Main Program

```
'Created by Jonathan Johnston and Adam Serafin
'This software is copyrighted and the rights are owned by us.
'This program takes a digital signal from the SBC and displays the results
'In a text box.  Then it takes the signal and displays the signal in the scope.

Option Explicit
Dim message As String
Dim InBuffer() As Byte                'buffer for the bytes
Dim i As Long
Dim LabelOffSet As Long
Private Const iHigh As Long = 9                          'Scope Trace High Line Value
Private Const iPW As Long = 9                            'Scope Trace Pulse Width
Private Const iDelta As Long = 36                        'Scope Trace Difference from High
To Low Line
Private Const sStartBitLabel As String = "S"             'Scope Trace Label For The Start
Bit
Private Const sStopBitLabel As String = "s"              'Scope Trace Label For The Stop
Bit
Private Const sParityBitLabel As String = "p"            'Scope Trace Label For The Parity
Bit

Private Type PortSettings
   Baud As String
   Parity As String
   DataBits As String
   StopBits As String
End Type

Private Type ScopeColor
   Picture As String
   Trace As Long
End Type
Private typDisplayColor As ScopeColor

Private Sub cmdClose_Click()
On Error Resume Next
MSComm1.PortOpen = False              'closes the port
cmdClose.Enabled = False              'disables the close button
cmdOpen.Enabled = True                'enables the open button
End Sub

Private Sub cmdOpen_Click()
On Error GoTo pOpen
If MSComm1.PortOpen = False Then
```

```
    MSComm1.PortOpen = True             'opens the port
    cmdOpen.Enabled = False             'disables the open button
    cmdClose.Enabled = True             'enables the close button
    Exit Sub
End If
pOpen:     MsgBox "The port is already open", vbInformation, "Port Open"
'this error is if the port is already open by another program
End Sub

Private Sub cmdshow_Click()
mnuInfo.Visible = True
End Sub

Private Sub Form_Load()
MSComm1.CommPort = 1                    'sets com port to 1
MSComm1.Settings = "38400,N,8,1"        'sets the baud rate, parity bit, bit size and stop
bit
MSComm1.InputLen = 0                    'sets length of buffer to 0
MSComm1.Handshaking = comNone           'no handshaking of port
MSComm1.InputMode = comInputModeBinary  'sets to read binary
MSComm1.NullDiscard = False             'accepts binary "0"
MSComm1.DTREnable = True                'sets the data terminal ready to true
cmdOpen.Enabled = True                  'enables the open button
cmdClose.Enabled = False                'disables the close button
With typDisplayColor                    'Set The Scope Trace To Blue
    .Picture = App.Path & "\ScopeGridBlue.bmp"
    .Trace = RGB(14, 181, 228)
    picScope.Picture = LoadPicture(.Picture)
End With
End Sub

Private Sub ReceiveBits()
Dim lScope() As Long
Dim iZ As Long
Dim Num As Integer
If MSComm1.InBufferCount = 0 Then       'checks for data in buffer and displays if not
    txtSensor.Text = "No Data in Buffer"
    Exit Sub
End If
InBuffer = MSComm1.Input                'pulls data from buffer
If UBound(InBuffer) < 3 Then
    ReDim lScope(UBound(InBuffer))      'resizes the scope to the size of the buffer
Else
    ReDim lScope(3)                     'resizes scope to 3
End If
For i = 0 To UBound(InBuffer)           'takes upper bounds of buffer
```

```
   message = Chr$(InBuffer(i) + 48)        'and puts it in a string
Next i
txtSensor.Text = "Sensor Reading= " & message     'writes message to text box
If chkScope.Value = vbChecked Then
   ScopeDisplay lScope
Else
   LabelOffSet = lblScope1(Num).Index * 35                   'One Control Array For All
Ports (offset accordingly)
