



## **USER MANUAL**

# **SecureMag Encrypted Magstripe Reader**

## **OPOS Reference Guide**

80096503-001  
Rev A. 10/07/09

## **ID TECH SOFTWARE LICENSE AGREEMENT**

ID TECH ("LICENSOR") IS WILLING TO LICENSE THIS SOFTWARE TO YOU ONLY IF YOU ACCEPT ALL OF THE TERMS IN THIS LICENSE AGREEMENT. PLEASE READ THE TERMS CAREFULLY BEFORE YOU AGREE BECAUSE YOU WILL BE BOUND BY THE TERMS OF THIS AGREEMENT. IF YOU DO NOT AGREE TO THESE TERMS, LICENSOR WILL NOT LICENSE THIS SOFTWARE TO YOU.

### Ownership of the Software

1. The Licensor software program ("Software") and any accompanying written materials are owned by Licensor [or its suppliers] and are protected by United States copyright laws, by laws of other nations, and by international treaties.

### Grant of License

2. Licensor grants the right to use the Software in conjunction with an ID TECH product. You may load one copy into permanent memory of one computer and may use that copy only on that same computer.

### Restrictions on Use and Transfer

3. The Software may not be copied, except that (1) one copy of the Software may be made solely for backup or archival purposes, and (2) the Software may be transfer to a single hard disk provided the original is kept solely for backup or archival purposes. The written materials may not be copied.

4. The Software may be permanently transferred and any accompanying written materials (including the most recent update and all prior versions) if no copies are retained and the transferee agrees to be bound by the terms of this Agreement. Such a transfer terminates your license. The software may not be rented or leased or otherwise transferred or assigned the right to use the Software, except as stated in this paragraph.

5. The software may not be reverse engineered, decompiled, or disassembled.

### **Limited Warranty**

6. If used in conjunction with an ID TECH product, Licensor warrants that the Software will perform substantially in accordance with the accompanying written materials for a period of 90 days from the date of your receipt of the Software. Any implied warranties on the Software are limited to 90 days. Some states and territories do not allow limitations on duration of an implied warranty, so the above limitation may not apply to you.

7. LICENSOR DISCLAIMS ALL OTHER WARRANTIES, EITHER EXPRESS OR

IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT, WITH RESPECT TO THE SOFTWARE AND ANY ACCOMPANYING WRITTEN MATERIALS. This limited warranty gives you specific legal rights. You may have others, which vary from state to state.

8. LICENSOR'S ENTIRE LIABILITY AND YOUR EXCLUSIVE REMEDY SHALL BE REPLACEMENT OF THE SOFTWARE THAT DOES NOT MEET LICENSOR'S LIMITED WARRANTY. Any replacement Software will be warranted for the remainder of the original warranty period or 30 days, whichever is longer.

9. This Limited Warranty is void if failure of the Software has resulted from modification, accident, abuse, or misapplication.

10. IN NO EVENT WILL LICENSOR BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY LOSS OF PROFITS, LOST SAVINGS, OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF YOUR USE OR INABILITY TO USE THE SOFTWARE. Because some states do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitation may not apply you.

11. This Agreement is governed by the laws of the state of California.

12. For any questions concerning this Agreement or to contact Licensor for any reason, please write: International Technologies & Systems Corporation, 10721 Walker Street, Cypress, CA 90630 or call (714) 761-6368.

13. U.S. Government Restricted Rights. The Software and documentation are provided with Restricted Rights. Use, duplication, or disclosure by the Government is subject to restrictions set forth in subparagraph (c)(1) of The Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 or subparagraphs (c)(1)(ii) and (2) of Commercial Computer Software - Restricted Rights at 48 CFR 52.227-19, as applicable. Supplier is ID TECH, 10721 Walker Street, Cypress, CA 90630.

Copyright 2009, International Technologies & Systems Corporation. All rights reserved. ID TECH is a registered trademark of International Technologies & Systems Corporation. Value through Innovation is a trademark of International Technologies & Systems Corporation.

### Revision History

Revision	Date	Description	By
A	10/07/2009	Initial Release	JW

### Table of Contents

<b>1</b>	<b>Description.....</b>	<b>5</b>
<b>2</b>	<b>Methods, Properties and Events of MSR.....</b>	<b>6</b>
2.1	Methods of MSR.....	6
2.2	Properties of MSR.....	8
2.3	Events of MSR.....	14
<b>3</b>	<b>Programming Examples.....</b>	<b>16</b>
3.1	Visual C++ 6.0 Programming Example.....	16
3.2	Visual Basic 6.0 Programming Example.....	19
3.3	Visual Studio 2005 C# Programming Example.....	19
<b>4</b>	<b>Result Code/Error Code List.....</b>	<b>19</b>

## 1 Description

The documentation describes the properties, methods, and events of the ID TECH SecureMag MSR OPOS component. The component includes two parts: a Control Object running on the upper level, which is an ActiveX control, and a Service Control running on the lower level, which is an OLE automation server. The properties, methods, and events are exposed by the Control Object. When the Control Object is imported into a project as an ActiveX control, you can see all the properties, methods, and events.

Target Device:

ID TECH SecureMag Encrypted MSR, USB-HID Interface

Applicable Platforms:

Microsoft Windows XP, 2000, 98, Vista

Service Object and Control Object:

Service Object Version: 1.10.000

Control Object Version: 1.10.000

Dll File Version: 2.75

## 2 Methods, Properties and Events of MSR

This section describes the methods, properties, and events for the Encrypted MSR.

