

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	1/49

//////////////////////////////////DLL SPECIFICATION//////////////////////////////////

File Name: CRT_310.dll

Version Number: v20110729

Functions: DLL for CRT-310

DLL could operate:

- Mifare one
- SLE4428、SLE4442
- 24C01、24C02、24C04、24C08、24C16、24C32、24C64
- AT88SC102、AT88SC1604、AT45D041、AT88SC1608
- Contact CPU Card (T=0/T=1)
- Contactless CPU (TYPE=A/TYPE=B)
- SAM/SIM Card (T=0/T=1)
- Magnetic Stripe card

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	2/49

//////////////////////////////////Functions//////////////////////////////////

Collect version information

1. int APIENTRY GetSysVerion(HANDLE ComHandle, char *strVerion);

//Parameter:

// ComHandle

// strVerion

//Return

// Successful: 0

// Failure: others

//////////////////////////////////

Retrieve error code

2. int APIENTRY GetErrCode(int *Errorcode);

//Parameter:

// Errorcode: The latest error code

Error Code:

Error code	Protocol Description	Description
-1		Communication Error==>Check cables and serial port setting
-2		Card Type error
-101		Serial port error==>Execute CommOpen or CommOpenWithBaut function to configure
-102		Baud Rate setting error==> Execute CommOpenWithBaut to open serial port
-200	0	Undefined parameter
-202	1	Command out of model==>Check if command support by the model number
-204	2	Command out of order==> Wrong command
-205	3	Power error==>Check power outputDC12V
-206	4	Over-short or long card inside
-207	5	Power off==>Check the power

//Return:

// Success: 0

// Failure: others

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	3/49

////////////////////////////////Serial Port Open //////////////////////////////////

Open Serial Port (Default 9600bps)

1. HANDLE APIENTRY CommOpen(char *Port);

//Parameter:

//Port

//E.g.: CommOpen("Com1");

//Return:

//Comhandle

//Notes: Please use this command before others

// Do not open the same serial port twice, and allow to open two different serial port

// CommClose() to close after all

//

Open Serial Port with defined baud rate

2. HANDLE APIENTRY CommOpenWithBaut(char *Port, BYTE _BaudOption);;

//Parameter:

//Port:

//_BaudOption=

// 1=> 1200bps

// 2=> 2400bps

// 3=> 4800bps

// 4=> 9600bps

// 5=> 19200bps

// 6=> 38400bps

// 7=> 57600bps

//E.g.:CommOpen("Com1",4);

//Return:

//Comhandle

//Notes: Please use this command before others

// Do not open the same serial port twice, and allow to open two different serial port

// CommClose() to close after all

//

Close serial port

3. int APIENTRY CommClose(HANDLE ComHandle);

//Parameter:

// ComHandle

//Return:

//Successful: 0

// Use it after all

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	5/49

//////////////////////////////////// CRT-310 Operation //////////////////////////////////////

CRT310 Reset

1. int APIENTRY CRT310_Reset(HANDLE ComHandle, BYTE _Eject);;

//Parameter:

// ComHandle

// _Eject: 0=No ejection 1=Front ejection 2=Rear ejection

//Return:

// Successful: 0

// Failure: others

////////////////////////////////////

//Read CRT-310 serial number

2. int APIENTRY CRT310_ReadSnr(HANDLE ComHandle, BYTE _SNData[], BYTE *_dataLen)

//Parameter:

// ComHandle

// _SNData: Serial Number

// _dataLen: Data Length

//Return:

// Successful: 0

// Failure: others

Notes: SN length (0 ~ 16 byte)

Format: A S C I I

E.g.:

“ 0x43,0x52,0x54,0x2D,0x33,0x31,0x30”

////////////////////////////////////

//Write CRT-310 Serial number

3. int APIENTRY CRT310_WriteSnr(HANDLE ComHandle, BYTE _SNData[], BYTE _dataLen)

//Parameter:

// ComHandle

// _SNData: Serial Number

// _dataLen: Data length

//Return:

// Successful: 0

// Failure: others

Notes: SN length (0 ~ 16 byte)

Format: A S C I I

E.g.:

“ 0x43,0x52,0x54,0x2D,0x33,0x31,0x30”

=0X49	Card exist in front with holding position
-------	---

SPECIFICATION

Model No.	CRT-310
Date	2011/07/29
Ver.	v20110729
Page	7/49

	=0X4A	Card in the reader
	=0X4B	Card in IC card operation position and contact with IC contact
	=0X4C	Card in rear side with holding position
	=0X4D	Card in rear side without holding position
	=0X4E	No card in the reader
// _frontSetting:	=0X49	Card in only with magnetic signal
	=0X4A	Card in only with switch signal
	=0X4B	Card in with magnetic signal
	=0X4E	Prohibit card in
// _rearSetting:	=0X4A	Card in from rear side
	=0X4E	Prohibit card in from rear side
	//Return:	
	// Success: 0	
	// Failure: others	

////////////////////////////////////

CRT310 Sensor Status

```
7.  int APIENTRY CRT310_SensorStatus (HANDLE ComHandle, BYTE *_PSS0, BYTE *_PSS1, BYTE *_PSS2, BYTE *_PSS3,
    BYTE *_PSS4, BYTE *_PSS5, BYTE *_CTSW, BYTE *_KSW)
```

```
//Parameter:
```

```
// ComHandle
```

