CRT-310N DLL Specification

Copyright：CREATOR (CHINA) TECH CO., LTD

Website：www.china-creator.com　E-Mail：hbhcb@china-creator.com

Tel: 0755-26710345 26710456

Document Version：V1.0　Date: 2010-02-26

WINDOWS DLL Specification:

Our company provides DLL for Windows32 and include following files:

1. CRT_310NV3.H	32 digit DLL header file
2. CRT_310NV3.LIB	32 digit input libraries
3. CRT_310NV3.DLL	WINDOWS 32 digit libraries

CRT-310N (V1.0) Support card type :

1. RFID Card: ISO/IEC 14443 TYPE A / B, MIFARE one (S50, S70, UL)
2. IC Card:

   SLE4428, SLE4442

   24C01A, 24C02, 24C04, 24C08, 24C16, 24C32, 24C64

   Contact CPU Card (T=0/T=1）

   SAM/SIM Card（T=0/T=1）
3. Magnetic Card

Notes:

1. Code examples of DLL running on VC6, VB6, DELPHI7, C++BUILD6, PB9, VB2005.NET, V#2005.NET environment are provided.
2. Technical support for DLL running on Linux/Unix (.so) can be provided for a large demand.
3. Technical support for JAVA environment can be provided for a large demand.
4. Technical support for OCX design and Web invocation can be provided for a large demand.
5. For optimal work effectiveness, we recommend you to use power-supply which is 24V and more than 2A(Above 2.5A for Hi-Co)

Contents

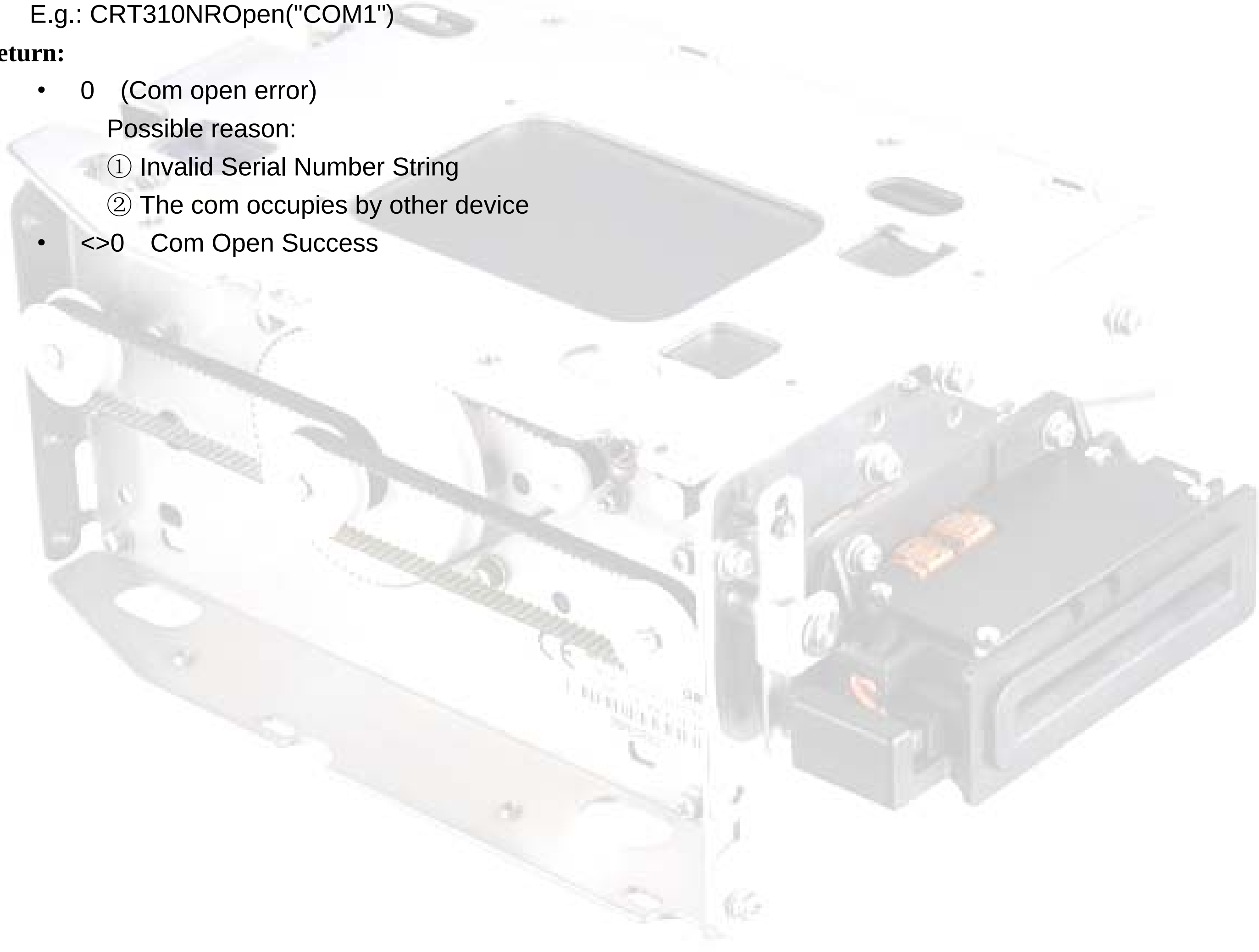
1. API Specification (RS232 Mode)

1.1 Open Com function (Default baud rate38400bps)

HANDLE APIENTRY CRT310NROpen (char *Port)

Parameter:

Port: Serial Number String
E.g.: CRT310NROpen("COM1")



Return:

- 0 (Com open error)
  Possible reason:
  ① Invalid Serial Number String
  ② The com occupies by other device
- <>0 Com Open Success

Notes: 1) Call this function before others.
2) Enable to open several com ports to get serial number string, but disable to open one port twice at same time.
3) Close com port by using CRT310NRClose() after the completion of all operations.

1.2 Open Com Port by Specified baud rate

HANDLE APIENTRY CRT310NROpenWithBaut (char *Port, unsigned int Baud rate);

Parameter:

Port: Serial port string

Baud rate: Baud rate

Baud rate=9600，19200，38400，57600，115200。

Eg: CRT310NROpenWithBaut ("COM1", 38400)

Return:

- 0 (Com open error)

  Possible reason:

  ① Invalid Serial Number String

  ② The com occupies by other device
- <>0 Com Open Success

Notes: 1) Call this function before others.

2) Enable to open several com ports to get serial number string, but disable to open one port twice at same time.

3) Close com port by using CRT310NRClose() after the completion of all operations.