   Num = Num + 1
      For iZ = LabelOffSet To LabelOffSet + 35
         lblScope(iZ).Caption = "-"
      Next
      picScope.Refresh
End If
End Sub

Private Sub ScopeDisplay(ByRef nScope() As Long)
On Error Resume Next

Dim iX As Long                                  'Use for the X Coordinate in the Scope Picture
Dim iY As Long                                  'Use for the Y Coordinate in the Scope Picture
Dim iZ As Long                                  'Multi use Index Counter
Dim iLow As Long
Dim hPic As Long
Dim iTraceColor As Long
Dim bOn As Boolean
Dim iBytes As Long
Dim bTrace() As Boolean
Dim iBits As Long
Dim iStop As Long
Dim PortSet As PortSettings
Dim iBitCount As Long
Dim iOnBits As Long
Dim bChkParity As Boolean

iLow = iHigh + iDelta           'Low Scope Trace Value
picScope.Refresh               'Clear The Scope Display
iBytes = UBound(nScope)         'No Of Bytes In The Data Array To Display
iZ = 0
With PortSet                              'Count The Additional Bits From The Ports
Settings
   If .Parity <> "n" Then
      iZ = 1
      bChkParity = True
   End If
   iBits = CLng(.DataBits)                    'Number Of Bits To Display From Port
Settings
   iStop = CLng(.StopBits)                    'Number Of Stop Bits To Add
   iZ = iZ + iBits + iStop                   'Sum Stop,Parity, and Data Bits
End With

iBitCount = (iZ * (iBytes + 1)) + (iBytes + 1)
      ReDim bTrace(iBitCount - 1)                     'Trace Array One less than Total
BitCount (array index)
      iZ = 0
      For iX = 0 To iBytes                           'Max iBytes is 3 (Four Byte Trace)
         bTrace(iZ) = True                          'Start Bit
         lblScope1(iZ + LabelOffSet).Caption = sStartBitLabel
         iZ = iZ + 1
         iOnBits = 0                              'Zero Count For On Bits To Check Parity
         For iY = 1 To iBits                        'Parse Each Data Bit
            bOn = BitOn(nScope(iX), iY)
            iOnBits = Abs(bOn) + iOnBits
            bTrace(iZ) = Not bOn                    'Invert as Negative Voltage on Scope is
Logical True
            lblScope1(iZ + LabelOffSet).Caption = CStr(iY - 1)
            iZ = iZ + 1
         Next
         If bChkParity Then                        'True For All But Parity Setting Of "None"
            Select Case PortSet.Parity
               Case "e"                         'Even Parity
                  If iOnBits Mod 2 Then              'OnBits is Odd Parity Bit Is On
                     bTrace(iZ) = False
                  Else
                     bTrace(iZ) = True              'OnBits is Even Parity Bit Is Off
                  End If
               Case "o"                         'Odd Parity
                  If iOnBits Mod 2 Then              'OnBits is Odd Parity Bit Is Off
                     bTrace(iZ) = True
                  Else
                     bTrace(iZ) = False             'OnBits is Even Parity Bit Is On
                  End If
               Case "m"                          'Mark Parity Bit is Always On
                  bTrace(iZ) = False
               Case "s"
                  bTrace(iZ) = True                 'Space Parity Bit is Always Off
            End Select
            lblScope1(iZ + LabelOffSet).Caption = sParityBitLabel
            iZ = iZ + 1
         End If
'Set The Scope Trace Stop Bits
```

```
          If iStop = 1 Then                         'One Stop Bit
             bTrace(iZ) = False                      'Stop Bit
             lblScope1(iZ + LabelOffSet).Caption = sStopBitLabel
             iZ = iZ + 1
          Else                                'Two Stop Bits
             bTrace(iZ) = False                      'Stop Bit
