/**********************************************************************

# Magnetic Strip Reader DLL User Guide

First Written in : 2009-4-10

Revised in : 2010-11-18

Revised in : 2011-05-11

**********************************************************************/

## Content：

1. Development Environment

2. Usage

3. Execution

4. API Reference

5. Program Flow Sheet

6.Example for VS 2005

## 1. Development Environment

(1) Install the provided SDK，OT110_SDK.msi .
(2) Execute Microsoft VS2005 Visual C++.
(3) Select “OT110_SDK” as the target of your application.
(4) Add “MSR_DLL.lib” in to the project.
(5) Include “MSR_DLL.h” in your source files.

## 2. Usage

(1) Library header
: “MSR_DLL.h” is the library header file. To use the Magnetic Stripe Reader(MSR) library, include the header file in your source header files.
(2) MSR API
: Call the API functions provided by the MSR library.
(3) Note
:Click “Start” >“Settings” “Control Panel”,double-click “Setting_Manager”;
<1>.Item “Modules” >“Enable MSR key wedge” should be not selected if user would use MSR modules individual.

## 3. Execution

(1) Copy the provided MSR_DLL.dll to the OT110 device. The location can be the same directory where an application exists or “＼Windows” directory.
(2) Copy the application that uses the MSR library in the device and execute it.

## 4. API Reference

[1] CODEAPI UINT WINAPI MSROpen(HWND);
Description :
Turns on the MSR power, opens the serial port and sets the window handle to receive data.
Return Values :
MSR_FAIL : Failed to open MSR reader
MSR_OK : Success
Parameter :
HWND: [in] the handle of the window to receive data

---

[2] CODEAPI UINT WINAPI MSRClose(void);
Description :
Turns off the MSR power, closes the serial port and releases the window handle.
Return Values :
MSR_FAIL : Failed to close the Card
MSR_OK : Success

---

[3] CODEAPI UINT WINAPI MSRBeepOn(BOOL bBeep);

Description :

Sets the beep sound to notify successful reading of the card data.

Return Value :

MSR_FAIL : Failed to on/off the beep sound..

MSR_OK : Success

MSR_ERROR_NOT_OPENED : The MSR has not been Opened.

Parameter :

BOOL bBeep: [in] TRUE Set the beep sound

FALSE Clear the beep sound

---

[4] CODEAPI UINT WINAPI MSRRead(int CardType, char *a_pData, int length);

Description :

Reads the card data.

Return Value :

MSR_ERROR_NOT_OPENED : The MSR has not been Opened.

UINT : the actual length of the data was read.

Parameters :

int CardType : [in] Track number

TRK_1, read track 1 data

TRK_2, read track 2 data

TRK_3, read track 3 data

ALL_DATA, read all data,include track1,track2,track3

char *a_pData : [in,out] pointer to the buffer which is used for storing the data you get from the card.

int length : [in] the length of the buffer which is used for storing the data you get from the card.

---

[5] CODEAPI UINT WINAPI MSRReadString(int CardType, TCHAR *a_tData, int length);

Description :

Reads the card data.

Return Value :

MSR_ERROR_NOT_OPENED : The MSR has not been Opened.

UINT : the actual length of the data was read.

Parameters :

int CardType : [in] Track number

TRK_1, read track 1 data

TRK_2, read track 2 data

TRK_3, read track 3 data

ALL_DATA, read all data,include track1,track2,track3

TCHAR *a_tData : [in,out] pointer to the buffer which is used for storing the data get from the card. **The buffer length of the user based on**

**their actual conditions, the general TRK_1 not less than 79, TRK_2 not less than 40, TRK_3 not less than 107, ALL_DATA not less than 226.**

int length : [in] the length of buffer which is used for storing the data get from the card.

---

## 5. Application Flow：

**1) Read card**

step1: Open the card

MSROpen(GetSafeHwnd());

step2: Wait message

WindowProc(UINT message,...);

a. track 1 data received

WM_MSR_TRACK_1_RECEIVE:

read track 1 data

MSRReadString(TRK_1,…,…);

b. track 2 data received

WM_MSR_TRACK_2_RECEIVE:

read track 2 data

MSRReadString(TRK_2,…,…);

c. track 3 data received

WM_MSR_TRACK_3_RECEIVE:

read track 3 data

MSRReadString(TRK_3,…,…);

d. all tracks data received

WM_COMM_RXCHAR_COMPLETE:

read all tracks data

MSRReadString(ALL_DATA,…,…);

e. fail to read track data

WM_MSR_ERROR_READ:

step3: Close the card

CardClose();

**2) Set beep**

step1: Open the card

MSROpen(GetSafeHwnd());

step2: Turn on the beep

MSRBeepOn(TRUE) ;

Turn off the beep

MSRBeepOn(FALSE);

step3: Close the card

MSRClose();

## 6. Example for VS2005 VC.

```
void CMSRTestProgramDlg::OnDestroy()
{
	MSRClose();
	CDialog::OnDestroy();
}

void CMSRTestProgramDlg::OnOpen()
{
	MSROpen(GetSafeHwnd());
}

void CMSRTestProgramDlg::OnClose()
{
	MSRClose();
}

void CMSRTestProgramDlg::OnBeepOn()
{
	MSRBeepOn(TRUE);
}

void CMSRTestProgramDlg::OnBeepOff()
{
	MSRBeepOn (FALSE);
}



LRESULT CMSRTestProgramDlg::WindowProc(UINT message, WPARAM
wParam, LPARAM lParam)
{
	TCHAR szTrack1[1024];
	TCHAR szTrack2[1024];
	TCHAR szTrack3[1024];
	TCHAR szRawdata[1024];

	switch(message)
	{
		case WM_COMM_RXCHAR_MESSAGE:	//receive data form serial port.
			break;
		case WM_MSR_TRACK_1_RECEIVE:	//track 1 data received.
```

```
                MSRReadString(TRK_1,szTrack1,1024);
                  memset(szTrack2,0,1024);
                    //      you can add extra codes here.
                break;
            case WM_CARD_TRACK_2_RECEIVE:
                MSRReadString(TRK_2,szTrack2,1024);
                 memset(szTrack2,0,1024);
                    // you can add extra codes here.
                break;
            case WM_CARD_TRACK_3_RECEIVE:
                MSRReadString(TRK_3,szTrack3,1024);
                 memset(szTrack3,0,1024);
                    // you can add extra codes here.
                break;
            case WM_COMM_RXCHAR_COMPLETE:
                MSRReadString(ALL_DATA,szRawdata,1024);
                 memset(szRawdata,0,1024);
                    // you can add extra codes here.
                break;
            case WM_CARD_ERROR_READ:
                    // you can add extra codes here.
                break;
            default:
                break;
        }
        return CDialog::WindowProc(message, wParam, lParam);
}
```