·Windows DLL Specification

Provide Windows32 DLL on windows environment platform, files' names as following：

1. CRT_711.H 32 digit dynamic library header file
2. CRT_711.LIB 32 digit input dynamic library
3. CRT_711.DLL Windows32 DLL file

·Applicable Card Type

1. RF Card: ISO/IEC 14443 TYPE A / B, MIFARE one (S50, S70, UL)
2. One-Dimension Barcode:
   39 code, EAN code, UPC code, 128 Code and 93 code.

Notes:

1. Code examples of DLL running on VC6, VB6, DELPHI7, C++BUILD6, PB9, VB2005.NET, V#2005.NET environment are provided.

Contents

1. Function Parameter Description

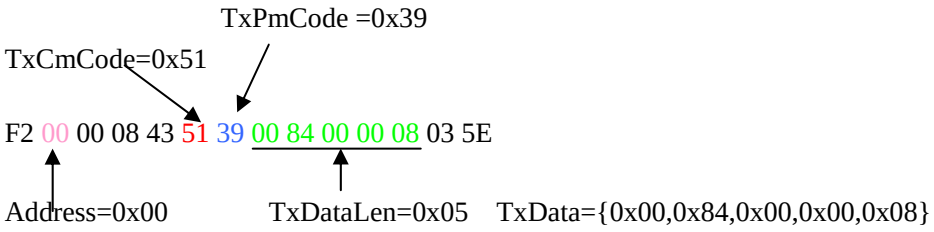
1.1 Device Address Setting

Address setting is implemented by 4 bit DIP switch

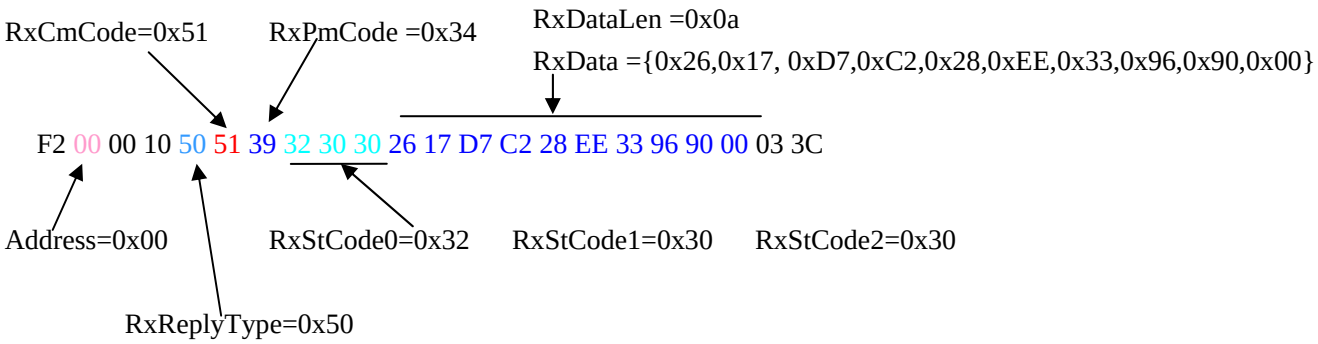
Dip Setting				Address
4	3	2	1	
ON	ON	ON	ON	0x00
ON	ON	ON	OFF	0x01
ON	ON	OFF	ON	0x02
ON	ON	OFF	OFF	0x03
ON	OFF	ON	ON	0x04
ON	OFF	ON	OFF	0x05
ON	OFF	OFF	ON	0x06
ON	OFF	OFF	OFF	0x07
OFF	ON	ON	ON	0x08
OFF	ON	ON	OFF	0x09
OFF	ON	OFF	ON	0x0A
OFF	ON	OFF	OFF	0x0B
OFF	OFF	ON	ON	0x0C
OFF	OFF	ON	OFF	0x0D
OFF	OFF	OFF	ON	0x0E
OFF	OFF	OFF	OFF	0x0F