### 2.1 Methods of MSR

These function declarations may be different when the Control Object (OPOSMSR.OCX) is imported into your application project. Please refer to the UnifiedPOS Specification for more detailed information on the Control Object.

#### 1) Open

**Syntax**     **LONG Open (BSTR DeviceName);**

**Remarks**    Call to open a device for subsequent I/O.

This method finds more parameters in the Windows Register Tables on key or subkeys: HKEY\_LOCAL\_MACHINE\Software\OLEforRetail\ServiceOPOS\MSR\IDTECH\_EM\_USBHID

#### USB HID interface:

Subkey: Connector:

Key value name: CONNECTOR

Key value: USBHID/0acd/2010

First field USBHID specify the type of the connector. 0acd is the USB device vendor ID, 2010 is the reader product ID for USB HID connector device.

#### 2) ClaimDevice

**Syntax**     **LONG ClaimDevice (LONG Timeout);**

**Remarks**    Call this method to request exclusive access to the device. The reader requires an application to claim it before it can be used.

#### 3) CheckHealth

**Syntax**     **LONG CheckHealth (LONG Level);**

**Remarks**    Called to test the state of the device.

#### Description

When CH\_INTERNAL is selected, SO returns the firmware version of the MSR device or returns a failure. CheckHealthText property will be "Internal HCheck: Successful" if the firmware version is read successfully.

When CH\_EXTERNAL is selected, SO displays a dialog for asking users to swipe a card. Once the card is swiped, the "Real data" of the card, including Start and End Sentinel, are shown in the dialog window. After the window is closed, CheckHealth Text property will be "External HCheck:: HCheck: Complete".

When CH\_INTERACTIVE is selected, SO will display a card swipe dialog with the firmware version shown. After the dialog is closed, CheckHealthText property would show “External HCheck:: HCheck: Complete”.

#### 4) ClearInput

**Syntax**     **LONG ClearInput ();**

**Remarks**    Called to clear all device input that has been buffered.

#### 5) DirectIO

**Syntax**     **LONG DirectIO (LONG Command, LONG\* pData, BSTR\* pString);**

**Remarks**    Call to communicate directly with the Service Object.

**Description** Currently this is partially implemented and to be improved in the next release.

#### 6) ReleaseDevice

**Syntax**     **LONG ReleaseDevice ();**

**Remarks**    Call this method to release exclusive access to the device.

#### 7) Close

**Syntax**     **LONG Close ();**

**Remarks**    Called to release the device and its resources.

#### 8) ClearInputProperties

**Syntax**     **void ClearInputProperties();**

**Remarks**    Sets all data properties that were populated as a result of firing a DataEvent or ErrorEvent back to their default values.

## 2.2 Properties of MSR

For detailed information, please refer the UnifiedPOS specification.

NOTE: CO --- Control Object

SO --- Service Object

AP or App --- the abbreviation of Application.

### Property Group1---Description

<i>Name</i>	<i>Type</i>	<i>Mutability</i>	<i>Use After</i>	<i>Description</i>	<i>Support?</i>
<b>DeviceControlDescription</b>	String	read-only	--	Identify the Control Object and the company that produced it	Yes
<b>DeviceControlVersion</b>	<i>int32</i>	read-only	--	hold the Control Object version number.	Yes
<b>DeviceServiceDescription</b>	<i>String</i>	read-only	open	identify the Service Object supporting the device and the company that produced it	Yes
<b>DeviceServiceVersion</b>	<i>int32</i>	read-only	open	hold the Service Object version number.	Yes
<b>PhysicalDeviceDescription</b>	<i>string</i>	read-only	open	identify the device and any pertinent information about it.	Yes
<b>PhysicalDeviceName</b>	<i>string</i>	read-only	open	identify the device and any pertinent information about it.	Yes

**Property Group2---Control**

<i>Name</i>	<i>Type</i>	<i>Mutability</i>	<i>Use After</i>	<i>Description</i>	<i>Support?</i>
<b>Claimed</b>	<i>Boolean</i>	read-only	open	The device must be claimed for exclusive use before access its methods and properties, and before any events to be fired. It is initialized to FALSE by the <b>Open</b> method. It is set to TRUE after the method <b>Claim</b> is successfully called.	Yes
<b>AutoDisable</b>	<i>Boolean</i>	read-write	open	When TRUE, as soon as an event <b>DataEvent</b> is received, then <b>DeviceEnabled</b> is automatically to FALSE. It is initialized to FALSE by the <b>Open</b> method.	Yes
<b>DeviceEnabled</b>	<i>Boolean</i>	read-write	open& claim	When FALSE, the reader has been disabled and any subsequent input will be discarded (No DataEvent could be received even if the card is swiped). It is initialized to FALSE by the <b>Open</b> method.	Yes
<b>FreezeEvents</b>	<i>boolean</i>	read-write	open	When TRUE, events are not required to be delivered and will be held by SO until events are unfrozen. It is initialized to FALSE by the <b>Open</b> method.	Yes
<b>DataEventEnabled</b>	<i>boolean</i>	read-write	open	When TRUE, a <b>DataEvent</b> or <b>ErrorEvent</b> will be delivered immediately when had. (Of course , <b>FreezeEvents=FALSE and DeviceEnabled=TRUE</b> is a prerequisite). It is initialized to FALSE by the <b>Open</b> method.	Yes
<b>CapPowerReporting</b>	<i>int32</i>	read-only	open	Identifies the reporting capabilities of the device about Power. It seems that reader doesn't support in the hardware.	No
<b>PowerNotify</b>	<i>int32</i>	read-write	open	Contains the type power notification selection made by the Application. is initialized to OPOS_PN_DISABLED by the <b>Open</b> method.	No
<b>PowerState</b>	<i>int32</i>	read-only	open	Contains the current power condition. It seems that the reader doesn't	No

