MWXUSB API Manual for MWXUSB.DLL Version 4.10

1. Overview

This API describes the usage of PrehKeyTec USB drivers to read MSR, Keylock, POSKey and TCO data from the keyboard. The API uses the MWXUSB.DLL and the KEYHOOK.DLL (USB version). The Dll supports all PrehKeyTec USB Keyboards with Firmware 605/3xxx.

2. Installation

The KEYHOOK.DLL needs to be installed and registered. This could be done with the Preh driver installation (today DrvPack1301.EXE) or manually.
To install manually you have to register the dll with the regsvr32.

Install the MWXUSB.DLL into the application directory.

3. Functions of MWXUSB.DLL

The MWXUSB.DLL loads the KEYHOOK.DLL and opens the USB keyboard. If the Preh USB keyboard is configured for OPOS than the MSR data is transmitted to the KEYHOOK.DLL. The MWXUSB.DLL gets the MSR data if opened and stores its content internally. The MWXUSB.DLL has OPEN, CLOSE, READ and CLEAR functions to control the MSR.
The current Version of the MWXUSB.DLL is 4.10.0.0
The current Version of the USB KEYHOOK.DLL is 1.0.0.26

3.1 Open USB Device

Description: Opens the keyboard communication channel and stores MSR data from the Keyboard.

Function: unsigned char Open_USB(void)

Return values:
- 0 = success
- 1 = failure

Before unloading the DLL Close USB Device has to be called.

3.2 Get Counter USB Device

Description: Gets the number of threads opened the keyhook.
Function: unsigned char Get_Counter_USB(void)

Return values:
- Number of opened threads.

Version: Implemented in Version 4.5

3.3 Check USB keyboard

Description: Check if a Preh USB Keyboard is plugged into the PC and recognized by USB.

Function: unsigned char Check_USB(void)

Return values:

0 = success
1 = failure (e.g. USB Device is not plugged in)

3.4 Close USB Device

Description: Close the keyboard communication channel. All stored data in the MWXUSB.DLL will be removed.

Function: unsigned char Close_USB(void)

Return values:
0 = success
1 = failure (e.g. USB Device was not opened)

3.5 Clear USB Device data

Description: Remove all data stored in the MWXUSB.DLL. Opened USB device remains open. The USB device has to be opened. This includes MSR, Keylock and TCO data.

Function: unsigned char Clear_USB(void)

Return values:
0 = success
1 = failure (e.g. USB Device was not opened)

3.6 Send data to USB Device

Description: Send a command to the USB keyboard.
For a parameter command set contact PrehKeyTec, because it can vary between models.

Function: unsigned char snddata_USB(unsigned char ucdata)

Parameter:
ucdata = data value

Return values:
0 = success
1 = failure (e.g. USB Device was not opened)

3.7 Read data from USB Device (obsolete)

Description: Reads data from the USB keyboard. Please do not use this Command.

Function: unsigned char readdata_USB(unsigned char *ucpData, DWORD dwLen, DWORD dwTime)

Parameters:

char *ucpData	Pointer to data buffer for return String.
DWORD dwLen	Length of ucpData data buffer-
DWORD dwTime	Timeout time in ms -

Return values:
0 = success
1 = failure (e.g. USB Device was not opened)

3.8 Read USB MSR data

Description: Read out MSR data. The function return immediately, if MSR data was already stored. Otherwise the DLL waits dwTime milliseconds for MSR data. If no card is swiped within time dwTime the return value is 15 (no data during time out time).

Function: unsigned char read_MSR_data_USB(char *ucpData1,int dwLen1,
char *ucpData2,int dwLen2,
char *ucpData3,int dwLen3,
DWORD dwTime)

Parameters:

char *ucpData1	Pointer to track 1 data buffer.
int dwLen1	Length of track 1 data buffer-
char *ucpData2	Pointer to track 2 data buffer.
int dwLen2	Length of track 2 data buffer-
char *ucpData3	Pointer to track 3 data buffer.
int dwLen3	Length of track 3 data buffer-
DWORD dwTime	Timeout time in ms

Return values:

0 = success
1 = USB Device was not opened
2 = no buffer for track 1
3 = no buffer for track 2
4 = no buffer for track 3
5 = dwTime is zero or undefined
6 = USB device initialisation error
7 = could not install timeout timer
8 = timeout timer not running
9 = track 1 buffer not sufficient for track 1 data
10 = internal communication error with track 1 data
11 = track 1 buffer not sufficient for track 2 data
12 = internal communication error with track 2 data
13 = track 1 buffer not sufficient for track 3 data
14 = internal communication error with track 3 data
15 = no data during timeout time

3.9 Read Keyboard Version

Description: Read Keyboard Information's. This function sends Commands to the keyboard to receive keyboard Version and other Information. The Information will be collected and returned. An internal Timeout of 5 sec. is used.

Function: unsigned char read_Version_USB(char *ucpData,int dwLen,int uLevel)

Parameters:

char *ucpData	Pointer to Information data buffer.
int dwLen	Length of Information data buffer-
int uLevel	0 = Complete Information String e.g. (C) 1990 - 2008 by PrehKeyTec GmbH 05129-605/3103 - MC 147 Alpha Jun 11 2008 11:10:45 BL 1.51 USB A ID MC147 C9 A07 G3 K4 M3 I2 SN 9132-1234567 154 = Firmware ID e.g. "605/3103" 155 = serial Number e.g. "9132-1234567" 156 = manufacture date e.g. "20090324" 160 = Type Information e.g. "MC147 C9A07G3K4M3I2" 161 = Product Number e.g. "90320-703/1800"

Return values:

0 = success
1 = USB Device was not opened
2 = no buffer for Informations
6 = USB device initialisation error
7 = could not install timeout timer
8 = timeout timer not running
9 = Information buffer ucpData not sufficient for Information data
15 = no data during timeout time
20 = Communication error first command byte
21 = Communication error second command byte
22 = Communication error third command byte
23 = Communication error fourths command byte

Version: Implemented in Version 4.7

3.9 Read TCO Data

Description: Read Keyboard TCO Information's.

Function: unsigned char read_TCO_data_USB(char *ucpData,int dwLen, DWORD dwTime)

Parameters:

char *ucpData	Pointer to Information data buffer.
int dwLen	Length of Information data buffer-
DWORD dwTime	Timeout time in ms

Return values:

0 = success
1 = USB Device was not opened
2 = no buffer for Informations
5 = dwTime is zero or undefined
6 = USB device initialisation error
7 = could not install timeout timer
8 = timeout timer not running
9 = Information buffer ucpData not sufficient for Information data
15 = no data during timeout time
16 = Close USB in Progress

Version: Implemented in Version 4.7

3.10 Read Keylock

Description: Send a command to the USB keyboard to read out the keylock position and receive the data and return it.

Function: int read_Keylock(void)

Parameter:

none

Return values:

0 to 4 Keylock Poition
-1 = failure (e.g. USB Device was not opened, no Keylock)

Version: Implemented in Version 4.7

3.11 Control ACCEPT LED

Description: Send a command to the USB keyboard to set the state of the ACCEPT Led

Function: unsigned char Accept_LED(int value)

Parameter:

Int value	0 = LED off
	1 = LED green
	2 = LED red

Return values:

0 = success
1 = USB Device was not opened
6 = USB device initialisation error
11 = Parameter value out of range
20 = Communication error first command byte
21 = Communication error second command byte
22 = Communication error third command byte
23 = Communication error fourths command byte

Version: Implemented in Version 4.7

3.12 Control FN Layer

Description: Send a command to the USB keyboard to set the state of the FN Layer

Function: unsigned char Set_FN_Layer(int value)

Parameter:

Int value	0 = FN Layer off
	1 = FN Layer on

Return values:

0 = success
1 = USB Device was not opened
6 = USB device initialisation error
11 = Parameter value out of range
20 = Communication error first command byte
21 = Communication error second command byte
22 = Communication error third command byte

Version: Implemented in Version 4.10

3.13 Init Sound Parameter

Description: Set Keyboard Sound Parameter. 4 different Frequencies are currently used (4800, 2400, 1200, 600)

Function: unsigned char Init_Sound_USB(int uFreq1, int uFreq2, int uFreq3,
int uVol1, int uVol2, int uVol3)

Parameters:

int uFreq1, int uFreq2, int uFreq3,	Frequency in Hz-(depends on Volume)
int uVol1, int uVol2, int uVol3	Volume in %

Return values:

- 0 = success
- 1 = USB Device was not opened
- 6 = USB device initialisation error
- 20..28 = Communication error command bytes
- 40 = uFreq1 = 0
- 41 = uFreq2 = 0
- 42 = uFreq3 = 0

Version: Implemented in Version 4.7

3.14 Sound

Description: Set Keyboard Sound Parameter.

