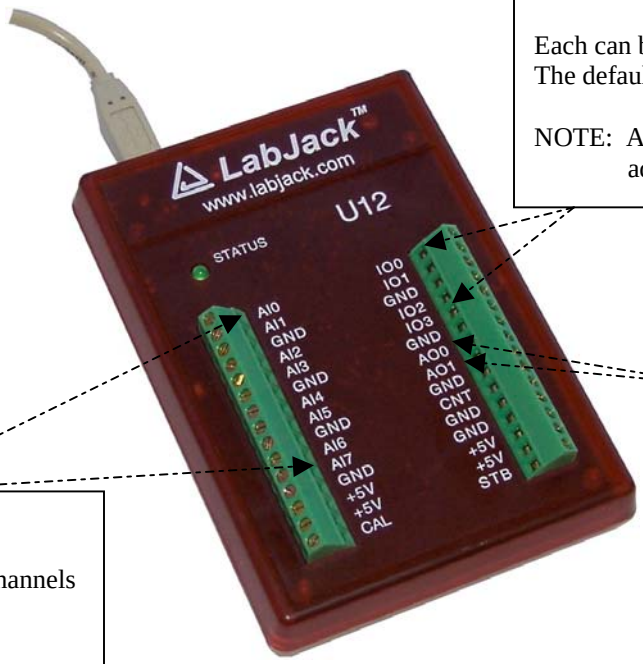
Using Visual Basic And The Labjack For PC to Outside World Interfacing

The terminal designations are indicated in the following diagram. A full description is given in the Labjack U12 users guide which can be accessed via blackboard or at the following internet address; www.labjack.com .



IO0 to IO3 are 4 digital input/output.

Each can be individually set to input or output, The default being input on power up or reset.

NOTE: An extra 16 I/Os (D0 - D15) can be added using a CB25 extension unit.

AO0 & AO1 are analogue outputs.

AI0 to AI7 8 analogue inputs

8 single ended OR 4 differential channels

OR any combination in between

CAL and STB are used during testing and calibration.

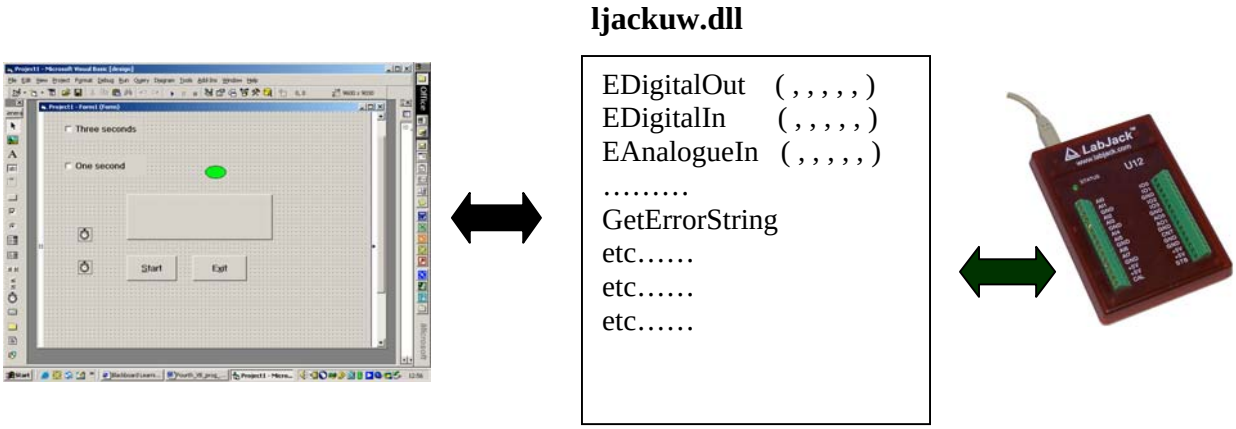
CNT is an input connection to a 32-bit counter.

GND and +5V

See user guide for complete description.

The Labjack cable provides both power from the PC and a communications link between PC and labjack. When the Labjack U12 is connected to the USB port on the PC the status led will blink four times and then stay off while the Labjack enumerates. Enumeration is the process where the PC's operating system gathers information from a USB device that describes it and it's capabilities. When connected to a PC for the very first time enumeration can take a few minutes, thereafter enumeration is relatively rapid. When enumeration is complete, the Labjack status LED will blink twice and remain on.

