

APPLICATION NOTE

HCR360 WIN32 API Reference Guide

(RS232 Interface Smart Card Reader)

Document SL042

Revision B

18 Nov. 2004

Contents

1. INTRODUCTION	4
2. DRIVER INFORMATION	4
3. SYSTEM REQUIREMENT	4
4. RELATED FILES	5
5. FUNCTIONS DESCRIPTION	6
General Functions	6
OpenCom ()	6
CloseCom ()	6
USI2 Low Level Functions	8
Usi2SendData ()	8
Usi2ReceiveData ()	8
Usi2Exchange ()	8
Usi2CheckCardPresence ()	9
Reader Commands	10
Usi2Retransmit ()	10
Usi2ReaderVerReport ()	10
Usi2SwitchStatus ()	11
Usi2SetVerboseMode ()	11
Usi2ReaderStatus ()	11
Usi2SetGreenLed ()	12
Usi2SetRedLed ()	13
Usi2Latch ()	13
Usi2Unlatch ()	14
Usi2ConfigReport ()	14
Usi2WarmReset ()	15
CPU Card	17
Usi2SelectIccConnector ()	17
Usi2CardPowerUp ()	17
Usi2CardPowerOff ()	18
Usi2SendApdu ()	18
Usi2ExecutePPS ()	18
Usi2SetIFSD ()	19
Usi2GetErrorCode ()	19
Memory Card	20
Usi2SelectMemCardType ()	20
Usi2SendMemCardApdu ()	21

Usi2MemoryVerify ()	21
Usi2MemoryAuthenticate ()	22
Usi2MemoryExtAuthenticate ()	22
Usi2MemoryReadData ()	22
Usi2MemoryWriteData ()	23
Usi2MemoryEraseData ()	23
Usi2MemoryRestoreData ()	24
Usi2MemoryWriteProtect ()	24
Usi2MemoryReadProtect ()	25
Usi2MemoryEraseUserArea ()	25
Usi2MemoryChangePSC ()	25
Usi2MemoryReadErrCount ()	26
Magnetic card command	27
Usi2ArmtoRead ()	27
Usi2CardMoveNotify ()	27
Usi2Abort ()	27
Usi2ReadMagData ()	28
Usi2ISOData ()	28
Usi2TransmitErrData ()	29
Usi2ForwardCusData ()	29
Usi2ReverseCusData ()	29
TLP224 Protocol	31
TLP224_SendData ()	31
TLP224_ReceiveData ()	31
TLP224_Exchange ()	31
TLP224 Turbo Protocol	33
TLP224Turbo_SendData ()	33
TLP224Turbo_ReceiveData ()	33
TLP224Turbo_Exchange ()	33
USI1 Protocol	35
USI1_SendData ()	35
UIS1_ReceiveData ()	35
TLP224Turbo_Exchange ()	35
Appendix A. Source Code of HCR360.bas	37

1. INTRODUCTION

This document describes the information related the Application Programming Interface (API) functions of HCR360 under MS Windows environment. It provides programmer with an efficient way for developing the magnetic stripe cards and IC (Integrate circuit) cards application.

HCR900 is hybrid card reader providing all the functions to manage the variety magnetic stripe cards and IC cards. Moreover, HCR360 is able to support any ISO7816 asynchronous smart card (T=0 or T=1 protocol) and some synchronous cards (I2C, S9, S10 and SLE4436) and variety magnetic stripe cards (ISO, AAMVA and CA old DMV card). It completely handles the communication layer between the card and the host system.

A specific protocol called “USI2” has been defined on the serial interface between HCR360 and the host system; it is developed by American Magnetics, which has features allowing it to be used for multi-dropping and for large (up to 64K bytes) messages. Though the Model HCR360 does not support multi-dropping at this time, a future variation of the device may. For further information on USI2, please refer to “HCR360 Programmer’s Manual”.

For function interface, this library supports two standard function interfaces, WIN32 API and VB Function. This will be much convenient for development of different application. In functions description, these two interfaces syntax will be listed together.

2. DRIVER INFORMATION

HCR360 also supports PC/SC driver. Since PCSC device driver will take the control of RS232 port. It is unable to access this library under PC/SC mode. Programmer should be aware of above situation to prevent the faults such as COM Port open error. Thus, the PCSC driver must not be installed if programmer is intended to use this library for HCR360.

3. SYSTEM REQUIREMENT

The HCR360 library requires IBM ATs or compatible running under Windows 95/98/2000/XP. Porting to different platforms such as Linux is also possible. Further information is provided upon request.

In addition to the libraries, samples program are provides the test the functionalities.

4. RELATED FILES

- 1.HCR360.dll: DLL file. Please place this file to Windows\System or Windows\System32 or place it in the same directory with the application file.
- 2.HCR360.h: Head file of HCR360.dll.
- 3.wsc.h: Head file for the low level functions of RS232. It had been included in HCR360.h.
- 4.HCR360.lib: Static library for VC++.
5. HCR360.bas: Re-defines all HCR360.dll supported API to Visual Basic format. According to VB programmer belabor, it changes the parameters of Binary Array of all WIN32 API to String format. This will be easy for VB developer to implement this file. This file should be added to the VB application project.

5. FUNCTIONS DESCRIPTION

General Functions

OpenCom ()

Syntax: int OpenCom (int Port, int Baudrate, int Databits)

Remark: Open specified COM port with specified baud rate.

Return: 0: Success

-1: Failure.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Baudrate: Baud rate ID.

0 => Baud110

1 => Baud300

2 => Baud1200

3 => Baud2400

4 => Baud4800

5 => Baud9600

6 => Baud19200

7 => Baud38400 (default)

8 => Baud57600

9 => Baud115200

DataBits: Parity and Data bits

0 => Even, 7bits

1 => Odd, 7bits

2 => Mark, 7bits

3 => Space, 7bits

4 => None, 8Bits (default)

VB Syntax:

Function vbOpenCom (Port As Long, Baudrate As Long, DataBits As Long) As Integer

CloseCom ()

Syntax: void CloseCom (int Port)

Remark: Closes the specified COM port.

Return: None.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

VB Syntax:

Sub vbCloseCom (Port As Long)

USI2 Low Level Functions

Usi2SendData ()

Syntax: void usi2SendData (int Port, BYTE Cmd, USHORT DataLen, BYTE DataBuf [])

Remark: Send one command to reader with USI2 protocol.

Return: None.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Cmd: Command code. Ex: '9', 09H.

DataLen: The data length that follow command.

DataBuf: Buffer for store data.

VB Syntax:

Sub vbSendData (Port As Long, strCmd as String, strDataBuf As String)

Usi2ReceiveData ()

Syntax: int usi2ReceiveData (int Port, BYTE *RcvData)

Remark: Waits for the response data from reader with USI2 protocol and receives it. This function will return timeout error if no data responded from the reader.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

RcvData: Data buffer address to receive.

VB Syntax:

Function vbReceiveData (Port As Long, strRcvData As String) As Long

Usi2Exchange ()

Syntax: int usi2Exchange (int Port, BYTE Cmd, USHORT SndDataLen, BYTE SndData [], BYTE RcvData [])

Remark: Send command and receive data from reader with USI2 protocol.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Cmd: Command code. Ex: '9', 09H.

SndDataLen: The data length that follow command.

SndData: Buffer for store data.

RcvData: Data buffer address to receive.

VB Syntax:

Function vbExchange (Port As Long, strCmd As String, strSndData As String, strRcvData As String) As Long

Usi2CheckCardPresence ()

Syntax: int usi2CheckCardPresence (int Port)

Remark: Check Card Presence

Return: 1: Card Inserted

0: Card Ejected

Else error occurred

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

VB Syntax:

N/A

Reader Commands

Usi2Retransmit ()

Syntax: int usi2Retransmit (int Port, BYTE RcvData [])

Remark: Retransmits the last message sent by the reader.

Return: Output data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

RcvData: Data buffer to receive.

VB Syntax:

Function vbRetransmit (Port As Long, strRcvData As String) As Long

Usi2ReaderVerReport ()

Syntax: int usi2ReaderVerReport (int Port, BYTE SubCmd, BYTE RcvData [])

Remark: Transmit version information.

Return: Output data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

SubCmd: Null, Version Report.

0(30H), Serial Number Report.

1(31H), Copyright Report.

2(32H), Reader Model Number Report.

3(33H), Reader PCB Number Report.

4(34H), Smart Card Library Version Report.

5(35H), F/D Parameters List Report.

6(36H), Boot loader Version Number Report.

7(37H), Reader Configuration Data Report.

8(38H), Reader Customer Configuration Data Report.

9(39H), Reader Manufacturing Configuration Data Report.

Others, Copyright Report.

RcvData: Data buffer to receive.

VB Syntax:

Function vbReaderVerReport (Port As Long, strCmd As String, strRcvData As String) As

Long

Usi2SwitchStatus ()

Syntax: int usi2SwitchStatus (int Port)

Remark: Report the status of the slot switches.

Return: Switch Status if return value ≥ 0 .

- 0 Neither switch actives. (No card inserted)
- 1 Front switch actives. (Card present)
- 2 Rear switch actives, but front switch isn't active. (Illegal condition)
- 3 Front and rear switches active. (Card seated)
- 1 Timeout, Reader no response.
- 2 LRC Error
- 3 Unknown error

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

VB Syntax:

Function vbusi2SwitchStatus (Port As Long) As Long

Usi2SetVerboseMode ()

Syntax: int usi2SetVerboseMode (int Port)

Remark: Most error responses until the reader receives a reset command will include a short descriptive message

Return: Success if ≥ 0 .

- 1 Timeout, Reader no response.
- 2 LRC Error

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

VB Syntax:

Function vbusi2SetVerboseMode (Port As Long) As Long

Usi2ReaderStatus ()

Syntax: int usi2ReaderStatus (int Port, BYTE RcvData [])

Remark: Interrogate the reader about its operational status. Two bytes of status information will be returned

Return: Success if ≥ 0 .