Function: unsigned char Sound_USB(int uNumber,)

Parameters:

int uNumber	Ton1 = 0x22, Ton2 = 0x26, Ton3 = 0x2A 0x2F = stop sound

Return values:

- 0 = success
- 1 = USB Device was not opened
- 6 = USB device initialisation error
- 20..24 = Communication error command bytes

Version: Implemented in Version 4.7

3.15 Get POS Keys

Description: Get Keyboard POS Keys. Number 1-128 is stored in *cpData[0]. If the Key is pressed *cpData[1] = 0, otherwise *cpData[1] = 1.

Function: unsigned char read_POS_key_USB(int *cpData, DWORD dwTime)
Parameters:
Return values:

- 0 = success
- 1 = USB Device was not opened
- 2 = no buffer for Informations
- 5 = dwTime is zero or undefined
- 6 = USB device initialisation error
- 7 = could not install timeout timer
- 8 = timeout timer not running
- 15 = no data during timeout time

3.16 Install Callback Function

Description: Install a Callback Function to a void Function (int data) Routine. Please return as quick as possible from this routine to avoid deadlocks.

Function: void Set_Callback (Function Pointer)

Callback Parameters:

Int data	1 = POS Keyboard down
	2 = POS Keyboard up
	3 = MSR Event
	4 = Keylock Event
	5 = Version, TCO Event
	8 = Scanner Event
	9 = Restart Keyhook Event

Return values: none

Version: Implemented in Version 4.7

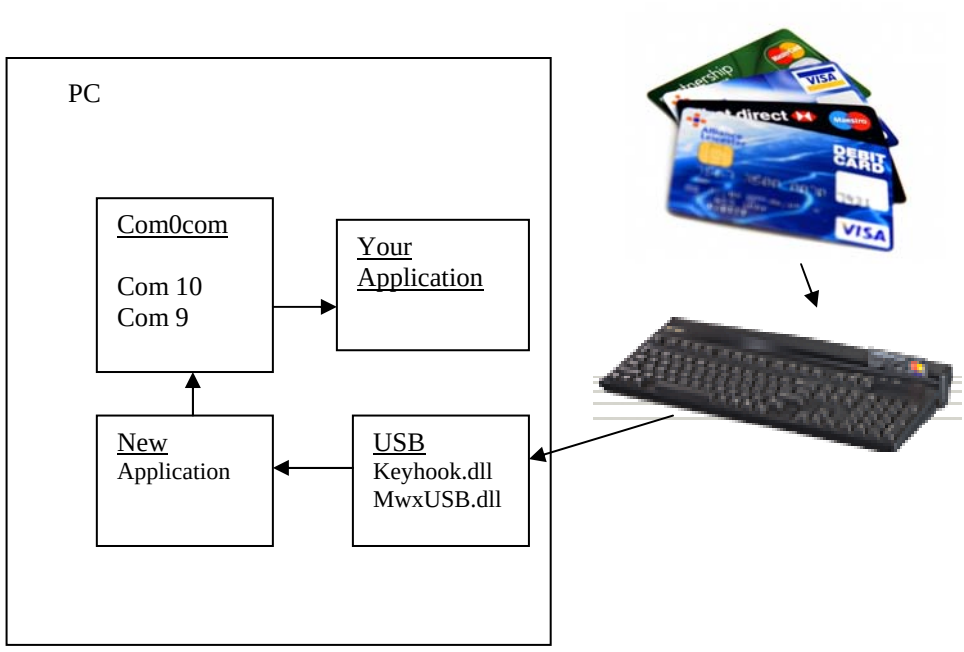
3.17 Install_COM

Description: MSR data can be transmitted to a COM port. This port must be defined in the mwxusb.ini file. The MSR data will be sent to this port.
With a COM0COM you will get the MSR data as an input serial data stream.