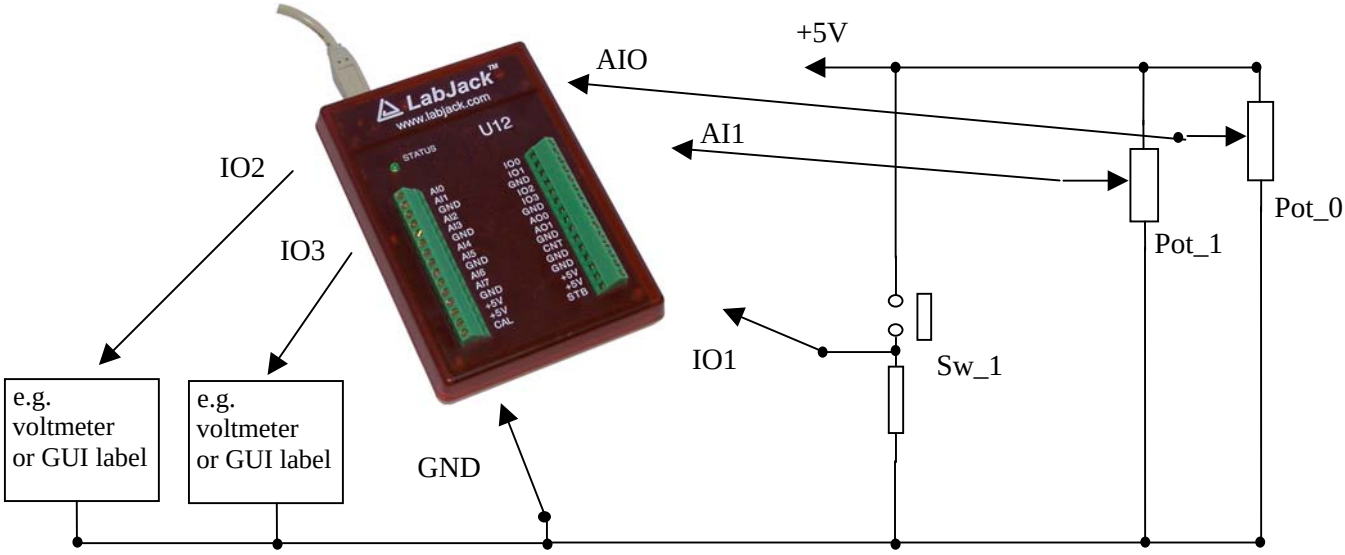
The Labjack installation disk includes the software necessary to enable the Visual Basic environment to interact with the Labjack hardware. The files necessary for this interaction, e.g. analogue I/O, single digital I/O , byte digital I/O, LJscope, LJcounter, etc are stored on the PC in a dynamic link library (DLL) called ***ljackuw.dll***



To introduce the use of Labjack we will use the three functions illustrated in the above diagram. The Labjack user's guide describes these as *easy* functions and they can be used to input and output (read/write) any 1-bit digital signal or to input (read) any single analogue input channel. Any of the available analogue and digital I/O can be written to or read with these easy functions.

The other function we may initially wish to employ is the GetErrorString function which results in the displaying of any error that occurs when accessing the labjack.

You are to employ the following GUI and program to familiarise yourself with an example use of Labjack. Take your time (up to 4 weeks) to do this exercise and make sure that you follow all aspects of this initial example project before you attempt the assignment for this module.



The programming reference can be found starting at page 25 of the Labjack U12 user's guide. For this exercise we have used the so called easy functions which are simplified versions of some of the Labjack functions. These easy functions have a six value parameter list so that data can be passed to and from calling procedures. A description of all available Labjack functions available in the ljackuw.dll library is given in the user's guide.

All external labjack functions to be used must be declared at the start of the program code in the general declarations section (use drop down menus). For this initial exercise we will only use four of the available labjack functions contained in ljackuw.dll, the details of which are detailed below.

The declarations for the external labjack functions are required to list those functions to be used and to inform VB that they are contained in the ljackuw.dll library installed on the PC hard drive from the labjack installation disk.