2.1 Combination of Function Parameter

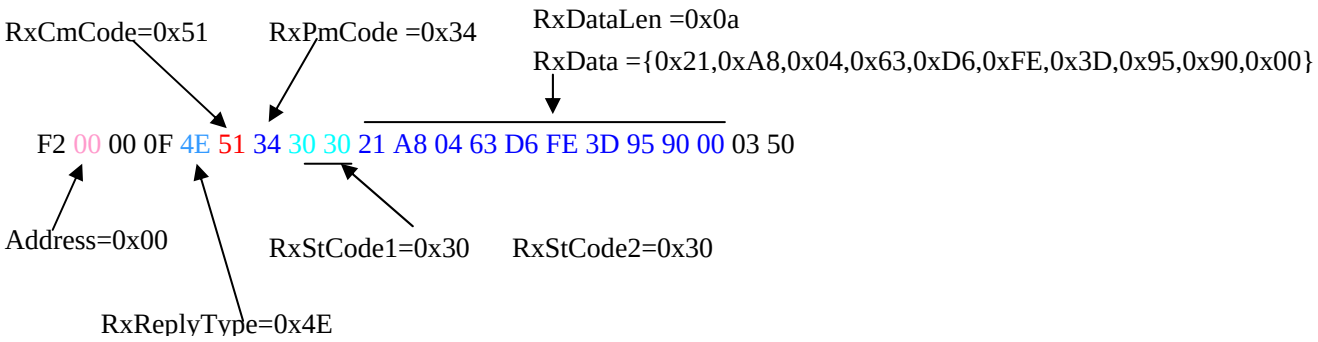
1. Send command format



2 Positive receive command format



3. Negative receive command format



Notes:

Positive Response：RxReplyType=0x50，RxStCode0，RxStCode1，RxStCode2 is status code
Negative Response：RxReplyType=0x4E，RxStCode1，RxStCode2 is error code

2 API Description

2.1 Open Com Port (Default baud rate38400bps)

HANDLE APIENTRY CommOpen(char *Port);

Parameter**:**

Port: Com port number string

E,g:CommOpen("COM1");

Return**:**

- 0 (Com open failure)

  Possible reason for Com port open failure:

  ① Invalid Com Port Number String

  ② This Com is occupied by other devices

- <>0 Com Open Success

Notes:

1) Call this function and return specified handle for Com port before calling of other functions.
2) Enable to open several com ports to get Com port number string for each, but disable to open one port twice simultaneously.
3) Close Com port by calling **CommClose()** after use.

2.2 Open Com Port by Defined Baud Rate

HANDLE APIENTRY CommOpenWithBaut(char *Port, unsigned int Baudrate);

Parameter:

Port: Com Port Number String

Baudrate: Baud Rate

Baud Rate=9600，19200，38400，57600,115200

E.g.: CommOpenWithBaut ("COM1",38400);

Return:

- 0 (Com open failure)

  Possible reason for Com port open failure:

  ① Invalid Com Port Number String

  ② This Com is occupied by other devices
- <>0 Com Open Success

Notes:

1) Call this function and return specified handle for Com port before calling of other functions.
2) Enable to open several com ports to get Com port number string for each, but disable to open one port twice simultaneously.
3) Close Com port by calling **CommClose()** after use.

2.3 Close Com Port

int APIENTRY CommClose(HANDLE ComHandle);;

Parameter:

ComHandle：Com Handel

Return:

=0 Success

<>0 Fail

Notes:

CommOpen () used together with CommClose (), call this function to close Com port

2.4 Execution Command Function

int APIENTRY ExecuteCommand(HANDLE ComHandle,BYTE TxAddr,
BYTE TxCmCode,BYTE TxPmCode,
int TxDataLen,BYTE TxData[],
BYTE *RxReplyType,
BYTE *RxStCode0,BYTE *RxStCode1,BYTE *RxStCode2,
int *RxDataLen,BYTE RxData[])