             lblScope1(iZ + LabelOffSet).Caption = sStopBitLabel
             bTrace(iZ + 1) = False                  'Stop Bit
             lblScope1(iZ + LabelOffSet + 1).Caption = sStopBitLabel
             iZ = iZ + 2
          End If
       Next
'Get The Proper Scope Trace To Match The Background Color
       iTraceColor = typDisplayColor.Trace
       iBits = 0                              'Zero The Bit Count Index
       bOn = bTrace(iBits)                          'Parse The Trace Array And Draw on
Scope Display
       For iX = 1 To picScope.Width                   'X Coordinate
          iY = iLow - (Abs(bOn) * iDelta)              'Y Coordinate
          Do While iX Mod iPW                       'Draw Horizontal Line To The Pulse
Width
             SetPixel hPic, iX, iY, iTraceColor
             iX = iX + 1
          Loop

          If iBits < UBound(bTrace) Then               'Check For End Of Trace
             If bTrace(iBits) <> bTrace(iBits + 1) Then  'Check To See If Next Bit Has
Changed State
                For iY = iHigh To iLow               'Next Bit Change In State Draw Vertical
Line
                   SetPixel hPic, iX, iY, iTraceColor
                Next
             End If
             iBits = iBits + 1                      'Get The Next Bit
             bOn = bTrace(iBits)
          Else                                'End Of Trace Data Set The Stop Bit
             iX = iX + 1                           'Skip a Bit for the Stop Bit
             iBits = iBits + 1 + LabelOffSet
             Exit For
          End If
       Next

       Do While iX < picScope.Width          'Run The Scope Trace Out
          SetPixel hPic, iX, iLow, iTraceColor
          iX = iX + 1
       Loop
```

```
        For iX = iBits To LabelOffSet + 47                  'Set The Remaining Scope Label
Captions
            lblScope1(iX).Caption = "-"
        Next
End Sub

Private Function BitOn(Number As Long, Bit As Long) As Boolean
        Dim iX As Long
        Dim iY As Long

        iY = 1
        For iX = 1 To Bit - 1
            iY = iY * 2
        Next
        If Number And iY Then BitOn = True Else BitOn = False
End Function

Private Sub Form_Unload(Cancel As Integer)
On Error Resume Next
MSComm1.PortOpen = False                'closes the com port
End Sub

Private Sub lblScope1_Click(Index As Integer)
'
End Sub

Private Sub lblShow_Click()
mnuInfo.Visible = True
End Sub

Private Sub mnuAbout_Click()
frmAbout.Show
End Sub

Private Sub mnuExit_Click()
On Error Resume Next
MSComm1.PortOpen = False
Unload Me
End Sub

Private Sub mnuInfo_Click()
MsgBox "This program is created by Jonathan Johnston!", vbOKOnly, "Info"
        MsgBox "I'm the best", vbExclamation, "BEST"
End Sub

Private Sub MSComm1_OnComm()
```

```
Select Case MSComm1.CommEvent
    Case comEvReceive                  'redefines the size of buffer then
        txtSensor.Text = ""
        ReDim InBuffer(MSComm1.InBufferCount)
        ReceiveBits                    'reads the bits from the port
    Case comEventRxOver                'redefines the size of buffer if
        ReDim InBuffer(MSComm1.InBufferCount)   'the buffer starts to overflow
End Select
End Sub
```

About Form

```
Option Explicit

' Reg Key Security Options...
Const READ_CONTROL = &H20000
Const KEY_QUERY_VALUE = &H1
Const KEY_SET_VALUE = &H2
Const KEY_CREATE_SUB_KEY = &H4
Const KEY_ENUMERATE_SUB_KEYS = &H8
Const KEY_NOTIFY = &H10
Const KEY_CREATE_LINK = &H20
Const KEY_ALL_ACCESS = KEY_QUERY_VALUE + KEY_SET_VALUE + _
                       KEY_CREATE_SUB_KEY + KEY_ENUMERATE_SUB_KEYS + _
                       KEY_NOTIFY + KEY_CREATE_LINK + READ_CONTROL

' Reg Key ROOT Types...