Should you need to employ any of the other available Labjack functions it is hoped that use of the user's guide and/or access to the internet (e.g. look for examples of VB used to control labjack) will enable you to do so.

Note : all parameters are declared as data type *long* which means that each contains 4 bytes of data within the range –2,147,483,648 to 2,147,483,648. Data type long also has the shortcut notation &.

Other data types include :

Integer, 2 bytes long, shortcut %, -32768 to 32,767.
String, depends on number of characters, shortcut $, can be up to 65,000 characters.

i. EAnalogIn

```
Private Declare Function EAnalogIn Lib "ljackuw.dll"_
  ( ByRef lpIDNum As Long, _
    ByVal lngDemo As Long, _
    ByVal alngChannel As Long, _
    ByVal alngGain As Long, _
    ByRef lngOverVoltage As Long, _
    ByRef asngVoltages As Single ) As Long
```

```
lpIDNum          ---     REM   Labjack ID number ( -1 if only one labjack )
lngDemo          ---     REM   0 for normal mode
alngChannel      ---     REM   channel number 0 to 7 ( single ended )
alngGain         ---     REM   0 gives Gain  =  1,  1 gives Gain  =  2
lngOverVoltage   ---     REM  Returns >0 if overvoltage on any analogue input
asngVoltages     ---     REM   Returns the input voltage magnitude
```

ii. EDigitalIn

```
Private Declare Function EDigitalIn Lib "ljackuw.dll" _
 ( ByRef lpIDNum As Long, _
   ByVal lngDemo As Long, _
   ByVal alngChannel As Long, _
   ByVal alngReadD As Long, _
   ByRef lngState As Long ) As Long
```

lpIDNum	---	Labjack ID number (-1 if only one labjack)
lngDemo	---	0 for normal mode
alngChannel	---	Channel to read IO 0 to 3 or D 0 to 15
alngReadD	---	If > 0 a D line is read instead of an IO line
lngState	---	If returns > 0, the line is set/true else line is clear/false

iii. EdigitalOut

```
Private Declare Function EDigitalOut Lib "ljackuw.dll" _
( ByRef lpIDNum As Long, _
  ByVal lngDemo As Long, _
  ByVal alngChannel As Long, _
  ByVal alngWriteD As Long, _
  ByVal lngState As Long ) As Long
```

lpIDNum	---	Labjack ID number (-1 if only one labjack)
lngDemo	---	0 for normal mode
alngChannel	---	Channel to read IO 0 to 3 or D 0 to 15
alngWriteD	---	If > 0 a D line is written to instead of an IO line.
lngState	---	If > 0, the line is set/true (5V) else line is clear/false (0V)

In the following code variables lngErrorcode, ChannelSelected and StateD are declared as local variables in the body of the appropriate procedures. Since they are local variables their names can be the same in all procedures but their values can be different in each procedure. Being able to use the same names but with different values saves having to think of variations of names describing equivalent functions.

In all procedures lngErrorcode is used to call the required function, EDigitalIn, EAnalogIn and EDigitalOut.

ChannelSelected is included in all parameter lists and the value 0/1/2/3 is appropriately set in each procedure and used to select channel number.

For EDigitalOut the code assigns StateD to 1 or 0 to set or clear the digital output.