**SecureMag OPOS User Manual**

				support in the hardware.	
<b>State</b>	<i>int32</i>	Read-only	--	Contains the current state of the Control. It can be set to one of the four values: Closed, Idle, Busy, or Error.	Yes
<b>DataCount</b>	<i>int32</i>	Read-only	open	Holds the number of enqueued <b>DataEvents</b> remained in the queue.	Yes
<b>CheckHealthText</b>	<i>string</i>	read-only	open	Holds the results of the most recent call to the <b>CheckHealth</b> method. Before the first <b>CheckHealth</b> method call, its value is uninitialized.	Yes

**Property Group3---Track Control**

<i>Name</i>	<i>Type</i>	<i>Mutability</i>	<i>Use After</i>	<i>Description</i>	<i>Support?</i>
<b>CapISO</b>	<i>boolean</i>	read-only	open	If TRUE, the reader supports ISO cards.	Yes
<b>CapJISOne</b>	<i>boolean</i>	read-only	open	If TRUE, the reader supports JIS Type-I cards. JIS-I cards are a superset of ISO cards. Therefore, if <b>CapJISOne</b> is true, it is implied that <b>CapISO</b> is also TRUE.	Yes
<b>CapJISTwo</b>	<i>boolean</i>	read-only	open	If TRUE, the reader supports JIS type-II cards.	Yes
<b>CapTransmitSentinels</b>	<i>boolean</i>	read-only	open	If TRUE, the reader is able to transmit the start and end sentinels. e.g. start sentinel could be ‘%’ or ‘;’, and stop sentinel could be ‘?’.	Yes
<b>DecodeData</b>	<i>boolean</i>	read-write	open	If TRUE, each byte of track data properties is mapped from its original encoded bit sequence (as it exists on the magnetic card) to its corresponding decoded ASCII bit sequence.	Yes
<b>ParseDecodeData</b>	<i>boolean</i>	read-write	open	When TRUE, the decoded data contained within the <b>Track1Data</b> and <b>Track2Data</b> properties is further separated into fields for access via various other properties. If <b>DecodeData=FALSE</b> , <b>ParseDecodeData must be false</b> .	Yes
<b>TransmitSentinels</b>	<i>boolean</i>	read-write	open	If TRUE, the <b>Track1Data</b> , <b>Track2Data</b> , <b>Track3Data</b> , and <b>Track4Data</b> properties contain start	Yes

**SecureMag OPOS User Manual**

				and end sentinel values. Otherwise only the track data between these sentinels.	
<b>TracksToRead</b>	<i>int32</i>	read-write	open	Indicate which track data that the App wishes to get following a card sweep.	Yes
<b>ErrorReportingType</b>	<i>int32</i>	Read-write	open	Holds the type of errors to report via <b>ErrorEvents</b> . This property has one of the following values: MSR_ERT_CARD or MSF_ERT_TRACK	Yes

**Property Group4---TrackData**

<i>Name</i>	<i>Type</i>	<i>Mutability</i>	<i>Use After</i>	<i>Description</i>	<i>Support?</i>
<b>Track1Data</b>	<i>binary</i>	read-only	open	Holds the track 1 data obtained from the most recently swept card. If <b>DecodeData</b> is true, then it has been decoded from the “raw” format. it may also be parsed into other properties when the <b>ParseDecodeData</b> property is set.	Yes
<b>Track1DiscretionaryData</b>	<i>binary</i>	read-only	open	Holds the track 1 discretionary data obtained from the most recently swept card. It may be NULL when: <b>1)</b> The field was not included in the track data obtained, or, <b>2)</b> The track data format was not supported, <b>3)</b> <b>ParseDecodeData</b> is false.	Yes
<b>Track2Data</b>	<i>binary</i>	read-only	open	Holds the track 2 data obtained from the most recently swept card. If <b>DecodeData</b> is true, then it has been decoded from the “raw” format. it may also be parsed into other properties when the <b>ParseDecodeData</b> property is set.	Yes
<b>Track2DiscretionaryData</b>	<i>binary</i>	read-only	open	Holds the track 2 discretionary data obtained from the most recently swept card. It may be NULL when: <b>1)</b> The field was not included in the track data obtained, or, <b>2)</b> The track data format was not supported, <b>3)</b> <b>ParseDecodeData</b> is false.	Yes

**SecureMag OPOS User Manual**

<b>Track3Data:</b>	<i>binary</i>	read-only	open	Holds the track 3 data obtained from the most recently swept card.	Yes
<b>Track4Data</b>	<i>binary</i>	read-only	open	Holds the track 4 data (JIS-II) obtained from the most recently swept card.	Yes