First Status Byte

Bit	Value: 0	Value: 1
0:0	Card not present	Card present
0:1	Card not seated	Card seated
0:2	Unlatched	Latched
0:3	ICC Power OFF	ICC Power ON
0:4	Non automatic slot switch report	Automatic slot switch report
0:5	No Magnetic Stripe Card data	Magnetic Stripe Card data
0:6	Not armed to read	Armed to read
0:7	Do not send auto ATR	Send auto ATR

Second Status Byte

Bit	Value: 00	Value: 01	Value: 10	Value: 11
1:0 - 1:1	Green LED OFF	Green LED ON	Green LED flash	Unused
1:2 - 1:3	Red LED OFF	Red LED ON	Red LED flash	Unused
1:4 - 1:7	Reserved for future use, always zero			

-1 Timeout, Reader no response.

-2 LRC Error

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

RcvData: Data buffer to receive.

VB Syntax:

Function vbusi2ReaderStatus (Port As Long, strRcvData As String) As Long

Usi2SetGreenLed ()

Syntax: int usi2SetGreenLed (int Port, BYTE LedStatus)

Remark: Set Green LED.

Return: Success if ≥ 0

-1 Timeout, Reader no response.

-2 LRC Error

-3 Led command error (In Self-Arm mode or other LedStatus)

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

LedStatus: 0(00H), LED Off.

1(01H), LED On.

2(02H), LED Flash.

Other, Error

VB Syntax:

Function vbGreenLedON (Port As Long) As Long

Function vbGreenLedOFF (Port As Long) As Long

Function vbGreenLedFLASH (Port As Long) As Long

Usi2SetRedLed ()

Syntax: int usi2SetRedLed (int Port, BYTE LedStatus)

Remark: Set Red LED.

Return: Success if ≥ 0

-1 Timeout, Reader no response.

-2 LRC Error

-3 Led command error (In Self-Arm mode or other LedStatus)

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

LedStatus: 0(00H), LED Off.

1(01H), LED On.

2(02H), LED Flash.

Other, Error

VB Syntax:

Function vbRedLedON (Port As Long) As Long

Function vbRedLedOFF (Port As Long) As Long

Function vbRedLedFLASH (Port As Long) As Long

Usi2Latch ()

Syntax: int usi2Latch (int Port, BYTE RcvData [])

Remark: Activate card latch.

Return: Received data length, if ≥ 0

-1 Timeout, Reader no response.

-2 LRC Error

-3 Latch Fail

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

RcvData: Data buffer to receive. Card latch success if RcvData [0] = '^'

VB Syntax:

Function vbusi2Latch (Port As Long, strRcvData As String) As Long

Usi2Unlatch ()

Syntax: int usi2Unlatch (int Port, BYTE RcvData [])

Remark: Deactivate card latch

Return: Received data length, if ≥ 0

-1 Timeout, Reader no response.

-2 LRC Error

-3 Unlatch Fail

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

RcvData: Data buffer to receive. Card unlatch success if RcvData [0] = '^'

VB Syntax:

Function vbusi2Unlatch (Port As Long, strRcvData As String) As Long

Usi2ConfigReport ()

Syntax: int usi2ConfigReport (int Port, BYTE RcvData [])

Remark: Return standard one byte or extended 16-byte string with configuration.

Return: 1 => return standard one configuration byte

Standard One Configuration Byte

Bit	Value: 0	Value: 1
0	Track 1 not present	Track 1 present
1	Track 2 not present	Track 2 present
2	Track 3 not present	Track 3 present
3	Latch not present	Latch present
4	ICC connector not present	ICC connector present
5	Power fail unlatch not present	Power Fail Unlatch present
6	Reserved for future use, always 0.	
7	Reserved for future use, always 0.	

16 => return extended configuration bytes

Extended Configuration Bytes (16 bytes)

Byte	0	1	2	3	4	5	6	7-15
Remark	Equip. 0	Equip. 1	Protocol	Speed	Address	Actual Memory	Default Memory	Reserved, set to 00H

						Card Type	Card Type	
--	--	--	--	--	--	-----------	-----------	--

Protocol: 0=USI2; 1=TLP-224; 2=TLP-224 Turbo.

Speed: 0=1200, 1=2400, 2=4800, 3=9600, 4=19.2k, 5=38.4k, 6=56k, and 7=115.2k bps.

Address: Always is 00H.

Actual Memory Card Type: See Memory Card Types table.

Default Memory Card Type: The memory card type after reader has been powered-up or reset.

Equip. 0 — Extended Configuration Byte 0

Bit	Value: 0	Value: 1
0	User card connector not present	User card connector present
1	Reserved for future use, always 0	
2	SAM Board not present	SAM Board present
3	CTS not present	CTS present
4	Track 1 not present	Track 1 present
5	Track 2 not present	Track 2 present
6	Track 3 not present	Track 3 present
7	Latch not present	Latch present

Equip. 1 — Extended Configuration Byte 1

Bit	Value: 0	Value: 1
0	Power fail not present	Power fail present
1 - 7	Reserved for future use, always 0.	

-1 => Timeout, Reader no response.

-2 => LRC Error

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

RcvData: Data buffer to receive.

VB Syntax:

Function vbusi2ConfigReport (Port As Long, strRcvData As String) As Long

Usi2WarmReset ()

Syntax: int usi2WarmReset (int Port)

Remark: Abort all current actions and cause the device to execute all initialization functions.

Return: Success if >= 0

-1 => Timeout, Reader no response.

-2 => LRC Error

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

VB Syntax:

Function vbusi2WarmReset (Port As Long) As Long

CPU Card

Usi2SelectIccConnector ()

Syntax: int usi2SelectIccConnector (int Port, BYTE IccConnector)

Remark: Select the SAM connector to be used with the following reader commands.

Return: Success if ≥ 0

-1 => Timeout, Reader no response.

-2 => LRC Error

-3 Select connector error

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

IccConnector:

0: User Card

1 ~ 4: SAM1 ~ SAM4

Other: Error

VB Syntax:

Function vbusi2SelectIccConnector (Port As Long, IccConnector As Byte) As Long

Usi2CardPowerUp ()

Syntax: int usi2CardPowerUp (int Port, BYTE Vcc, BYTE EmvComply, BYTE Atr [])

Remark: Power up a smart card.

Return: ATR length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Vcc: 1(01H) => Power up the card at a VCC of 1.8V.

3(03H) => VCC = 3V.

5(05H) => VCC = 5V. (Default)

EmvComply: 0 => Power up with ISO mode. (Default)

1 => Power up with EMV compliant mode. If the card having an ATR which does not comply with EMV3.1.1 requirements will be rejected.

Atr: The ATR data.

VB Syntax:

Function vbCardPowerUp (Port As Long, Vcc As Integer, EmvComply As Integer, strAtr As String) As Long

Usi2CardPowerOff ()

Syntax: int usi2CardPowerOff (int Port)

Remark: Deactivates smart card, whatever CPU card or Memory card are used.

Return: 0 Success.

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

VB Syntax:

Function vbusi2CardPowerOff (Port As Long) As Long

Usi2SendApdu ()

Syntax: int usi2SendApdu (int Port, USHORT ApduLen, BYTE ApduCmd [],
BYTE RcvData [], BYTE SW [])

Remark: Sends an APDU command to card, whatever T=0 or T=1 are used.

Return: Received data length, if returned value >= 0.

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

ApduLen: The length of APDU command

ApduCmd: Card command under APDU format.

RcvData: Received data buffer.

SW: Status words that responded from the smart card.

VB Syntax:

Function vbSendApdu (Port As Long, strApdu As String, strRcvData As String, strSW As String) As Long

Usi2ExecutePPS ()

Syntax: int usi2ExecutePPS(int Port, BYTE ICprtcl, BYTE Fi, BYTE Di, BYTE RcvData[])

Remark: Executes PPS.

Return: Success, if return 6 bytes.

Else, return 1 byte error code.

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

-3 ICC isn't activated.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

ICprtcl: For IC card protocol.

FiDi: Set Fi/Di parameter

RcvData: Received data buffer.

VB Syntax:

Function vbExecutePPS(Port As Long, ICprtcl As Integer, Fi As Integer, Di As Integer, strRcvData As String) As Long

Usi2SetIFSD ()

Syntax: int usi2SetIFSD(int Port, BYTE IFSDsize)

Remark: Set IFSD.

Return: Success, return <00H>,

Else, return error code.

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

-3 ICC isn't activated.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

IFSDsize: Set the IFSD size.

VB Syntax:

Function vbSetIFSD(Port As Long, IFSDsize As Integer) As Long

Usi2GetErrorCode ()

Syntax: int usi2GetErrorCode (int Port)

Remark: Get error code.

Return: If no error, return <00H>,

Else, return error code.