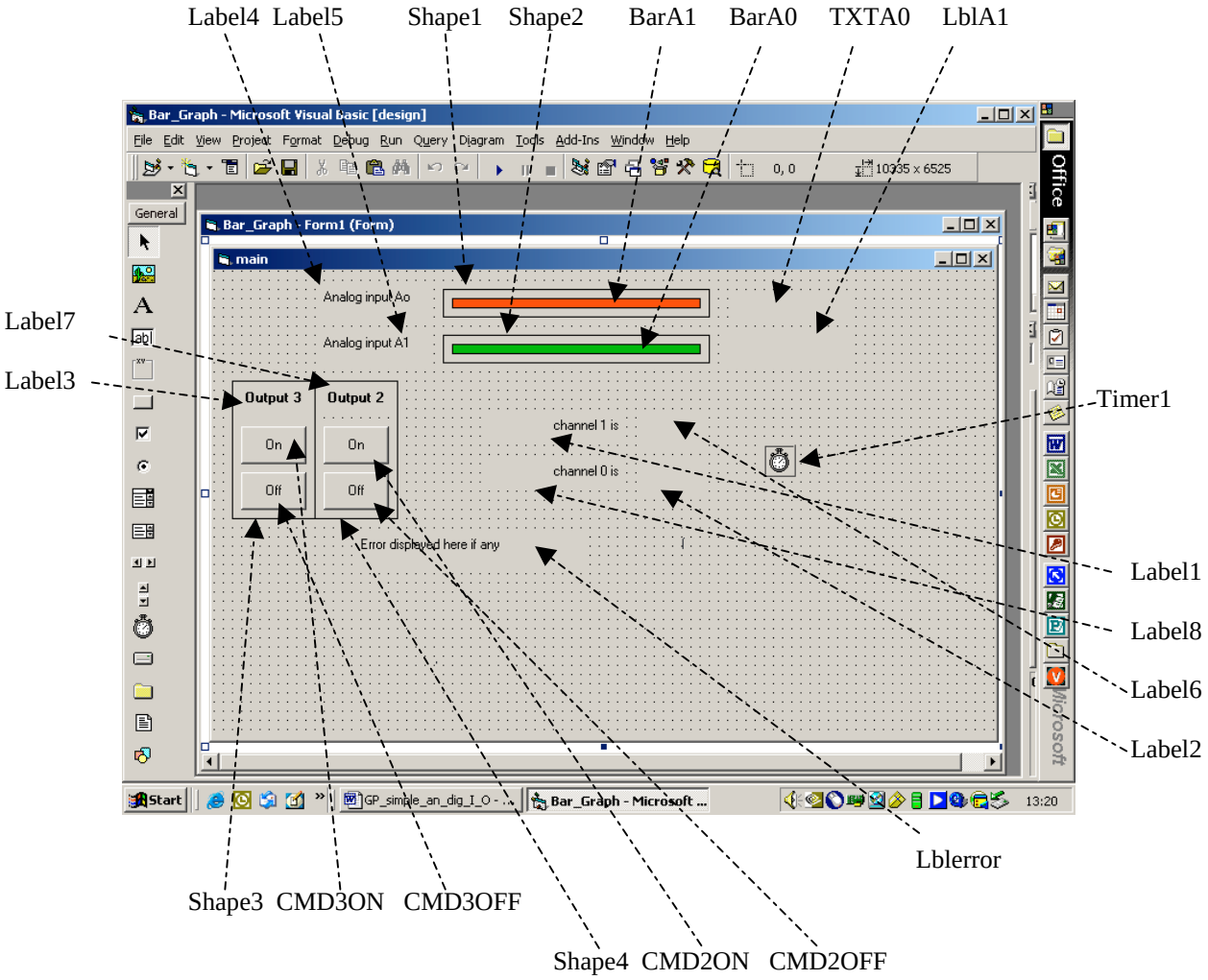
EDigitalIn sets StateD to 1 if there is a voltage present on the a digital input or to 0 if there is no voltage present on the digital input.

Test Hardware

You will be provided with the Labjack and a UWIC produced PCB containing 2 analogue inputs variable between 0 – 5 volts by the use of adjustable pots RV1 and RV2, and three switched digital inputs, SWA, SWB, and SWC. Outputs will need to be monitored by a combination of GUI and any external indicators (e.g voltmeter, logic probe etc)

The GUI

Instead of a properties table, the following diagram is labelled to illustrate the make-up of the GUI. This GUI controls and displays the condition of 2 analogue inputs, 2 digital inputs, and 2 digital outputs.



Note 1 : All captions as on above diagram, all toolbox object types are indicated either by their full names or obvious abbreviations.

Note 2 : BarA0 and BarA1 are shapes whose Fillcolours are as shown and whose Fillstyle is solid. The width of each bar is controlled by program code and determined by value of analogue voltage connected to the analogue inputs.

Physical Connections

Digital Input IO0 --- connect SWA (Tml A) to Labjack IO0
Digital Input IO1 --- connect SWB (Tml B) to Labjack IO1
Digital Output IO2 --- no connection required
Digital Output IO3 --- no connection required

Analogue Input AI0 --- connect RV1 (Tml AV1) to Labjack AI0
Analogue Input AI1 --- connect RV2 (Tml AV2) to Labjack AI1

Procedure (Not necessarily in the order presented here)

i. Use the Visual Basic development tool to build the required GUI.

ii. Edit the required code as listed below.

iii. Connect the UWIC built analogue/digital I/O pcb to the labjack unit.

iv. Connect the Labjack to a PC USB port and wait for enumeration to take.

v. Test the operation of the program.