_ PSS0—PSS5:	Infrared sensor:	PSSx=0X30	No Card
		PSSx=0X31	Have card
_CTSW:	Shutter sensor	CTSW=0X30	Shutter Closed
		CTSW=0X31	Shutter Opened
_KSW :	Card entry switch sensor	KSW=0X30	No card in signal
		KSW=0X31	Card in signal

```
//Return:
```

```
// Successful: 0
// Failure: others
```

////////////////////////////////////

CRT310 Card movement

8. int APIENTRY CRT310_MovePosition(HANDLE ComHandle, BYTE _Position)

```
//Parameter:
```

```
// ComHandle
// _Position
```

=0x1	Move card to front card without holding position
=0x2	Move card to front card with holding position
=0x3	Move card to RF operation position

////////////////////////////////////

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	9/49

//////////////////////////////////// Magnetic Card Functions////////////////////////////////////

Read Magnetic data

1. int APIENTRY MC_ReadTrack(HANDLE ComHandle, BYTE _mode, BYTE _track, int *_BlockDataLen, BYTE _BlockData[])

//Parameter:

// ComHandle。

// _mode: Data Mode

Read Mode: 0x30 ASCII
0x31 Binary

// _track: Track combination

Track: 0x30 No track reading
0x31 Read track 1
0x32 Read track 2
0x33 Read track 3
0x34 Read tracks 1, 2
0x35 Read tracks 2, 3
0x36 Read tracks 1, 3
0x37 Read track 1,2,3

// _BlockDataLen: Data length

// _BlockData: Data

Track Format:

Beginning sentinel+Status code+Magnetic data information

Beginning sentinel: 0x1F

Status Code: 0x59 Read data correctly
0x4E Read data incorectly
0x4F Inavailabe to ready, magnetic data is 0xE0

Error Code:

0xE1 No STX
0xE2 No ETX
0xE3 VRC Error
0xE4 LRC Error
0xE5 Blank track

*Binary Mode:

Track 1: b0,b1,b2,b3,b4,P

Track2, 3: b0,b1,b2 b3,P

*Notes:

ASCII mode will transfer date to ASCII code

E.g.: HEX Data: 0x03 (HEX)

Transfer to ASCII mode: 0x33 (ASCII)

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	10/49

Binary Mode will transfer Hex data to binary data

E.g. HEX data: 0x03 (HEX)

Transferred binary code: 0x0 0x3

//Return:

// Successful 0

// Failure Others

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	11/49

//////////////////////////////// Mafare 1////////////////////////////////

Detect M1

1. int APIENTRY RF_DetectCard(HANDLE ComHandle);

//Parameter:

// ComHandle

//Return P:

// Successful 0

// Failure Others

P= 'N' (0X4E) Detection Failure

P= 'E' (0X45) No card in the reader

P= 'W' (0X57) Card is not on the operation position

////////////////////////////////

Acquire M1 serial number

2. int APIENTRY RF_GetCardID(HANDLE ComHandle, BYTE *_CardIDLen, BYTE _CardID[]);

//Parameter:

// ComHandle

// _CardIDLen: SN length

// _CardID[]: M1 ID number

//Return:

// Successful 0

// Failure Others

P= 'N' (0X4E) Acquire SN failure, return (0X00, 0X00, 0X00, 0X00)

P= 'E' (0X45) No card in the reader

////////////////////////////////

Verify Password

3. int APIENTRY RF_LoadSecKey(HANDLE ComHandle, BYTE _Sec, BYTE _KEYType, BYTE _KEYLen, BYTE _KEY[]);

//Parameter:

// ComHandle

// _Sec: Sector number

// _KEYType: Password Selection 0=KEYA; 1=KEYB

// _KEYLen: 6 (Password length)

// _KEY[]: Password (6 byte)

//Return P:

// Successful: 0

// Failure: Others

P= '0' (0X30) Fail to find RF card

P= '3' (0X33) Password error

P= 'E' (0X45) No card in the reader


```
//  _BlockDatalen: 16*_BlockNumber (Block Data Length)
```


	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	16/49

P= 'W' (0X57) Card is not placed correct

//Notes: Before increment value and decrement value operaton, value initialization is highly recommended to avert error.

Initialization Value: Data format as the following:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Value				/Value				Value				Adr	/Adr	Adr	/Adr

Value: 4 byte Value (Initialization value) , Low byte before high byte

/Value: Neglect 4 byte value

Adr: Address for initialization: Adr= Sector X 4 + Block

/Adr: Neglect address for initialization

E.g: Set block0 of sector 5 as value 10 (Please refer to .net sample source code)

“ 0x0A, 0x00, 0x00, 0x00, 0xF5, 0xFF,0xFF ,0xFF , 0x0A, 0x00, 0x00, 0x00, 0x14, 0xEB, 0x14, 0xEB ”

int P;

BYTE value[4];

memset(value,0,4);

value[0]=0xA;

P=RF_InitValue(hCom,5,0,value);

Notes: KEY A and KEY B are restored in Block 3 of 0-31 sector (S50) and Block 15 of 32-39 sectors (S70). Keep these blocks unchange

