

```
Public ConfigurationMode As Byte ' See SetConfiguration comment
```

```
Public Sub Form_Load()  
Call SelectMode_Click ' Set DOT mode  
End Sub
```

```
Private Sub Form_MouseDown(Button%, Shift%, Xpos As Single, Ypos As Single)  
' Determine which LED the mouse is over  
xValue& = Int((Xpos - 120) / 240): yValue& = Int((Ypos - 1440) / 240)  
If (xValue& > -1) And (xValue& < 40) And InDotMode Then  
    Select Case yValue ' Toggle the LED if clicked while in DOT mode  
        Case 0: LED0(xValue&).Visible = Not LED0(xValue).Visible  
        Case 1: LED1(xValue&).Visible = Not LED1(xValue).Visible  
        Case 2: LED2(xValue&).Visible = Not LED2(xValue).Visible  
        Case 3: LED3(xValue&).Visible = Not LED3(xValue).Visible  
        Case 4: LED4(xValue&).Visible = Not LED4(xValue).Visible  
        Case 5: LED5(xValue&).Visible = Not LED5(xValue).Visible  
        Case 6: LED6(xValue&).Visible = Not LED6(xValue).Visible  
    End Select  
End If  
End Sub
```

```
Private Sub SelectMode_Click()  
' Toggle the select mode  
If SelectMode.Left = 4200 Then  
    SelectMode.Left = 4560: EntryText.Visible = False ' Dot mode  
    Dot.ForeColor = RGB(255, 255, 255): Text.ForeColor = RGB(42, 42, 42) ' Grey  
    Call SetConfiguration(2)  
Else: SelectMode.Left = 4200: EntryText.Visible = True ' Text mode  
    Dot.ForeColor = RGB(42, 42, 42): Text.ForeColor = RGB(255, 255, 255) ' White  
    Call SetConfiguration(1)  
End If  
End Sub
```

```
Private Function InDotMode() As Boolean  
InDotMode = (SelectMode.Left = 4560)  
End Function
```

```
Private Sub UpdateIO_Click()  
' Update the real IO device  
Dim Buffer(40) As Byte  
Buffer(0) = ConfigurationMode  
For i& = 0 To 39  
    If InDotMode Then ' Fill the buffer with the DOT pattern  
        Buffer(i& + 1) = (LED0(i&).Visible And 1) + (LED1(i&).Visible And 2) + (LED2(i&).Visible And 4) +  
(LED3(i&).Visible And 8) + _  
        (LED4(i&).Visible And 16) + (LED5(i&).Visible And 32) + (LED6(i&).Visible And 64)  
    Else ' In Text Mode. Fill the buffer with text  
        If i& < Len(EntryText.Text) Then  
            Buffer(i& + 1) = Asc(Mid(EntryText.Text, i& + 1, 1))  
        Else: Buffer(i& + 1) = 0: End If  
    End If ' In DotMode  
Next i&  
If InDotMode Then ' Send the DOT pattern to the I/O device  
    Call WriteUSBdevice(AddressFor(Buffer(0)), 41)  
Else ' Send the text and retrieve the dot pattern  
    Call WriteUSBdevice(AddressFor(Buffer(0)), 41)  
    Call ReadUSBdevice(AddressFor(Buffer(0)), 40)  
    For i& = 0 To 39  
        LED6(i&).Visible = Buffer(i&) And 1: LED5(i&).Visible = Buffer(i&) And 2: LED4(i&).Visible = Buffer(i&)  
And 4  
        LED3(i&).Visible = Buffer(i&) And 8: LED2(i&).Visible = Buffer(i&) And 16: LED1(i&).Visible = Buffer(i&)  
And 32  
        LED0(i&).Visible = Buffer(i&) And 64  
    Next i&  
End If  
End Sub
```

```
Public Sub SetConfiguration(NewConfiguration As Byte)  
' Get the Current Configuration  
Dim ConfigurationData As ConfigurationDataType  
Dim Buffer(100) As Long  
' PROBLEM: during system testing it was discovered that Windows 98 DOES NOT SUPPORT the  
' HidD_GetConfiguration system call - this means that an application program (which is what  
' this is) CANNOT send a SetConfiguration packet to an I/O device; the call is supported  
' by Windows 2000 and by Windows 98 for kernel level (ie device driver) software.  
' The "Reader Board" I/O device was tested independantly for its correct dual-configuration  
' operation.  
' This example, therefore, uses an ALTERNATE method of implementing "dual-configuration"
```

```
' I extended the Report Buffer by one byte and prepend the data with the "mode" byte.  
' This works, not as elegant as I would have wanted, but it does implement "dual-modes"  
,  
' Success& = HidD_GetConfiguration(HidHandle&, AddressFor(Buffer(0)), UBound(Buffer))  
' If (Success& = 0) Then ErrorExit ("Could not get CurrentConfiguration")  
ConfigurationMode = NewConfiguration  
End Sub
```