Program Function

The purpose of this program is described by the following steps :

i. Command buttons CMD2ON and CMD2OFF when *clicked* will cause IO2 to be switched to 5V (logic 1) and 0V (logic 0) respectively.

ii. Command buttons CMD3ON and CMD3OFF when *clicked* will cause IO3 to be switched to 5V (logic 1) and 0V (logic 0) respectively.

The timer, Timer1, will cause the analogue inputs to be updated every 0.2 seconds, i.e. Timer1 is set to an interval of 200 ms in its' property box. Also the state of the digital inputs IO1 and IO2 will be read every 2 seconds and the switch setting (ON or OFF) is displayed using label2 and label6.

iii. EAnalogIn is used to read the value of the analogue voltage on inputs AI0 and AI1.

iv. The value of the variable BarWidth is determined by the magnitude of the voltage on analogue input channels 0 and 1. A formula is used to calculate the value of BarWidth which ranges from width 0 for 0 V to width 3255 for 5V.

The value of the variable BarWidth is then used to control the width of the red and green bars between width 0 and width 3255.

v. Variables VoltageAI0 and VoltageAI1 are set by step iii. and are used to display the values of the analogue voltages present on AI0 and AI1 in text box TXTbox and label LblA1 which are situated to the right of the coloured horizontal bars. (*textbox and label used to illustrate use of both*)

vi. The variable StateD is used in conjunction with EDigitalIn to store the state of the digital inputs IO0 and IO3 (and hence the switch settings) and to determine the messages (ON/OFF) placed into labels Label2 and Label6.

Build the GUI, enter the code, and try the program yourself. Take your time to understand the program operation and the code before you proceed onto the assignment.

