MTMICRImage.OCX
PROGRAMMING REFERENCE MANUAL

PART NUMBER 99875457-1

DECEMBER 2009

MAGTEK®
REGISTERED TO ISO 9001:2008
1710 Apollo Court
Seal Beach, CA 90740
Phone: (562) 546-6400
FAX: (562) 546-6301
Technical Support: (651) 415-6800
www.magtek.com

REVISIONS

Rev Number	Date	Notes
1.01	9 Dec 09	Initial Release

SOFTWARE LICENSE AGREEMENT

IMPORTANT: YOU SHOULD CAREFULLY READ ALL THE TERMS, CONDITIONS AND RESTRICTIONS OF THIS LICENSE AGREEMENT BEFORE INSTALLING THE SOFTWARE PACKAGE. YOUR INSTALLATION OF THE SOFTWARE PACKAGE PRESUMES YOUR ACCEPTANCE OF THE TERMS, CONDITIONS, AND RESTRICTIONS CONTAINED IN THIS AGREEMENT. IF YOU DO NOT AGREE WITH THESE TERMS, CONDITIONS, AND RESTRICTIONS, PROMPTLY RETURN THE SOFTWARE PACKAGE AND ASSOCIATED DOCUMENTATION TO THE ABOVE ADDRESS, ATTENTION: CUSTOMER SUPPORT.

TERMS, CONDITIONS, AND RESTRICTIONS

MagTek, Incorporated (the "Licensor") owns and has the right to distribute the described software and documentation, collectively referred to as the "Software".

LICENSE: Licensor grants you (the "Licensee") the right to use the Software in conjunction with MagTek products. LICENSEE MAY NOT COPY, MODIFY, OR TRANSFER THE SOFTWARE IN WHOLE OR IN PART EXCEPT AS EXPRESSLY PROVIDED IN THIS AGREEMENT. Licensee may not decompile, disassemble, or in any other manner attempt to reverse engineer the Software. Licensee shall not tamper with, bypass, or alter any security features of the software or attempt to do so.

TRANSFER: Licensee may not transfer the Software or license to the Software to another party without the prior written authorization of the Licensor. If Licensee transfers the Software without authorization, all rights granted under this Agreement are automatically terminated.

COPYRIGHT: The Software is copyrighted. Licensee may not copy the Software except for archival purposes or to load for execution purposes. All other copies of the Software are in violation of this Agreement.

TERM: This Agreement is in effect as long as Licensee continues the use of the Software. The Licensor also reserves the right to terminate this Agreement if Licensee fails to comply with any of the terms, conditions, or restrictions contained herein. Should Licensor terminate this Agreement due to Licensee's failure to comply, Licensee agrees to return the Software to Licensor. Receipt of returned Software by the Licensor shall mark the termination.

LIMITED WARRANTY: Licensor warrants to the Licensee that the disk(s) or other media on which the Software is recorded are free from defects in material or workmanship under normal use.

THE SOFTWARE IS PROVIDED AS IS. LICENSOR MAKES NO OTHER WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Because of the diversity of conditions and PC hardware under which the Software may be used, Licensor does not warrant that the Software will meet Licensee specifications or that the operation of the Software will be uninterrupted or free of errors.

IN NO EVENT WILL LICENSOR BE LIABLE FOR ANY DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS, OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE, OR INABILITY TO USE, THE SOFTWARE. Licensee's sole remedy in the event of a defect in material or workmanship is expressly limited to replacement of the Software disk(s) if applicable.

GOVERNING LAW: If any provision of this Agreement is found to be unlawful, void, or unenforceable, that provision shall be removed from consideration under this Agreement and will not affect the enforceability of any of the remaining provisions. This Agreement shall be governed by the laws of the State of California and shall inure to the benefit of MagTek, Incorporated, its successors or assigns.

ACKNOWLEDGMENT: LICENSEE ACKNOWLEDGES THAT HE HAS READ THIS AGREEMENT, UNDERSTANDS ALL OF ITS TERMS, CONDITIONS, AND RESTRICTIONS, AND AGREES TO BE BOUND BY THEM. LICENSEE ALSO AGREES THAT THIS AGREEMENT SUPERSEDES ANY AND ALL VERBAL AND WRITTEN COMMUNICATIONS BETWEEN LICENSOR AND LICENSEE OR THEIR ASSIGNS RELATING TO THE SUBJECT MATTER OF THIS AGREEMENT.