////////////////////////////////////

Read Value

10. int APIENTRY RF_ReadValue(HANDLE ComHandle, BYTE _Sec, BYTE _Block, BYTE * _DataLen, BYTE _Data[]);

//Parameter:

// ComHandle

// _Sec: Sector number

// _Block: Block number

// **_DataLen: 4 bytes Data length**

// _Data: Data

//Return P:

// Successful 0

// Failure Others

P= '0' (0X30) Fail to find RF

P= '1' (0X31) Operation Sector number (Without key verification)

P= '2' (0X32) S/N number error

P= '3' (0X33) Password verification error

P= '4' (0X34) Block data format error (The format should be as value)

P= '5' (0X35) Incremental value flow-out

P= 'E' (0X45) No card in the reader

P= 'W' (0X57) On the non-operatable position

Notes: Before reading value, recommended to executeRF_InitValue function

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	17/49

Notes 2: Sector number= 0x00 0x01 0x020x28

S50 sector number: 0x00 0x01 0x020x0F, S70 sector number: 0x00 0x01 0x020x28

Block number = 0x00 0x01 0x020x0F

Each S50 has 4 blocks: 0x00 0x01 0x02 0x03,

Each S70 has 4 blocks for sector 0-3: 0x00 0x01 0x02 0x03,

16 blocks for 32-39 sector: 0x00 0x01 0x02.....0x0F

Notes3: In value operation, the following sector is for reserve password, so operation is avert

S50, S70 Sector 0-31, block 3; S70 Sector 32-39 Block 15

////////////////////////////////////

Value Increment

11. int APIENTRY RF_Increment(HANDLE ComHandle, BYTE _Sec, BYTE _Block, BYTE _DataLen, BYTE _Data[]);

//Parameter:

// ComHandle

// _Sec: Sector

// _Block

// _DataLen: 4 (Data length)

// _Data: Value

//Return P:

// Successful 0

// Failure Others

P= '0' (0X30) Fail to find RF

P= '1' (0X31) Password sector error (Without password verification)

P= '2' (0X32) S/N number error

P= '3' (0X33) Password

P= '4' (0X34) Block data format error (Not to format as value)

P= '5' (0X35) Incremental value flow out

P= 'E' (0X45) No card in the reader

P= 'W' (0X57) Not on the operatable position

Notes: Before reading value, recommended to executeRF_InitValue function

Notes 2: Sector number= 0x00 0x01 0x020x28

S50 sector number: 0x00 0x01 0x020x0F, S70 sector number: 0x00 0x01 0x020x28

Block number = 0x00 0x01 0x020x0F

Each S50 has 4 blocks: 0x00 0x01 0x02 0x03,

Each S70 has 4 blocks for sector 0-3: 0x00 0x01 0x02 0x03,

16 blocks for 32-39 sectors: 0x00 0x01 0x02.....0x0F

Notes3: In value operation, the following sector is for reserve password, so operation is avert

S50, S70 Sector 0-31, block 3; S70 Sector 32-39 Block 15

////////////////////////////////////

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	18/49

Value Decrement

12. int APIENTRY RF_Decrement(HANDLE ComHandle, BYTE _Sec, BYTE _Block, BYTE _DataLen, BYTE _Data[]);

//Parameter:

// ComHandle
 // _Sec: Sector number
 // _Block: Block number
 // _DataLen: 4 (Datalength)
 // _Data: Value

//Return P:

// Successful 0
 // Failure Others
 P= '0' (0X30) Fail to find RF
 P= '1' (0X31) Password sector error (Without password verification)
 P= '2' (0X32) S/N number error
 P= '3' (0X33) Password
 P= '4' (0X34) Block data format error (Not to format as value)
 P= '5' (0X35) Incremental value flow out
 P= 'E' (0X45) No card in the reader
 P= 'W' (0X57) Not on the operatable position

Notes: Before reading value, recommended to executeRF_InitValue function

Notes 2: Sector number= 0x00 0x01 0x020x28

S50 sector number: 0x00 0x01 0x020x0F, S70 sector number: 0x00 0x01 0x020x28

Block number = 0x00 0x01 0x020x0F

Each S50 has 4 blocks: 0x00 0x01 0x02 0x03,

Each S70 has 4 blocks for sector 0-3: 0x00 0x01 0x02 0x03,

16 blocks for 32-39 sectors: 0x00 0x01 0x02.....0x0F

Notes3: In value operation, the following sector is for reserve password, so operation is avert

S50, S70 Sector 0-31, block 3; S70 Sector 32-39 Block 15

////////////////////////////////////

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	19/49

//////////////////RF CPU Card Function (TYPE=A / TYPE=B) //////////////////

/////

Notes: 1> Check card type before (type A or type B)

2> After successfully check card type, execute APDU

Obtain CPU Card S/N number (Only TYPE A)

1. int APIENTRY CPU_GetRfCardID(HANDLE ComHandle, BYTE *_CardIDLen, BYTE _CardID[]);

//Parameter:

// ComHandle

// _CardIDLen: S/N length

// _CardID[]:M1S/N (4 byte)

//Return:

// Successful 0

// Failure Others

P= 'N' (0X4E) Obtain SN Error, Return (0X00, 0X00, 0X00, 0X00)

P= 'E' (0X45) No card in the reader

////////////////////////////////////

Search RF CPU Card (Reset)