1.3 Close Com Port Function

int APIENTRY CRT310NRClose (HANDLE ComHandle)

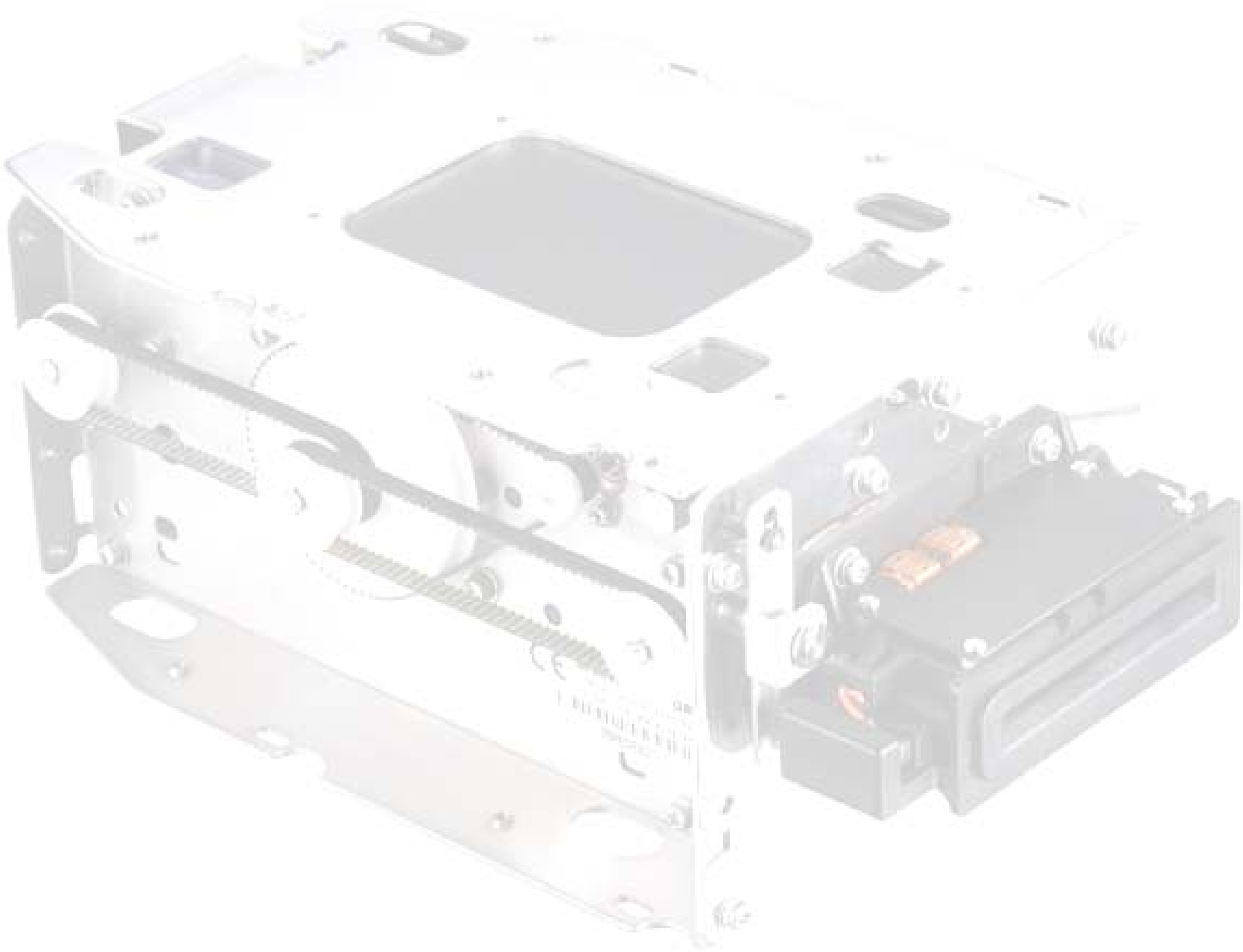
Patameter:

ComHandle：Com Handle

Return:

=0 Success

<>0 Fail



Notes: Combined application with CRT310NROpen () or CRT310NRopenWithBaut function，Close com port after usage of com port.

1.4 Command Function

int APIENTRY RS232_ExeCommand (HANDLE ComHandle, BYTE TxCmCode, BYTETxPmCode, int TxDataLen, BYTE TxData, BYTE *RxReplyType, BYTE *RxStCode1,BYTE *RxStCode0, int *RxDataLen, BYTE RxData);

Function：

Execute command and return the result

Parameter:

ComHandle：Com handle

TxCmCode: Command Code

TxPmCode: Parameter Code

TxDataLen: Extra command data length

TxData: Extra command data package

RxReplyType: Reply Type

0x50 : Execute successful

0x4E : Execute fail

0x10 : Communication is cancel by slave (NAK)

0x20 : Communication failure

0x30 : Command is cancel by HOST (DLE, EOT)

RxStCode1: Return State code 1 // Pls see more detain from Status code and error code of communication protocol

RxStCode0: Return State code 0 //Pls see more detain from Status code and error code of communication protocol

RxDataLen: Return Package length

RxData: Return package data

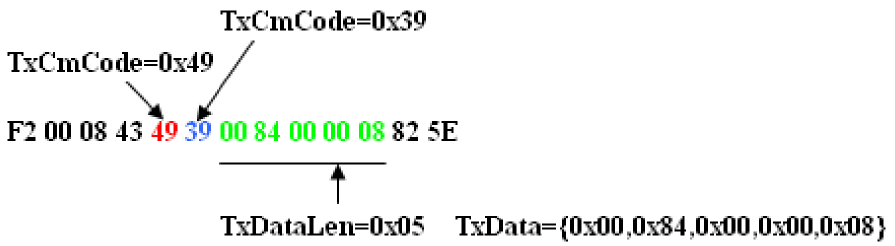
Return:

=0 Successful

<>0 Fail

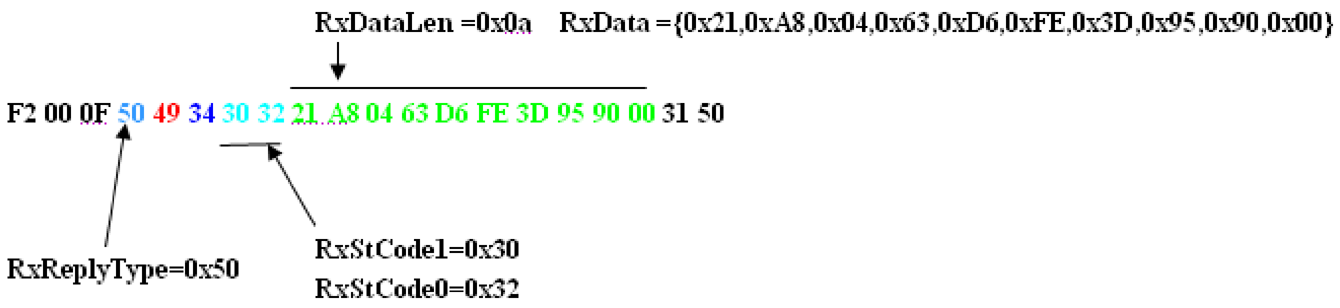
Parameter example base on communication protocol