Const HKEY_LOCAL_MACHINE = &H80000002
Const ERROR_SUCCESS = 0
Const REG_SZ = 1                         ' Unicode nul terminated string
Const REG_DWORD = 4                      ' 32-bit number

Const gREGKEYSYSINFOLOC = "SOFTWARE\Microsoft\Shared Tools Location"
Const gREGVALSYSINFOLOC = "MSINFO"
Const gREGKEYSYSINFO = "SOFTWARE\Microsoft\Shared Tools\MSINFO"
Const gREGVALSYSINFO = "PATH"

Private Declare Function RegOpenKeyEx Lib "advapi32" Alias "RegOpenKeyExA"
(ByVal hKey As Long, ByVal lpSubKey As String, ByVal ulOptions As Long, ByVal
samDesired As Long, ByRef phkResult As Long) As Long
Private Declare Function RegQueryValueEx Lib "advapi32" Alias "RegQueryValueExA"
(ByVal hKey As Long, ByVal lpValueName As String, ByVal lpReserved As Long,
ByRef lpType As Long, ByVal lpData As String, ByRef lpcbData As Long) As Long
Private Declare Function RegCloseKey Lib "advapi32" (ByVal hKey As Long) As Long
Private Sub cmdSysInfo_Click()
  Call StartSysInfo
End Sub

Private Sub cmdOK_Click()
Unload Me
End Sub

Private Sub Form_Load()
    Me.Caption = "About " & App.Title
    lblVersion.Caption = "Version " & App.Major & "." & App.Minor & "." &
App.Revision
    lblTitle.Caption = App.Title
End Sub

Public Sub StartSysInfo()
    On Error GoTo SysInfoErr

    Dim rc As Long
    Dim SysInfoPath As String

    ' Try To Get System Info Program Path\Name From Registry...
    If GetKeyValue(HKEY_LOCAL_MACHINE, gREGKEYSYSINFO,
gREGVALSYSINFO, SysInfoPath) Then
    ' Try To Get System Info Program Path Only From Registry...
    ElseIf GetKeyValue(HKEY_LOCAL_MACHINE, gREGKEYSYSINFOLOC,
gREGVALSYSINFOLOC, SysInfoPath) Then
        ' Validate Existance Of Known 32 Bit File Version
        If (Dir(SysInfoPath & "\MSINFO32.EXE") <> "") Then
            SysInfoPath = SysInfoPath & "\MSINFO32.EXE"

        ' Error - File Can Not Be Found...
        Else
            GoTo SysInfoErr
        End If
    ' Error - Registry Entry Can Not Be Found...
    Else
        GoTo SysInfoErr
    End If

    Call Shell(SysInfoPath, vbNormalFocus)

    Exit Sub
SysInfoErr:
    MsgBox "System Information Is Unavailable At This Time", vbOKOnly
End Sub
```

```
Public Function GetKeyValue(KeyRoot As Long, KeyName As String, SubKeyRef As
String, ByRef KeyVal As String) As Boolean
   Dim i As Long                                ' Loop Counter
   Dim rc As Long                               ' Return Code
   Dim hKey As Long                             ' Handle To An Open Registry Key
   Dim hDepth As Long                           '
   Dim KeyValType As Long                       ' Data Type Of A Registry Key
   Dim tmpVal As String                         ' Tempory Storage For A Registry Key
Value
   Dim KeyValSize As Long                       ' Size Of Registry Key Variable
   '------------------------------------------------------------
   ' Open RegKey Under KeyRoot {HKEY_LOCAL_MACHINE...}
   '------------------------------------------------------------
   rc = RegOpenKeyEx(KeyRoot, KeyName, 0, KEY_ALL_ACCESS, hKey) ' Open
Registry Key

   If (rc <> ERROR_SUCCESS) Then GoTo GetKeyError          ' Handle Error...

   tmpVal = String$(1024, 0)                    ' Allocate Variable Space
   KeyValSize = 1024                            ' Mark Variable Size

   '------------------------------------------------------------
   ' Retrieve Registry Key Value...
   '------------------------------------------------------------
   rc = RegQueryValueEx(hKey, SubKeyRef, 0, _
                KeyValType, tmpVal, KeyValSize)    ' Get/Create Key Value

   If (rc <> ERROR_SUCCESS) Then GoTo GetKeyError          ' Handle Errors

   If (Asc(Mid(tmpVal, KeyValSize, 1)) = 0) Then          ' Win95 Adds Null Terminated
String...
      tmpVal = Left(tmpVal, KeyValSize - 1)             ' Null Found, Extract From String
   Else                                         ' WinNT Does NOT Null Terminate String...
      tmpVal = Left(tmpVal, KeyValSize)                 ' Null Not Found, Extract String
Only
   End If
   '------------------------------------------------------------
   ' Determine Key Value Type For Conversion...