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

VB Syntax:

Function vbGetErrorCode(Port As Long) As Long

Memory Card

Usi2SelectMemCardType ()

Syntax: int usi2SelectMemCardType (int Port, BYTE MemCardType, BYTE PageSize)

Remark: Select Memory Card Type

Return: Success if ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

-3 The specified memory card type is not supported

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

MemCardType:

TYPE	CARDS
00H	Telephone Card: Siemens SLE4406, SLE4436(E), SGS Thomson ST1305, Solaic E192B, GPM103
01H	S=9 Card: Siemens SLE4418, SLE4428
02H	416 Type Card: Siemens SLE4404, GemPlus GPM416-5V, GPM896, Atmel AT88SC101, AT88SC102, Incard MS101, MS102, AMMI 101 and 102
03H	S=10 Card: Siemens SLE4432, SLE4442
04H	3 Byte I2C Protocol, Atmel 24C01A, 24C02, 24C04, 24C08, 24C16, Orga OMC256T, OMC2048; SGC Thomson ST14C02C; ZeitControl ZCM02M, ZCM16M
05H	4 Byte I2C Protocol, Atmel AT24C64, AT24C256; AT24C512, Microchip 24LC65; ZeitControl ZCM64M

PageSize: Used for I2C card only for write data byte. This API will using default value is

PageSize = 0

Page Sizes on a Variety of I2C Cards

Description	Card Type	Page Size
AT24C01A	4	8 bytes (8H)
AT24C02	4	8 bytes (8H)
AT24C04	4	16 bytes (10H)
AT24C08	4	16 bytes (10H)
AT24C16	4	16 bytes (10H)
AT24C64	5	32 bytes (20H)
AT24C128	5	64 bytes (40H)
AT24C256	5	64 bytes (40H)
AT24C512	5	128 bytes (80H)

VB Syntax:

Function vbusi2SelectMemCardType (Port As Long, CardType As Byte, PageSize As Byte) As

Long

Usi2SendMemCardApu ()

Syntax: int usi2SendMemCardApu (int Port, USHORT ApduLen, BYTE ApduCmd [], BYTE RcvData [], BYTE SW [])

Remark: Send APDU command to Memory Card.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

ApduLen: The length of APDU command

ApduCmd: Card command under APDU format.

RcvData: Received data buffer.

SW: Status words that responded from the smart card.

VB Syntax:

Function vbusi2SendMemCardApu (Port As Long, strApdu As String, strRcvData As String, strSW As String) As Long

Usi2MemoryVerify ()

Syntax: int usi2MemoryVerify (int Port, USHORT Nums, BYTE PassWord [], BYTE SW [])

Remark: To verify memory card for type0 to type3.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Nums: Length of password.

PassWord: Verify code.

SW: Status words that responded from the reader.

VB Syntax:

Function vbusi2MemoryVerify (Port As Long, strPassWord As String, strSW As String) As Long

Usi2MemoryAuthenticate ()

Syntax: int usi2MemoryAuthenticate (int Port, USHORT Nums, BYTE DataBuf [], BYTE SW [])

Remark: Basic authentic for SLE4436/5536.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Nums: Length for challenge buffer

DataBuf: Send data buffer for challenge.

SW: Status words that responded from the reader.

VB Syntax:

Function vbusi2MemoryAuthenticate (Port As Long, strDataBuf As String, strSW As Byte) As Long

Usi2MemoryExtAuthenticate ()

Syntax: int usi2MemoryExtAuthenticate (int Port, USHORT Nums, BYTE DataBuf [], BYTE SW[])

Remark: Extended authentic for SLE5536.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Nums: Length for challenge buffer

DataBuf: Send data buffer for challenge.

SW: Status words that responded from the reader.

VB Syntax:

Function vbusi2MemoryExtAuthenticate (Port As Long, strDataBuf As String, strSW As Byte) As Long

Usi2MemoryReadData ()

Syntax: int usi2MemoryReadData (int Port, USHORT Adds, USHORT Nums, BYTE RcvData [], BYTE SW [])

Remark: Read byte data for all kind of memory card.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Adds: Byte address of memory card.

Nums: Length for data buffer.

RcvData: Receive data buffer.

SW: Status words that responded from the reader.

VB Syntax:

Function vbusi2MemoryReadData (Port As Long, Adds As Integer, rcvLen As Integer, strRcvData As String, strSW As String) As Long

Usi2MemoryWriteData ()

Syntax: int usi2MemoryWriteData (int Port, USHORT Adds, USHORT Nums, BYTE DataBuf [], BYTE SW [])

Remark: Write Byte data for all kind of memory card.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Adds: Byte address of memory card.

Nums: Length for data buffer.

DataBuf: Write data.

SW: Status words that responded from the reader.

VB Syntax:

Function vbusi2MemoryWriteData (Port As Long, Adds As Integer, strDataBuf As String, strSW As String) As Long

Usi2MemoryEraseData ()

Syntax: int usi2MemoryEraseData (int Port, USHORT Adds, BYTE SW [])

Remark: Erase binary data for type0 and type1 memory card.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Adds: Byte address of memory card.

SW: Status words that responded from the reader.

VB Syntax:

Function vbusi2MemoryEraseData (Port As Long, Adds As Integer, strSW As String) As Long

Usi2MemoryRestoreData ()

Syntax: int usi2MemoryRestoreData (int Port, USHORT Adds, BYTE SW [])

Remark: Restore Data for type 0 memory card.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Adds: Byte address of memory card.

SW: Status words that responded from the reader.

VB Syntax:

Function vbusi2MemoryRestoreData (Port As Long, Adds As Integer, strSW As String) As Long

Usi2MemoryWriteProtect ()

Syntax: int usi2MemoryWriteProtect (int Port, USHORT Adds, USHORT Nums, BYTE DataBuf [], BYTE SW [])

Remark: Write Binary with Protect for type1 and type3 memory card.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Adds: Byte address of memory card.

Nums: length for data buffer to send data

DataBuf: Write data buffer.

SW: Status words that responded from the reader.

VB Syntax:

Function vbusi2MemoryWriteProtect (Port As Long, Adds As Integer, strDataBuf As String, strSW As String) As Long

Usi2MemoryReadProtect ()

Syntax: int usi2MemoryReadProtect (int Port, USHORT Adds, USHORT Num, BYTE RcvData [], BYTE SW [])

Remark: Read Binary with Protect for type1 and type3 memory card.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Adds: Byte address of memory card.

Num: length for data buffer to send data

RcvData: Receive data buffer.

SW: Status words that responded from the reader.

VB Syntax:

Function vbusi2MemoryReadProtect (Port As Long, Adds As Integer, rcvLen As Integer, strRcvData As String, strSW As String) As Long

Usi2MemoryEraseUserArea ()

Syntax: int usi2MemoryEraseUserArea (int Port, USHORT Num, BYTE ErsPSC [], BYTE SW [])

Remark: Erase User Area for type2 memory card.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Num: Length for erase password.

ErsPSC: Erase Password.

SW: Status words that responded from the reader.

VB Syntax:

Function vbusi2MemoryEraseUserArea (Port As Long, strErsPSC As String, strSW As String) As Long

Usi2MemoryChangePSC ()

Syntax: int usi2MemoryChangePSC (int Port, USHORT Num, BYTE NewPSC [], BYTE SW [])

Remark: Change password for type1 to type3 memory card.

Return: Success if = 0.

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Nums: length for new password.

NewPSC: New Password.

SW: Status words that responded from the reader.

VB Syntax:

Function vbusi2MemoryChangePSC (Port As Long, strNewPSC As String, strSW As String)

As Long

Usi2MemoryReadErrCount ()

Syntax: int usi2MemoryReadErrCount (int Port, BYTE RcvData [], BYTE SW [])

Remark: To read Error counter.

Return: Success if = 0.

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

RcvData: For receive error count.

SW: Status words that responded from the reader.

VB Syntax:

Function vbusi2MemoryReadErrCount (Port As Long, strRcvData As String, strSW As String)

As Long

Magnetic card command

Usi2ArmtoRead ()

Syntax: int usi2ArmtoRead (int Port)

Remark: Arm to read.

Return: Success if = 0.

- 1 Timeout, Reader no response.
- 2 LRC Error
- 3 Invalid Command or Unavailable this request

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

VB Syntax:

Function vbusi2ArmtoRead (Port As Long) As Long

Usi2CardMoveNotify ()

Syntax: int usi2CardMoveNotify (int Port, BYTE *RcvData)

Remark: Checks the reader's card notification. The smart card reader will send a notification command automatically when a card inserted or extracted without any command coming from the system.

Return: Output data length, if returned value > 0. (For Self-Arm mode)

- 0: Read Msg. card OK. (For arm to read command and auto ATR report)
- 1: Timeout, Reader no response.
- 2: LRC Error
- 3: Isn't Msg. card.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

RcvData: Data buffer address to receive.

VB Syntax:

Function vbusi2CardMoveNotify (Port As Long, strRcvData As String) As Long

Usi2Abort ()

Syntax: int usi2Abort (int Port)

Remark: Abort 'Arm to read'.

Return: Success if >= 0.

- 1 Timeout, Reader no response.
- 2 LRC Error

-3 Invalid Command or Unavailable this request

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

VB Syntax:

Function vbusi2Abort (Port As Long) As Long

Usi2ReadMagData ()

Syntax: int usi2ReadMagData (int Port, BYTE selectTK, BYTE RcvData [])

Remark: Read Magnetic data.

Return: Output data length, if returned value ≥ 0 .

-1 Timeout, Reader no response.

-2 LRC Error

-3 Invalid Command or Unavailable this request

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

selectTK: select track via bit 0,1,2 (tk1/tk2/tk3)

RcvData: Receive response data

VB Syntax:

Function vbusi2ReadMagData (Port As Long, selectTK As Integer, strRcvData As String) As Long

Usi2ISOData ()

Syntax: int usi2ISOData (int Port, BYTE Track, BYTE viaISO, BYTE RcvData [])

Remark: Read Track data via ISO decode

Return: Output data length, if returned value ≥ 0 .

-1 Timeout, Reader no response.

-2 LRC Error

-3 Invalid Command or Unavailable this request

-4 parameter error.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Track: select Track (01h, 02h, 03h or 'q', 'r', 's').

viaISO: select decode character.(01h, 02h, 03h or 31h,32h,33h)

RcvData: Receive response data

VB Syntax:

Function vbusi2ISOData (Port As Long, Track As Integer, viaISO As Integer, strRcvData As

String) As Long

Usi2TransmitErrData ()

Syntax: int usi2TransmitErrData (int Port, BYTE RcvData [])

Remark: Transmit "Error Data".

Return: Output data length, if returned value ≥ 0 .

-1 Timeout, Reader no response.

-2 LRC Error

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

RcvData: Receive response data

VB Syntax:

Function vbusi2TransmitErrData(port As Long, strRcvData As String) As Long

Usi2ForwardCusData ()

Syntax: int usi2ForwardCusData (int Port, BYTE Track, BYTE viaNull, BYTE chrNums, BYTE RcvData [])

Remark: Read forward customer data.