1. (Send command format)



TxCmCode=0x39
TxCmCode=0x49
F2 00 08 43 49 39 00 84 00 00 08 82 5E
TxDataLen=0x05
TxData={0x00,0x84,0x00,0x00,0x08}

2. (Receive command format)



RxDataLen =0x0a
RxData ={0x21,0xA8,0x04,0x63,0xD6,0xFE,0x3D,0x95,0x90,0x00}
F2 00 0F 50 49 34 30 32 21 A8 04 63 D6 FE 3D 95 90 00 31 50
RxReplyType=0x50
RxStCode1=0x30
RxStCode0=0x32

1.5 IC Card Transportation Command

int APIENTRY RS232_ICCardTransmit (HANDLE ComHandle, BYTE TxCmCode, BYTE TxPmCode, int TxDataLen, BYTE TxData, BYTE *RxReplyType, BYTE *RxStCode1,BYTE *RxStCode0, BYTE *RxCmCode, BYTE *RxPmCode, int *RxDataLen, BYTE RxData)

Function：

Execute command and return

Notes：Comparing to command function（RS232_ExeCommand）, it add two parameters(RxCmCode and RxPmCode)

Parameters:

Parameter:

ComHandle：Com handle

TxCmCode: Command Code

TxPmCode: Parameter Code

TxDataLen: Extra command data length

TxData: Extra command data package

RxReplyType: Reply Type

0x50 : Execute successful

0x4E : Execute fail

0x10 : Communication is cancel by slave (NAK)

0x20 : Communication failure

0x30 : Command is cancel by HOST (DLE, EOT)

RxStCode1: Return State code 1 // Please see more detain from Status code and error code of communication protocol

RxStCode0: Return State code 0 //Please see more detain from Status code and error code of communication protocol

RxDataLen: Return Package length

RxData: Return package data

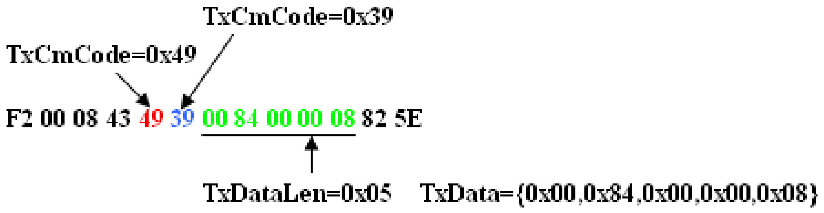
Return:

=0 Successful

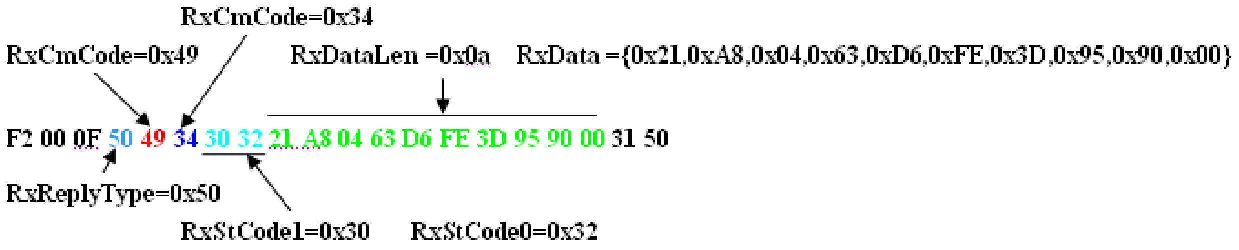
<>0 Fail

Parameter example base on communication protocol

1 (Send command format)


TxCmCode=0x39
TxCmCode=0x49
F2 00 08 43 49 39 00 84 00 00 08 82 5E
TxDataLen=0x05 TxData={0x00,0x84,0x00,0x00,0x08}

2 (Receive command format)


RxCmCode=0x34
RxCmCode=0x49
RxDataLen =0x0a RxData ={0x21,0xA8,0x04,0x63,0xD6,0xFE,0x3D,0x95,0x90,0x00}
F2 00 0F 50 49 34 30 32 21 A8 04 63 D6 FE 3D 95 90 00 31 50
RxReplyType=0x50
RxStCode1=0x30 RxStCode0=0x32

1.6 Cancel Command Function

int APIENTRY CancelCommand (HANDLE ComHandle);

Function：

Cancel recent operation, finish following operation

① Send "DLE,EOT" control code to slave and waiting for "DLE,EOT"

② Cancel the command. For example, card-in command, it can be cancel when the card is inserting by this command.

Parameter:

ComHandle：Com Handle

Return:

=0 Success

=-1 Fail

2 API Specification (USB Mode)

2.1 Open USB Function

HANDLE APIENTRY CRT310NUOpen ();

Parameter:

Eg: CRT310NUopen();

Return:

0 Com open fail

Possible Reason:

① Equipment is not connected, please check the power and communication cable

② Power is not compromised, Please check if the power supply is 24V and above 2A(Write Hi-co need 2.5A)

<>0 Open Com successful

Notes: 1) Call this function before others.

2) Enable to open several com ports to get serial number string, but disable to open one port twice at same time.

3) Close com port by using CRT310NUClose() after the completion of all operations.

2.1 Close USB Function

int APIENTRY CRT310NUClose (HANDLE ComHandle)

Parameter:

ComHandle：Com Handle

Return:

=0 Success

<>0 Fail

Notes : Combine with CRT310NUOpen () function and use this command to close the equipment after operation.