The Code (Eliminate or add comments [REM or ‘] as you see fit)

```
'This example calls EDigitalIn, EDigitalOut, EAnalogIn, and GetErrorString,
'directly from the LabJack U12 DLL, rather than using the ActiveX control.

'If you have installed the LabJack software distribution or drivers_only,
‘ ljackuw.dll should have been put in your system directory,where
‘VB will be able to find it.

Option Explicit      ‘All variables must now be declared

‘Now follows the declaration for the four external labjack functions used

Private Declare Function EAnalogIn Lib "ljackuw.dll" _
   ( ByRef lpIDNum As Long, _
     ByVal lngDemo As Long, _
     ByVal alngChannel As Long, _
     ByVal alngGain As Long, _
     ByRef lngOverVoltage As Long, _
     ByRef asngVoltages As Single) As Long

 '================================
Private Declare Function EDigitalIn Lib "ljackuw.dll" _
   ( ByRef lpIDNum As Long, _
     ByVal lngDemo As Long, _
     ByVal alngChannel As Long, _
     ByVal alngReadD As Long, _
     ByRef lngState As Long) As Long

'==================================
```

```
Private Declare Function EDigitalOut Lib "ljackuw.dll" _
   ( ByRef lpIDNum As Long, _
     ByVal lngDemo As Long, _
     ByVal alngChannel As Long, _
     ByVal alngWriteD As Long, _
     ByVal lngState As Long) As Long

'================================

Private Declare Sub GetErrorString Lib "ljackuw.dll" _
   ( ByVal lngErrorcode As Long, ByVal lpErrorString As String )

'Note that Strings are passed ByVal since they are inherently pointers. Also,
‘functions with no return value, are called a "Sub" in VB, rather than a “Function”,
‘and when you call a "Sub" you don't put parentheses around the parameters.

'=========================================================

Private Sub CMD2OFF_Click()

   Dim ChannelSelected, stateD, lngErrorcode As Long

   'set digital output channel2 OFF ( i.e. 0V )

   ChannelSelected  = 2        '0=IO0, 1=IO1, 2=IO2, 3=IO3

   stateD = 0

   lngErrorcode = EDigitalOut(-1, 0, ChannelSelected, 0, stateD)

End Sub

‘-------------------------------------------------------------------------------

Private Sub CMD2ON_Click()

   Dim ChannelSelected, stateD, lngErrorcode As Long

   'set digital output channel2 ON ( i.e. 5V )

   ChannelSelected  = 2        '0=IO0, 1=IO1, 2=IO2, 3=IO3

   stateD = 1

   lngErrorcode = EDigitalOut(-1, 0, ChannelSelected, 0, stateD)

End Sub

‘-------------------------------------------------------------------------------
```

```
Private Sub CMD3OFF_Click()
    Dim ChannelSelected, stateD, lngErrorcode As Long
    'set digital output channel3 OFF ( i.e. 0V )
    ChannelSelected  = 3      '0=IO0, 1=IO1, 2=IO2, 3=IO3
    stateD = 0
    lngErrorcode  = EDigitalOut(-1, 0, ChannelSelected, 0, stateD)
End Sub

‘-------------------------------------------------------------------------------

Private Sub CMD3ON_Click()
    Dim ChannelSelected, stateD, lngErrorcode As Long
    'set digital output channel 3 ON (i.e. 5V )
    ChannelSelected  = 3      '0=IO0 , 1=IO1, 2=IO2, 3=IO3
    stateD = 1
    lngErrorcode = EDigitalOut(-1, 0, ChannelSelected, 0, stateD)
End Sub

‘-------------------------------------------------------------------------------

Private Sub Timer1_Timer()

    Dim lngErrorcode As Long
    Dim strError As String * 50
    Dim overvoltage As Long
    Dim VoltageAI0 As Single
    Dim VoltageAI1 As Single
    Dim ChannelSelected As Long
    Dim stateD As Long
    Dim BarWidth As Single

    ' Read analogue input AI0.  Value will be stored in voltageA0
```

```
ChannelSelected = 0

lngErrorcode = EAnalogIn(-1, 0, ChannelSelected, 0, overvoltage, VoltageAI0)

' Read analogue input AI1.  Value will be stored in VoltageAI1

ChannelSelected = 1

lngErrorcode = EAnalogIn(-1, 0, ChannelSelected, 0, overvoltage, VoltageAI1)

'++++++++++++++++++++++++++++++++++++++++++

'Decipher error code if any and set appropriate textbox

GetErrorString lngErrorcode, strError

LblError.Caption = StrError

'Set bargraph displays

' max width from properties is 3255. Decided by drawing the full length
' of bar then checking the width property.  The following code prevents the
'  barwidth becomming zero or less

BarWidth = Int((VoltageAI0 / 5) * 3255)

If  BarWidth < 1 Then
      BarA0.Width = 1

   Else
      BarA0.Width = BarWidth

End If

TXTA0.Text = Str(VoltageAI0) + "   Volts "  'using a text box for display here


BarWidth = Int((VoltageAI1 / 5) * 3255)

If BarWidth < 1 Then
      BarA1.Width = 1

   Else
      BarA1.Width = BarWidth

End If

LblA1.Caption = VoltageAI1    ‘using a label box for display here

'++++++++++++++++++++++++++++++++++++++++++++++++
```

```
    'Read state of digital input channel 1 and store result ( 0/1) in stateD
    ChannelSelected = 1           ' 0 = IO0 , 1 = IO1, 2 = IO2, 3 = IO3
    lngErrorcode = EDigitalIn(-1, 0, ChannelSelected, 0, stateD)
    ' display the state of digital channel 1 ( switch closed or open )
    If stateD = 0 Then Label6.Caption = "OFF" Else Label6.Caption = "ON"
    '++++++++++++++++++++++++++++++++++++++++++++++++++++++
     'Read state of digital input channel 0 and store result ( 0/1) in stateD
    ChannelSelected = 0           ' 0=IO0 , 1=IO1, 2=IO2, 3=IO3
    lngErrorcode = EDigitalIn(-1, 0, ChannelSelected, 0, stateD)
    ' display the state of channel 0 (switch closed or open )
    If stateD = 0 Then Label2.Caption = "OFF" Else Label2.Caption = "ON"
    '++++++++++++++++++++++++++++++++
End Sub
' ***********     END OF PROGRAM  ********************
```