Return: Output data length, if returned value ≥ 0 .

-1 Timeout, Reader no response.

-2 LRC Error

-3 Parameter error

-4 Invalid Command or Unavailable this request

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Track: select Track, '1'/'2'/'3'

viaNull: with NULL character, !=0

chrNums: with NULL character, '3'~'8'

RcvData: Receive response data

VB Syntax:

Function vbusi2ForwardCusData (Port As Long, Track As Integer, viaNull As Integer, chrNums As Integer, strRcvData As String) As Long

Usi2ReverseCusData ()

Syntax: int usi2ReverseCusData (int Port, BYTE Track, BYTE viaNull, BYTE chrNums, BYTE RcvData [])

Remark: Read reverse customer data.

Return: Output data length, if returned value ≥ 0 .

- 1 Timeout, Reader no response.
- 2 LRC Error
- 3 Parameter error
- 4 Invalid Command or Unavailable this request

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Track: select Track, '1'/'2'/'3'

viaNull: with NULL character, !=0

chrNums: with NULL character, '3'~'8'

RcvData: Receive response data

VB Syntax:

Function vbusi2ReverseCusData (Port As Long, Track As Integer, viaNull As Integer, chrNums As Integer, strRcvData As String) As Long

TLP224 Protocol

TLP224_SendData ()

Syntax: void TLP224_SendData (int Port, BYTE Cmd, BYTE DataLen, BYTE DataBuf [])

Remark: Send one command to reader with TLP224 protocol.

Return: None.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Cmd: Command code. Ex: '9', 09H.

DataLen: The data length that follow command.

DataBuf: Buffer for store data.

VB Syntax:

Sub vbTLP224_SendData (Port As Long, strCmd as String, strDataBuf As String)

TLP224_ReceiveData ()

Syntax: int TLP224_ReceiveData (int Port, BYTE *RcvData, int DelayTime)

Remark: Waits for the response data from reader with TLP224 protocol and receives it. This function will return timeout error if no data responded from the reader.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

-4 Over buffer. (512 bytes)

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

RcvData: Data buffer address to receive.

DelayTime: Wait receives response time. (ms)

VB Syntax:

Function vbTLP224_ReceiveData (Port As Long, strRcvData As String, lDelayTime as long)
As Long

TLP224_Exchange ()

Syntax: int TLP224_Exchange (int Port, BYTE Cmd, BYTE SndDataLen, BYTE SndData [],
BYTE RcvData [], int DelayTime)

Remark: Send command and receive data from reader with TLP224 protocol.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

-4 Over buffer. (512 bytes)

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Cmd: Command code. Ex: '9', 09H.

SndDataLen: The data length that follow command.

SndData: Buffer for store data.

RcvData: Data buffer address to receive.

DelayTime: Wait receives response time. (ms)

VB Syntax:

Function vbTLP224_Exchange (Port As Long, strCmd As String,
strSndData As String, strRcvData As String, lDelayTime as long) As Long

TLP224 Turbo Protocol

TLP224Turbo_SendData ()

Syntax: void TLP224Turbo_SendData (int Port, BYTE Cmd, BYTE DataLen,
BYTE DataBuf [])

Remark: Send one command to reader with TLP224Turbo protocol.

Return: None.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Cmd: Command code. Ex: '9', 09H.

DataLen: The data length that follow command.

DataBuf: Buffer for store data.

VB Syntax:

Sub vbTLP224Turbo_SendData (Port As Long, strCmd as String, strDataBuf As String)

TLP224Turbo_ReceiveData ()

Syntax: int TLP224Turbo_ReceiveData (int Port, BYTE *RcvData, int DelayTime)

Remark: Waits for the response data from reader with TLP224Turbo_ReceiveData protocol and receives it. This function will return timeout error if no data responded from the reader.

Return: Received data length, if returned value >= 0.

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

-4 Over buffer. (512 bytes)

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

RcvData: Data buffer address to receive.

DelayTime: Wait receives response time. (ms)

VB Syntax:

Function vbTLP224Turbo_ReceiveData (Port As Long, strRcvData As String, lDelayTime as
long) As Long

TLP224Turbo_Exchange ()

Syntax: int TLP224Turbo_Exchange (int Port, BYTE Cmd, BYTE SndDataLen, BYTE SndData [],
BYTE RcvData [], int DelayTime)

Remark: Send command and receive data from reader with TLP224 Turbo protocol.

Return: Received data length, if returned value >= 0.

- 1 Timeout, Reader had no response.
- 2 LRC error of the response data.
- 4 Over buffer. (512 bytes)

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Cmd: Command code. Ex: '9', 09H.

SndDataLen: The data length that follow command.

SndData: Buffer for store data.

RcvData: Data buffer address to receive.

DelayTime: Wait receives response time. (ms)

VB Syntax:

Function vbTLP224Turbo_Exchange (Port As Long, strCmd As String,
strSndData As String, strRcvData As String, lDelayTime as long) As Long

USI1 Protocol

USI1_SendData ()

Syntax: void UIS1_SendData (int Port, BYTE Cmd, USHORT DataLen, BYTE DataBuf [])

Remark: Send one command to reader with USI1 protocol.

Return: None.

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Cmd: Command code. Ex: '9', 09H.

DataLen: The data length that follow command.

DataBuf: Buffer for store data.

VB Syntax:

Sub vbUIS1_SendData (Port As Long, strCmd as String, strDataBuf As String)

UIS1_ReceiveData ()

Syntax: int UIS1_ReceiveData (int Port, BYTE *RcvData, int DelayTime)

Remark: Waits for the response data from reader with USI1 protocol and receives it. This function will return timeout error if no data responded from the reader.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

-4 Over buffer. (512 bytes)

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

RcvData: Data buffer address to receive.

DelayTime: Wait receives response time. (ms)

VB Syntax:

Function vbUIS1_ReceiveData (Port As Long, strRcvData As String, lDelayTime as long) As
Long

TLP224Turbo_Exchange ()

Syntax: int UIS1_Exchange (int Port, BYTE Cmd, USHORT SndDataLen, BYTE SndData [],
BYTE RcvData [], int DelayTime)

Remark: Send command and receive data from reader with USI1 protocol.

Return: Received data length, if returned value ≥ 0 .

-1 Timeout, Reader had no response.

-2 LRC error of the response data.

-4 Over buffer. (512 bytes)

Parameters:

Port: COM port ID. 0 is COM1, 1 is COM2.... The maximal value is 15(COM16).

Cmd: Command code. Ex: '9', 09H.

SndDataLen: The data length that follow command.

SndData: Buffer for store data.

RcvData: Data buffer address to receive.

DelayTime: Wait receives response time. (ms)

VB Syntax:

Function vb UIS1_Exchange (Port As Long, strCmd As String,
strSndData As String, strRcvData As String, lDelayTime as long) As Long

Appendix A. Source Code of HCR360.bas

```
'=====
'File Name: HCR360.bas
'Version   : 2A
'=====

Public Declare Function OpenCom Lib "hcr360.dll" (ByVal port As Long, ByVal baud As Long,
    ByVal DataBits As Long) As Long
Public Declare Sub CloseCom Lib "hcr360.dll" (ByVal port As Long)

'Usi2
Public Declare Sub usi2SendData Lib "hcr360.dll" (ByVal port As Long, ByVal Cmd As Byte,
    ByVal DataLen As Integer, DataBuf As Byte)
Public Declare Function usi2ReceiveData Lib "hcr360.dll" (ByVal port As Long, RcvData As Byte)
    As Long
Public Declare Function usi2Exchange Lib "hcr360.dll" (ByVal port As Long, ByVal Cmd As Byte,
    ByVal SndDataLen As Integer, SndData As Byte, RcvData As Byte) As Long
Public Declare Function usi2CheckCardPresence Lib "hcr360.dll" (ByVal port As Long) As Long
Public Declare Function usi2CardMoveNotify Lib "hcr360.dll" (ByVal port As Long, RcvData As
    Byte) As Long

'Reader command
Public Declare Function usi2Retransmit Lib "hcr360.dll" (ByVal port As Long, RcvData As Byte)
    As Long
Public Declare Function usi2ReaderVerReport Lib "hcr360.dll" (ByVal port As Long, ByVal
    SubCmd As Byte, RcvData As Byte) As Long
Public Declare Function usi2SwitchStatus Lib "hcr360.dll" (ByVal port As Long) As Long
Public Declare Function usi2SetVerboseMode Lib "hcr360.dll" (ByVal port As Long) As Long
Public Declare Function usi2ReaderStatus Lib "hcr360.dll" (ByVal port As Long, RcvData As Byte)
    As Long
Public Declare Function usi2SetGreenLed Lib "hcr360.dll" (ByVal port As Long, ByVal LedStatus
    As Byte) As Long
Public Declare Function usi2SetRedLed Lib "hcr360.dll" (ByVal port As Long, ByVal LedStatus
    As Byte) As Long
Public Declare Function usi2Latch Lib "hcr360.dll" (ByVal port As Long, RcvData As Byte) As
    Long
Public Declare Function usi2Unlatch Lib "hcr360.dll" (ByVal port As Long, RcvData As Byte) As
```

Long

Public Declare Function usi2ConfigReport Lib "hcr360.dll" (ByVal port As Long, RcvData As Byte) As Long

Public Declare Function usi2WarmReset Lib "hcr360.dll" (ByVal port As Long) As Long

'IC card

Public Declare Function usi2CardPowerUp Lib "hcr360.dll" (ByVal port As Long, Vcc As Byte, ByVal EmvComply As Byte, Atr As Byte) As Long

Public Declare Function usi2CardPowerOff Lib "hcr360.dll" (ByVal port As Long) As Long