2. int APIENTRY CPU_SelRfCard(HANDLE ComHandle,BYTE _CPUType,BYTE _exData[], int *_exdataLen);

//Parameter:

// ComHandle

// _CPUType: CPU Type

=0x0 TYPE A

=0x1 TYPE B

// _exData: Return Data

// _exdataLen: Return Data length

//Return:

// Successful 0

// Failure Others

= 'N' (0X4E) Opearion unsucessful

= 'E' (0X45) No Card in reader

////////////////////////////////////

C-APDU Command

3. int APIENTRY CPU_SendRfAPDU(HANDLE ComHandle, int _dataLen, BYTE _APDUData[], BYTE _exData[], int *_exdataLen);

//Parameter:

// ComHandle

// _dataLen: C-APDU Data length

```
// Failure Others
```

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	21/49

//////////////////////////////////// IC Card Operation Function //////////////////////////////////////

Power On

1. int APIENTRY CRT_IC_CardOpen(HANDLE ComHandle);

//Parameter:

// ComHandle

//Return:

// Successful 0

// Failure Others

////////////////////////////////////

Power Down

2. int APIENTRY CRT_IC_CloseCard(HANDLE ComHandle);

//Parameter:

// ComHandle

//Return:

// Successful 0

// Failure Others

////////////////////////////////////

Auto-test Card Type

3. int APIENTRY CRT_R_DetectCard(HANDLE ComHandle,BYTE *_CardType,BYTE *_CardInfor)

//Parameter:

// ComHandle

// CardType: Card Type

// CardInfor: Card Information

CardType	CardInfor	Contents
'N'	'0'	No card in the reader
	'1'	Unknown card type
	'2'	Card is no on right position
'0'	'0'	S50
	'1'	S70
	'2'	UL
	'4'	TYPE A CPU
	'5'	TYPE B CPU
'1'	'0'	T=0 CPU
	'1'	T=1 CPU
'2'	'0'	24C01
	'1'	24C02
	'2'	24C04

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	22/49

	‘3’	24C08
	‘4’	24C16
	‘5’	24C32
	‘6’	24C64
‘3’	‘0’	SL4442
	‘1’	SL4428
‘4’	‘0’	AT88S102
	‘1’	AT88S1604
	‘2’	AT45D041
	‘3’	AT88S1608

//Return:

// Successful 0

// Failure Others

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	23/49

//////////////////////////////////// CPU Card Function (T=0/ T=1) //////////////////////////////////////

CPU Reset

1. int APIENTRY CPU_ColdReset(HANDLE ComHandle, BYTE _CPUMode, BYTE *_CPUType, BYTE _exData[], int *_exdataLen);

//Parameter:

// ComHandle

// _CPUMode: Reset Mode

=0x0 EMV Mode

=0x1 ISO7816 Mode

// _CPUType: Return CPU Card type

=0x0 T=0 CPU

=0x1 T=1 CPU

// _exData: Return Data

// _exdataLen: Return Data Length

//Return:

// Successful 0

// Failure Others

= 'N' (0X4E) Unsuccessful

= 'E' (0X45) No card in the reader

= 'W' (0X57) Card is no on the right position

E.g.: Please refer to .net sample source code

int rc;

BYTE exData[300]; //Reserve data length is 300, Actual valid length related "&lenASK"

int lenASK;

BYTE _CPUType;

rc=CPU_Reset(hCom,0, &_CPUType,exData,&lenASK);

//Reset Successful

//rc=0

//&lenASK Return data length

//exData Return data

CString str,t;

for(int i=0; i<lenASK; i++)

{

t.Format("%02x ",exData[i]);

str += t;

t="";

}

t.Format("CPU Card Type T%d \n:",_CPUType);

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	24/49

```

m_list1.AddString (t);

t.Format("Return data length as cold reset %d \n, such as :",lenASK);

m_list1.AddString (t);

m_list1.AddString (str); //Return str data

m_list1.AddString ("\n");

```

//

CPU Hot reset

2. int APIENTRY CPU_WarmReset(HANDLE ComHandle, BYTE *_CPUType, BYTE _exData[], int *_exdataLen);

//Parameter:

```

// ComHandle

// _CPUType: Return CPU Type

           =0x0    T=0

           =0x1    T=1

```

// _exData: Return data

// _exdataLen: Return data length

//Return:

```

// Successful 0

// Failure Others

    = 'N' (0X4E)    Failure

    = 'E' (0X45)    No card in the reader

    = 'W' (0X57)    Card on un-operative position

```

E.g.: Refer to CPU Reset function in sample source code

//

CPU (T=0) C-APDU Command

3. int APIENTRY CPU_T0_C_APDU(HANDLE ComHandle, int _dataLen, BYTE _APDUData[], BYTE _exData[], int *_exdataLen);

//Parameter:

```

// ComHandle

// _dataLen: C-APDU data length

// _APDUData: CPU C-APDU Data

// _exData: Return Data

// _exdataLen: Return Data length

```

//Return:

```

// Successful 0

// Failure Others

    = 'N' (0X4E)    Failure

    = 'E' (0X45)    No card in the reader

    = 'W' (0X57)    Card on un-operative position

```


	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	25/49

E.g.:

BYTE DataBlock[5]; //Reserve C-APDU command, data length is 5 byte

BYTE exData[300]; //Return data according to C-APDU

int lenASK; //Return data length according to C-APDU

DataBlock[0]=0x00;