If an error occurs when accessing the labjack it can be displayed using the following two lines.

```
    GetErrorString lngErrorcode, strError
    'Label7.Caption = strError
```

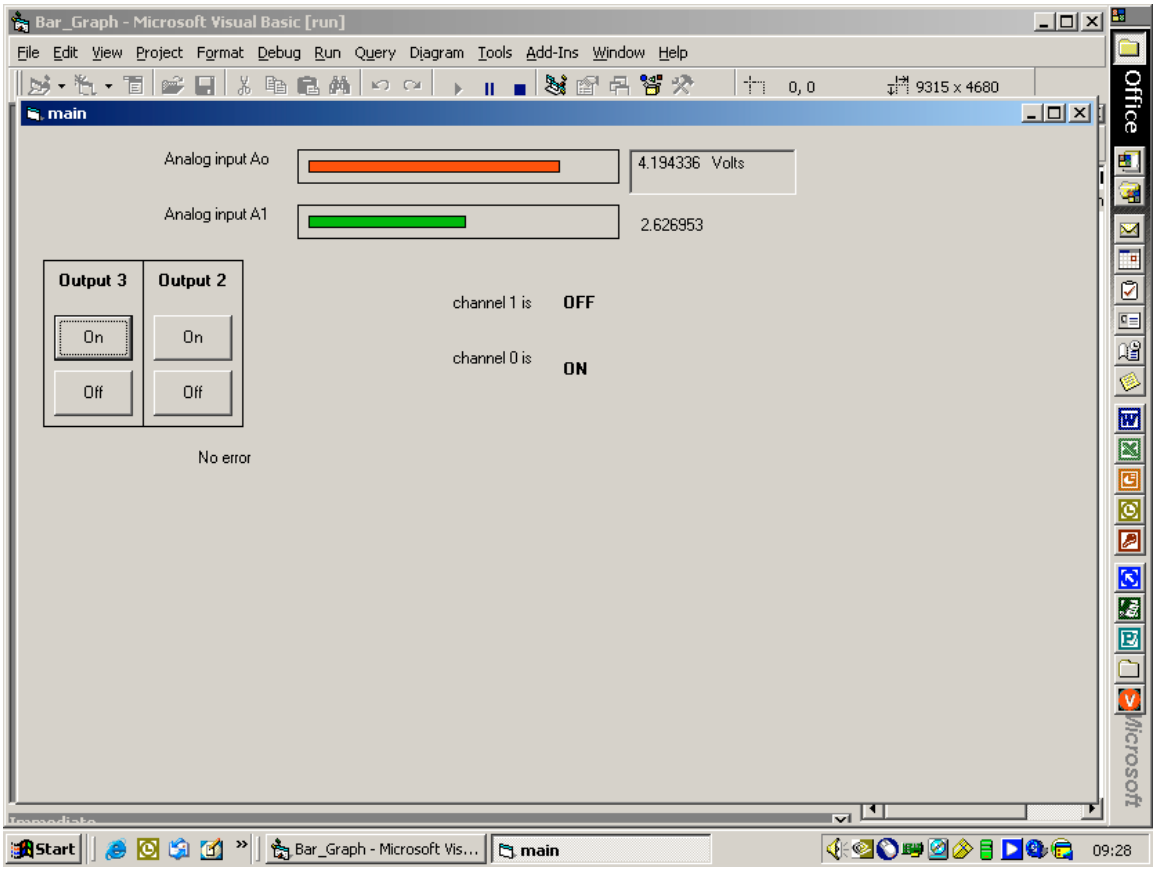
These lines are included in the above code. As a test, leave the Labjack U12 disconnected and run the program, label Lblerror should display *no labjack found*.

This is **not** an assignment so look for as much help as you find necessary and from **anyone** you consider able and willing to provide it. The important thing is that you are able to follow what the program does and how the program code makes it do it.

If anyone wants some extra supervised time in A207 then see me (G.P) to arrange it. In any case you are advised to take advantage of periods for which no timetabled classes are using the room.

The Program Run

The following image shows a screen dump which illustrates the running of the program.



NOTE :

If you use the Microsoft Visual Basic development system to create an executable file, then any PC you attempt to run the program on would **not** need to have Visual Basic installed but would have to have the Labjack dynamic link library ljackuw.dll installed.