   '------------------------------------------------------------
   Select Case KeyValType                       ' Search Data Types...
   Case REG_SZ                                  ' String Registry Key Data Type
      KeyVal = tmpVal                           ' Copy String Value
   Case REG_DWORD                               ' Double Word Registry Key Data Type
      For i = Len(tmpVal) To 1 Step -1          ' Convert Each Bit
         KeyVal = KeyVal + Hex(Asc(Mid(tmpVal, i, 1)))   ' Build Value Char. By Char.
```

```
        Next
        KeyVal = Format$("&h" + KeyVal)                  ' Convert Double Word To String
    End Select

    GetKeyValue = True                                ' Return Success
    rc = RegCloseKey(hKey)                            ' Close Registry Key
    Exit Function                                   ' Exit

GetKeyError:      ' Cleanup After An Error Has Occured...
    KeyVal = ""                                     ' Set Return Val To Empty String
    GetKeyValue = False                               ' Return Failure
    rc = RegCloseKey(hKey)                            ' Close Registry Key
End Function
```

API Module

```
Option Explicit
'   File:
'       API.Bas
'   Author:
'       Jonathan Johnston
'   Description:
'       This Module Contains The API Functions To Create, Read, and Write To The
Windows Registry As Well As The
'   Required Constants.
'------------------------------------------------------------------------------------------------------
------------
'   Revisions:
'       Original 5/22/2002
'------------------------------------------------------------------------------------------------------
------------
'   Functions And Subroutines:
'       1.  Sleep
'       2.  SleepEx
'       3.  RegOpenKeyEx              Alias: RegOpenKeyExA
'       4.  RegCloseKey
'       5.  RegQueryValueEx           Alias: RegQueryValueExA
'       6.  RegCreateKeyEx            Alias: RegCreateKeyExA
'       7.  RegDeleteKey              Alias: RegDeleteKeyA
'       8.  RegSetValueEx             Alias: RegSetValueExA
'       9.  RegEnumKey                Alias: RegEnumKeyA
'       10. RegDeleteValue            Alias: RegDeleteValueA
'       11. RegEnumValue              Alias: RegEnumValueA
'       12. CopyMemory                Alias: RtlMoveMemory
'       13. FormatMessage             Alias: FormatMessageA
'       14. keybd_event
'       15. GetVersionEx                Alias: GetVersionExA
'-----------------------------------------------------------------------------------------------------------
------------
'   Properties:
'-----------------------------------------------------------------------------------------------------------
------------
'   Required Functions,Subroutines,Properties,Variables,Etc.:
'
'-----------------------------------------------------------------------------------------------------------
------------
'   Variables:
'       Public:
'
'-----------------------------------------------------------------------------------------------------------
------------
'   Types:
'       Public
'-----------------------------------------------------------------------------------------------------------
------------
Public Type POINTAPI
        x As Long
        y As Long
End Type
'   Constants:
'       Public:
'-----------------------------------------------------------------------------------------------------------
------------
Public Const SYNCHRONIZE = &H100000
Public Const READ_CONTROL = &H20000
Public Const STANDARD_RIGHTS_READ = (READ_CONTROL)
Public Const STANDARD_RIGHTS_WRITE = (READ_CONTROL)
Public Const STANDARD_RIGHTS_ALL = &H1F0000
'-----------------------------------------------------------------------------------------------------------
------------
Public Const KEY_QUERY_VALUE = &H1
Public Const KEY_ENUMERATE_SUB_KEYS = &H8
Public Const KEY_NOTIFY = &H10
Public Const KEY_SET_VALUE = &H2
Public Const KEY_CREATE_SUB_KEY = &H4
Public Const KEY_READ = ((READ_CONTROL Or KEY_QUERY_VALUE Or
KEY_ENUMERATE_SUB_KEYS Or KEY_NOTIFY) And (Not SYNCHRONIZE))
Public Const KEY_WRITE = ((STANDARD_RIGHTS_WRITE Or KEY_SET_VALUE
Or KEY_CREATE_SUB_KEY) And (Not SYNCHRONIZE))
'-----------------------------------------------------------------------------------------------------------