Install_COM Implemented in Version 4.8

Function: unsigned char Install_COM(void)
Parameters:
Return values:

0 = success
1 = USB Device was not opened
2 = could not open COM port
3 = could not get settings of COM port
4 = could not change settings of COM port
5= could not set timeouts of COM port



Installation:

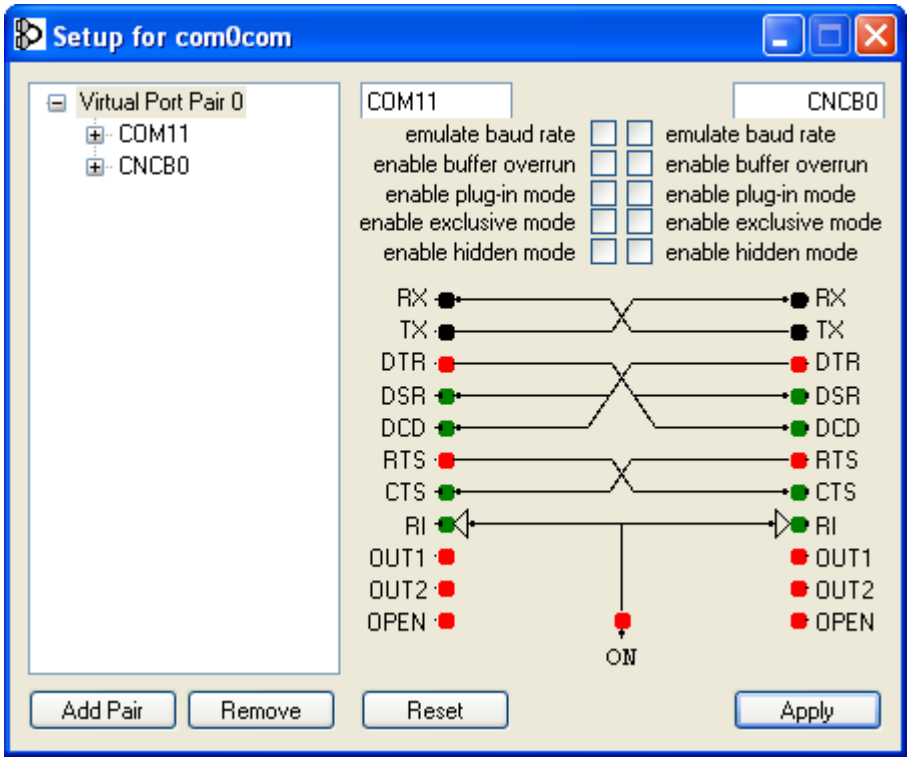
Based on a GPL Software com0com you can install 2 virtual com ports on your PC.

1. COM0COM
   Install COM0COM or add a new pair if you have already installed COM0COM.
   See: http://com0com.sourceforge.net/
2. Choose the port names you want to use. One for your Application and another for the internal use (e.g. CNCB0).
3. Extract the USB2COM.zip Package in a folder. This Package does not need any installation. You can run it directly within this folder.
4. Click on registerkeyhook.bat – this will register a USB Communication DLL (Keyhook.dll) for PrehKeyTec keyboards.
5. Modify mwxusb.ini
   In this ini, you can set the baud rate, parity, Headers Terminators for each track and enable and disable tracks. See MWXUSB.ini
6. Run Startmwxusb.exe to start the virtual com bridge.

MWXUSB.ini

Section [COM_INIT]

COMPort = “CNCB0”
Choose the COM Port you have defined for the internal use in the COM0COM



Settings = "baud=9600 parity=E data=7 stop=1"
Choose the COM Port parameters.

Section [TRACK1], [TRACK2], [TRACK3]

Enabled = 0
You can switch off each track. Use 0 or false

Checksum = true
You can switch on the Checksum of the MSR. Use on, true or 1

Sentinels = true
You can switch on the Sentinels of the MSR. Use on, true or 1

HeaderText = "S" or TerminatorText
This will add Text before the data (Header) and behind the data (Terminator)

HeaderHex = 0x24 or TerminatorHex
Here you can define Hex values.