**Property Group5---ParsedData**

<i>Name</i>	<i>Type</i>	<i>Mutability</i>	<i>Use After</i>	<i>Description</i>	<i>Support?</i>
<b>AccountNumber</b>	<i>string</i>	read-only	Open	Holds the account number obtained from the most recently swept card. it is initialized to NULL if: <b>1)</b> The field was not included in the track data obtained, or, <b>2)</b> The track data format was not supported, or, <b>3)</b> <b>ParseDecodeData</b> is false.	Yes
<b>ExpirationData</b>	<i>string</i>	read-only	Open	Holds the expiration date obtained from the most recently swept card. Others are same as <b>AccountNumber</b> .	Yes
<b>FirstName</b>	<i>string</i>	read-only	Open	Holds the first name obtained from the most recently swept card. Others are same as <b>AccountNumber</b> .	Yes
<b>MiddleInitial</b>	<i>string</i>	read-only	Open	Holds the middle initial obtained from the most recently swept card. Others are same as <b>AccountNumber</b> .	Yes
<b>Surname</b>	<i>string</i>	read-only	Open	Holds the surname obtained from the most recently swept card. Others are same as <b>AccountNumber</b> .	Yes
<b>Title</b>	<i>string</i>	read-only	Open	Holds the title obtained from the most recently swept card.. Others are same as <b>AccountNumber</b> .	Yes
<b>Suffix</b>	<i>string</i>	read-only	Open	Holds the suffix obtained from the most recently swept card.. Others are same as <b>AccountNumber</b> .	Yes
<b>ServicCode</b>	<i>string</i>	read-only	Open	Holds the service code obtained from the most recently swept card. Others are same as <b>AccountNumber</b> .	Yes

**Property Group6--- Statistic**

<i>Name</i>	<i>Type</i>	<i>Mutability</i>	<i>Use After</i>	<i>Expected Result</i>	<i>Test Result</i>
<i>CapStatisticsReporting</i>	<i>boolean</i>	read-write	Open	If true ,the SO can get device information to a XML statistics	No
<i>CapUpdateStatistics</i>	<i>boolean</i>	read-write	Open	If true ,the SO can update the XML statistics	No

**Property Group7---Firmware**

<i>Name</i>	<i>Type</i>	<i>Mutability</i>	<i>Use After</i>	<i>Expected Result</i>	<i>Test Result</i>
<i>CapCompareFirmwar eVersion</i>	<i>boolean</i>	read-write	Open	If true ,the SO can compare the Firmware version	No
<i>CapUpdateFirmware</i>	<i>boolean</i>	read-write	Open	If true ,the SO can update the firmware of the device	No
<i>CapWritableTracks</i>	<i>Int32</i>	Read_only	Open	This capability indicates if the MSR device supports the writing of track data - and which tracks are supported.	NONE
<i>EncodingMaxLength</i>	<i>Int32</i>	Read_only	Open	The maximum length of data that can be written by the MSR to the track(s) .	NO
<i>TracksToWrite</i>	<i>Int32</i>	Read-Write	Open	Holds the MSR track(s) that will be written.	NO

## 2.3 Events of MSR

These events are fired by the Service Object when it is necessary. The following functions are, in fact, the event-handlers that can be added into the applications. Then the applications can receive these events and do some processing accordingly. Please refer to the UnifiedPOS specification for detailed information.

### 1) DataEvent

**Syntax**     **void DataEvent (LONG Status);**

The *Status* parameter contains the input status. Its value is Control-dependent, and may describe the type or qualities of the input.

**Remarks**    Fired to present input data from the device to the application.

#### Description

a **DataEvent** can be received when a magnetic card is swiped if the three conditions are all met:

- 1) **DeviceEnabled** = TRUE
- 2) **FreezeEvents** = FALSE
- 3) **DataEventEnabled** = TRUE.

The track data can be obtained, and the parsed data can also be obtained if **ParseDecodeData** is TRUE.

### 2) DirectIO Event

**Syntax**     **void DirectIOEvent (LONG EventNumber, LONG\* pData, BSTR\* pString);**

#### Parameter Description

*EventNumber*    Event number. Specific values are assigned by the Service Object.

*pData*            Pointer to additional numeric data. Specific values vary by *EventNumber* and the Service Object.

*pString*          Pointer to additional string data. Specific values vary by *EventNumber* and the Service Object.

**Remarks**    Fired by a Service Object to communicate directly with the application.

**Description** The event **DirectIOEvent** is used for special communication between one SO and an application. Currently it is implemented incompletely.

### 3) Error Event

**Syntax**    **void ErrorEvent (LONG *ResultCode*, LONG *ResultCodeExtended*,  
LONG *ErrorLocus*, LONG\* *pErrorResponse*);**

**Parameter Description**

*ResultCode*            Result code causing the error event. See **ResultCode** for values.

*ResultCodeExtended*    Extended result code causing the error event. See **ResultCodeExtended** for values.

*ErrorLocus*            Location of the error. See values below.

*PErrorResponse*        Pointer to the error event response. See values below.

when **ErrorReportingType** property is MSR\_ERT\_TRACK, and *ErrorCode* is E\_EXTENDED, then *ErrorCodeExtended* contains Track-level status, broken down as the followings:

Byte3	Byte2	Byte1	Byte0
Track 4	Track 3	Track 2	Track 1

**Remarks** Fired when an error is detected and the Control's **State** transitions into the error state.

### 3 Programming Examples

There are three simple programming examples provided in this section including VC++6.0, VB6.0, and VS2005 C#. In each of the example it demonstrates how to read the track 1 data from a card.