2.3 Command Function

int APIENTRY USB_ExeCommand (HANDLE ComHandle, BYTE TxCmCode, BYTE TxPmCode, int TxDataLen, BYTE TxData, BYTE *RxReplyType, BYTE *RxStCode1, BYTE *RxStCode0, int *RxDataLen, BYTE RxData)

Function：

Execute command and return

Parameter:

ComHandle：Com handle

TxCmCode: Command Code

TxPmCode: Parameter code

TxDataLen: Extra Command data length

TxData: Extra Command data package

RxReplyType: Reply Type

0x50 : Successful

0x4E : Fail

RxStCode1: Return Code 1 // Please see more detain from Status code and error code of communication protocol

RxStCode0: Return Code 0 // Please see more detain from Status code and error code of communication protocol

RxDataLen: Return package length

RxData: Return Data

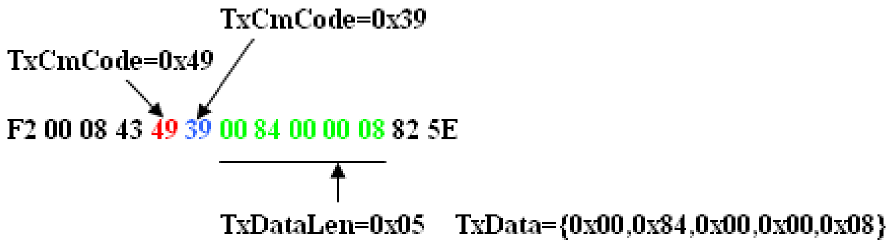
Return:

=0 Success

<>0 Fail (Close USB communication automatically，need to rerun CRT310NUopen to connect equipment)

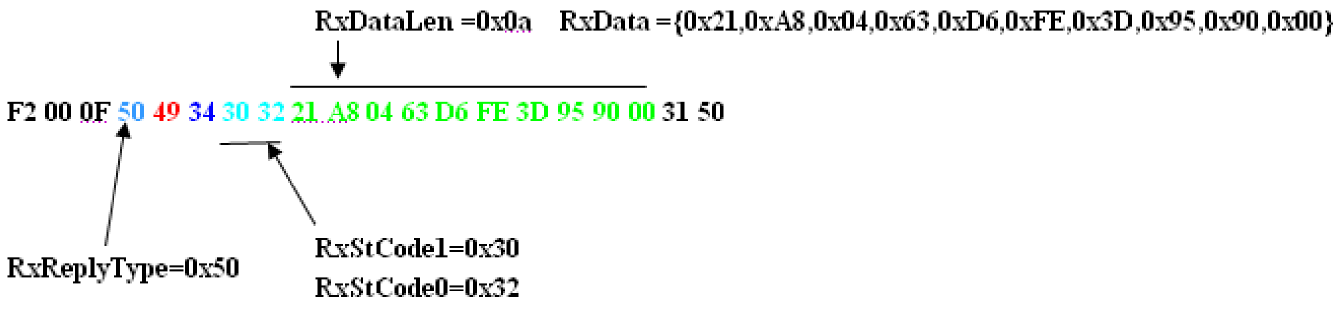
Parameter example is shown as below：

1. (Send command format)



TxCmCode=0x39
TxCmCode=0x49
F2 00 08 43 49 39 00 84 00 00 08 82 5E
TxDataLen=0x05
TxData={0x00,0x84,0x00,0x00,0x08}

2. (Receive command format)



RxDataLen =0x0a
RxData ={0x21,0xA8,0x04,0x63,0xD6,0xFE,0x3D,0x95,0x90,0x00}
F2 00 0F 50 49 34 30 32 21 A8 04 63 D6 FE 3D 95 90 00 31 50
RxReplyType=0x50
RxStCode1=0x30
RxStCode0=0x32

2.4 IC Card Transportation Comman

int APIENTRY USB_ICCardTransmit (HANDLE ComHandle, BYTE TxCmCode, BYTE TxPmCode, int TxDataLen, BYTE TxData, BYTE *RxReplyType, BYTE *RxStCode1,BYTE *RxStCode0, BYTE *RxCmCode, BYTE *RxPmCode, int *RxDataLen, BYTE RxData);

Function：

Execute command and return

Notes：Comparing to（USB_ExeCommand）it add two parameters (RxCmCode and RxPmCode)

Parameter:

ComHandle：Com handle

TxCmCode: Command Code

TxPmCode: Parameter code

TxDataLen: Extra Command data length

TxData: Extra Command data package

RxReplyType: Reply Type

0x50 : Successful

0x4E : Fail

RxStCode1: Return Code 1 // Please see more detain from Status code and error code of communication protocol

RxStCode0: Return Code 0 // Please see more detain from Status code and error code of communication protocol

RxDataLen: Return package length

RxData: Return Data

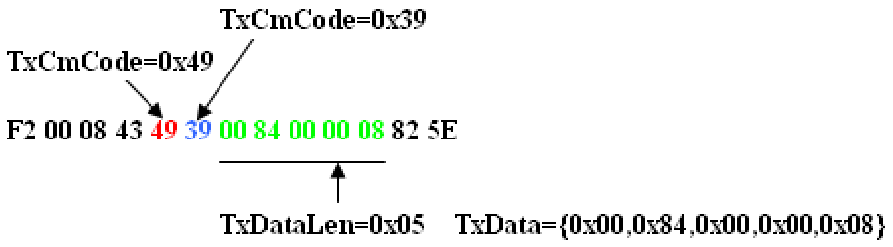
Return:

=0 Success

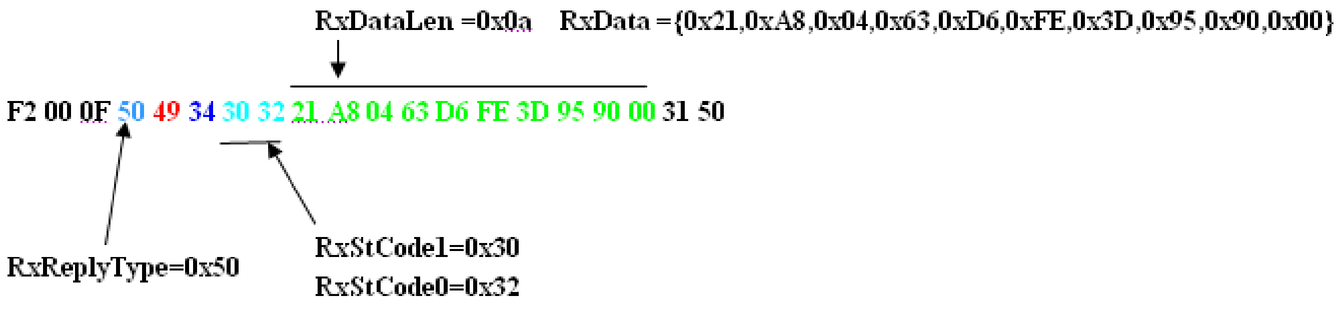
<>0 Fail (Close USB communication automatically，need to rerun CRT310NUopen to connect equipment)

Parameter example is shown as below：

1. (Send command format)


TxCmCode=0x39
TxCmCode=0x49
F2 00 08 43 49 39 00 84 00 00 08 82 5E
TxDataLen=0x05
TxData={0x00,0x84,0x00,0x00,0x08}

2. (Receive command format)


RxDataLen =0x0a
RxData ={0x21,0xA8,0x04,0x63,0xD6,0xFE,0x3D,0x95,0x90,0x00}
F2 00 0F 50 49 34 30 32 21 A8 04 63 D6 FE 3D 95 90 00 31 50
RxReplyType=0x50
RxStCode1=0x30
RxStCode0=0x32

3 Code Example

3.1 Call Function Specification

USB interface is as the same as RS232

1、Equipment Connection

（USB interface call CRT310NUopen function and RS232 interface call CRT310NRopen function）

2、Communication Procedure

USB interface call USB_ExeCommand function and RS232 interface call RS232_ExeCommand function.