HeaderDec = 111 12 or TerminatorDec
Here you can define decimal values.

If you define Text and Hex and Dec there is a defined structure:

HeaderHex, HeaderDec, HeaderText, msr track data, TerminatorHex, TerminatorDec and TerminatorText

Example of MWXUSB.ini:

```
[COM_INIT]
COMPort = COM9
Settings = "baud=9600 parity=E data=8 stop=1"
[TRACK1]
Sentinels = true
HeaderText = "S"
HeaderHex = 0x02 0x39
[TRACK2]
Checksum = 1
HeaderDec = 02 39
TerminatorDec = 03 114
[TRACK3]
Enabled = 1
Checksum = 1
HeaderDec = 02 39
TerminatorHex = 03 14
TerminatorText = "F"
```

3.18 Use cryptographic transmission

Description: Keyboard data can be encrypted to establish a secure data transmission between keyboard and application. With those functions your software will exceed PCI requirements.

Cryptographic functions are implemented in Version 4.9

To switch on this functionality you need a special firmware and a mwxusb.ini file.

Please change the example key for your security.

The debug functionality is switched off for security reasons. For debug options contact PrehKeyTec.

Example of MWXUSB.ini:

[CRYPT]
Enabled = 1
Key = “12345678901234567890123456789012” This is a very unsecure Key `example`
SerialNumber = “8124-0000173525”

The firmware use One Time Pad, AES with 256 bit and ARCFOUR with 256 bit. The way of authentication and encryption is secret and will not be published.

With **enabled = 1** you can switch on the encrypted functions. After Open USB Device (3.1 Open USB Device) the encryption will start and install the encryption in the keyboard.

With **Key** you can set up your individual key. This key needs to be downloaded to the keyboard as well. Use the Preh Windows Programmer and the BadReadString to configure the key.

In Keyboards with encryption functions reading of the keytable is not possible.

With **SerialNumber** you can restrict the use of the DLL to one specific keyboard with its unique serial number.

With **MSRTrack1 = 1, MSRTrack2 = 1, MSRTrack3 = 1, Keylock = 1, Keyboard = 1** the encryption can be switched on for the specific modules.

Example of MWXUSB.ini
[DEBUG]
Logfile = "mwxusb.log"
[CRYPT]
Enabled = 1
MSRTrack1 = 1
MSRTrack2 = 0
MSRTrack3 = 1
Keylock = 1
Keyboard = 1
Key = "A#O$^u[TQN37>t>^D7{v>?n*pR:MC8tH"
sSerialNumber= "7421-0002001"

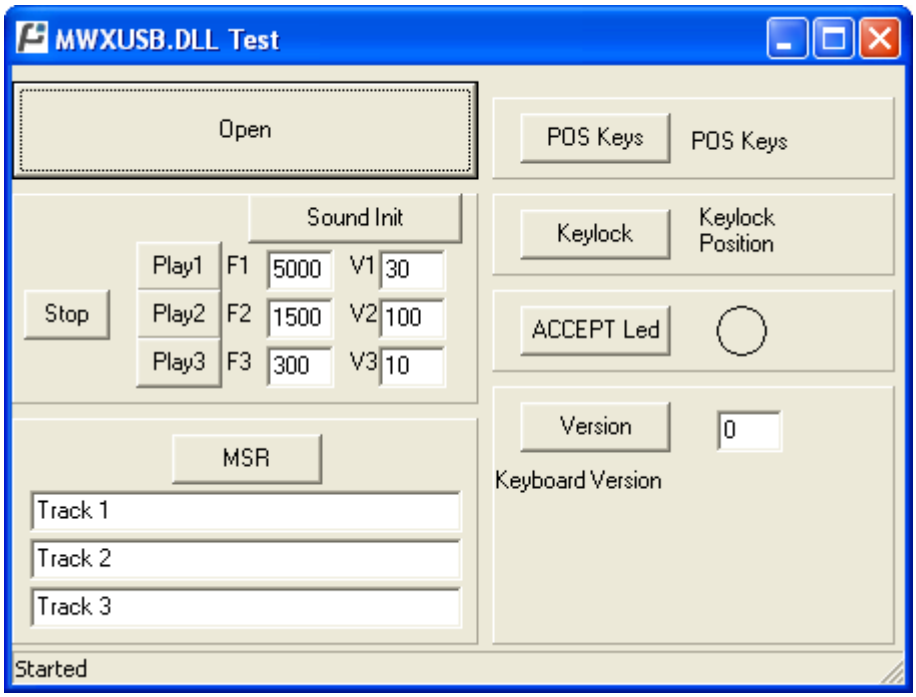
4. Examples

4.1 C++ Application MWXUSB Test.exe

This is a demo application with source code to show the functionality of the MWXUSB.DLL.