DataBlock[1]=0x84;

DataBlock[2]=0x00;

DataBlock[3]=0x00;

DataBlock[4]=0x08;

rc=CPU_C_APDU(hCom,5,DataBlock,exData,&lenASK);

if(rc==1) AfxMessageBox("No Find Card!");

else if(rc!=0) AfxMessageBox("ERR!");

else

{

CString str;

for(int i=0; i<lenASK; i++)

{

t.Format("%02x",exData[i]);

str += t;

t="";

}

}

//Return 16 byte string

//

CPU(T=1)卡 C-APDU 命令。

4. int APIENTRY CPU_T1_C_APDU(HANDLE ComHandle, int _dataLen, BYTE _APDUData[], BYTE _exData[], int *_exdataLen);

//Parameter:

// ComHandle

// _dataLen: C-APDU Command Data length

// _APDUData: C-APDU Command

// _exData: Return Data

// _exdataLen: Return Data length

//Return:

// Successful 0

// Failure Others

= 'N' (0X4E) Failure

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	26/49

= 'E' (0X45) No card in the reader

= 'W' (0X57) Card on un-operative position

E.g.: Please refer to sample source code

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	27/49

//////////////////////////////////// SIM Card Functions (T=0/ T=1) //////////////////////////////////////

SIM Reset

1. int APIENTRY SIM_Reset(HANDLE ComHandle, BYTE _VOLTAGE, BYTE _SIMNo, BYTE *_SIMTYPE, BYTE _exData[], int *_exdataLen);

//Parameter:

// ComHandle

// _VOLTAGE

= 0x2E Reset for 1.8 V SIM

= 0x2F Reset for 3.0 V SIM

= 0x30 Reset for 5.0 V SIM

// SIMNo: SIM Number

SIM Number =0x30 SIM1

=0x31 SIM2

=0x32 SIM3

=0x33 SIM4

=0x34 SIM5

=0x35 SIM6

=0x36 SIM7

=0x37 SIM8

// _SIMTYPE: Return SIM card type

=0x0 T=0

=0x0 T=1

// _exData: Return Data

// _exdataLen: Return Data length

//Return:

// Successful 0

// Failure Others

////////////////////////////////////

2. SIM (T=0)C-APDU Command

int APIENTRY SIM_T0_C_APDU(HANDLE ComHandle, BYTE SIMNo, int _dataLen, BYTE _APDUData[], BYTE _exData[], int *_exdataLen);

//Parameter:

// ComHandle

// SIMNo: SIM Number

SIM Number =0x30 Operate SIM1

=0x31 Operate SIM2

=0x32 Operate SIM3

=0x33 Operate SIM4

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	28/49

=0x34 Operate SIM5

=0x35 Operate SIM6

=0x36 Operate SIM7

=0x37 Operate SIM8

// _dataLen: C-APDU Command Length

// _APDUData: CPU C-APDU Command

// _exData: Return data

// _exdataLen: Return data length

//Return:

// Successful 0

// Failure: Others

//

SIM (T=1) C-APDU D Command

3. int APIENTRY SIM_T1_C_APDU(HANDLE ComHandle, BYTE SIMNo, int _dataLen, BYTE _APDUData[], BYTE _exData[], int *_exdataLen);

//Parameter:

// ComHandle

// SIMNo: SIM Number

SIM Number =0x30 Operate SIM1

=0x31 Operate SIM2

=0x32 Operate SIM3

=0x33 Operate SIM4

=0x34 Operate SIM5

=0x35 Operate SIM6

=0x36 Operate SIM7

=0x37 Operate SIM8

// _dataLen: C-APDU Command Length

// _APDUData: CPU C-APDU Command

// _exData: Return data

// _exdataLen: Return data length

//Return:

// Successful 0

// Failure: Others

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	29/49

//////////////////////////////// SLE4428 Functions //////////////////////////////////

SEL4428 Reset

1. int APIENTRY SLE4428_Reset(HANDLE ComHandle)

//Parameter:

// ComHandle

//Return:

// Successful 0

// Failure Others

= 'E' (0X45) No Card

= 'W' (0X57) Card is not on the operatable position

////////////////////////////////

Read Block Data from address in "_Address" and data length reserve in address of "_ BlockData" at length of "_ dataLen"

2. int APIENTRY SLE4428_Read(HANDLE ComHandle, int _Address, BYTE _dataLen, BYTE _BlockData[])

//Parameter:

// ComHandle

// _Address

// Starting Address

// _dataLen

// Data Length

// _BlockData

// Pointer reserved inside and read data save in the address _Block data point

// Pointer which reserved in _ BlockData and cover the length of _dataLen

//Return:

// Successful 0

// Failure Others

= 'E' (0X45) No card in the reader

= 'W' (0X57) Card is on non-operatable

////////////////////////////////

Write Block Data from address in "_Address" and data length reserve in address of "_ BlockData" at length of "_ dataLen"

3. int APIENTRY SLE4428_Write(HANDLE ComHandle, int _Address, BYTE _dataLen, BYTE _BlockData[])

//Parameter:

// ComHandle

// _Address

// Starting Address

// _dataLen

// Data Length

// _BlockData

// Pointer reserved inside and read data save in the address _Block data point

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	30/49

```
// Successful 0
// Failure Others
// =‘E’ (0X45) No card in the reader
// =‘W’ (0X57) Card is on non-operatable
```

//

Read SLE4442 protection zone (0X01 means non-protect which is available to modify and 0X00 means protected which can't modify)

4. int APIENTRY SLE4428_ReadP(HANDLE ComHandle, int _Address, BYTE _dataLen, BYTE _BlockData[])

//Parameter:

```
// ComHandle
// _Address
// Starting Address
// _dataLen
// Data length
// _BlockData
// Pointer reserved inside and read data save in the address _Block data point
```

//Return:

```
Successful 0
// Failure Others
// =‘E’ (0X45) No card in the reader
// =‘W’ (0X57) Card is on non-operatable
```

//

Write the data' _ BlockData' appointed to address in '_ Address'

5. int APIENTRY SLE4428_WriteP(HANDLE ComHandle, int _Address, BYTE _dataLen, BYTE _BlockData[])

//Parameter:

```
// ComHandle
// _Address
// Defined starting address
// _dataLen
// Appointed data length
// _BlockData
// Pointer which reserved in _ BlockData and cover the length of _dataLen
```

//Return:

```
// Successful 0
// Failure Others
// =‘E’ (0X45) No card in the reader
// =‘W’ (0X57) Card is on non-operatable
```

//

Verify Password

////////////////////////////////////

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	32/49

//////////////////////////////// SLE4442 Function////////////////////////////////

SEL4442 Reset

1. int APIENTRY SLE4442_Reset(HANDLE ComHandle);

//Parameter:

// ComHandle

//Return:

// Successful 0

// Failure Others

////////////////////////////////

SEL4442 Read Data

2. int APIENTRY SLE4442_Read(HANDLE ComHandle, BYTE _Address, BYTE _dataLen, BYTE _BlockData[]);

//Parameter:

// ComHandle:

// _Address: Address

// _dataLen: Data Length

// _BlockData: Data

//Return:

// Successful 0

// Failure Others

////////////////////////////////

SEL4442 Read Protection Data

3. int APIENTRY SLE4442_ReadP(HANDLE ComHandle, BYTE _BlockDataLen, BYTE _BlockData[]);

//Parameter:

// ComHandle

// _BlockDataLen: Block Data Length, Fix as 32

// _BlockData: Block Data

//Return:

// Successful 0

// Failure Others

////////////////////////////////

SEL4442 Read Security Data

4. int APIENTRY SLE4442_ReadS(HANDLE ComHandle, BYTE _BlockDataLen, BYTE _BlockData[]);

//Parameter:

// ComHandle

// _BlockDataLen: Block data length, fix as 4

// _BlockData: Data

//Return:

// Successful 0

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	33/49

// Failure Others

//

SEL4442 Verification

5. int APIENTRY SLE4442_VerifyPWD(HANDLE ComHandle, BYTE _PWDataLen, BYTE _PWData[]);

//Parameter:

// ComHandle

// **_PWDataLen:** Password Data Length

// **_PWData:** Password, 3 byte

//Return:

// Successful 0

// Failure Others

//

SEL4442 Data Writing

6. int APIENTRY SLE4442_Write(HANDLE ComHandle, BYTE _Address, BYTE _dataLen, BYTE _BlockData[]);

//Parameter:

// ComHandle

// **_Address:** Address

// **_dataLen:** Data Length

// **_BlockData:** Data

//Return:

// Successful 0

// Failure Others

//

SEL4442 Write Protection Data

7. int APIENTRY SLE4442_WriteP(HANDLE ComHandle, BYTE _Address, BYTE _dataLen, BYTE _BlockData[]);

//Parameter:

// ComHandle

// **_dataLen:** Data Length

// **_BlockData:** Data

//Return:

// Successful 0

// Failure Others

//

SEL4442 Password Modification

8. int APIENTRY SLE4442_WritePWD(HANDLE ComHandle, BYTE _PWDataLen, BYTE _PWData[]);

//Parameter:

// ComHandle:

// **_PWDataLen:** Length of password, fixed as 3

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	34/49

// _PWData: Password data

//Return:

// Successful 0

// Failure Others

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	35/49

////////////////////////////////////24Cxx Serial Functions////////////////////////////////////

Check 24Cxx

1. int APIENTRY IC_DetectCard(HANDLE ComHandle, BYTE *_CardType);

//Parameter:

// ComHandle

// _CardType: Card Type

=0	24C01	128BYTE	ADR=0x0000—0x007F
=1	24C02	256BYTE	ADR=0x0000—0x00FF
=2	24C04	512BYTE	ADR=0x0000—0x01FF
=3	24C08	1K BYTE	ADR=0x0000—0x03FF
=4	24C16	2K BYTE	ADR=0x0000—0x07FF
=5	24C32	4K BYTE	ADR=0x0000—0x0FFF
=6	24C64	8K BYTE	ADR=0x0000—0x1FFF

//Return:

// Successful 0

// Failure 0

P= 'N' (0X4E) Failure

P = 'E' (0X45) No Card in the machine

P = 'W' (0X57) Card is on non-operatable position

////////////////////////////////////

Read Defined Address Data

2. int APIENTRY IC_ReadBlock(HANDLE ComHandle, BYTE _CardType, int _Address, BYTE _dataLen, BYTE _BlockData[]);