------------
Public Const ERROR_SUCCESS = 0&
```

```
'-----------------------------------------------------------------------------------------------------
------------
Public Const REG_SZ = 1
Public Const REG_BINARY = 3
Public Const REG_DWORD = 4
'-----------------------------------------------------------------------------------------------------
------------
Public Const HKEY_CLASSES_ROOT = &H80000000
Public Const HKEY_CURRENT_USER = &H80000001
Public Const HKEY_LOCAL_MACHINE = &H80000002
Public Const HKEY_USERS = &H80000003
Public Const HKEY_PERFORMANCE_DATA = &H80000004
Public Const HKEY_CURRENT_CONFIG = &H80000005
Public Const HKEY_DYN_DATA = &H80000006
'-----------------------------------------------------------------------------------------------------
------------
Public Const REG_CREATED_NEW_KEY = &H1
Public Const REG_OPENED_EXISTING_KEY = &H2
'-----------------------------------------------------------------------------------------------------
------------
Public Const PS_SOLID = 0
Public Const PS_DASH = 1                    '  -------
Public Const PS_DOT = 2                     '  .......
Public Const PS_DASHDOT = 3                 '  _._._._
Public Const PS_DASHDOTDOT = 4              '  _.._.._
Public Const PS_NULL = 5
Public Const PS_INSIDEFRAME = 6
'-----------------------------------------------------------------------------------------------------
------------
Public Const FLOODFILLBORDER = 0
Public Const FLOODFILLSURFACE = 1
'-----------------------------------------------------------------------------------------------------
------------
Public Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)
'-----------------------------------------------------------------------------------------------------
------------
Public Declare Function SleepEx Lib "kernel32" (ByVal dwMilliseconds As Long,
ByVal bAlertable As Long) As Long
'-----------------------------------------------------------------------------------------------------
------------
Public Declare Function RegOpenKeyEx Lib "advapi32.dll" Alias "RegOpenKeyExA"
(ByVal hKey As Long, ByVal lpSubKey As _
    String, ByVal ulOptions As Long, ByVal samDesired As Long, phkResult As Long)
As Long
'-----------------------------------------------------------------------------------------------------
------------
```

```
Public Declare Function RegCloseKey Lib "advapi32.dll" (ByVal hKey As Long) As
Long
'-------------------------------------------------------------------------------------------------------------
------------
Public Declare Function RegQueryValueEx Lib "advapi32.dll" Alias
"RegQueryValueExA" (ByVal hKey As Long, ByVal _
   lpValueName As String, ByVal lpReserved As Long, lpType As Long, lpData As Any,
lpcbData As Long) As Long
'-------------------------------------------------------------------------------------------------------------
------------
Public Declare Function RegCreateKeyEx Lib "advapi32.dll" Alias "RegCreateKeyExA"
(ByVal hKey As Long, ByVal lpSubKey _
   As String, ByVal Reserved As Long, ByVal lpClass As Long, ByVal dwOptions As
Long, ByVal samDesired As Long, ByVal _
   lpSecurityAttributes As Long, phkResult As Long, lpdwDisposition As Long) As Long
'-------------------------------------------------------------------------------------------------------------
------------
Public Declare Function RegDeleteKey Lib "advapi32.dll" Alias "RegDeleteKeyA"
(ByVal hKey As Long, ByVal lpSubKey As _
   String) As Long
'-------------------------------------------------------------------------------------------------------------
------------
Public Declare Function RegSetValueEx Lib "advapi32.dll" Alias "RegSetValueExA"
(ByVal hKey As Long, ByVal _
   lpValueName As String, ByVal Reserved As Long, ByVal dwType As Long, lpData
As Any, ByVal cbData As Long) As Long
'-------------------------------------------------------------------------------------------------------------
------------
Public Declare Function RegEnumKey Lib "advapi32.dll" Alias "RegEnumKeyA"
(ByVal hKey As Long, ByVal dwIndex As Long, _
   ByVal lpName As String, ByVal cbName As Long) As Long
'-------------------------------------------------------------------------------------------------------------
------------