In general, there are two steps to work with the OPOS control object:

1. Insert the OPOS Control Object (CO) into the project
2. Add an event handle

#### 3.1 Visual C++ 6.0 Programming Example

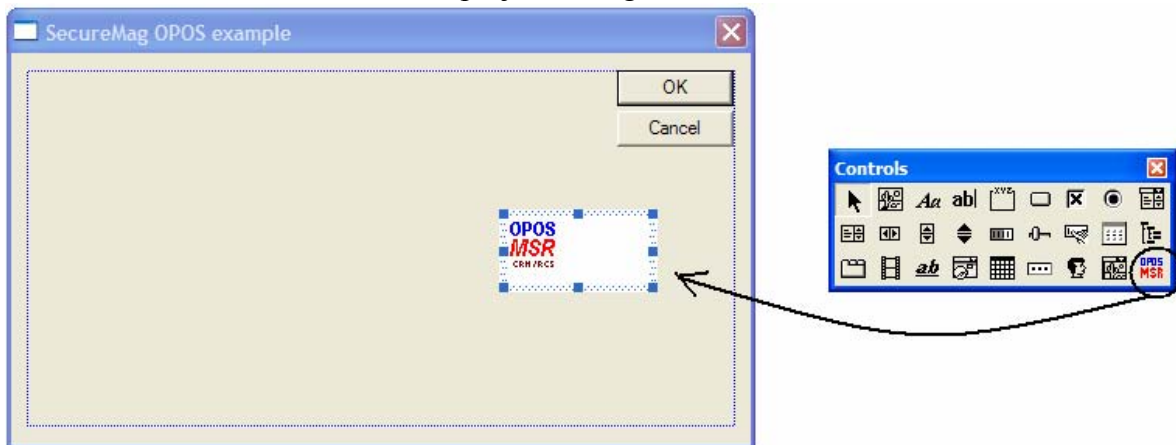
Programming Environment:

Windows XP Professional, Visual C++ 6.0, OPOS CO 1.10, ID TECH SO 1.10

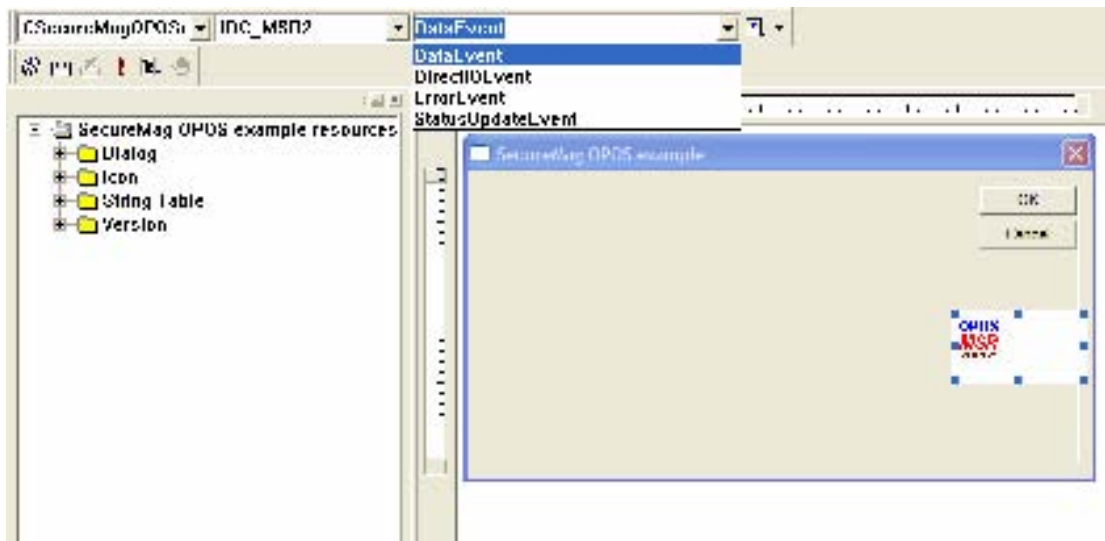
1. Install the OPOS driver downloaded from the ID TECH website ([www.idtechproducts.com](http://www.idtechproducts.com)). Make sure the OPOS demo is functioning on the reader.
2. In Visual C++, create a Dialog Based MFC application using MFC Application Wizard with ActiveX supports.
3. Go to Project->Add to Project->Components and Controls. From the “Registered ActiveX Controls” folder, select “OPOS MSR Control 1.9.00” and insert this ActiveX control into the project. An icon for OPOS MSR will be added in the Controls toolbar.



