	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	1/33


CRT Card Operation DLL Specification

CONTENT

1. Magnetic Card Function	2
2. Mifare 1 Card Function.....	6
3. IC Card Operation Communal Function	11
4. CPU Card Function (T=0/T=1).....	13
5. SIM Card Function (T=0/T=1).....	15
6. SLE4428 Card Function	17
7. SLE4442 Card Function.....	22
8. 24CXX Series Card Function.....	25
9. AT88SC102 Card Function	27
10. AT88SC1604 Card Function.....	30
11. AT45D041 Card Function	33

Note:

All the function in grey are old version, it is not suggested to be used anymore. But in order to ensure the compatibility, DLL provides both two kinds of functions.

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	2/33

//////////////////// Magnetic Card Function //////////////////////

Read magnetic tracks data.

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

//parameter:

// ComHandle: Handle of Com.

// _mode: Data mode.

Read card mode: 0x30 ASCII code read data
 0x31 binary code read data

// _track: Track.

Appointed track code: 0x30 do not read any track

 0x31 read track 1

 0x32 read tracks 2

 0x33 read tracks 3

 0x34 read track 1, 2

 0x35 read track 2, 3

 0x36 read 1, 3 tracks

 0x37 read all 3 tracks

// _BlockData: Data in the block.

Data package format of each tracks:

Starting byte + reading status byte + data of tracks

Starting byte: 0x1F

Reading status byte: 0x59 correct data information of track if reading successful

 0x4E wrong information if reading error

 0x4F can not read, track data is 0xE0

Error information:

 0xE1 error, no STX

 0xE2 error, no EXT

 0xE3 error, no VRC

 0xE4 error, wrong LRC

 0xE5 error, blank track

*The format of binary code read data is:


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

Track 2, 3: b0, b1, b2, b3, P

Notice:

When it is set in ASCII code reading mode, data of each track is uploaded in one byte ASCII code.

Eg, First byte of track 1 0x03 (HEX)

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	3/33

Uploading data package is 0x33 (ASCII)

When it is set in binary code reading mode, data of each track is uploaded in 4 bit per unit in ASCII code.

Eg, First byte of track 1 0x03 (HEX)

Uploading data package is 0x30 0x33

//Return:

// Success: 0

// Failure: not 0

P='E' (0x45) No card in the reader

P='W' (0x57) Card is not at the permitted operation position

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

Re-Reading the appointed tracks data according to the appointed mode

2. int APIENTRY MC_ReReadTrack(HANDLE ComHandle, BYTE _mode, BYTE _track, BYTE _BlockData[])

// parameter:

// ComHandle: Handle of Com.

// _mode: Data mode.

Read card mode: 0x30 ASCII code read data
0x31 binary code read data

// _track: Track.

Appointed track code: 0x30 do not read any track

0x31 read track 1

0x32 read tracks 2

0x33 read tracks 3

0x34 read track 1, 2

0x35 read track 2, 3

0x36 read 1, 3 tracks

0x37 read all 3 tracks

// _BlockData: Data in the block.

Data package format of each tracks:

Starting byte + reading status byte + data of tracks

Starting byte: 0x1F


Reading status byte: 0x59 correct data information of track if reading successful

0x4E wrong information if reading error

0x4F can not read, track data is 0xE0

Error information:

0xE1 error, no STX

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	4/33

0xE2 error, no EXT
0xE3 error, no VRC
0xE4 error, wrong LRC
0xE5 error, blank track

*The format of binary code read data is:

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

Track 2, 3: b0, b1, b2, b3, P

Notice:

When it is set in ASCII code reading mode, data of each track is uploaded in one byte ASCII code.

Eg, First byte of track 1 0x03 (HEX)

Uploading data package is 0x33 (ASCII)

When it is set in binary code reading mode, data of each track is uploaded in 4 bit per unit in ASCII code.

Eg, First byte of track 1 0x03 (HEX)

Uploading data package is 0x30 0x33

//Return:

// Success: 0

// Failure: not 0

P='E' (0x45) No card in the reader

P='W' (0x57) Card is not at the permitted operation position

P='N' (0x4E) Operation failed

Set the reading mode of CRT-287

3. int WINAPI CRT287_MCSSetting(HANDLE ComHandle, BYTE _mode, BYTE _ctrl, BYTE _track)

// parameter:

// ComHandle: Handle of Com.