Function:

Execute Command and Return Result

Parameter:

ComHandle: Com Handle

TxAddr: Device Address; Address Range: 0x00~0x0f

TxCmCode: CM Code

TxPmCode: PM Code

TxDataLen: Command Data Length

TxData: Command Additional package

RxReplyType: Return Type

0x50 : Execute Successful

0x4E : Execute failure

0x10 : Lower machine cancel communication (NAK)

0x20 : Communication failure

0x30 : Upper machine cancel command (EOT)

RxStCode0: Status code 0 **// Specific Information Please Refer to Communication Protocol**

RxStCode1: Status code 1 **// Specific Information Please Refer to Communication Protocol**

RxStCode2: Status code 2 **// Specific Information Please Refer to Communication Protocol**

RxDataLen: Return package length

RxData: Return data

Return:

=0 Successful

<>0 Failure

Notes:

If RxReplyType value is 0x50，RxStCode0，RxStCode1，RxStCode2 is status code

If RxReplyType value is 0x4E，RxStCode1，RxStCode2 is error code, RxStCode0 is pointless

3 Command List

Cm: Command code; Pm: Parameters

Chapter	Command	Function	CM	PM	Description
4.1	INITIALIZE	Initialize CRT-711	30H	30H	If card is inside, move card to front side card holding position
				31H	If card is inside, retract card to error card bin
				33H	If card is inside, does not move the card.
				34H	Same as 30H and eject card/error card bin counter will work.
				35H	Same as 31H and eject card/error card bin counter will work.
				37H	Same as 33H and eject card/error card bin counter will work.
4.2	STATUS REQUEST	Inquire status	31H	30H	Report machine status
				31H	Report sensor status
4.3	CARD MOVE	Card movement	32H	30H	Move card to front side card holding position
				33H	Move card to error card bin
				39H	Move card to front side without card holding position
4.4	CARD ENTRY		33H	30H	Enable card entry from front side
				31H	Disable card entry from front side
4.5	RFID CARD TYPE		50H	31H	Auto check RFID Card Type
4.6	SAM CARD CONTROL	SAM Card Operation	52H	30H	SAM Card cold reset
				31H	SAM Card power down
				32H	SAM Card status check
				33H	T=0 SAM Card APDU data exchange
				34H	T=1 SAM Card APDU data exchange
				38H	SAM Card hot reset
				39H	Auto distinguish T=0/T=1 SAM Card APDU data exchange
				40H	Select SAM card stand

4.7	RFID CARD CONTROL (13.56 MHz)	Mifare standard card Type A & B T=CL protocol operation	60H	30H	RFID Card initialization
				31H	RFID Card power down and release
				32H	RFID Card operation status check
				33H	Mifare standard Card read/write
				34H	Type A standard T=CL Card APDU data exchange
				35H	Type B standard T=CL Card APDU data exchange
				39H	RFID card sleep/wake-up
4.8	Barcode Scan		70H	30H	Barcode Scan
4.9	Read Machine CONFIG information		A3H	30H	Read machine configuration information
4.10	READ machine VERSION		A4H	30H	Read machine software version information
4.11	Error card bin counter		A5H	30H	Read counter of error card bin
				31H	Initiate error card bin counter
4.12	Eject card counter		A6H	30H	Read counter of eject card
				31H	Initiate counter of eject card