C++ Borland Builder 4.0

This test Application shows the Events of MSR, Keylock, POS Keys and internal function Callbacks. You can read devices and control LED and Sound.



Requirement: Windows Vista, Windows XP, Windows 2000

4.2 Console Application MSRdemo.exe

This is a demo console application with source code to simulate the Preh RS232 MSR.

C++ Borland Builder 4.0

The MSRdemo has a parameter for the timeout time for reading a MSR in seconds.

MSRdemo 10
waits 10 seconds for a MSR swipe.

Steps of the console application

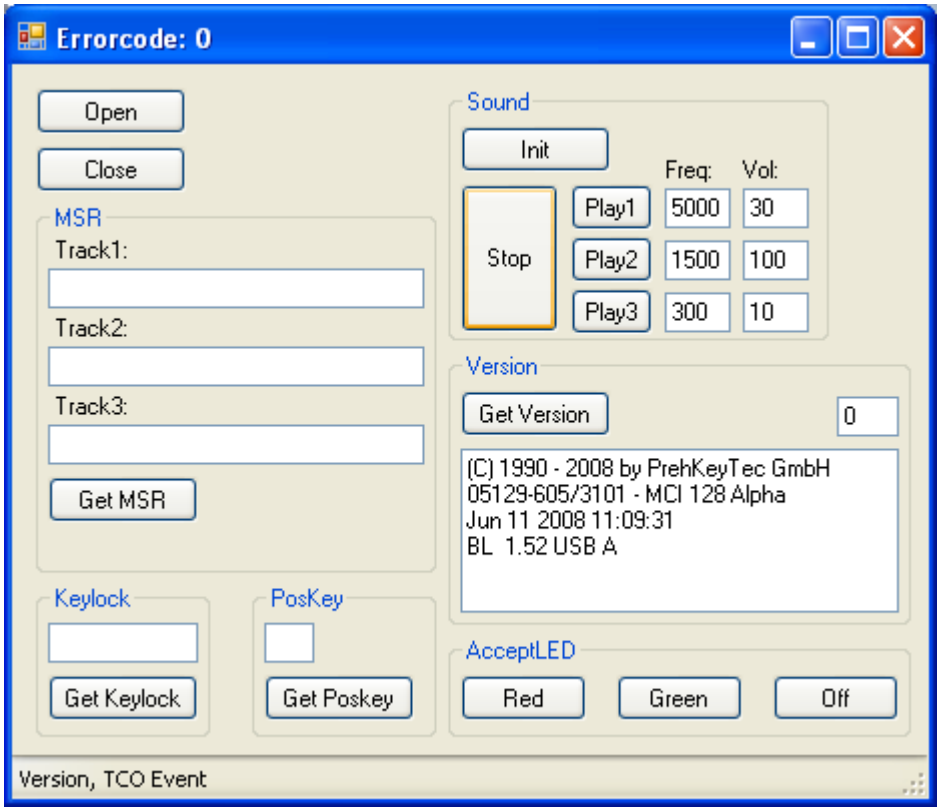
- load the MWXUSB.DLL
- open the USB device
- switch ACCEPT LED to green
- wait for a card read
- switch ACCEPT LED to red if error read
- switch ACCEPT LED off if good read
- close the USB device
- close the MWXUSB.DLL
- write MSR data to MSR.TXT
- exit application

Requirement: Windows Vista, Windows XP, Windows 2000

4.3 Visual Basic Application MWXDLL.exe

This is a demo application with source code to show the functionality of the MWXUSB.DLL.