QUESTIONS REGARDING THIS AGREEMENT SHOULD BE ADDRESSED IN WRITING TO MAGTEK, INCORPORATED, ATTENTION: CUSTOMER SUPPORT, AT THE ABOVE ADDRESS, OR E-MAILED TO support@magtek.com.

Table of Contents

SECTION 1. INTRODUCTION

This object presents the application programmer with a set of Properties, Methods, and Events which allow for complete control of all functions of the MICRImage reader. The OCX object takes care of all “low-level” tasks related to reader-specific commands, as well as initialization and setup of communication ports. Thus, the OCX object dramatically simplifies the programming process, and assists the application programmer in quickly developing a working software interface to the MICRImage reader. Also, because the OCX object is compiled as a “binary file”, it can be used as a component in almost any Windows programming environment (e.g. Visual Basic, Visual C++, Delphi, etc.)

SECTION 2. PROPERTIES

COMMPORT PROPERTY

Sets or gets the communications port number.

Syntax

MICRImage.**CommPort** [= *Value*]

The **CommPort** property syntax has these parts:

Part	**Description**
Value	An integer value specifying the port number.

Remarks

You can set value to any number between 1 and (n) at design time (the default is 1).

(n) is Operating System dependent: Windows 95 Limits the Maximum number to 16, while NT and above allow up to 255.

This property should only be changed when the CommPort is not opened (*MICRImage*.**PortOpen** = False).

Data Type

Read/Write

Integer

DSRHOLDING PROPERTY

Gets the status of the DSRHolding property.

Syntax

Value = *MICRImage*.**DSRHolding**

The **DSRHolding** property syntax has these parts:

Part	**Description**
Value	A Boolean value specifying the status.

Remarks

If the **DSRHolding** property is True and the **PortOpen** property is True then there is a MICR attached. If the MICR is disconnected then the **DSRHolding** property will be False.

Data Type

Read Only

Boolean

MICRDATA PROPERTY

Gets the raw data (MICR line or Magstripe) from the last read from the device.

Syntax

MICRImage.**MicrData** [= *value*]

The **MicrData** property syntax has these parts:

Part	**Description**
Value	A string value containing the last MICR data line read. If the Device has a Magnetic Stripe Reader, the **MicrData** Property will have the last Magnetic Stripe Data read.

Data Type

Read/Write

String

MICRTIMEOUT PROPERTY

Sets the amount of time (in seconds) to wait for a command to timeout when there is no response from the device.

Syntax

MICRImage.**MicrTimeOut** [= *value*]

The **MicrTimeOut** property syntax has these parts:

Part	**Description**
Value	A long integer containing the time out for the MICRImage.

Remarks

Legal range is 1 to 60.

Default is 2 seconds.

This property indicates how long to wait before timing out from a MICR command. In versions prior to 1.05, it was also used to indicate how long to wait for an image transfer. The image transfer now uses a watchdog timer.

Data Type

Read/Write

Long

PORTOPEN PROPERTY

Opens or closes the connection to the MICRImage.

Syntax

MICRImage.**PortOpen** [= *value*]

The **PortOpen** property syntax has these parts:

Part	**Description**
Value	A Boolean value specifying the status.

Example

MICRImage.**PortOpen** = True ‘Will open the device

MICRImage.**PortOpen** = False ‘Will close the device

Data Type

Read/Write

Boolean

SETTINGS PROPERTY

Sets or gets the baud rate, parity, data bit, and stop bit parameters.

Syntax

MICRImage.**Settings** [= *value*]

The **Settings** property syntax has these parts:

Part	**Description**
Value	A string expression representing the communications port settings, as described below.

Remarks

If value is not valid when the port is opened, the MICRImage control generates error 380 (Invalid property value).

Value is composed of four settings separated by commas:

BaudRate,Parity,DataBits,StopBits

The default value is: ’115200,N,8,1’

Data Type

Read/Write

String

SECTION 3. METHODS

ABOUT METHOD

Displays a message box containing a copyright message.

Syntax

MICRImage.**About**()

ADDTAG METHOD

Adds a TIFF tag to the Currently Scanned File.

Syntax

MICRImage.**AddTag**(*strTag* As String)

The **AddTag** method syntax has these parts:

Part	**Description**
StrTag	Tag number and data

Example

'Adding a tiff tag.