USB will close the communication if function call is failure, and need to rerun CRT310NUopen

RS232will not close the communication if function call is failure

3、Close Com Port

（USB interface call CRT310NUClose function and RS232 interface call CRT310NRClose function）

3.2 Code Example 1 (Magnetic card operation)

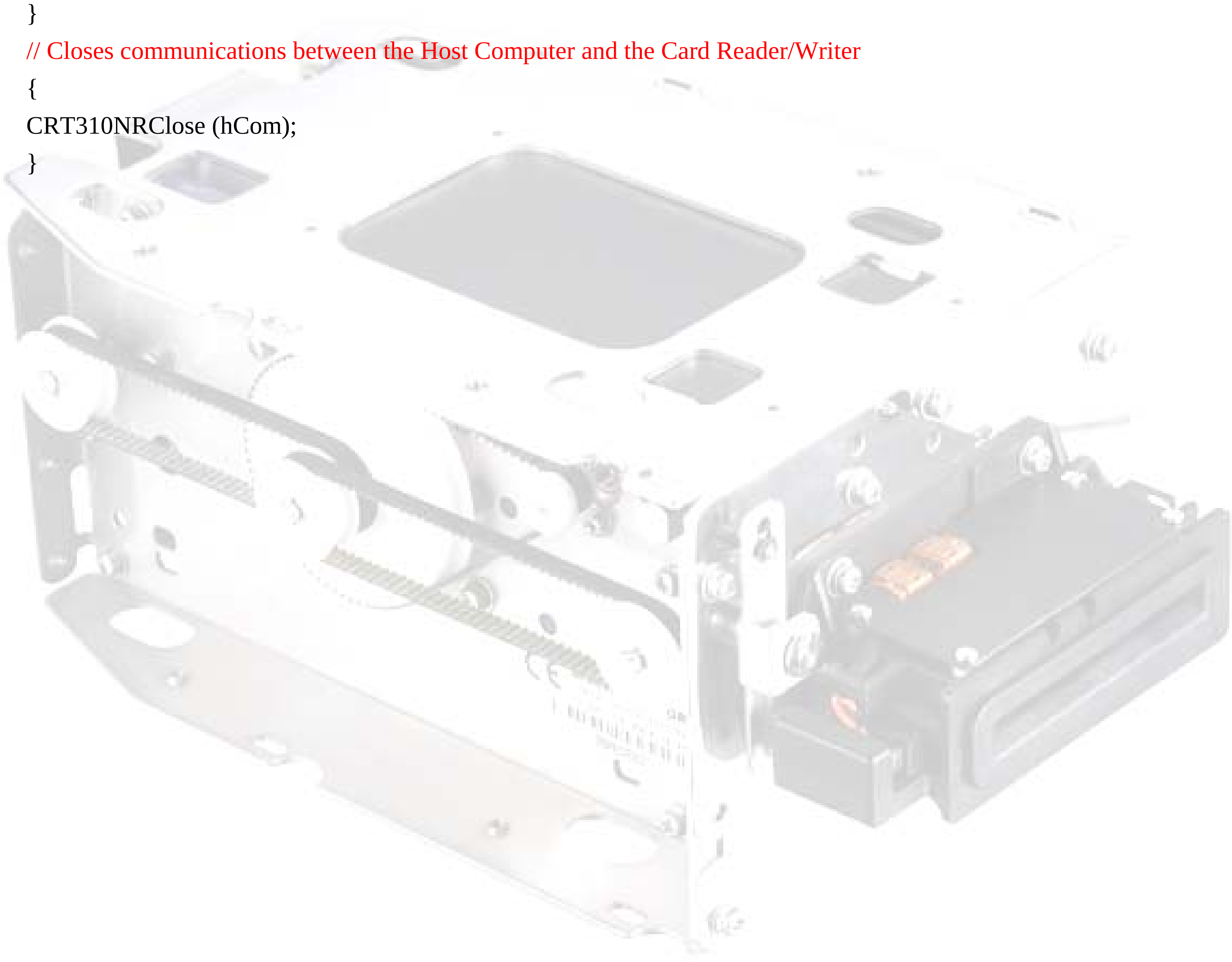
```
        HANDLE hCom;        // Target device's HANDLE
        int rc=0;                    //Result
        unsigned char CmCode;
        unsigned char PmCode;
        int    CmDataLen;
        unsigned char CmData[1024];

        unsigned char ReType;
        unsigned char St1;
        unsigned char St0;
        int    ReDataLen;
        unsigned char ReData[1024];
    // Open Comm. port
    {
        hCom=CRT310NROpenWithBaut (“COM1”,38400);
        if(hCom<=0)
        {
            // failed
            …
        }
        else
        {
            // successfully.
            …
        }
    }
    // Initialize
    {
        memset(CmData,0x00,sizeof(CmData));
        CmCode=0x30; // Initialize command
        PmCode=0x36; // Parameter code
        CmDataLen=11; // Data size (bytes)
        CmData[0]=0x33;
        CmData[1]=0x32;
        CmData[2]=0x34;
        CmData[3]=0x30;
        CmData[4]=0x30;// fm Not use. Always 30H
        CmData[5]=0x33;// Pd
        CmData[6]=0x30;// Wv
        CmData[7]=0x30;// Sh
        CmData[8]=0x31;// Ds
        CmData[9]=0x30;// Ty
        CmData[10]=0x31;// Cp

        rc=RS232_ExeCommand(hCom,CmCode,PmCode,CmDataLen,CmData,
                              &ReType,&St1,&St0,&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 by Mag card
    {
        memset(CmData,0x00,sizeof(CmData));
        CmCode= 0x3a; // Card In
        PmCode=0x32; // by Mag card
        CmDataLen=0; // Data size
        rc=RS232_ExeCommand(hCom,CmCode,PmCode,CmDataLen,CmData,
                               &ReType,&St1,&St0,&ReDataLen,ReData);
        if(rc!=0) || ReType != 0x50)
        {
            // Command sending failed or command execution failed
                ResErrMsg(St1,St0);       // ResErrMsg is a function to show the reason of error
      // Details please see from error code of communication protocol
                …
        }
    }
    // Request Status
    {
        memset(CmData,0x00,sizeof(CmData));
        CmCode= 0x31; // Status request command
    PmCode= 0x30; // Parameter code
    CmDataLen=0; // Data size
    rc=RS232_ExeCommand(hCom,CmCode,PmCode,CmDataLen,CmData,
                        &ReType,&St1,&St0,&ReDataLen,ReData);
    if(rc==0)
        if (ReType==0x50)
        {
            //Execute Ok
              if(St1 == '0' && St1 == '2')
              {
                  // status code="02"
                  // Detected a card inside of Card Reader/Writer
                  ...
              }
        }
        else (ReType==0x4e)
        {
            //Command execution failed
            ResErrMsg(St1,St0);
            …
        }
    else
        {
                //Communication Error
                …
        }
}
// Read Track 1 of Mag card’s tracks
{
    memset(CmData,0x00,sizeof(CmData));
    CmCode= 0x36; // Read Mag Card
    PmCode=0x31; // Track 1
    CmDataLen=0; // Data size
    rc=RS232_ExeCommand(hCom,CmCode,PmCode,CmDataLen,CmData,
                        &ReType,&St1,&St0,&ReDataLen,ReData);
    if(rc==0)
        if (ReType==0x50)
        {
            //Read track data OK
            if (ReData[0]==0x50)
            {
                CString Tra1Buf,t;
                int n;
                for(n=1; n<ReDataLen; n++)
```

```
                    {
                        t.Format("%c",ReData[n]);
                        Tra1Buf += t;
                        t="";
                    }
                //Tra1Buf is the buffer of track 1 data
            }
            else if   (ReData[0]==0x4e)
            {
                //Read track data Error
                switch(ReData[2])
                {
                    case 36 :     //No start bits (STX)
                        break;
                    case 37       //No stop bits (ETX)
                        break;
                    case 30 :     //Byte Parity Error(Parity))
                        break;
                    case 38 :       //Parity Bit Error(LRC)
                        break;
                    case 34 :       //Card Track Data is Blank
                    case 33 :       //Only(SS-ES-LRC)
                        break;
                }
            }
        }
        else (ReType==0x4e)
        {
            // Command execution failed
            ResErrMsg(St1,St0);
        }
    else
        {
        //Communication Error
        }
}
// Read Track 2 of Mag card's tracks
{
    memset(CmData,0x00,sizeof(CmData));
    CmCode= 0x36; // Read Mag Card
    PmCode=0x32; // Track 2
    CmDataLen=0; // Data size
    rc=RS232_ExeCommand(hCom,CmCode,PmCode,CmDataLen,CmData,
                        &ReType,&St1,&St0,&ReDataLen,ReData);
```