Public Declare Function usi2SendApu Lib "hcr360.dll" (ByVal port As Long, ByVal ApduLen As Integer, ApduCmd As Byte, RcvData As Byte, SW As Byte) As Long

Public Declare Function usi2SelectIccConnector Lib "hcr360.dll" (ByVal port As Long, ByVal IccConnector As Byte) As Long

Public Declare Function usi2ExecutePPS Lib "hcr360.dll" (ByVal port As Long, ByVal ICprtcl As Byte, ByVal Fi As Byte, ByVal Di As Byte, RcvData As Byte) As Long

Public Declare Function usi2SetIFSD Lib "hcr360.dll" (ByVal port As Long, ByVal IFSDsize As Byte) As Long

Public Declare Function usi2GetErrorCode Lib "hcr360.dll" (ByVal port As Long) As Long

'Memory card

Public Declare Function usi2SelectMemCardType Lib "hcr360.dll" (ByVal port As Long, ByVal MemCardType As Byte, ByVal PageSize As Byte) As Long

Public Declare Function usi2SendMemCardApu Lib "hcr360.dll" (ByVal port As Long, ByVal ApduLen As Integer, ApduCmd As Byte, RcvData As Byte, SW As Byte) As Long

Public Declare Function usi2ReportMemCardType Lib "hcr360.dll" (ByVal port As Long) As Long

Public Declare Function usi2MemoryVerify Lib "hcr360.dll" (ByVal port As Long, ByVal Nums As Integer, PassWord As Byte, SW As Byte) As Long

Public Declare Function usi2MemoryAuthenticate Lib "hcr360.dll" (ByVal port As Long, ByVal Nums As Integer, DataBuf As Byte, SW As Byte) As Long

Public Declare Function usi2MemoryExtAuthenticate Lib "hcr360.dll" (ByVal port As Long, ByVal Nums As Integer, DataBuf As Byte, SW As Byte) As Long

Public Declare Function usi2MemoryReadData Lib "hcr360.dll" (ByVal port As Long, ByVal Adds As Integer, ByVal Nums As Integer, RcvData As Byte, SW As Byte) As Long

Public Declare Function usi2MemoryWriteData Lib "hcr360.dll" (ByVal port As Long, ByVal Adds As Integer, ByVal Nums As Integer, DataBuf As Byte, SW As Byte) As Long

Public Declare Function usi2MemoryEraseData Lib "hcr360.dll" (ByVal port As Long, ByVal Adds As Integer, SW As Byte) As Long

Public Declare Function usi2MemoryRestoreData Lib "hcr360.dll" (ByVal port As Long, ByVal

Adds As Integer, SW As Byte) As Long

Public Declare Function usi2MemoryWriteProtect Lib "hcr360.dll" (ByVal port As Long, ByVal Adds As Integer, ByVal Nums As Integer, DataBuf As Byte, SW As Byte) As Long

Public Declare Function usi2MemoryReadProtect Lib "hcr360.dll" (ByVal port As Long, ByVal Adds As Integer, ByVal Nums As Integer, RcvData As Byte, SW As Byte) As Long

Public Declare Function usi2MemoryEraseUserArea Lib "hcr360.dll" (ByVal port As Long, ByVal Nums As Integer, ErsPSC As Byte, SW As Byte) As Long

Public Declare Function usi2MemoryChangePSC Lib "hcr360.dll" (ByVal port As Long, ByVal Nums As Integer, NewPSC As Byte, SW As Byte) As Long

Public Declare Function usi2MemoryReadErrCount Lib "hcr360.dll" (ByVal port As Long, ErrCount As Byte, SW As Byte) As Long

'Mag card cmd

Public Declare Function usi2ArmtoRead Lib "hcr360.dll" (ByVal port As Long) As Long

Public Declare Function usi2Abort Lib "hcr360.dll" (ByVal port As Long) As Long

Public Declare Function usi2ReadMagData Lib "hcr360.dll" (ByVal port As Long, ByVal selectTK As Byte, RcvData As Byte) As Long

Public Declare Function usi2TK1Data Lib "hcr360.dll" (ByVal port As Long, RcvData As Byte) As Long

Public Declare Function usi2TK2Data Lib "hcr360.dll" (ByVal port As Long, RcvData As Byte) As Long

Public Declare Function usi2TK3Data Lib "hcr360.dll" (ByVal port As Long, RcvData As Byte) As Long

Public Declare Function usi2ISOData Lib "hcr360.dll" (ByVal port As Long, ByVal Track As Byte, ByVal viaISO As Byte, RcvData As Byte) As Long

Public Declare Function usi2TransmitErrData Lib "hcr360.dll" (ByVal port As Long, RcvData As Byte) As Long

Public Declare Function usi2ForwardCusData Lib "hcr360.dll" (ByVal port As Long, ByVal Track As Byte, ByVal viaNull As Byte, ByVal chrNums As Byte, RcvData As Byte) As Long

Public Declare Function usi2ReverseCusData Lib "hcr360.dll" (ByVal port As Long, ByVal Track As Byte, ByVal viaNull As Byte, ByVal chrNums As Byte, RcvData As Byte) As Long

'TLP224Turbo Protocol

Public Declare Sub TLP224Turbo_SendData Lib "hcr360.dll" (ByVal port As Long, ByVal Cmd As Byte, ByVal DataLen As Integer, DataBuf As Byte)

Public Declare Function TLP224Turbo_ReceiveData Lib "hcr360.dll" (ByVal port As Long, RcvData As Byte, ByVal lDelayTime as Long) As Long

Public Declare Function TLP224Turbo_Exchange Lib "hcr360.dll" (ByVal port As Long, ByVal

Cmd As Byte, ByVal SndDataLen As Integer, SndData As Byte, RcvData As Byte, ByVal
lDelayTime as Long) As Long

'TLP224 Protocol

Public Declare Sub TLP224_SendData Lib "hcr360.dll" (ByVal port As Long, ByVal Cmd As Byte,
ByVal DataLen As Integer, DataBuf As Byte)

Public Declare Function TLP224_ReceiveData Lib "hcr360.dll" (ByVal port As Long, RcvData As
Byte, ByVal lDelayTime as Long) As Long

Public Declare Function TLP224_Exchange Lib "hcr360.dll" (ByVal port As Long, ByVal Cmd As
Byte, ByVal SndDataLen As Integer, SndData As Byte, RcvData As Byte, ByVal
lDelayTime as Long) As Long

'USI1 Protocol

Public Declare Sub USI1_SendData Lib "hcr360.dll" (ByVal port As Long, ByVal Cmd As Byte,
ByVal DataLen As Integer, DataBuf As Byte)

Public Declare Function USI1_ReceiveData Lib "hcr360.dll" (ByVal port As Long, RcvData As
Byte, ByVal lDelayTime as Long) As Long

Public Declare Function USI1 Lib "hcr360.dll" (ByVal port As Long, ByVal Cmd As Byte, ByVal
SndDataLen As Integer, SndData As Byte, RcvData As Byte, ByVal lDelayTime as Long)
As Long

' COM port codes

Global Const COM1 = 0

Global Const COM2 = 1

Global Const COM3 = 2

Global Const COM4 = 3

Global Const COM5 = 4

Global Const COM6 = 5

Global Const COM7 = 6

Global Const COM8 = 7

Global Const COM9 = 8

Global Const COM10 = 9

Global Const COM11 = 10

Global Const COM12 = 11

Global Const COM13 = 12

Global Const COM14 = 13

Global Const COM15 = 14

Global Const COM16 = 15

' Parity Codes

Global Const Even7Parity = 0

Global Const Odd7Parity = 1

Global Const Mark7Parity = 2

Global Const Space7Parity = 3

Global Const No8Parity = 4

' baud codes

Global Const Baud1200 = 2

Global Const Baud2400 = 3

Global Const Baud4800 = 4

Global Const Baud9600 = 5

Global Const Baud19200 = 6

Global Const Baud38400 = 7

Global Const Baud57600 = 8

Global Const Baud115200 = 9

' Define IC Card VCC

Public Const V1_8 As Byte = 1

Public Const V3 As Byte = 3

Public Const V5 As Byte = 5

' Geneneral Purpose Functions

' Bin_ary [31][32][33][34][35] -> Hex_str "3132333435"

Public Function BinToHexStr(baInPut() As Byte, ByVal nLen As Integer) As String

Dim buf As String

Dim i As Integer

Dim Value As Byte

For i = 0 To (nLen - 1)

Value = baInPut(i)

If Value <= 15 Then

buf = buf + "0" + Hex\$(Value)

Else

```

        buf = buf + Hex$(Value)
    End If
Next i
BinToHexStr$ = buf 'Modify by Steven
End Function

```

```

'-----
' Hex_str "3132333435" -> Bin_ary [31][32][33][34][35]
'-----

```

```
Public Function HexStrToBin(strInput As String, baOutPut() As Byte) As Integer
```

```
    Dim i As Integer, j As Integer
```

```
    Dim nStrLen As Integer
```

```
    Dim buf As String
```

```
    nStrLen = Len(strInput)
```

```
    j = 0
```

```
    For i = 1 To nStrLen Step 2
```

```
        buf$ = "&H" + Mid$(strInput$, i, 2)
```

```
        baOutPut(j) = Val(buf$)
```

```
        j = j + 1
```

```
    Next i
```

```
    HexStrToBin% = nStrLen / 2
```

```
End Function
```

```

'-----
' Bin_ary [31][32][33][34][35][00] -> Asc_str "12345<00>"
'-----