MICRImage.**AddTag**("T32768=This was added by the Demo Program")

Will insert the TAG "This was added by the Demo Program" in TAG Number 32768.

Data Type

String

CLEARBUFFER METHOD

Clears the OCX buffered MICR Data.

Syntax

MICRImage.**ClearBuffer**()

The **ClearBuffer** method syntax has no parameters.

Example

MICRImage.**ClearBuffer**()

Data Type

None

FILEMEMORYERASE METHOD

Returns Current Memory Status.

Syntax

MICRImage.**FileMemoryErase**() As String

The **FileMemoryErase** method syntax has no parameters.

Data Type

String

FILEMEMORYRESETPOINTER METHOD

Resets the Pointer to the first Image on board.

Returns Current Memory Status.

Syntax

MICRImage.**FileMemoryResetPointer**() As String

The **FileMemoryResetPointer** method syntax has no parameters.

Data Type

String

FILEMEMORYSTATUS METHOD

Returns Current Memory Status.

Syntax

MICRImage.**FileMemoryStatus**() As String

The **FileMemoryStatus** method syntax has no parameters.

Data Type

String

FINDELEMENT METHOD

Returns the specified parsed card data.

Syntax

MICRImage.**FindElement**(*TrackNum* As Integer, *RefChar* As String, *Displacement* As Integer, *NumDigits* As String, Optional *DirectionBack* As Variant = False) As String

The **FindElement** method syntax has these parts:

Part	Description
TrackNum	Track 0 = MicrData Track 1 = Mag Stripe Track 1 Track 2 = Mag Stripe Track 2 Track 3 = Mag Stripe Track 3 that uses + as a start sentinel Track 4 = Mag Stripe Track 3 that uses # as a start sentinel Track 5 = Mag Stripe Track 3 that uses ! as a start sentinel
RefChar	The Character to Find. If N characters long, then the OCX will look for the Nth instance of that Character.
Displacement	The Number of characters to displace from the *RefChar*.
NumDigits	An Integer or Character to Find to end the string.
DirectionBack	(optional) If set to True, the OCX looks Right to Left; if set to False, the OCX looks Left to Right. If not specified, the OCX looks Left to Right unless the *RefChar* specified is an End Sentinel on Magnetic Track Data ("?"), in which case the OCX looks Right to Left.

Example

Assume the **MICRData** from a Check Returned:
"T123456789T987654A1234"

MICRImage.**FindElement**(0,"T",0,"TT")
will return "123456789"

MICRImage.**FindElement**(0,"TT",0,"A")
will return "987654"

MICRImage.**FindElement**(0,"TT",0,"2")
will return "98"

Data Type

String

FORMATCHANGE METHOD

Returns "OK".

Syntax

MICRImage.**FormatChange**(*strFormat* As String) As String

The **FormatChange** method syntax has these parts:

Part	**Description**
StrFormat	See Programmers Reference Manual for a list of Format Codes and associated Outputs

Example

Response = *MICRImage*.**FormatChange**(“6200”)

Data Type

String

FORMATSHOW METHOD

Returns the Current Format of the *MICRImage.*

Syntax

MICRImage.**FormatShow**() As String

The **FormatShow** method syntax has no parameters.

Example

Response = *MICRImage*.**FormatShow**()

MsgBox Response

Data Type

String

GETDEFSETTING METHOD

Returns the Value Stored under the associated key or returns the Default value passed in. Can be used to retrieve a value located by a specific “Key”.

Syntax

MICRImage.**GetDefSetting**(ByVal *Key* As String, Optional *Default* As Variant) As String

The **GetDefSetting** method syntax has these parts:

Part	**Description**
Key	The Key being searched
Default	The Default value to return if the key does not exist

Example

Return = *MICRImage*.**GetDefSetting**(“MICRCommPort”, “1”)

Return will contain the string “1” or the string that had been previously stored using the *MICRImage*.**SaveDefSetting** command.

Data Type

String

GETFNAME METHOD

Returns the First Name from the track data.

Syntax

MICRImage.**GetFName**()

The **GetFName** method syntax has no parameters.

Remarks

Returns the First Name on an ABA type ‘A’ or’ ‘B’ formatted card or returns an empty string.

Example

txtFirstName.Text = *MICRImage*.**GetFName**()

Data Type

String

GETLNAME METHOD

Returns the Last Name from the track data.