4. Add the OPOS MSR CO to the project dialog.

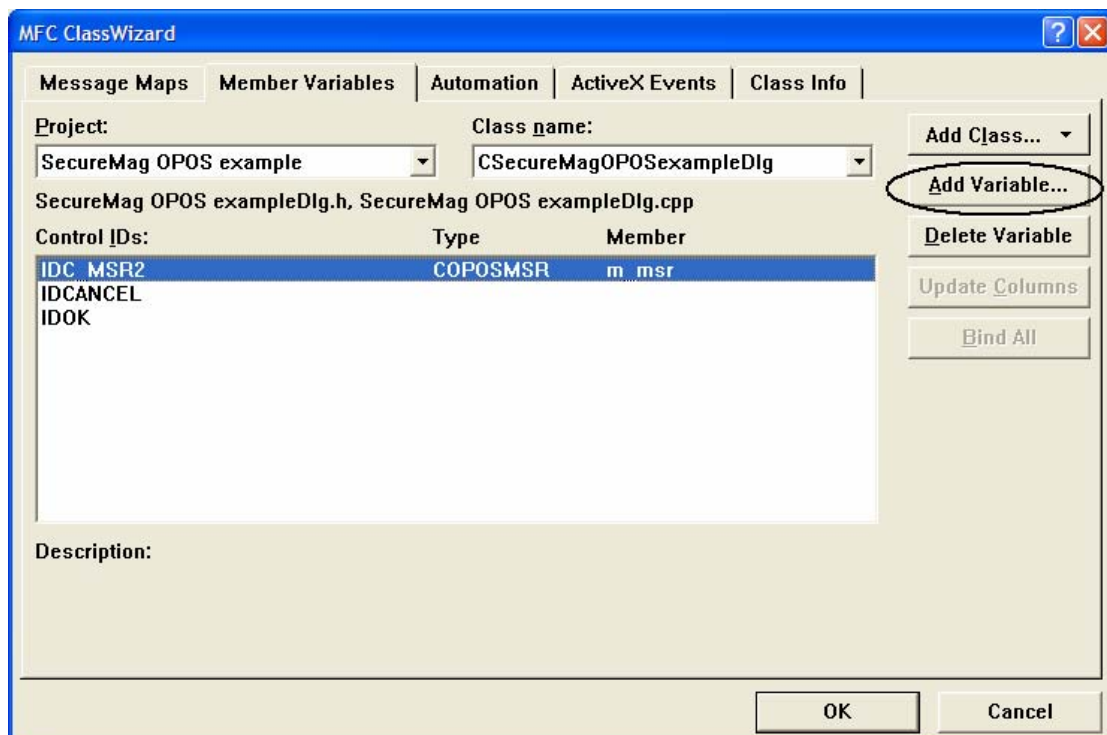


5. Add DataEvent and ErrorEvent handle



```
void CSecureMagOPOSexampleDlg::OnDataEventMsrl(long Status)
```

6. Go the View->ClassWizard and select the “Member Variables” tab. Select IDC\_MSR and add a member variable of type “COPOSMSR”, name it *m\_msr*.



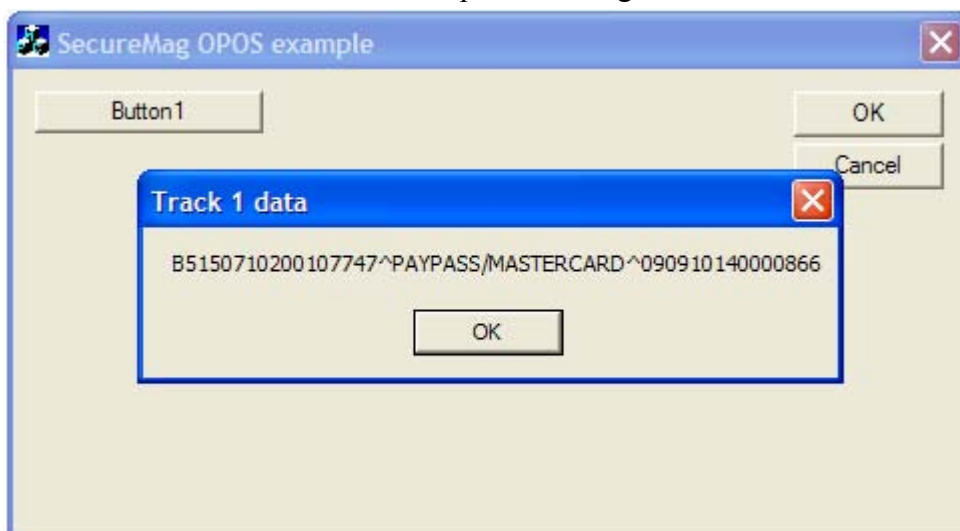
7. Create a button on the form and add the following initialization code:

```
void CMfc_diagDlg::OnButton1()
{
    if (m_msr.Open("IDTECH_EM_USBHID") == 0 /*OPOS_SUCCESS*/)
    {
        m_msr.ClaimDevice(100);
        m_msr.SetDeviceEnabled(TRUE);
        m_msr.SetDataEventEnabled(TRUE);
    }
    else {
        // error handling code...
    }
}
```

8. Add code for DataEvent handle

```
void CMfc_diagDlg::OnDataEventMsr1(long Status)
{
    MessageBox(m_msr.GetTrack1Data(), "Track 1 data");
    m_msr.SetDataEventEnabled(TRUE); // prepare the next
    event.
}
```

9. Compile and run the program. Click on "Button1" to initialize the reader and swipe a card. Track 1 data will show up in a message box.



### 3.2 Visual Basic 6.0 Programming Example

Programming Environment:

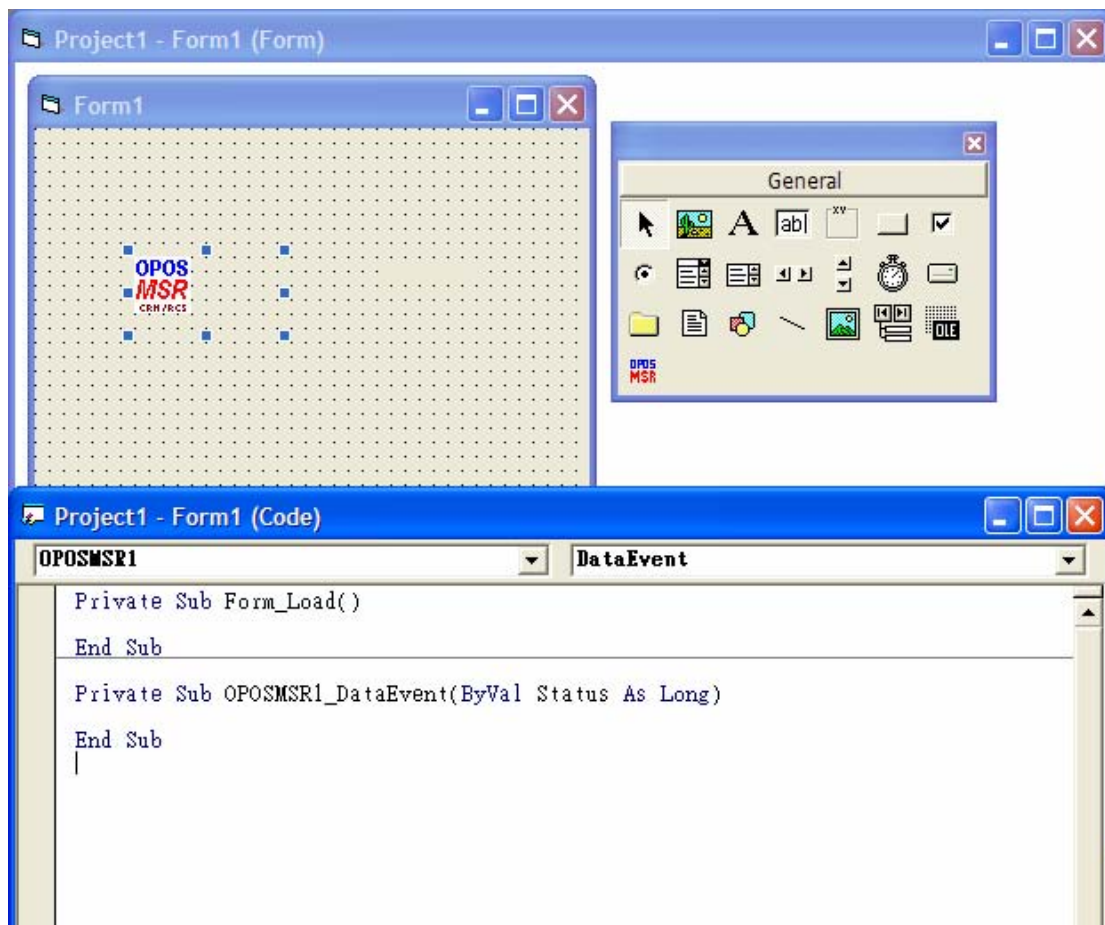
Windows XP Professional, Visual Basic 6.0, OPOS CO 1.10, ID TECH SO 1.10

1. Create a new project of type “Standard EXE”.
2. From Project->Components, select “OPOS MSR Control 1.10.000” and click “apply”.

The OPOS MSR icon will be added to the control toolbar.