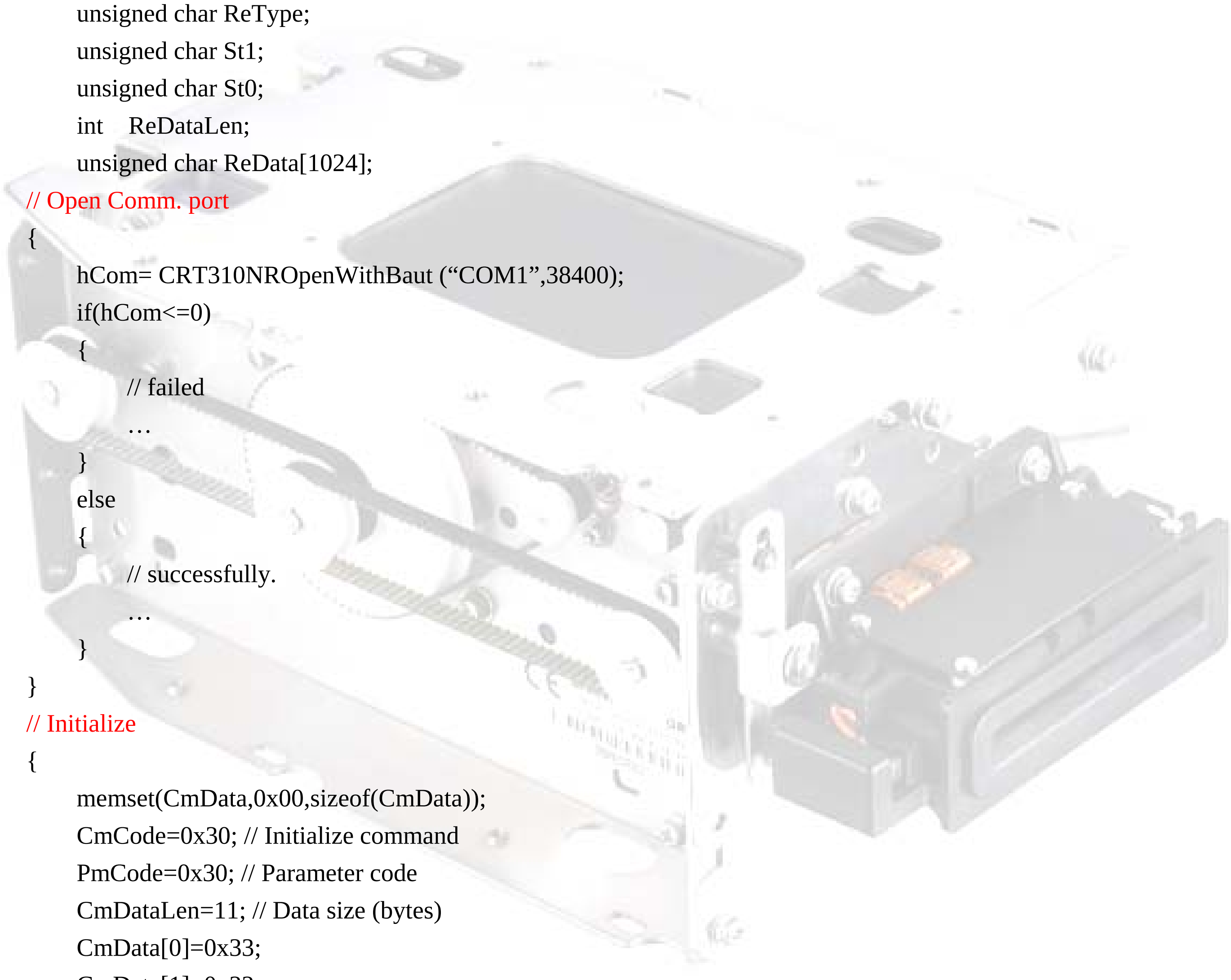
```
if(rc==0)
    if (ReType==0x50)
    {
        //Read track data OK
        if (ReData[0]==0x50)
        {
            CString Tra1Buf,t;
            int n;
            for(n=1; n<ReDataLen; n++)
                {
                    t.Format("%c",ReData[n]);
                    Tra1Buf += t;
                    t="";
                }
            //Tra1Buf is the buffer of track 2 data
        }
        else if    (ReData[0]==0x4e)
        {
            //Read track data Error
            switch(ReData[2])
            {
                case 36 :     //No start bits (STX)
                    break;
                case 37       //No stop bits (ETX)
                    break;
                case 30 :     //Byte Parity Error(Parity))
                    break;
                case 38 :      //Parity Bit Error(LRC)
                    break;
                case 34 :      //Card Track Data is Blank
                case 33 :      //Only(SS-ES-LRC)
                    break;
            }
        }
    }
    else (ReType==0x4e)
    {
        // Command execution failed
        ResErrMsg(St1,St0);
    }
else
    {
    //Communication Error
    }
```

```
}
// Read Track 3 of Mag card's tracks
{
    memset(CmData,0x00,sizeof(CmData));
    CmCode= 0x36; // Read Mag Card
    PmCode=0x33; // Track 3
    CmDataLen=0; // Data size
    rc=RS232_ExeCommand(hCom,CmCode,PmCode,CmDataLen,CmData,
                        &ReType,&St1,&St0,&ReDataLen,ReData);
    if(rc==0)
        if (ReType==0x50)
        {
            //Read track data OK
            if (ReData[0]==0x50)
            {
                CString Tra1Buf,t;
                int n;
                for(n=1; n<ReDataLen; n++)
                    {
                        t.Format("%c",ReData[n]);
                        Tra1Buf += t;
                        t="";
                    }
                //Tra1Buf is the buffer of track 3 data
            }
            else if    (ReData[0]==0x4e)
            {
                //Read track data Error
                switch(ReData[2])
                {
                    case 36 :     //No start bits (STX)
                        break;
                    case 37       //No stop bits (ETX)
                        break;
                    case 30 :     //Byte Parity Error(Parity))
                        break;
                    case 38 :      //Parity Bit Error(LRC)
                        break;
                    case 34 :      //Card Track Data is Blank
                    case 33 :      //Only(SS-ES-LRC)
                        break;
                }
            }
        }
```