```

```
Public Function BinAryToAsciiStr(baInPut() As Byte, ByVal nLen As Integer) As String
```

```
    Dim buf As String
```

```
    Dim i As Integer
```

```
    Dim Value As Byte
```

```
    For i = 0 To (nLen - 1)
```

```
        Value = baInPut(i)
```

```
        If Value <= 15 Then
```

```
            buf = buf + "<0" + Hex$(Value) + ">"
```

```
        ElseIf Value <= 31 Or Value >= 127 Then
```

```
        buf = buf + "<" + Hex$(Value) + ">"
    Else
        buf = buf + Chr(Value)
    End If
Next i
BinAryToAsciiStr = buf 'Modify by Jay
End Function

'-----
' Hex_str "313233343500" -> Asc_str "12345<00>"
'-----

Public Function HexStrToAsciiStr(HexStr As String) As String
    Dim buf As String
    Dim i As Integer
    Dim Value As Byte

    For i = 1 To Len(HexStr) Step 2
        Value = Val("&H" + Mid(HexStr, i, 2))
        If Value <= 15 Then
            buf = buf & "<0" & Hex(Value) & ">"
        ElseIf Value <= 31 Or Value >= 127 Then
            buf = buf + "<" + Hex(Value) + ">"
        Else
            buf = buf + Chr(Value)
        End If
    Next i
    HexStrToAsciiStr = buf 'Modify by Jay
End Function

'-----
' communication
'-----

Public Function vbOpenCom(port As Long, baud As Long, DataBits As Long) As Long
    vbOpenCom = OpenCom(port, baud, No8Parity)
End Function

Public Sub vbCloseCom(port As Long)
    Call CloseCom(port)
End Sub
```

,

' Usi2 functions for VB

,

'-----

'Usi2 Send data, receive data, exchange data

'-----

Public Sub vbSendData(port As Long, strCmd As String, strDataBuf As String)

 Dim bCmd As Byte

 Dim baSend(512) As Byte

 Dim DataLen As Integer

 Dim Rtn As Long

 bCmd = Val("&H" & strCmd)

 DataLen = HexStrToBin(strDataBuf, baSend)

 Call usi2SendData(port, bCmd, DataLen, baSend(0))

End Sub

Public Function vbReceiveData(port As Long, strRcvData As String) As Long

 Dim baReceive(512) As Byte

 Dim Rtn As Long

 Rtn = usi2ReceiveData(port, baReceive(0))

 If Rtn >= 0 Then

 strRcvData = BinToHexStr(baReceive, Rtn)

 Rtn = Rtn * 2 'String length unpack to 2 times

 End If

 vbReceiveData = Rtn

End Function

```
Public Function vbExchange(port As Long, strCmd As String, _  
                           strSndData As String, strRcvData As String) As Long
```

```
    Dim bCmd As Byte  
    Dim baSend(512) As Byte  
    Dim baReceive(512) As Byte  
    Dim DataLen As Integer  
    Dim Rtn As Long
```

```
    bCmd = Val("&H" & strCmd)
```

```
    DataLen = HexStrToBin(strSndData, baSend)
```

```
    Rtn = usi2Exchange(port, bCmd, DataLen, baSend(0), baReceive(0))
```

```
    If Rtn >= 0 Then  
        strRcvData = BinToHexStr(baReceive, Rtn)  
        Rtn = Rtn * 2    'String length unpack to 2 times  
    End If
```

```
    vbExchange = Rtn
```

```
End Function
```

```
Public Function vbCheckCardPresence(port As Long) As Long
```

```
    vbCheckCardPresence = usi2CheckCardPresence(port)
```

```
End Function
```

```
Public Function vbusi2CardMoveNotify(port As Long, strRcvData As String) As Long
```

```
    Dim baReceive(512) As Byte
```

```
    Dim Rtn As Long
```

```
    Rtn = usi2CardMoveNotify(port, baReceive(0))
```

```
    If Rtn >= 0 Then  
        strRcvData = BinToHexStr(baReceive, Rtn)  
        Rtn = Rtn * 2    'String length unpack to 2 times
```

End If

Vbusi2CardMoveNotify = Rtn

End Function

'-----
'Usi2 Reader command
'-----

Public Function vbReaderVerReport(port As Long, strCmd As String, strRcvData As String) As Long

'strCmd is Hex string -> "39"

Dim bCmd As Byte

Dim baReceive(128) As Byte

Dim Rtn As Long

If strCmd <> "" Then

 bCmd = Val("&H" & strCmd)

Else

 bCmd = 0

End If

Rtn = usi2ReaderVerReport(port, bCmd, baReceive(0))

If Rtn >= 0 Then

 strRcvData = BinToHexStr(baReceive, Rtn)

 Rtn = Rtn * 2 'String length unpack to 2 times

End If

vbReaderVerReport = Rtn

End Function

Public Function vbRetransmit(port As Long, strRcvData As String) As Long

Dim baReceive(512) As Byte

Dim Rtn As Long

Rtn = usi2Retransmit(port, baReceive(0))

```
If Rtn >= 0 Then
    strRcvData = BinToHexStr(baReceive, Rtn)
    Rtn = Rtn * 2    'String length unpack to 2 times
End If
```

```
vbRetransmit = Rtn
End Function
```

```
Public Function vbusi2SwitchStatus(port As Long) As Long
```

```
    Vbusi2SwitchStatus = usi2SwitchStatus(port)
End Function
```

```
Public Function vbusi2SetVerboseMode(port As Long) As Long
```

```
    Vbusi2SetVerboseMode = usi2SetVerboseMode(port)
End Function
```

```
Public Function vbusi2ReaderStatus(port As Long, strRcvData As String) As Long
Dim baReceive(4) As Byte
Dim Rtn As Long
```

```
    Rtn = usi2ReaderStatus(port, baReceive(0))

    If Rtn >= 0 Then
        strRcvData = BinToHexStr(baReceive, Rtn)
        Rtn = Rtn * 2    'String length unpack to 2 times
    End If
```

```
    Vbusi2ReaderStatus = Rtn
End Function
```

```
Public Function vbGreenLedON(port As Long) As Long
```

```
    vbGreenLedON = usi2SetGreenLed(port, 1) '1=ON
End Function
```

```
Public Function vbGreenLedOFF(port As Long) As Long
```

```
vbGreenLedOFF = usi2SetGreenLed(port, 0) '0=off
```

```
End Function
```

```
Public Function vbGreenLedFLASH(port As Long) As Long
```

```
vbGreenLedFLASH = usi2SetGreenLed(port, 2) '2=flash
```

```
End Function
```

```
Public Function vbRedLedON(port As Long) As Long
```

```
vbRedLedON = usi2SetRedLed(port, 1)
```

```
End Function
```

```
Public Function vbRedLedOFF(port As Long) As Long
```

```
vbRedLedOFF = usi2SetRedLed(port, 0)
```

```
End Function
```

```
Public Function vbRedLedFLASH(port As Long) As Long
```

```
vbRedLedFLASH = usi2SetRedLed(port, 2)
```

```
End Function
```

```
Public Function vbusi2Latch(port As Long, strRcvData As String) As Long
```

```
Dim baReceive(50) As Byte
```

```
Dim Rtn As Long
```

```
Rtn = usi2Latch(port, baReceive(0))
```

```
If Rtn >= 0 Then
```

```
    strRcvData = BinToHexStr(baReceive, Rtn)
```

```
    Rtn = Rtn * 2    'String length unpack to 2 times
```

```
End If
```

```
Vbusi2Latch = Rtn
```

```
End Function
```

Public Function vbusi2Unlatch(port As Long, strRcvData As String) As Long

Dim baReceive(50) As Byte

Dim Rtn As Long

Rtn = usi2Unlatch(port, baReceive(0))

If Rtn >= 0 Then

strRcvData = BinToHexStr(baReceive, Rtn)

Rtn = Rtn * 2 'String length unpack to 2 times

End If

Vbusi2Unlatch = Rtn

End Function

Public Function vbusi2ConfigReport(port As Long, strRcvData As String) As Long

Dim baReceive(32) As Byte

Dim Rtn As Long

Rtn = usi2ConfigReport(port, baReceive(0))

If Rtn >= 0 Then

strRcvData = BinToHexStr(baReceive, Rtn)

Rtn = Rtn * 2 'String length unpack to 2 times

End If

vbusiConfigReport = Rtn

End Function

Public Function vbusi2WarmReset(port As Long) As Long

Vbusi2WarmReset = usi2WarmReset(port)