//Parameter:

// ComHandle

// _Address: Starting Address

// _dataLen: Data length

// _BlockData: Data

//Return:

// Successful 0

// Failure Others

////////////////////////////////////

Write Defined Address Data

3. int APIENTRY IC_WriteBlock(HANDLE ComHandle, BYTE _CardType, int _Address, BYTE _dataLen, BYTE _BlockData[]);

//Parameter:

// ComHandle

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	36/49

```
//  _Address: Starting Address
//  _dataLen: Data Length
//  _ BlockData: Data

//Return:
//  Successful 0
//  Failure Others
```

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	37/49

//////////////////////////////////// AT88SC102 Functions////////////////////////////////////

Reset

1. int APIENTRY AT88SC102_Reset(HANDLE ComHandle);

//Parameter:

// ComHandle

//Return P

// Successful 0

// Failure Others

P= 'N' (0X4E) Failure

P = 'E' (0X45) No Card in the reader

P = 'W' (0X57) Card isn't on operatable position

////////////////////////////////////

Verify Password

2. int APIENTRY AT88SC102_VerifyPWD(HANDLE ComHandle, BYTE _PWDataLen, BYTE _PWData[]);

//Parameter:

// ComHandle

// _PWDataLen: Password data length, fixed as 2

// _PWData: Password, 2 byte

//Return P:

// Successful 0

// Failure Others

P= 'N' (0X4E) Failure

P= 'E' (0X45) No card in the reader

P= 'F' (0X46) Card is crashed (Password dead lock)

P= 'W' (0X57) Card is on the non-operatable position

Under security level 1, all unit could be readable

Under security level 2, besides password information, else could be readable

////////////////////////////////////

Password Modification

3. int APIENTRY AT88SC102_WritePWD(HANDLE ComHandle, BYTE _PWIndex, BYTE _PWData[]);

//Password:

// ComHandle

// _PWIndex: Password Type 0=Main Password; 1=Erase password 1; 2=Erase password 2

// _PWData: Password

Block =0 Modify control password 2 byte

=1 Modify application zone 1 password 6 byte

=2 Modify application zone 2 password 4 byte

[illegible]

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	39/49

安全模式 2 下擦除应用区一。

6. **int APIENTRY AT88SC102_Security2ClearApp1(HANDLE ComHandle, BYTE _Security2App1PWLen, BYTE _Security2App1PW[]);**

//Parameter:

// ComHandle

// _Security2App1PWLen: Password length, fix as 6

// _Security2App1PW: Erase Password

//Return P:

// Successful 0

// Failure Others

= 'N' (0X4E) Failure

= 'E' (0X45) No card in the reader

= 'W' (0X57) Card on non-operatable position

//

Erase application 2 under security mode

7. **int APIENTRY AT88SC102_Security2ClearApp2(HANDLE ComHandle, BYTE _EC2, BYTE _Security2App2PWLen, BYTE _Security2App2PW[]);**

//Parameter:

// ComHandle

// EC2: Fuse status operation character

EC2 = 0 Erase application zone 2 which fuse is burndown

EC2 = 1 Erase application zone 2 which furs is conected

// _Security2App1PWLen: Passowrd length, fixed as 4

// _Security2App2PW: Erase Password

//Return P:

// Successful 0

// Failure Others

P= 'N' (0X4E) Failure

P= 'E' (0X45) No card

P= 'F' (0X46) Zone 2 is crashed , and out of operation

P= 'W' (0X57) Card on non-operatable position

//

Write Defined Address

8. **int APIENTRY AT88SC102_Write(HANDLE ComHandle, BYTE _Index, BYTE _Address, BYTE _dataLen, BYTE _BlockData[]);**

//Parameter:

// ComHandle

// _Index: Zone Selection 0=Control Zone; 1=Application Zone 1; 2= Application Zone 2

// _Address: Starting Address

// _dataLen: Data Length

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	40/49

// _BlockData: Data

//Return P:

// Successful 0

// Failure Others

P= 'N' (0X4E) Failure

P= 'E' (0X45) No card

P= 'W' (0X57) Card on non-operatable position

//

Personization, enter security mode 2

9. int APIENTRY AT88SC102_InitSecurity2(HANDLE ComHandle, BYTE _CtrlMode);

//Parameter:

// ComHandle

// _CtrlMode: **Control Mode**

_CtrlMode =0x30 Enter security mode 2 for testing

_CtrlMode =0x31 Enter simulated security mode 2 and recover to mode 1

_CtrlMode =0x32 Enter security mode 2 totoally and unrecoverable to mode 1

//Return P:

// Successful 0

// Failure Others

P= 'N' (0X4E) Failure

P= 'E' (0X45) No card

P= 'W' (0X57) Card on non-operatable position

Setting application 1 and 2 password before enter security mode 2, the data in application 1 (0x16) and application 2 (0x5C) shouldn't modify, because they control the write enable or not

After entering security mode 2, befor erase, please reserve the data to alert data missing and also reserve the fuse status. The fuse could onlyoperater 128times and the card only could be write 100,00 times after out of life cycle of fuse

//

Erase operation character EC2 in application zone 2 and setting as invalid

10. int APIENTRY AT88SC102_DisableEC2(HANDLE ComHandle);