3. Add an OPOS MSR control to the form. Double click on the control to add “DataEvent” handle.



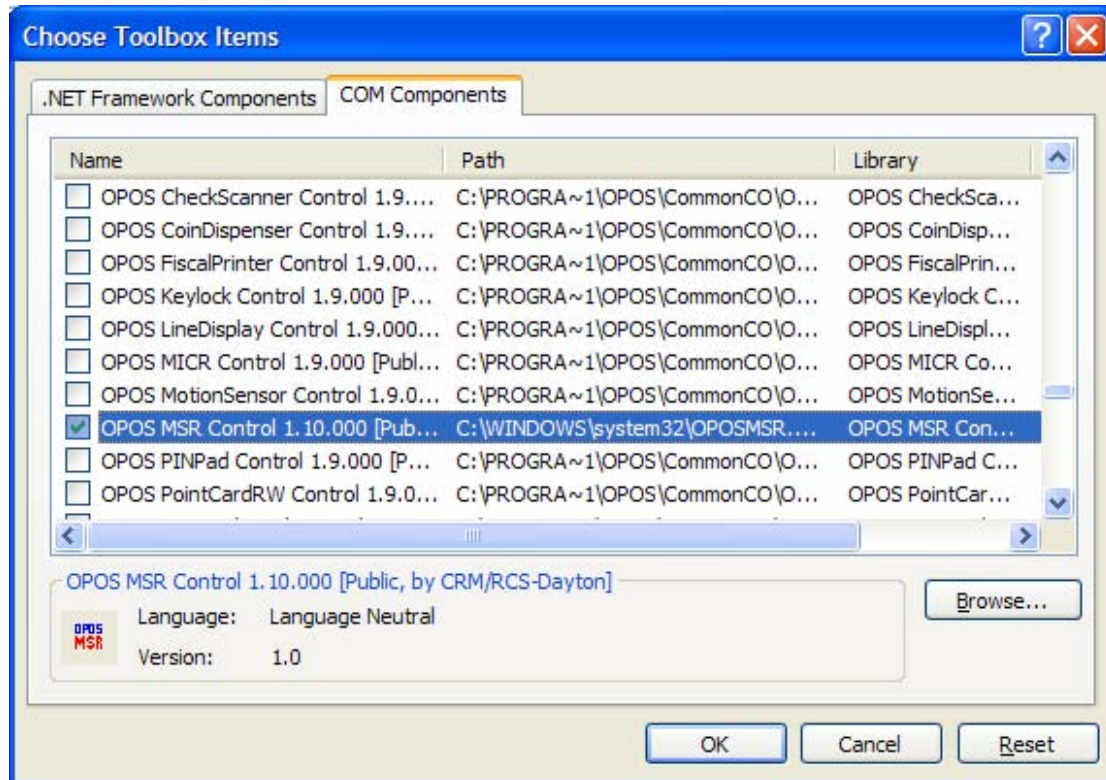


### 3.3 Visual Studio 2005 C# Programming Example

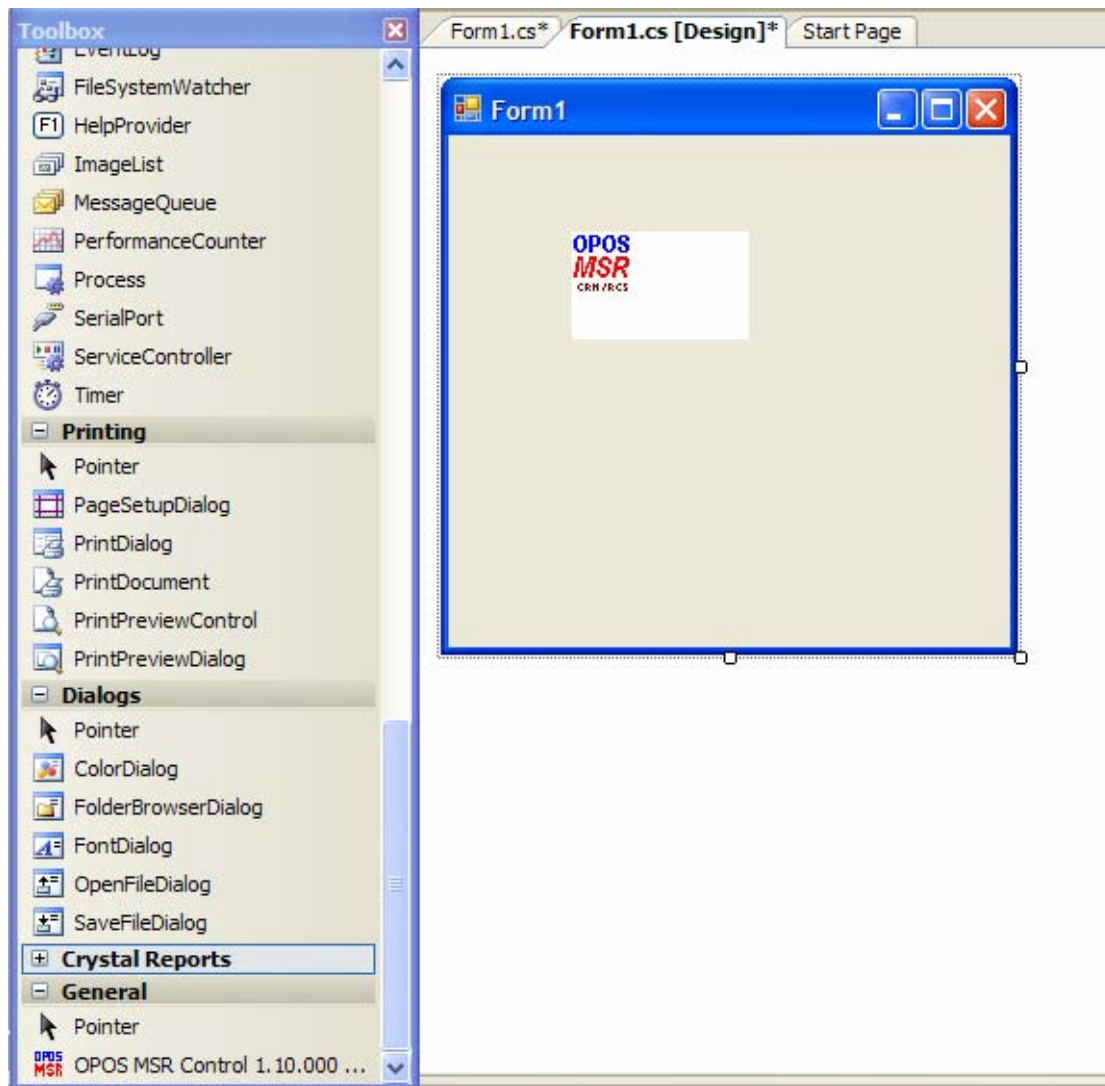
Programming Environment:

Windows XP Professional, Visual Studio 2005 C# OPOS CO 1.10, ID TECH SO 1.10

1. Create a “Windows Application” Project.
2. Right click on the “Toolbox” tool bar, select “Choose item ...”. Under “COM Components” tab, select “OPOS MSR Control 1.10.000” and click okay.



## 3. Add “OPOS MSR Control” to “Form1”



## 4. Double click on the OPOS MSR Control to add DataEvent handler code:

```

private void Form1_Load(object sender, EventArgs e)
{
    if (axOPOSMSR1.Open("IDTECH_MMII_USBHID") == 0 /*
OPOS_SUCCESS */)
    {
        axOPOSMSR1.ClaimDevice(100);
        axOPOSMSR1.DeviceEnabled = true;
        axOPOSMSR1.DataEventEnabled = true;
    }
}

private void axOPOSMSR1_DataEvent(object sender,
AxOposMSR_1_9_Lib._IOPOSMSREvents_DataEventEvent e)
{

```