End Function

'-----

'Usi2 Magnetic card

'-----

Public Function vbusi2ArmtoRead(port As Long) As Long

```
Vbusi2ArmtoRead = usi2ArmtoRead(port)
```

```
End Function
```

```
Public Function vbusi2Abort(port As Long) As Long
```

```
    Vb2usiAbort = usi2Abort(port)
```

```
End Function
```

```
Public Function vbusi2ReadMagData(port As Long, selectTK As Integer, strRcvData As String) As Long
```

```
Dim baReceive(256) As Byte
```

```
Dim Rtn As Long
```

```
    Rtn = usi2ReadMagData(port, CByte(selectTK), baReceive(0))
```

```
    If Rtn >= 0 Then
```

```
        strRcvData = BinToHexStr(baReceive, Rtn)
```

```
        Rtn = Rtn * 2    'String length unpack to 2 times
```

```
    End If
```

```
    Vbusi2ReadMagData = Rtn
```

```
End Function
```

```
Public Function vbusi2TK1Data(port As Long, strRcvData As String) As Long
```

```
Dim baReceive(256) As Byte
```

```
Dim Rtn As Long
```

```
    Rtn = usi2TK1Data(port, baReceive(0))
```

```
    If Rtn >= 0 Then
```

```
        strRcvData = BinToHexStr(baReceive, Rtn)
```

```
        Rtn = Rtn * 2    'String length unpack to 2 times
```

```
    End If
```

```
    Vbusi2TK1Data = Rtn
```

```
End Function
```

```
Public Function vbusi2TK2Data(port As Long, strRcvData As String) As Long
```

```
Dim baReceive(256) As Byte
```

```
Dim Rtn As Long
```

```
Rtn = usi2TK2Data(port, baReceive(0))
```

```
If Rtn >= 0 Then
```

```
    strRcvData = BinToHexStr(baReceive, Rtn)
```

```
    Rtn = Rtn * 2    'String length unpack to 2 times
```

```
End If
```

```
Vbusi2TK2Data = Rtn
```

```
End Function
```

```
Public Function vbusi2TK3Data(port As Long, strRcvData As String) As Long
```

```
Dim baReceive(256) As Byte
```

```
Dim Rtn As Long
```

```
    Rtn = usi2TK3Data(port, baReceive(0))
```

```
If Rtn >= 0 Then
```

```
    strRcvData = BinToHexStr(baReceive, Rtn)
```

```
    Rtn = Rtn * 2    'String length unpack to 2 times
```

```
End If
```

```
Vbusi2TK3Data = Rtn
```

```
End Function
```

```
Public Function vbusi2ISOData(port As Long, Track As Integer, viaISO As Integer, _  
                                strRcvData As String) As Long
```

```
Dim baReceive(256) As Byte
```

```
Dim Rtn As Long
```

```
    Rtn = usi2ISOData(port, CByte(Track), viaISO, baReceive(0))
```

```
If Rtn >= 0 Then
```

```
    strRcvData = BinToHexStr(baReceive, Rtn)
```

```
    Rtn = Rtn * 2    'String length unpack to 2 times
```

```
End If
```

```
Vbusi2ISOData = Rtn
```

```
End Function
```

```
Public Function vbusi2TransmitErrData(port As Long, strRcvData As String) As Long
```

```
Dim baReceive(1024) As Byte
```

```
Dim Rtn As Long
```

```
    Rtn = usi2TransmitErrData(port, baReceive(0))
```

```
    If Rtn >= 0 Then
```

```
        strRcvData = BinToHexStr(baReceive, Rtn)
```

```
        Rtn = Rtn * 2    'String length unpack to 2 times
```

```
    End If
```

```
    Vbusi2TransmitErrData = Rtn
```

```
End Function
```

```
Public Function vbusi2ForwardCusData(port As Long, Track As Integer, viaNull As Integer, _
```

```
    chrNums As Integer, strRcvData As String) As Long
```

```
Dim baReceive(256) As Byte
```

```
Dim Rtn As Long
```

```
    Rtn = usi2ForwardCusData(port, CByte(Track), CByte(viaNull), CByte(chrNums),  
        baReceive(0))
```

```
    If Rtn >= 0 Then
```

```
        strRcvData = BinToHexStr(baReceive, Rtn)
```

```
        Rtn = Rtn * 2    'String length unpack to 2 times
```

```
    End If
```

```
    Vbusi2ForwardCusData = Rtn
```

```
End Function
```

```
Public Function vbusi2ReverseCusData(port As Long, Track As Integer, viaNull As Integer, _
```

```
    chrNums As Integer, strRcvData As String) As Long
```

```
Dim baReceive(256) As Byte
```

```
Dim Rtn As Long
```

```
    Rtn = usi2ReverseCusData(port, CByte(Track), CByte(viaNull), CByte(chrNums),  
baReceive(0))
```

```
    If Rtn >= 0 Then
```

```

        strRcvData = BinToHexStr(baReceive, Rtn)
        Rtn = Rtn * 2    'String length unpack to 2 times
    End If

```

```

        Vbusi2ReverseCusData = Rtn
    End Function

```

```

'-----
'Usi2 CPU card
'-----
'IccConnector = 0~4
Public Function vbusi2SelectIccConnector(port As Long, IccConnector As String) As Long
    Vbusi2SelectIccConnector = usi2SelectIccConnector(port, CByte(Val(IccConnector)))
End Function

```

```

Public Function vbCardPowerUp(port As Long, Vcc As Integer, EmvComply As Integer, strAtr As
String) As Long
    Dim baAtr(64) As Byte
    Dim Rtn As Long
'BYTE Vcc[] : (1/3/5) V1_8/V3/V5, two byte
'Emv_PPSComply : x0: ISO Mode, x1: EMV Mode
'
                0x: PPS T=0, 1X: PPS T=1

```

```

    Rtn = usi2CardPowerUp(port, CByte(Vcc), CByte(EmvComply), baAtr(0)) 'Vcc = V5 or V3

```

```

    If Rtn >= 0 Then
        strAtr = BinToHexStr(baAtr, Rtn)
        Rtn = Rtn * 2    'String length unpack to 2 times
    End If

```

```

        vbCardPowerUp = Rtn
    End Function

```

```

Public Function vbCardPowerOff(port As Long) As Long
    vbCardPowerOff = usi2CardPowerOff(port)
End Function

```

Public Function vbSendApdu(port As Long, strApdu As String, strRcvData As String, strSW As String) As Long

Dim baApdu(264) As Byte

Dim baReceive(512) As Byte

Dim baSW(8) As Byte

Dim nAPDULen As Integer

Dim Rtn As Long

nAPDULen = HexStrToBin(strApdu, baApdu)

Rtn = usi2SendApdu(port, nAPDULen, baApdu(0), baReceive(0), baSW(0))

If Rtn >= 0 Then

strRcvData = BinToHexStr(baReceive, Rtn)

strSW = BinToHexStr(baSW, 2)

Rtn = Rtn * 2 'String length unpack to 2 times

End If

vbSendApdu = Rtn

End Function

Public Function vbExecutePPS(port As Long, ICprctl As Integer, Fi As Integer, Di As Integer, strRcvData As String) As Long

Dim baReceive(12) As Byte

Dim Rtn As Long

Rtn = usi2ExecutePPS(port, CByte(ICprctl), CByte(Fi), CByte(Di), baReceive(0))

If Rtn >= 0 Then

strRcvData = BinToHexStr(baReceive, Rtn)

Rtn = Rtn * 2 'String length unpack to 2 times

End If

vbExecutePPS = Rtn

End Function

Public Function vbSetIFSD(port As Long, IFSDsize As Integer) As Long

'strIFSDsize = 0 ~ 255

```
vbSetIFSD = usi2SetIFSD(port, CByte(IFSDsize))
```

```
End Function
```

```
Public Function vbGetErrorCode(port As Long) As Long
```

```
vbGetErrorCode = usi2GetErrorCode(port)
```

```
End Function
```

```
'-----
```

```
'Usi2 Memory card
```

```
'-----
```

```
Public Function vbusi2SelectMemCardType(port As Long, CardType As Integer, PageSize As Integer) As Long
```

```
Vbusi2SelectMemCardType = usi2SelectMemCardType(port, CByte(CardType), CByte(PageSize))
```

```
End Function
```

```
Public Function vbusi2SendMemCardApdu(port As Long, strApdu As String, strRcvData As String, strSW As String) As Long
```

```
Dim baApdu(264) As Byte
```

```
Dim baReceive(512) As Byte
```

```
Dim baSW(8) As Byte
```

```
Dim nAPDULen As Integer
```

```
Dim Rtn As Long
```

```
nAPDULen = HexStrToBin(strApdu, baApdu)
```

```
Rtn = usi2SendMemCardApdu(port, nAPDULen, baApdu(0), baReceive(0), baSW(0))
```

```
If Rtn >= 0 Then
```

```
strRcvData = BinToHexStr(baReceive, Rtn)
```

```
strSW = BinToHexStr(baSW, 2)
```

```
Rtn = Rtn * 2 'String length unpack to 2 times
```

```
End If
```

Vbusi2SendMemCardApdu = Rtn

End Function

Public Function vbusi2MemoryVerify(port As Long, strPassWord As String, strSW As String) As Long

Dim baPassWord(16) As Byte

Dim baSW(8) As Byte

Dim nPSWLen As Integer

Dim Rtn As Long

nPSWLen = HexStrToBin(strPassWord, baPassWord)

Rtn = usi2MemoryVerify(port, nPSWLen, baPassWord(0), baSW(0))

If Rtn >= 0 Then

strSW = BinToHexStr(baSW, 2)

Rtn = Rtn * 2 'String length unpack to 2 times

End If

Vbusi2MemoryVerify = Rtn

End Function

Public Function vbusi2MemoryAuthenticate(port As Long, strDataBuf As String, strSW As Byte) As Long

Dim baDataBuf(264) As Byte

Dim baSW(8) As Byte

Dim nDataLen As Integer

Dim Rtn As Long

nDataLen = HexStrToBin(strDataBuf, baDataBuf)

Rtn = usi2MemoryAuthenticate(port, nDataLen, baDataBuf(0), baSW(0))

If Rtn >= 0 Then

strSW = BinToHexStr(baSW, 2)

```
        Rtn = Rtn * 2    'String length unpack to 2 times
    End If
```

```
Vbusi2MemoryAuthenticate = Rtn
```

End Function

```
Public Function vbusi2MemoryExtAuthenticate(port As Long, strDataBuf As String, strSW As
String) As Long
```

```
    Dim baDataBuf(264) As Byte
```

```
    Dim baSW(8) As Byte
```

```
    Dim nDataLen As Integer
```

```
    Dim Rtn As Long
```

```
    nDataLen = HexStrToBin(strDataBuf, baDataBuf)
```

```
    Rtn = usi2MemoryExtAuthenticate(port, nDataLen, baDataBuf(0), baSW(0))
```

```
    If Rtn >= 0 Then
```

```
        strSW = BinToHexStr(baSW, 2)
```

```
        Rtn = Rtn * 2    'String length unpack to 2 times
```

```
    End If
```

```
Vbusi2MemoryExtAuthenticate = Rtn
```