// _mode: Data mode.

Read card mode: 0x30 ASCII code read data

0x31 binary code read data

_ctrl: Uploading control: 0x30 Permit initiative uploading. (Suggest not using reading command to read the data under this mode.)


0x31 Prohibit initiative uploading. (Under this mode, use the magnetic card reading command to attain information from card)

// _track: Track.

Appointed track code: 0x30 do not read any track

0x31 read track 1

0x32 read tracks 2

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	5/33


0x33 read tracks 3
0x34 read track 1, 2
0x35 read track 2, 3
0x36 read 1, 3 tracks
0x37 read all 3 tracks

//Return:

// Success: 0

// Failure: -1

CREATOR

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	6/33

//////////////////// Mifare 1 Card Function //////////////////////

//M1 card identify.

1. int APIENTRY RF_DetectCard(HANDLE ComHandle);

//Parameter:

// None

//Return Value:

// Success: 0

// Failure: not 0

P= 'N' (0X4E) Fail to seek card

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

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

//Capture the series No. of M1 card.

2. int APIENTRY RF_GetCardID(HANDLE ComHandle, BYTE _CardID[4]);

//Parameter:

// _CardID[4]: Series No. of M1 card.

//Return Value:

// Success: 0.

// Failure: not 0.

P= 'N' (0X4E) fail to capture card S/N and return empty S/N (0X00, 0X00, 0X00, 0X00)

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

Load Section Password.

3. int APIENTRY RF_LoadSecKey(HANDLE ComHandle, BYTE _Sec, BYTE _KEYType , BYTE _KEY[6]);

//Parameter:

// _Sec: Sector number.

// _KEYType: Password option: 0=KEYA; 1=KEYB.

// _KEY: Password.


//Return Value:

// Success: 0.

// Failure: not 1.

P= '0' (0X30) no RF card in

P= '3' (0X33) password error

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	7/33

P= 'E' (0X45) no card in

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

//

Read data from appointed block.

**4. int APIENTRY RF_ReadBlock(HANDLE ComHandle, BYTE _Sec, BYTE _Block, BYTE
_BlockData[16]);**

//Parameter:

// _Sec: Sector number.

// _Block: Block number.

// _BlockData: Data content in the block.

//Return Value:

// Success: 0.

// Failure: not 0

P= '0' (0X30) no RF card in

P= '1' (0X31) operated sector No. is wrong (not the sector by password checked)

P= '2' (0X32) S/N of operated card error

P= '3' (0X33) password error

P= '4' (0X34) data read error

P= 'E' (0X45) no card in

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

//

Write data to appointed block.

**5. int APIENTRY RF_WriteBlock(HANDLE ComHandle, BYTE _Sec, BYTE _Block, BYTE
_BlockData[16]);**

//Parameter:

// _Sec: Sector number.

// _Block: Block number.

// _BlockData: Data content in the block.

//Return Value:

// Success: 0.


// Failure: not 0.

P= '0' (0X30) no RF card in

P= '1' (0X31) operated sector No. is wrong (not the sector by password checked)

P= '2' (0X32) S/N of operated card error

P= '3' (0X33) password error

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	9/33

Read value: Executed by read sector block data command, for the 16 byte data format, it should be in MIFARE card value data format. If yes, read the value, if not, reading error alert (error data format).

When processing a value operation, block 3 of each sector cannot save a value data. And notice the address range of the sector when initializing value, increment, decrement, read value.

//

//Read value

7. int APIENTRY RF_ReadValue(HANDLE ComHandle, BYTE _Sec, BYTE _Block, BYTE _Data[4]);

// Parameter:

// _Sec: Sector number.

// _Block: Block number.

// _Data: initial value.

//Return Value:

// Success: 0.

// Failure: not 0.

P= '0' (0X30) no RF card in

P= '1' (0X31) operated sector No. is wrong (not the sector by password checked)

P= '2' (0X32) S/N of operated card error

P= '3' (0X33) password error

P= '4' (0X34) format of block data error (not written in a value format)

P= '5' (0X35) increment/decrement over load

P= 'E' (0X45) no card inside

P= 'Y' (0X59) operation successful

P= 'W' (0X57) card is not at the permitted operation position

//Note: Before execute the Increment or Decrement, you should first use the write data function to initialize the data in the appointed sector, or there will be error occurred.