Syntax

MICRImage.**GetLName**()

The **GetLName** method syntax has no parameters.

Remarks

Returns the Last Name on an ABA type 'A' or' 'B' formatted card or returns an empty string.

Example

txtLastName.Text = *MICRImage*.**GetLName**()

Data Type

String

GETTRACK METHOD

Returns the specified track data from the **MicrData** property.

Syntax

MICRImage.**GetTrack**(*TrackNum* as Integer) As String

The **GetTrack** method syntax has these parts:

Part	**Description**
TrackNum	An integer specifying which Track Number from **MicrData** to Return: Track 0 = MicrData Track 1 = Mag Stripe Track 1 Track 2 = Mag Stripe Track 2 Track 3 = Mag Stripe Track 3 that uses + as a start sentinel Track 4 = Mag Stripe Track 3 that uses # as a start sentinel Track 5 = Mag Stripe Track 3 that uses ! as a start sentinel

Example

txtTrack1.Text = *MICRImage*.**GetTrack**(1)

txtTrack2.Text = *MICRImage*.**GetTrack**(2)

txtTrack3.Text = *MICRImage*.**GetTrack**(3)

Data Type

String

MICRCOMMAND METHOD

Syntax

MICRImage.**MicrCommand**(ByVal *strCommand* As String, Optional *bReturnData* As Variant = True)

The **MicrCommand** method syntax has these parts:

Part	**Description**
StrCommand	The command to send the MICR
bReturnData	If the Command returns data, then set this flag to True; if the command has no response (per the programmers reference manual), then set this flag to False

Remarks

Can be used to Send Commands not specified in the OCX to the MICR.

Example

MICRImage.**MicrCommand**("SWA 00100010", False)
‘setting the SWA command has no response.

Response = *MICRImage*.**MicrCommand**("SWA", True)
‘getting the SWA command has a response.

Data Type

String

RESET METHOD

Sets the MICRImage back to its power on conditions as defined in its EEPROM.

Returns "OK".

Syntax

MICRImage.**Reset**() As String

The **Reset** method syntax has no parameters.

SAVE METHOD

Returns "OK".

Syntax

MICRImage.**Save**() As String

The **Save** method syntax has no parameters.

Remarks

Saves any changes to the MICRImage's temporary parameters to its EEPROM.

SAVEDEFSETTING METHOD

Can be used to Save a value using a Key.

Syntax

MICRImage.**SaveDefSetting**(*Key* As String, *Setting* As String) As String

The **SaveDefSetting** method syntax has these parts:

Part	**Description**
Key	The Key Identifier
Setting	The string value of the setting to Save

Example

Return = MICRImage.SaveDefSetting("MICRCommPort", "1")

This will save the string "1" under the Key named "MICRCommPort"

Data Type

String

SENDNEXTIMAGE METHOD

Returns:
SendNextImage = 0
StatusMsg "OK"

SendNextImage = -1
StatusMsg "Failed - TimeOut"

SendNextImage = -2
StatusMsg "Failed - File Exists"

SendNextImage = -3
StatusMsg "Failed - No Image Ready"

or VB Err.Number
and StatusMsg = Err. Description

Syntax

MICRImage.**SendNextImage**(ByVal *ImgFileName* As String, ByRef *StatusMsg* As String, Optional *MTParams* As Variant) As Long

The **SendNextImage** method syntax has these parts:

Part	**Description**
ImgFileName	The File name specified (can be UNC)
StatusMsg	The Status Message variable passed in. Used to return the Status message of the operation
MTParams	(Optional) can be used to specify snippets or Tags. Do not use it to override the image transfer properties MAIN or AUX or the Transfer type. These are handled by the OCX

Example

Status = *MICRImage*.**SendNextImage**(ImageFileName, StatusMsg)

Data Type

Long

STOREIMAGE METHOD

Stores an Image in the MicrImage Device

Syntax

MICRImage.**StoreImage**(Optional *strTag* As Variant) As String

The **StoreImage** method syntax has these parts:

Part	Description
StrTag	See Programmers Reference for optional **strTag** Usage

Data Type

String

VERSION METHOD

Returns the Firmware Version Number.

Syntax

MICRImage.**Version**() As String

The **Version** method syntax has no parameters.