End Function

```
Public Function vbusi2MemoryReadData(port As Long, Adds As Integer, rcvLen As Integer, _
                                     strRcvData As String, strSW As String) As Long
```

```
    Dim baReceive(512) As Byte
```

```
    Dim baSW(8) As Byte
```

```
    Dim Rtn As Long
```

```
    Rtn = usi2MemoryReadData(port, Adds, rcvLen, baReceive(0), baSW(0))
```

```
    If Rtn >= 0 Then
```

```
        strRcvData = BinToHexStr(baReceive, Rtn)
```

```
        strSW = BinToHexStr(baSW, 2)
```

```
        Rtn = Rtn * 2    'String length unpack to 2 times
    End If
```

```
    Vbusi2MemoryReadData = Rtn
End Function
```

```
Public Function vbusi2MemoryWriteData(port As Long, Adds As Integer, _
                                         strDataBuf As String, strSW As String) As Long
```

```
    Dim baDataBuf(512) As Byte
    Dim baSW(8) As Byte
    Dim nDataLen As Integer
    Dim Rtn As Long
```

```
    nDataLen = HexStrToBin(strDataBuf, baDataBuf)
```

```
    Rtn = usi2MemoryWriteData(port, Adds, nDataLen, baDataBuf(0), baSW(0))
```

```
    If Rtn >= 0 Then
        strSW = BinToHexStr(baSW, 2)
        Rtn = Rtn * 2    'String length unpack to 2 times
    End If
```

```
    Vbusi2MemoryWriteData = Rtn
End Function
```

```
Public Function vbusi2MemoryEraseData(port As Long, Adds As Integer, strSW As String) As
Long
```

```
    Dim baSW(8) As Byte
    Dim Rtn As Long
```

```
    Rtn = usi2MemoryEraseData(port, Adds, baSW(0))
```

```
    If Rtn >= 0 Then
        strSW = BinToHexStr(baSW, 2)
        Rtn = Rtn * 2    'String length unpack to 2 times
    End If
```

```
    Vbusi2MemoryEraseData = Rtn
```

End Function

```
Public Function vbusi2MemoryRestoreData(port As Long, Adds As Integer, strSW As String) As Long
```

```
Dim baSW(8) As Byte
```

```
Dim Rtn As Long
```

```
    Rtn = usi2MemoryRestoreData(port, Adds, baSW(0))
```

```
    If Rtn >= 0 Then
```

```
        strSW = BinToHexStr(baSW, 2)
```

```
        Rtn = Rtn * 2    'String length unpack to 2 times
```

```
    End If
```

```
    Vbusi2MemoryRestoreData = Rtn
```

End Function

```
Public Function vbusi2MemoryWriteProtect(port As Long, Adds As Integer, strDataBuf As String,  
    strSW As String) As Long
```

```
Dim baDataBuf(512) As Byte
```

```
Dim baSW(8) As Byte
```

```
Dim nDataLen As Integer
```

```
Dim Rtn As Long
```

```
    nDataLen = HexStrToBin(strDataBuf, baDataBuf)
```

```
    Rtn = usi2MemoryWriteProtect(port, Adds, nDataLen, baDataBuf(0), baSW(0))
```

```
    If Rtn >= 0 Then
```

```
        strSW = BinToHexStr(baSW, 2)
```

```
        Rtn = Rtn * 2    'String length unpack to 2 times
```

```
    End If
```

```
    Vbusi2MemoryWriteProtect = Rtn
```

End Function

Public Function vbusi2MemoryReadProtect(port As Long, Adds As Integer, rcvLen As Integer, _
strRcvData As String, strSW As String) As Long

Dim baReceive(512) As Byte

Dim baSW(8) As Byte

Dim Rtn As Long

Rtn = usi2MemoryReadProtect(port, Adds, rcvLen, baReceive(0), baSW(0))

If Rtn >= 0 Then

strRcvData = BinToHexStr(baReceive, Rtn)

strSW = BinToHexStr(baSW, 2)

Rtn = Rtn * 2 'String length unpack to 2 times

End If

Vbusi2MemoryReadProtect = Rtn

End Function

Public Function vbusi2MemoryEraseUserArea(port As Long, strErsPSC As String, strSW As String)
As Long

Dim baDataBuf(16) As Byte

Dim baSW(8) As Byte

Dim Rtn As Long

Dim nDataLen As Integer

nDataLen = HexStrToBin(strErsPSC, baDataBuf)

Rtn = usi2MemoryEraseUserArea(port, nDataLen, baDataBuf(0), baSW(0))

If Rtn >= 0 Then

strSW = BinToHexStr(baSW, 2)

Rtn = Rtn * 2 'String length unpack to 2 times

End If

Vbusi2MemoryEraseUserArea = Rtn

End Function

Public Function vbusi2MemoryChangePSC(port As Long, strNewPSC As String, strSW As String)

As Long

Dim baDataBuf(12) As Byte

Dim baSW(8) As Byte

Dim Rtn As Long

Dim nDataLen As Integer

nDataLen = HexStrToBin(strNewPSC, baDataBuf)

Rtn = usi2MemoryChangePSC(port, nDataLen, baDataBuf(0), baSW(0))

If Rtn >= 0 Then

 strSW = BinToHexStr(baSW, 2)

 Rtn = Rtn * 2 'String length unpack to 2 times

End If

Vbusi2MemoryChangePSC = Rtn

End Function

Public Function vbusi2MemoryReadErrCount(port As Long, strRcvData As String, strSW As String) As Long

Dim baReceive(20) As Byte

Dim baSW(8) As Byte

Dim Rtn As Long

Dim nDataLen As Integer

Rtn = usi2MemoryReadErrCount(port, baReceive(0), baSW(0))

If Rtn >= 0 Then

 strRcvData = BinToHexStr(baReceive, Rtn)

 strSW = BinToHexStr(baSW, 2)

 Rtn = Rtn * 2 'String length unpack to 2 times

End If

Vbusi2MemoryReadErrCount = Rtn

End Function

'TLP224Turbo Protocol: Send data, receive data, exchange data

Public Sub vbTLP224Turbo_SendData(port As Long, strCmd As String, strDataBuf As String)

Dim bCmd As Byte

Dim baSend(512) As Byte

Dim DataLen As Integer

Dim Rtn As Long

Dim strSendData as String

strSendData = strCmd + strDataBuf

DataLen = HexStrToBin(strSendData, baSend)

Call TLP224Turbo_SendData(port, baSend(0), DataLen, baSend(1))

End Sub

Public Function vbTLP224Turbo_ReceiveData(port As Long, strRcvData As String,
lDelayTime as Long) As Long

Dim baReceive(512) As Byte

Dim Rtn As Long

Rtn = TLP224Turbo_ReceiveData(port, baReceive(0), lDelayTime)

If Rtn >= 0 Then

strRcvData = BinToHexStr(baReceive, Rtn)

Rtn = Rtn * 2 'String length unpack to 2 times

End If

vbTLP224Turbo_ReceiveData = Rtn

End Function

Public Function vbTLP224Turbo_Exchange(port As Long, strCmd As String, _
strSndData As String, strRcvData As String, _
lDelayTime as Long) As Long

```
Dim bCmd As Byte
Dim baSend(512) As Byte
Dim baReceive(512) As Byte
Dim DataLen As Integer
Dim Rtn As Long
Dim strSendData as String
```

```
strSendData = strCmd + strDataBuf
```

```
DataLen = HexStrToBin(strSendData, baSend)
```

```
Rtn = TLP224Turbo_Exchange(port, baSend(0), DataLen, baSend(1), baReceive(0), _
                           lDelayTime)
```

```
If Rtn >= 0 Then
    strRcvData = BinToHexStr(baReceive, Rtn)
    Rtn = Rtn * 2    'String length unpack to 2 times
End If
```

```
vb TLP224Turbo_Exchange = Rtn
```

```
End Function
```

```
'-----
'TLP224 Protocol: Send data, receive data, exchange data
'-----
```

```
Public Sub vbTLP224T_SendData(port As Long, strCmd As String, strDataBuf As String)
```

```
    Dim bCmd As Byte
    Dim baSend(512) As Byte
    Dim DataLen As Integer
    Dim Rtn As Long
    Dim strSendData as String
```

```
    strSendData = strCmd + strDataBuf
```

```
    DataLen = HexStrToBin(strSendData, baSend)
```

End Sub

End Function

Page 64

```
If Rtn >= 0 Then
    strRcvData = BinToHexStr(baReceive, Rtn)
    Rtn = Rtn * 2    'String length unpack to 2 times
End If
```

```
VbTLP224_Exchange = Rtn
```

```
End Function
```

```
'-----
'USI1 Protocol: Send data, receive data, exchange data
'-----
```

```
Public Sub vbUSI1_SendData(port As Long, strDataBuf As String)
```

```
    Dim bCmd As Byte
    Dim baSend(512) As Byte
    Dim DataLen As Integer
    Dim Rtn As Long
```

```
    DataLen = HexStrToBin(strDataBuf, baSend)
```

```
    Call USI1_SendData(port, DataLen, baSend(0))
```

```
End Sub
```

```
Public Function vbUSI1_ReceiveData(port As Long, strRcvData As String,
                                     lDelayTime as Long) As Long
```

```
    Dim baReceive(512) As Byte
    Dim Rtn As Long
```

```
    Rtn = USI1_ReceiveData(port, baReceive(0), lDelayTime)
```

```
If Rtn >= 0 Then
    strRcvData = BinToHexStr(baReceive, Rtn)
    Rtn = Rtn * 2    'String length unpack to 2 times
```

End If

VbUSI1_ReceiveData = Rtn

End Function

```
Public Function vbUSI1_Exchange(port As Long, _  
                                strSndData As String, strRcvData As String, _  
                                lDelayTime as Long) As Long
```

```
    Dim bCmd As Byte
```

```
    Dim baSend(512) As Byte
```

```
    Dim baReceive(512) As Byte
```

```
    Dim DataLen As Integer
```

```
    Dim Rtn As Long
```

```
    DataLen = HexStrToBin(strSndData, baSend)
```

```
    Rtn = USI1_Exchange(port, DataLen, baSend(0), baReceive(0), _  
                        lDelayTime)
```

```
    If Rtn >= 0 Then
```

```
        strRcvData = BinToHexStr(baReceive, Rtn)
```

```
        Rtn = Rtn * 2    'String length unpack to 2 times
```

```
    End If
```

```
    VbUSI1_Exchange = Rtn
```

End Function