Visual Basic .NET 2005



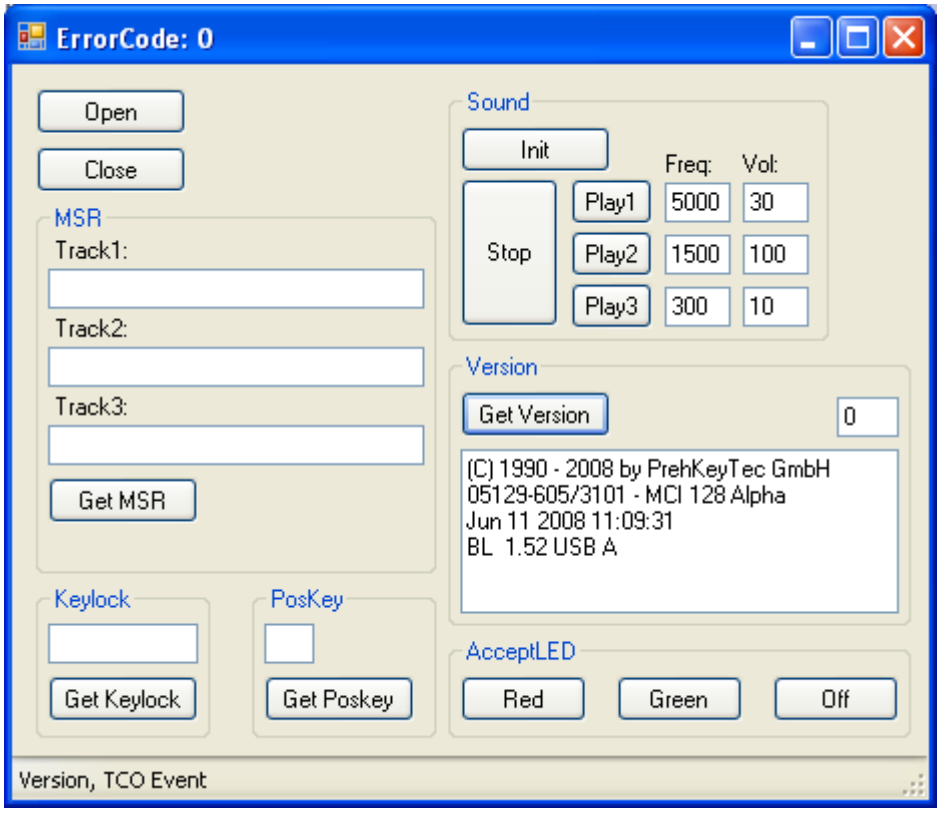
This test Application shows the Events of MSR, Keylock, POS Keys and internal function Callbacks. You can read devices and control LED and Sound.

Requirement: Windows Vista, Windows XP, Windows 2000
Framework 2.0 or later installed

4.4 C# Application MWXDLL.exe

This is a demo application with source code to show the functionality of the MWXUSB.DLL.

Visual C# 2008



This test Application shows the Events of MSR, Keylock, POS Keys and internal function Callbacks. You can read devices and control LED and Sound.

Requirement: Windows Vista, Windows XP, Windows 2000
Framework 2.0 or later installed

4.5 Utility StartMWXUSB.exe

This exe will start the dll and open a COM port for transferring MSR data to the COM Port. The COM port must be available and a MWXUSB.ini file with all settings must be located in the same directory of the MWXUSB.dll.

5. Version history and releases

Overview about MWXUSB.DLL versions and revision changes.

Version 4.1.0, date 21.11.2002
Version 4.2.0, date 01.03.2005
Version 4.3.0, date 13.06.2007 read USB data added
Version 4.4.0, date 19.09.2007 Multithread deadlock
Version 4.5.0, date 26.09.2007 Multithread deadlock
Version 4.6.0, date 27.09.2007 Multithread deadlock
Version 4.7.0, date 12.08.2008, TCO version and callback function added
Version 4.8.1, date 08.10.2008, COM Interface and MWXUSB.ini added
Version 4.8.2, date 07.11.2008, Winprog download Problem, Event handling
Version 4.9.0, date 02.12.2008, Cryptographic transmission added
Version 4.10.0, date 12.03.2009, Cryptographic, POSKeys, FN Function added