Example

strVersion = *MICRImage*.**Version**()

Data Type

String

TRANSMITCURRENTIMAGE METHOD

Returns:
TransmitCurrentImage = 0
StatusMsg "OK"

TransmitCurrentImage = -1
StatusMsg "Failed - TimeOut"

TransmitCurrentImage = -2
StatusMsg "Failed - File Exists"

TransmitCurrentImage = -3
StatusMsg "Failed - No Image Ready"

or VB Err.Number
and StatusMsg = Err. Description

Syntax

MICRImage.**TransmitCurrentImage**(ByVal *ImgFileName* As String, ByRef *StatusMsg* As String, Optional *MTParams* As Variant) As Long

The **TransmitCurrentImage** method syntax has these parts:

Part	**Description**
ImgFileName	The File name specified (can be UNC)
StatusMsg	The Status Message variable passed in. Used to return the Status message of the operation
MTParams	(Optional) can be used to specify snippets or Tags. Do not use it to override the image transfer properties MAIN or AUX or the Transfer type. These are handled by the OCX

Example

Status = *MICRImage*.**TransmitCurrentImage**(ImageFileName, StatusMsg)

Data Type

String

ENUMTIFFTAGS METHOD

Returns a 1 based Variant Array of all the TIFF Tags in the IFD (Inter-File Directory)

Syntax

MICRImage.**EnumTiffTags**(*FileToSearch* As String, ByVal *IFDNumber* As Long) As String

The **EnumTiffTags** method syntax has these parts:

Part	**Description**
FileToSearch	The path and file name of the file to be searched (can be UNC)
IFDNumber	The Inter File Directory to be searched. Files with 1 image will have 1 IFD. Files with n Images will have n IFDs

Example

ReturnVal = *MICRImage*.**EnumTiffTags**(txtFileName.Text, txtIFD.Text)

If IsArray(ReturnVal) Then

  FieldCount = UBound(ReturnVal)

Data Type

String

GETTIFFTAGBYNUMBER METHOD

Returns a string representation of the value stored in the TIFF Tag specified or an empty String.

Syntax

MICRImage.**GetTiffTagByNumber**(*FileToSearch* As String, ByVal *TagNum* As Long, ByVal *IFDNumber* As Long) As String

The **GetTiffTagByNumber** method syntax has these parts:

Part	**Description**
FileToSearch	The path and file name of the file to be searched (can be UNC)
TagNum	The specific TagNumber to return
IFDNumber	The Inter-File Directory to search. Files with 1 image will have 1 IFD. Files with n Images will have n IFDs

Example

txtTagOutput.Text = *MICRImage*.**GetTiffTagByNumber**(txtFileName.Text, txtTagNum.Text, txtIFD.Text)

Data Type

String

GETTIFFTAGNUMBYINDEX METHOD

Returns a Long Integer containing the Tag Number specified by its Index Number (1 based) or 0 if the Tag does not exist.

Syntax

MICRImage.**GetTiffTagByNumber**(*FileToSearch* As String, ByVal *IndexNum* As Long, ByVal *IFDNumber* As Long) As String

The **GetTiffTagByNumber** method syntax has these parts:

Part	**Description**
FileToSearch	The path and file name of the file to be searched (can be UNC)
IndexNum	The specific TagIndex to return (1 Based)
IFDNumber	The Inter File Directory to be searched. Files with 1 image will have 1 IFD. Files with n Images will have n IFDs

Example

TagNum = *MICRImage*.**GetTiffTagNumByIndex**(txtFileName.Text, i, txtIFD.Text)

Data Type

String

SECTION 4. EVENTS

MICRDATARECEIVED EVENT

Fires each time a check is passed through the Check Reader or a Magnetic stripe is passed through the MagStripe Reader.