//

// Increment.

8. int APIENTRY RF_Increment(HANDLE ComHandle, BYTE _Sec, BYTE _Block, BYTE _Data[4]);

//parameter:


// _Sec: Sector number.

// _Block: Block number.

// _Data: Data.

//Return Value:

// Success: 0.

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	10/33

// Failure: not 0.

P= '0' (0X30)	no RF card in
P= '1' (0X31)	operated sector No. is wrong (not the sector by password checked)
P= '2' (0X32)	S/N of operated card error
P= '3' (0X33)	password error
P= '4' (0X34)	format of block data error (not written in a value format)
P= '5' (0X35)	increment over load
P= 'E' (0X45)	no card inside
P= 'Y' (0X59)	operation successful
P= 'W' (0X57)	card is not at the permitted operation position

//Note: Before execute the Increment or Decrement, you should first use the write data function to initialize the data in the appointed sector, or there will be error occurred.

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

// Decrement.

9. int APIENTRY RF_Decrement(HANDLE ComHandle, BYTE _Sec, BYTE _Block, BYTE _Data[4]);

//parameter:

// _Sec: sector number.

// _Block: block number.

// _Data: data.


//Return Value:

// Success: 0.

// Failure: not 0.

P= '0' (0X30)	no RF card in
P= '1' (0X31)	operated sector No. is wrong (not the sector by password checked)
P= '2' (0X32)	S/N of operated card error
P= '3' (0X33)	password error
P= '4' (0X34)	format of block data error (not written in a value format)
P= '5' (0X35)	decrement over load
P= 'E' (0X45)	no card inside
P= 'Y' (0X59)	operation success
P= 'W' (0X57)	card is not at the permitted operation position

//Note: Before execute the Increment or Decrement, you should first use the write data function to initialize the data in the appointed sector, or there will be error occurred.

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	11/33

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

// Use the function to power on the IC card before operating on it

1. int APIENTRY CRT_IC_CardOpen(HANDLE ComHandle);

int APIENTRY IC_OpenCard(HANDLE ComHandle, BYTE *_bATR);//Function for the old version

//parameter:

// _bATR

// character pointer, point to reset response string attained from card memory section and return to application.

Under HEXADECIMAL mode, application should have given at least 4 bytes room to _bATR before use it; under ASCIISTRING mode, should have given at least 8 byte room to _bATR

//return value:

// success: 0. and the content of _bATR is the reset response string attained from card.

// failure: not 0.

////////////////////
//Use the function to power off the IC card before pulling it out after finishing all operations.

2. int APIENTRY CRT_IC_CloseCard(HANDLE ComHandle);

int APIENTRY IC_CloseCard(HANDLE ComHandle);//Function for the old version

//parameter:

// none

//return value:

// success: 0.

// failure: not 0.

////////////////////
//Auto test card type in card reader.

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

int APIENTRY CRT_R1_DetectCard(HANDLE ComHandle,BYTE *_CardType,BYTE *_CardInfor);


//Function for the old version

//parameter:

// CardType: Card type

// CardInfor: Card information

CardType	CardInfor	Instruction
'N'	'0'	No card inside
	'1'	Unknown card type
'0'	'0'	Contactless RF card


	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	12/33

'1'	'0'	Contact CPU card T=0
	'1'	Contact CPU card T=1
'2'	'0'	24C01 card
	'1'	24C02 card
	'2'	24C04 card
	'3'	24C08 card
	'4'	24C16 card
	'5'	24C32 card
	'6'	24C64 card
	'6'	24C64 card
'3'	'0'	SL4442 card
	'1'	SL4428 card
'4'	'0'	AT88S102 card
	'1'	AT88S1604 card
	'2'	AT45D041 card

//return value:

// Success: 0.

// Failure: not 0.

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	13/33

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

CPU card reset.

1. int APIENTRY CPU_Reset(HANDLE ComHandle, BYTE _exData[], int _exdataLen);

//Parameter:

// ComHandle: Handle of COM.