//Parameter:

// ComHandle

// Return P:

// Successful 0

// Failure Others

P= 'N' (0X4E) Failure

P = 'E' (0X45) No card

P = 'W' (0X57) Card on non-operatable position

Use this function to unlimit erase time under security mode w before personalization operation

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	41/49

Otherwise, after setting mode2, the erase time as default will be 128 times. It will be unavailable to set the unlimited of erase time. And please reserve the related parameter of EC2

//////////////////////////////////// AT88SC1604 **Function** //////////////////////////////////////

Reset

1. **int APIENTRY AT88SC1604_Reset(HANDLE ComHandle)**

//Parameter:

// ComHandle

//Return P:

// Successful 0

// Failure Others

= 'N' (0X4E) Failure

= 'E' (0X45) No card in the reader

= 'W' (0X57) Card on non-operatable position

////////////////////////////////////

Password Verification

2. **int APIENTRY AT88SC1604_VerifyPWD(HANDLE ComHandle, BYTE _PWIndex, BYTE _PWData[])**

//Parameter:

// ComHandle.

// PWIndex: Password Index

Password Type : = 0 Verify Main Password

= 1 Verify Application Zone 1

= 2 Verify Application Zone 1 Erase Password

= 3 Verify Application Zone 2

= 4 Verify Application Zhone 1 Erase Password

= 5 Verify Application Zone 3

= 6 Verify Application Zone 1 Erase Password

= 7 Verify Application Zone 4 Password

= 8 Verify Application Zone 4 Erase Password

// _PWData: Password Data

// Return P:

// Successful 0

// Failure Others

P= 'N' (0X4E) Failure

P= 'E' (0X45) No Card in the machine

P= 'W' (0X57) Card on non-operatable position

P= 'F' (0X46) Card Crashed


```
//  _Address: Starting Address
```

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	44/49

// _dataLen; Data Length

//Return P:

// Successful 0

// Failure Others

P= 'N' (0X4E) Failure

P= 'E' (0X45) No card in the reader

P= 'W' (0X57) Card is not on the operatable position

////////////////////////////////////

Personalization (Security Mode 2)

7. **int APIENTRY AT88SC1604_Personalization(HANDLE ComHandle, BYTE _data);**

//Parameter:

// ComHandle

// _data: Operation Number = 0x30 Soft personalization (Imitate personalization and enter security mode 2 for testing)
= 0x31 Quit soft personalization
= 0x32 Total personalization

//Return P:

// Successful 0

// Failure Others

P= 'N' (0X4E) Failure

P= 'E' (0X45) No card in the reader

P= 'W' (0X57) Card is not on the operatable position

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	45/49

//////////////////////////////////// AT88SC1608 **Function**////////////////////////////////////

//Reset

1. int APIENTRY AT88SC1608_Reset(HANDLE ComHandle);

//Parameter:

// ComHandle

//Return P:

// Successful 0

// Failure Others

= 'N' (0X4E) Failure

= 'E' (0X45) No card in the reader

= 'W' (0X57) Card on non-operatable position

////////////////////////////////////

//Verify Password

2. int APIENTRY AT88SC1608_VerifyPWD(HANDLE ComHandle, BYTE _PWIndex, BYTE _PWDData[],BYTE _PWDDataLen);

//Parameter:

// ComHandle:

// PWIndex: Password Index

= 0 Verify Area1 password

= 1 Verify Area2 password

= 2 Verify Area3 password

= 3 Verify Area4 password

= 4 Verify Area5 password

= 5 Verify Area6 password

= 6 Verify Area7 password

= 7 Verify Area8 password

= 8 Verify Application1 password

= 9 Verify Application2 password

=10 Verify Application3 password

=11 Verify Application4 password

=12 Verify Application5 password

= 13 Verify Application6 password

= 14 Verify Application7 password

= 15 Verify Application8 password

// _PWDData: Password 3 byte

// **_PWDDataLen: Length of password**

//Return P:

// Successful 0

```
//Parameter:
```


////////////////////////////////////

	SPECIFICATION	Model No.	CRT-310
		Date	2011/07/29
	DLL	Ver.	v20110729
		Page	49/49

//////////////////////////////// AT45D041 **Function**////////////////////////////////

Reset

1. **int APIENTRY AT45D041_Reset(HANDLE ComHandle)**

//Parameter:

// ComHandle

//Return:

// Successful 0

// Failure Others

////////////////////////////////

Read Defined Page Data

2. **int APIENTRY AT45D041_Read(HANDLE ComHandle, int _Address, BYTE _BlockData[])**

//Parameter:

// ComHandle

// _Address: Page Address: 0x0000---0x07FF

// _BlockData: Block data

//Return P:

// Successful 0, together with _BlockData

// Failure Others

= 'E' (0X45) No card in the machine

= 'W' (0X57) Card on non-operatable position

////////////////////////////////

Write Defined Page Data

3. **int APIENTRY AT45D041_Write(HANDLE ComHandle, int _Address, BYTE _BlockData[])**

//参数:

// ComHandle

// _Address: Page Address: 0x0000---0x07FF

// _BlockData: Writing data

//Return P:

// Successful 0

// Failure Others

= 'E' (0X45) No card in the reader

= 'W' (0X57) Card on non-operatable position