Example

```
  Private Sub LogStatus(ByVal InfoToLog As String)
    txtStatus.Text = txtStatus.Text & InfoToLog & vbCrLf
    txtStatus.SelLength = 0
    txtStatus.SelStart = Len(txtStatus.Text)
  End Sub

  Private Sub cmdClear_Click()
    txtStatus.Text = ""
    txtMagStripeData.Text = ""
    txtFirstName.Text = ""
    txtLastName.Text = ""
    txtMonth.Text = ""
    txtYear.Text = ""
    txtTrack1.Text = ""
    txtTrack2.Text = ""
    txtTrack3.Text = ""
    txtAccountNum.Text = ""
    txtMicrData.Text = ""
    txtMicrAccountNum.Text = ""
    txtCheckNum.Text = ""
    txtTransit.Text = ""

    txtFileName.Text = ""
    txtIFD.Text = ""
    txtTagNum.Text = ""
    txtTagOutput.Text = ""
    cmdGetTagByNum.Enabled = False
    cmdGetAllTags.Enabled = False

  End Sub

  Private Sub cmdExit_Click()
  Unload Me

  End Sub
```

```
Private Sub cmdGetAllTags_Click()
Dim ReturnVal As Variant
Dim FieldCount As Integer
Dim i As Integer
Dim TagNum As Long

LogStatus "Getting All Tags in file " & txtFileName.Text & ": "
ReturnVal = MICRImage.EnumTiffTags(txtFileName.Text, txtIFD.Text)

If IsArray(ReturnVal) Then
  FieldCount = UBound(ReturnVal)
  LogStatus "There are " & FieldCount & " Tags"

  For i = 1 To FieldCount
    TagNum = MICRImage.GetTiffTagNumByIndex(txtFileName.Text, i, txtIFD.Text)
    LogStatus "    Tag # " & TagNum & "= " &
MICRImage.GetTiffTagByNumber(txtFileName.Text, TagNum, txtIFD.Text)
  Next
Else
  LogStatus "There are No Tags in IFD " & txtIFD.Text & " in file " & txtFileName.Text
End If

End Sub

Private Sub cmdGetTagByNum_Click()
  LogStatus "Getting Tag By Tag Number:"
  txtTagOutput.Text = MICRImage.GetTiffTagByNumber(txtFileName.Text,
txtTagNum.Text, txtIFD.Text)
End Sub

Private Sub cmdPortOpen_Click()

If Not (MICRImage.PortOpen) Then
  MICRImage.CommPort = txtCommPort.Text
  MICRImage.Settings = txtSettings.Text
End If
MICRImage.PortOpen = Not MICRImage.PortOpen

If MICRImage.PortOpen Then

   LogStatus "Port Opened"
   cmdPortOpen.Caption = "Close Port"
   If MICRImage.DSRHolding Then
     LogStatus "Device Attached"
     'Displays Current Switch Settings
     'If you use the MICRImage.Save command then these do not need to be sent
     'every time you open the device

     MICRImage.MicrTimeOut = 1
     LogStatus "These are the Current Switch Settings"
     LogStatus "    Switch A: " & MICRImage.MicrCommand("SWA", True)
     LogStatus "    Switch B: " & MICRImage.MicrCommand("SWB", True)
     LogStatus "    Switch C: " & MICRImage.MicrCommand("SWC", True)
     LogStatus "    Switch D: " & MICRImage.MicrCommand("SWD", True)
     LogStatus "    Switch E: " & MICRImage.MicrCommand("SWE", True)
     LogStatus "    Switch I: " & MICRImage.MicrCommand("SWI", True)
     LogStatus "    Switch HW: " & MICRImage.MicrCommand("HW", True)

     'Sets Switch Settings
     'If you use the MICRImage.Save command then these do not need to be sent
     'every time you open the device

     MICRImage.MicrCommand("SWA 00100010", False)
     MICRImage.MicrCommand("SWB 00100010", False)
     MICRImage.MicrCommand("SWC 00100000", False)
     MICRImage.MicrCommand("HW 00111100", False)
     MICRImage.MicrCommand("SWE 00000010", False)
     MICRImage.MicrCommand("SWI 00000000", False)
     'MICRImage.Save

     'Displays New Settings
     'If you use the MICRImage.Save command then these do not need to be sent
     'every time you open the device
     LogStatus "These are the New Switch Settings:"
     LogStatus "    Switch A: " & MICRImage.MicrCommand("SWA", True)
     LogStatus "    Switch B: " & MICRImage.MicrCommand("SWB", True)
     LogStatus "    Switch C: " & MICRImage.MicrCommand("SWC", True)
     LogStatus "    Switch D: " & MICRImage.MicrCommand("SWD", True)