Public Declare Function RegDeleteValue Lib "advapi32.dll" Alias "RegDeleteValueA"
(ByVal hKey As Long, ByVal _
   lpValueName As String) As Long
'-------------------------------------------------------------------------------------------------------------
------------
Public Declare Function RegEnumValue Lib "advapi32.dll" Alias "RegEnumValueA"
(ByVal hKey As Long, ByVal dwIndex As _
   Long, ByVal lpValueName As String, lpcbValueName As Long, ByVal lpReserved
As Long, lpType As Long, lpData As Any, _
lpcbData As Long) As Long
'-------------------------------------------------------------------------------------------------------------
------------
```

```
Public Declare Sub CopyMemory Lib "kernel32" Alias "RtlMoveMemory" (dest As Any,
Source As Any, ByVal numBytes As Long)
'-----------------------------------------------------------------------------------------------------------
------------
Public Declare Function LineTo Lib "gdi32" (ByVal hdc As Long, ByVal x As Long,
ByVal y As Long) As Long
'-----------------------------------------------------------------------------------------------------------
------------
Public Declare Function Ellipse Lib "gdi32" (ByVal hdc As Long, ByVal X1 As Long,
ByVal Y1 As Long, ByVal X2 _
As Long, ByVal Y2 As Long) As Long
'-----------------------------------------------------------------------------------------------------------
------------
Public Declare Function MoveToEx Lib "gdi32" (ByVal hdc As Long, ByVal x As Long,
ByVal y As Long, lpPoint _
As POINTAPI) As Long
'-----------------------------------------------------------------------------------------------------------
------------
Public Declare Function CreatePen Lib "gdi32" (ByVal nPenStyle As Long, ByVal
nWidth As Long, _
ByVal crColor As Long) As Long
'-----------------------------------------------------------------------------------------------------------
------------
Public Declare Function SelectObject Lib "gdi32" (ByVal hdc As Long, ByVal hObject
As Long) As Long
'-----------------------------------------------------------------------------------------------------------
------------
Public Declare Function DeleteObject Lib "gdi32" (ByVal hObject As Long) As Long
'-----------------------------------------------------------------------------------------------------------
------------
Public Declare Function CreateSolidBrush Lib "gdi32" (ByVal crColor As Long) As
Long
'-----------------------------------------------------------------------------------------------------------
------------
Public Declare Function ExtFloodFill Lib "gdi32" (ByVal hdc As Long, ByVal x As
Long, ByVal y As Long, _
ByVal crColor As Long, ByVal wFillType As Long) As Long
'-----------------------------------------------------------------------------------------------------------
------------
Public Declare Function GetPixel Lib "gdi32" (ByVal hdc As Long, ByVal x As Long,
ByVal y As Long) As Long
'-----------------------------------------------------------------------------------------------------------
------------
Public Declare Function SetPixel Lib "gdi32" (ByVal hdc As Long, ByVal x As Long,
ByVal y As Long, _
ByVal crColor As Long) As Long
```

```
'---------------------------------------------------------------------------------------------------------
------------
Public Declare Function CreateCompatibleBitmap Lib "gdi32" (ByVal hdc As Long,
ByVal nWidth As Long, _
 ByVal nHeight As Long) As Long
'---------------------------------------------------------------------------------------------------------
------------
Public Declare Function CreateCompatibleDC Lib "gdi32" (ByVal hdc As Long) As
Long
'---------------------------------------------------------------------------------------------------------
------------
Public Declare Function DeleteDC Lib "gdi32" (ByVal hdc As Long) As Long
'---------------------------------------------------------------------------------------------------------
------------
Public Declare Function BitBlt Lib "gdi32" (ByVal hDestDC As Long, ByVal x As
Long, ByVal y As Long, ByVal nWidth _
As Long, ByVal nHeight As Long, ByVal hSrcDC As Long, ByVal xSrc As Long,
ByVal ySrc As Long, ByVal dwRop As Long) _
As Long
'---------------------------------------------------------------------------------------------------------
------------
```