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-253 **Version 1.00**

Using DAPIO32 Interface with Visual Basic .NET

The interface files DapIo32.vb and DapIoErr.vb are compatible with Visual Basic .NET. The syntax and function calls to the DAPIO32 library are the same in VB.NET as they were in previous versions of Visual Basic. If you have used the DAPIO32 interface with previous versions of VB before, please note the following:

- A variable of type `Short` in VB.NET is the same as one of type `Integer` in previous versions of VB. In VB.NET a `Short` is 2-byte and an `Integer` or a `Long` is 4-byte.
- The name of the buffer array, instead of the first element of the array, should be passed to the DAPIO32 functions that require one. For example, in previous versions of VB, the function call to `DapBufferGet` might be `DapBufferGet(hDapBinGet, BytesGet, BufData(0))`. In VB.NET, the last parameter should be simply `BufData`.

To start a new VB.NET project for the DAP board, add DapIo32.vb and DapIoErr.vb as existing items to the project. The two files can be found in the DapDev directory after the DAP Development Software is installed from a DAPtools CD version 4.20 or later.

DVM Example

Shown below is the source code of the form DVM for the Digital VoltMeter example in VB.NET. The project can be loaded by opening the file DVM.vbproj. The example does the following:

- Allows the user to start and stop the DAP board with two different buttons, `CmdStart` and `CmdStop`,
- Read a data block from the DAP board per timer event of `Timer1`, and
- Update the label, `lblMeter`, with the last value in the retrieved data block.

```
Option Strict Off
Option Explicit On

Public Class DVM
    Inherits System.Windows.Forms.Form

#Region " Windows Form Designer generated code "
    ' Generated by VB automatically. Not shown here.
#End Region

    Const ON_TRUE = 1
    Const ON_FALSE = 0
    Const MAX_VALUES = 4096
    Const BINARY_TO_VOLT = 0.0001525    ' = 5/32768

    Dim hdapSysPut As Integer
    Dim hdapBinGet As Integer
```

```
Dim BufGetEx As TDapBufferGetEx
Dim iData(MAX_VALUES - 1) As Short

' Start the DAP board.
Private Sub CmdStart_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
                          Handles CmdStart.Click
    If (DapInitialize() = ON_TRUE) Then
        Timer1.Enabled = True
        CmdStart.Visible = False
        CmdStop.Visible = True
    End If
End Sub

' Stop the DAP board. Reset the controls.
Private Sub CmdStop_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
                          Handles CmdStop.Click
    DapTerminate()
    Timer1.Enabled = False
    CmdStop.Visible = False
    CmdStart.Visible = True
End Sub

' Read a block of data from the DAP board.
Function DapBinaryRead(ByRef iNumBytesRead As Integer) As Integer

    DapBinaryRead = ON_TRUE

    iNumBytesRead = DapBufferGetEx(hdapBinGet, BufGetEx, iData)
    If iNumBytesRead < 0 Then
        MsgBox("Error reading data from DAP.")
        DapBinaryRead = ON_FALSE
    End If
End Function

' Establish communication with the DAP board and download the DAPL configuration.
Function DapInitialize() As Integer
    On Error GoTo ErrorHandler

    DapInitialize = ON_TRUE

    ' Open DAP handles
    hdapSysPut = DapHandleOpen("\\.\DAP0\$SYSIN", DAPOPEN_WRITE)
    hdapBinGet = DapHandleOpen("\\.\DAP0\$BINOUT", DAPOPEN_READ)

    If (hdapSysPut = 0) Or (hdapBinGet = 0) Then
        MsgBox("Error opening DAP handle(s).")
        GoTo ErrorHandler
    End If

    ' Flush any data from $BINOUT
    DapInputFlush(hdapBinGet)

    ' The startup path = output path set under Project | DVM Properties
   If DapConfig(hdapSysPut, System.Windows.Forms.Application.StartupPath & "\DVM.DAP")
      = False Then
        MsgBox("Error downloading DAPL configuration to DAP.")
        GoTo ErrorHandler
    End If

    Exit Function
```

```
    ErrorHandler:
        DapTerminate()
        DapInitialize = ON_FALSE
    End Function

    ' Disable the timer and stop the DAP. Close the Accel32 handles.
    Sub DapTerminate()
        Timer1.Enabled = False
        If hdapSysPut <> 0 Then
            DapLinePut(hdapSysPut, "STOP")
            DapHandleClose(hdapSysPut)
        End If

        If hdapBinGet <> 0 Then DapHandleClose(hdapBinGet)
    End Sub

    ' Stop the DAP board before unloading form.
    Private Sub DVM_Closed(ByVal sender As System.Object, ByVal e As System.EventArgs)
                          Handles MyBase.Closed
        DapTerminate()
    End Sub

    ' Initialize the controls when the form loads.
    Private Sub DVM_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
                        MyBase.Load
        CmdStart.Visible = True
        CmdStop.Visible = False
        Timer1.Interval = 1000
        Timer1.Enabled = False

        ' Initialize the structure for DapBufferGetEx
        DapStructPrepare(BufGetEx, Len(BufGetEx))
        BufGetEx.iBytesGetMin = 2       ' Read at least one data point (one short = 2 bytes)
        BufGetEx.iBytesGetMax = MAX_VALUES * 2
        BufGetEx.dwTimeWait = 1000      ' 1000 milliseconds
        BufGetEx.dwTimeOut = 0
        BufGetEx.iBytesMultiple = 2
    End Sub

    ' Read a block of data from the DAP and display the most recent value from the block.
    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs)
                           Handles Timer1.Tick
        On Error GoTo ErrorHandler
        Dim iNumBytesRead As Integer
        Dim fValue As Single

        If DapBinaryRead(iNumBytesRead) = ON_FALSE Then
            DapTerminate()
        Else
            If iNumBytesRead <> 0 Then
                fValue = CSng(iData(iNumBytesRead / 2 - 1)) * BINARY_TO_VOLT
                lblMeter.Text = Format(fValue, "0.00") & " V"
            End If
        End If
        Exit Sub
    ErrorHandler:
        Timer1.Enabled = False
    End Sub

End Class
```