//Return Value:

// _exData: Return data

// _exdataLen: Return the data length

// return value:

// Success: 0

// Failure: not 0

P= 'N' (0X4E) Operation failed

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

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

CPU Card C-APDU command.

2. int APIENTRY CPU_C_APDU(HANDLE ComHandle, BYTE _dataLen, BYTE _APDUData[], BYTE _exData[], int _exdataLen);

//Parameter:

// _dataLen: C-APDU command length.

// _APDUData: CPU Card C-APDU command.

// _exData: Return data

// _exdataLen: Return the data length

// return value:


// Success: 0

// Failure: not 0

P= 'N' (0X4E) Operation failed

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

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

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	14/33

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

CPU card reset.

1. int APIENTRY CPU_T1_Reset(HANDLE ComHandle, BYTE _exData[], int _exdataLen);

//Parameter:

// ComHandle: Handle of COM.

//Return Value:

// _exData: Return data

// _exdataLen: Return the data length

// return value:

// Success: 0

// Failure: not 0

P= 'N' (0X4E) Operation failed

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

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

CPU Card C-APDU command.

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

//Parameter:

// _dataLen: C-APDU command length.

// _APDUData: CPU Card C-APDU command.

// _exData: Return data

// _exdataLen: Return the data length

// return value:


// Success: 0

// Failure: not 0

P= 'N' (0X4E) Operation failed

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

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

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	15/33

////////////////////////////////////// SIM Card Function (T=0) //

SIM card reset.

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

//parameter:

// ComHandle: Handle of Com.

// SIMNo: SIM card connector number.

// _exData: Return data

// _exdataLen: Return the data length

// return value:

// Success: 0

// Failure: not 0

SIM card C-APDU command.

2. int APIENTRY SIM_C_APDU(HANDLE ComHandle, BYTE SIMNo, int _dataLen, BYTE _APDUData[], BYTE _exData[], int _exdataLen);

//parameter:

// SIMNo: SIM card connector number.

SIM card connector No.=0x30 operation SIM card 1

=0x31 operation SIM card 2

=0x32 operation SIM card 3

=0x33 operation SIM card 4

=0x34 operation SIM card 5

=0x35 operation SIM card 6

=0x36 operation SIM card 7

=0x37 operation SIM card 8

// _dataLen: C-APDU command length.

// _APDUData: CUP card C-APDU command.


// _exData: Return data

// _exdataLen: Return the data length

//Return:

// Success: 0

// Failure: not 0

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	16/33

////////////////////////////////////// SIM Card Function (T=1) //

SIM card reset.

1. int APIENTRY SIM_T1_Reset(HANDLE ComHandle, BYTE _SIMNo, BYTE _exData[], int _exdataLen);

//parameter:

// ComHandle: Handle of Com.

// SIMNo: SIM card connector number.

// _exData: Return data

// _exdataLen: Return the data length

// return value:

// Success: 0

// Failure: not 0

SIM card C-APDU command.

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

//parameter:

// SIMNo: SIM card connector number.

SIM card connector No.=0x30 operation SIM card 1

=0x31 operation SIM card 2

=0x32 operation SIM card 3

=0x33 operation SIM card 4

=0x34 operation SIM card 5

=0x35 operation SIM card 6

=0x36 operation SIM card 7

=0x37 operation SIM card 8

// _dataLen: C-APDU command length.

// _APDUData: CUP card C-APDU command.


// _exData: Return data

// _exdataLen: Return the data length

//Return:

// Success: 0

// Failure: not 0

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	17/33

//////////////////////////////////// SLE4428 Card Function //////////////////////////////////////

//Attain _wLength long character string from address of _wAddr in the card and saved in _bReadData pointer and return to application.

1. int APIENTRY SLE4428_ReadChar(HANDLE ComHandle, WORD _wAddr, WORD _wLength, BYTE *_bReadData);

//parameter:

// _wAddr

// point to the starting address which will be read in the memory section.

// _wLength

// point to the length of character string which will be read. Length of SLE4428 memory section is 1024 bytes.

// _bReadData

// character pointer, point to character string attained from card memory section and return to application.

Under HEXADECIMAL mode, application should have given at least _wLength byte room to _bReadData before use it; under ASCIISTRING mode, at least (_wLength * 2) byte room.