     LogStatus "    Switch E: " & MICRImage.MicrCommand("SWE", True)
     LogStatus "    Switch I: " & MICRImage.MicrCommand("SWI", True)
     LogStatus "    Switch HW: " & MICRImage.MicrCommand("HW", True)

     'The OCX will work with any Micr Format.  You just need to know which
     'format is being used to parse it using the FindElement Method
     LogStatus "Changing Format to 6200 for this Demo"
     MICRImage.FormatChange("6200")
     LogStatus "Version: " & MICRImage.Version
     MICRImage.MicrTimeOut = 5
   Else
     LogStatus "Device Not Attached"
   End If
Else
   LogStatus "Port Closed"
   cmdPortOpen.Caption = "Open Port"
End If

End Sub
```

```
Private Sub Form_Load()
txtCommPort.Text = MICRImage.GetDefSetting("CommPort", "1")
txtSettings.Text = MICRImage.GetDefSetting("Settings", "115200,N,8,1")
lblCaption(0).Caption = App.ProductName
lblCaption(1).Caption = App.ProductName
End Sub

Private Sub MicrImage1_MicrDataReceived()
Dim ImagePath As String
Dim ImageFileName As String
Dim ImageIndex As String
Dim Status As Long
Dim StatusMsg As String
Dim bOpStatus As Boolean

If MICRImage.GetTrack(1) & MICRImage.GetTrack(2) & MICRImage.GetTrack(3) <> ""
Then
  LogStatus "Event Fired: MagStripe Data"
  txtMagStripeData.Text = MICRImage.MicrData
  txtFirstName.Text = MICRImage.GetFName()
  txtLastName.Text = MICRImage.GetLName()
  txtMonth.Text = MICRImage.FindElement(2, "=", 2, "2", False)
  txtYear.Text = MICRImage.FindElement(2, "=", 0, "2", False)
  txtTrack1.Text = MICRImage.GetTrack(1)
  txtTrack2.Text = MICRImage.GetTrack(2)
  txtTrack3.Text = MICRImage.GetTrack(3)
  txtAccountNum.Text = MICRImage.FindElement(2, ";", 0, "=", False)
Else
  LogStatus "Event Fired:  MicrData"
  txtMicrData.Text = MICRImage.MicrData
  txtMicrAccountNum.Text = MICRImage.FindElement(0, "TT", 0, "A", False)
  txtCheckNum.Text = MICRImage.FindElement(0, "A", 0, "12", False)
  txtTransit.Text = MICRImage.FindElement(0, "T", 0, "TT", False)

  ImagePath = MICRImage.GetDefSetting("ImagePath", "C:\")

  'This sets up an index number so that we can deal with same check being
  'inserted over and over.  The TransmitCurrentImage Method with fail if the file
  'already exists.  This is to ensure that no check image is overwritten.  By keeping
  'an ImageIndex and incrementing it we ensure that the same file name will not be
  'generated below.  You are free to name the file anything that is considered to be
  'a valid file name.
  ImageIndex = MICRImage.GetDefSetting("ImageIndex", "0")

  ImageIndex = CStr(CInt(ImageIndex) + 1)
  MICRImage.SaveDefSetting "ImageIndex", ImageIndex

  ImageFileName = ImagePath & "MI" & txtTransit.Text & txtMicrAccountNum.Text &
txtCheckNum.Text & ImageIndex & ".TIF"
```

```
  LogStatus "Acquiring File: " & ImageFileName & " ..."

  'Adding a tiff tag.
  MICRIMage.AddTag("T32768=This was added by the Demo Program")

  'Transmitting current image
  Status = MICRImage.TransmitCurrentImage(ImageFileName, StatusMsg)

  'Logging status of TransmitCurrentImage
  LogStatus "TransmitCurrentImage: " & Status

  If Status = "0" Then
   bOpStatus = ShellEx(ImageFileName, , , , , Me.hWnd)
    If bOpStatus = True Then
      LogStatus "Shell Successful"

      'setting up the Image Info
      txtFileName.Text = ImageFileName
      txtIFD.Text = "1"
      txtTagNum.Text = "270"
      cmdGetTagByNum.Enabled = True
      cmdGetAllTags.Enabled = True
    Else
      LogStatus "Shell Failed"
      cmdGetTagByNum.Enabled = False
      cmdGetAllTags.Enabled = False
    End If
  End If
End If

MICRImage.ClearBuffer
End Sub

Private Sub mnuAbout_Click()
frmAbout.Show vbModal

End Sub

Private Sub mnuExit_Click()
Unload Me
End Sub
```