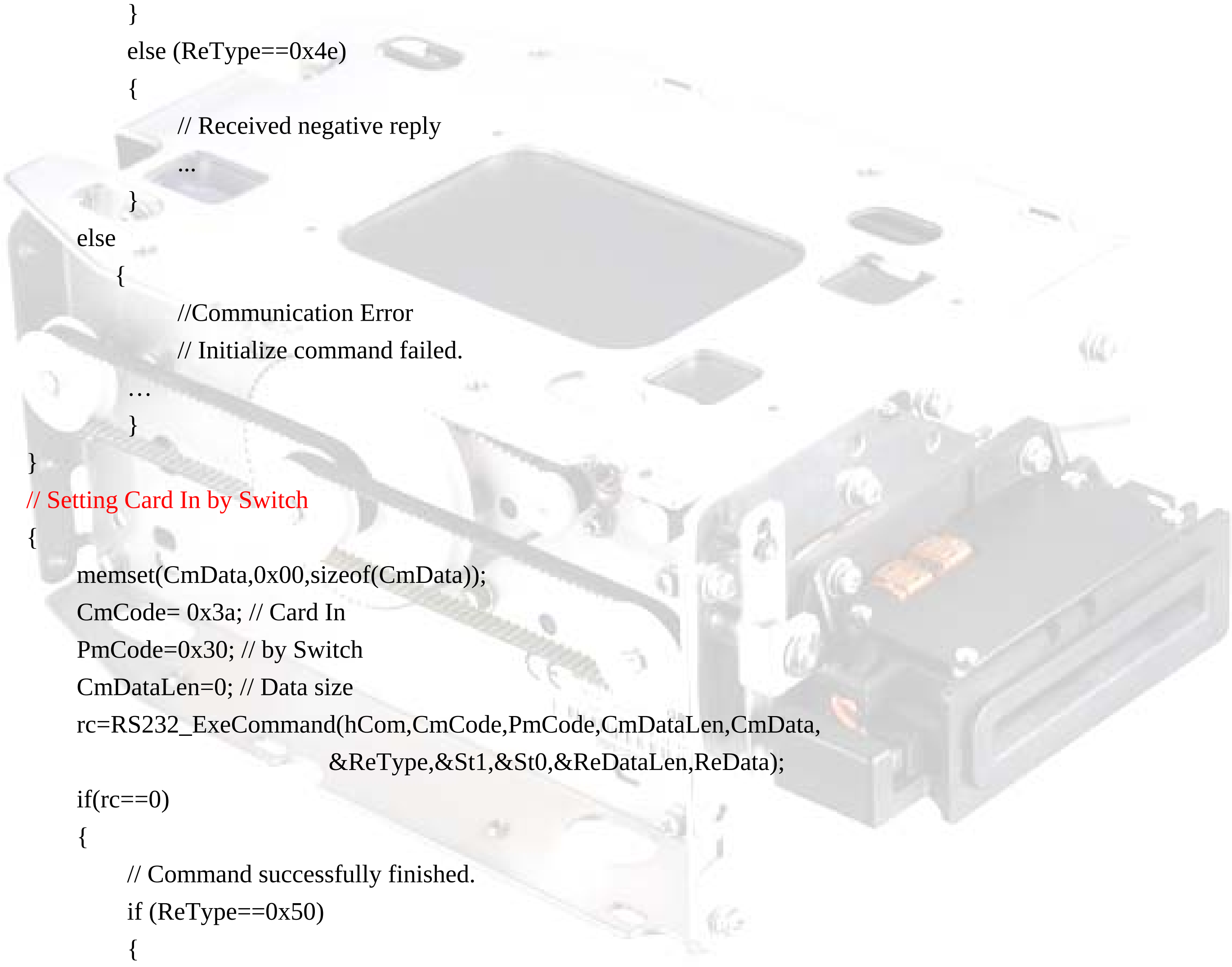
```
            else (ReType==0x4e)
            {
                // Command execution failed
                ResErrMsg(St1,St0);
            }
        else
            {
            //Communication Error
            }
}
// Closes communications between the Host Computer and the Card Reader/Writer
{
CRT310NRClose (hCom);
}
```