3.1 Sample Code (Barcode Operation)

```
        HANDLE hCom;        // Target device's HANDLE
        int rc=0;                    //Result
        unsigned char Addr;
        unsigned char CmCode;
        unsigned char PmCode;
        int    CmDataLen;
        unsigned char CmData[1024];

        unsigned char ReType;
        unsigned char St2;
        unsigned char St1;
        unsigned char St0;
        int    ReDataLen;
        unsigned char ReData[1024];
    // Open Comm. port
    {
        hCom=CommOpenWithBaut("COM1",38400);
        if(hCom<=0)
        {
            // failed
            …
        }
        else
        {
            // successfully.
            …
        }
    }
    // Initialize
    {
        memset(CmData,0x00,sizeof(CmData));
        Addr=0x00;
        CmCode=0x30; // Initialize command
        PmCode=0x34; // Parameter code
        CmDataLen=0; // Data size (bytes)
        rc=ExecuteCommand(hCom,Addr,CmCode,PmCode,CmDataLen,CmData,
                          &ReType,&St0,&St1,&St2,&ReDataLen,ReData);
        if(rc==0)
            // Initialize command successfully finished.
            if (ReType==0x50)
```

```
            {
                // Received positive reply
                …
            }
            else (ReType==0x4e)
            {
               // Received negative reply
               ...
            }
        else
            {
             //Communication Error
               // Initialize command failed.
            …
            }
}

// Setting Card In
{
        memset(CmData,0x00,sizeof(CmData));
        CmCode= 0x33; // Card In
        PmCode= 0x30; //    Enable
        CmDataLen= 0; // Data size
        rc=ExecuteCommand(hCom,Addr,CmCode,PmCode,CmDataLen,CmData,
                                &ReType,&St0,&St1,&St2,&ReDataLen,ReData);
        if(rc!=0) || ReType != 0x50)
        {
            // Command sending failed or command execution failed
                ResErrMsg(St1,St2);       // ResErrMsg is a function to see error reason
                                           //Please refer to communication protocol to see error description,
                …
        }
}
// Request Status
{
        memset(CmData,0x00,sizeof(CmData));
        CmCode= 0x31; // Status request command
        PmCode= 0x30; // Parameter code
        CmDataLen=0; // Data size
        rc=ExecuteCommand(hCom,Addr,CmCode,PmCode,CmDataLen,CmData,
                                &ReType,&St0,&St1,&St2,&ReDataLen,ReData);
        if(rc==0)
            if (ReType==0x50)
```

```
        {
            //Execute Ok
              if(St0 == '2')       // There is a card in ICRW
              {
                  // status code="2xx"
                  // Detected a card inside of ICRW
                  ...
              }
        }
        else (ReType==0x4e)
        {
            //Command execution failed
            ResErrMsg(St1,St2);
            …
        }
    else
        {
              //Communication Error
              …
        }
}
// Scan bar code data
{
    memset(CmData,0x00,sizeof(CmData));
    CmCode= 0x70; // Bar_code card
    PmCode=0x30; // Scan
    CmDataLen=0; // Data size
    rc=ExecuteCommand(hCom,Addr,CmCode,PmCode,CmDataLen,CmData,
                          &ReType,&St0,&St1,&St2,&ReDataLen,ReData);
    if(rc!=0)
    {
        // Transmit failed
        ...
    }
    else
    {
        if (ReType==0x50)
        {
            CString t;
            CString TempBuf="";
            for(int n=0; n<ReDataLen; n++)    // bar code data= start from n=0
            {
                t.Format("%02x",ReData[n]);
```

```
                    TempBuf += t;
                    t="";
                }
                // TempBuf    is the buffer of bar code data
            }
            else //(ReType==0x4e)
            {
                // Command execution failed
                ErrMsg(St1,St2);
            }
        }
    }
    // Move card to front
    {
        memset(CmData,0x00,sizeof(CmData));
        CmCode=0x32; // Move card
        PmCode=0x30; // to front
        CmDataLen=0; // Data size
        rc=ExecuteCommand(hCom,Addr,CmCode,PmCode,CmDataLen,CmData,
                          &ReType,&St0,&St1,&St2,&ReDataLen,ReData);
        if(rc!=0) || ReType != 0x50)
        {
            // Command sending failed or command execution failed
            …
        }
    }
    _EXIT1:
    // Closes communications between the Host Computer and the Card Reader/Writer
    CommClose(hCom);
    _EXIT:
```