3.4 Code Example 2 (CPU Card Operation)



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

    unsigned char ReType;
    unsigned char St1;
    unsigned char St0;
    int    ReDataLen;
    unsigned char ReData[1024];
// Open Comm. port
{
    hCom= CRT310NROpenWithBaut (“COM1”,38400);
    if(hCom<=0)
    {
        // failed
        …
    }
    else
    {
        // successfully.
        …
    }
}
// Initialize
{
    memset(CmData,0x00,sizeof(CmData));
    CmCode=0x30; // Initialize command
    PmCode=0x30; // Parameter code
    CmDataLen=11; // Data size (bytes)
    CmData[0]=0x33;
    CmData[1]=0x32;
    CmData[2]=0x34;
    CmData[3]=0x30;
    CmData[4]=0x30;// fm Not use. Always 30H
    CmData[5]=0x33;// Pd
    CmData[6]=0x30;// Wv
    CmData[7]=0x30;// Sh
    CmData[8]=0x31;// Ds
    CmData[9]=0x30;// Ty
```



```
    CmData[10]=0x31;// Cp
    rc=RS232_ExeCommand(hCom,CmCode,PmCode,CmDataLen,CmData,
                          &ReType,&St1,&St0,&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 by Switch
{
    memset(CmData,0x00,sizeof(CmData));
    CmCode= 0x3a; // Card In
    PmCode=0x30; // by Switch
    CmDataLen=0; // Data size
    rc=RS232_ExeCommand(hCom,CmCode,PmCode,CmDataLen,CmData,
                           &ReType,&St1,&St0,&ReDataLen,ReData);
    if(rc==0)
    {
        // Command successfully finished.
        if (ReType==0x50)
        {
            // Received positive reply
            …
        }
        else (ReType==0x4e)
        {
          // Received negative reply
            ResErrMsg(St1,St0); // ResErrMsg is a function ,please see the error code of communication protocol
            …
        }
    else
      {
            //Communication Error, Command failed.
        …
        }
    }
}
// Request Status
{
    memset(CmData,0x00,sizeof(CmData));
    CmCode= 0x31; // Status request command
    PmCode= 0x30; // Parameter code
    CmDataLen=0; // Data size
    rc=RS232_ExeCommand(hCom,CmCode,PmCode,CmDataLen,CmData,
                        &ReType,&St1,&St0,&ReDataLen,ReData);
    if(rc==0)
        if (ReType==0x50)
        {
            //Execute Ok
              if(St1 == '0' && St1 == '2')
              {
                  // status code="02"
                  // Detected a card inside of Card Reader/Writer
                  ...
              }
        }
        else (ReType==0x4e)
        {
            //Command execution failed
            ResErrMsg(St1,St0);
            …
        }
    else
        {
                //Communication Error
                …
        }
}
// Switches IC contact on
{
    memset(CmData,0x00,sizeof(CmData));
    CmCode= 0x40; // IC contact
    PmCode=0x30; // IC contact set
    CmDataLen=0; // Data size
```

```
        rc=RS232_ExeCommand(hCom,CmCode,PmCode,CmDataLen,CmData,
                                        &ReType,&St1,&St0,&ReDataLen,ReData);
        if(rc!=0) || ReType != 0x50)
        {
                // Command sending failed or command execution failed
                goto _EXIT1;
        }
}
// Activates CPU card
{
        memset(CmData,0x00,sizeof(CmData));
        CmCode= 0x49; // IC card control
        PmCode=0x30; // Activate
        CmDataLen=1; // Data size
        If (true)           //Choice Mode
        {
                CmData[0]=0x30;         //EMV2000 V4.0 Vcc=5v
        }
        else
        {
                CmData[0]=0x33;        // ISO/IEC7816-3
        }
        rc=RS232_ExeCommand(hCom,CmCode,PmCode,CmDataLen,CmData,
                                        &ReType,&St1,&St0,&ReDataLen,ReData);
        if(rc!=0))
        {
                // Command sending failed or command execution failed
                goto _EXIT2;
        }
        else
        {
                if (ReType==0x50)
                {
                        CString t;
                        CString TempBuf="";
                        for(int n=1; n<ReDataLen; n++)    //    ATR= start from n= 1
                        {
                                t.Format("%02x",ReData[n]);
                                TempBuf += t;
                                t="";
                        }
                        //TempBuf    is the buffer of ATR data
                }
                else //(ReType==0x4e)
        {
            // Command execution failed
            SANKYOErrMsg(St1,St0);
        }
    }
}
// Exchanges data between the Host Computer and IC card
{
    memset(CmData,0x00,sizeof(CmData));
    CmCode= 0x49; // IC card control
    PmCode=0x39; // Deactivate
    CmDataLen=5; // Data size
    CmData[0]=0x00;
    CmData[1]=0x84;
    CmData[2]=0x00;
    CmData[3]=0x00;
    CmData[4]=0x08;
    rc=RS232_ExeCommand(hCom,CmCode,PmCode,CmDataLen,CmData,
                        &ReType,&St1,&St0,&ReDataLen,ReData);
    if(rc!=0)
    {
        // ICCardTransmit failed
        ...
    }
    else
    {
        if (ReType==0x50)
        {
            CString t;
            CString TempBuf="";
            for(int n=0; n<ReDataLen; n++)    // R-APDU= start from n=0
            {
                t.Format("%02x",ReData[n]);
                TempBuf += t;
                t="";
            }
            // TempBuf    is the buffer of R-APDU data
        }
        else //(ReType==0x4e)
        {
            // Command execution failed
            SANKYOErrMsg(St1,St0);
        }
    }
```

```
}
// Deactivates IC card
{
        memset(CmData,0x00,sizeof(CmData));
        CmCode= 0x49; // IC card control
        PmCode=0x31; // Deactivate
        CmDataLen=0; // Data size
        rc=RS232_ExeCommand(hCom,CmCode,PmCode,CmDataLen,CmData,
                                        &ReType,&St1,&St0,&ReDataLen,ReData);
        if(rc!=0) || ReType != 0x50)
        {
                // Command sending failed or command execution failed
                …
        }
}
_EXIT2:
// Switches IC contact off
{
        memset(CmData,0x00,sizeof(CmData));
        CmCode= 0x40; // IC contact
        PmCode=0x32; // IC contact release
        CmDataLen=0; // Data size
        rc=RS232_ExeCommand(hCom,CmCode,PmCode,CmDataLen,CmData,
                                        &ReType,&St1,&St0,&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
CRT310NRClose (hCom);
_EXIT:
```