//return value:

// success: 0. and the content of _bReadData is character string attained from card.

// failure: not 0. No card=1; card type error=2

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

//Writes character string appointed by _bWriteData into _wAddr address; _wLength byte in total. To ensure the write operation successfully, the correct appointed content and length of _bWriteData should be guaranteed.

2. int APIENTRY SLE4428_WriteChar(HANDLE ComHandle, WORD _wAddr, WORD _wLength, BYTE *_bWriteData);

//parameter:

// _wAddr

// point to the re-written starting address in the memory section.

// _wLength

// point to the length of character string which will be written. Length of SLE4428 memory section is 1024 bytes.

// _bWriteData


// character pointer. Application should save the character string which is going to be written in the card into

_bWriteData pointer. The string will cover _wLength bytes starting from _wAddr address in the card. Under

HEXADECIMAL mode, the character string appointed by _bWriteData should be _wLength byte; under

ASCIISTRING mode, the string appointed by _bWriteData should be (_wLength * 2) bytes.

// return value:

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	18/33

// success: 0.
 // failure: not 0. No card=1; card type error=2; write card (protect bit) error =3

//

// Read memory section in SLE4428 card. Under HEXADECIMAL mode, 0x00 shows correspond data section has been protected; 0x01 shows unprotected, can be modified after password parity correct.

3. int APIENTRY SLE4428_PReadChar(HANDLE ComHandle, WORD _wAddr, WORD _wLength, BYTE *_bPReadData);


// parameter:
 // _wAddr
 // point to the starting address which will be read in protect section.
 // _wLength
 // point to the length of character string which will be read in protect section. Length of SLE4428 memory section is 1024 bytes.
 // _bPReadData
 // character pointer, point to character string attained from protect section and return to application. Under HEXADECIMAL mode, application should have given at least _wLength byte room to _bPReadData pointer before use it; under ASCIISTRING mode, at least (_wLength * 2) byte room.
 // return value:
 // success: 0, and content of _bPReadData is character string attained from card.
 // failure: not 0.

//

// Write the character string pointed by _bWriteData into _wAddr address of the card, _wLength byte in total, meantime write corresponding protect bit.

4. int APIENTRY SLE4428_PWriteChar(HANDLE ComHandle, WORD _wAddr, WORD _wLength, BYTE *_bWriteData);

// parameter:
 // _wAddr
 // point to the re-written starting address in the memory section.
 // _wLength
 // point to the length of character string which will be written. The range of SLE4428 protected memory section is from address 0 to 1023, 1024 bytes in total.
 // _bWriteData
 // character pointer. Application should save the character string which is going to be written in the card into _bWriteData pointer. The string will cover _wLength bytes starting from _wAddr address in the card. Under HEXADECIMAL mode, the character string appointed by _bWriteData should be _wLength byte; under

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	19/33

ASCIISTRING mode, the string appointed by _bWriteData should be (_wLength * 2) bytes.

// return value:

// success: 0.

// failure: not 0.

//

// From _wAddr byte address, read a non-symbol long integer value ranged from 0x00000000 to 0xFFFFFFFF, then save in the room appointed by _dwValue pointer and return to application.

5. int APIENTRY SLE4428_ReadValue(HANDLE ComHandle, WORD _wAddr, DWORD *_dwValue);

// parameter:

// _wAddr

// point to the starting address of value in card memory section which will be read

// _dwValue

// non-symbol long integer pointer, content is the the value read from card.

// return value:

// success: 0. and save the value to be read into _dwValue and return to application.

// failure: not 0.

//

// Write _dwValue value, ranged from 0x00000000 to 0xFFFFFFFF, into _wAddr byte address in the card.

6. int APIENTRY SLE4428_WriteValue(HANDLE ComHandle, WORD _wAddr, DWORD _dwValue);

// parameter:

// _wAddr

// point to the address in card memory section which will be written.

// _dwValue

// the value will be written into card memory section

// return value:

// success: 0.


// failure: not 0.

//

// Password parity needed before application modify the memory section of SLE4428 card; only when password parity correct can memory written be available.

7. int APIENTRY SLE4428_VerifyPWD(HANDLE ComHandle, BYTE _bPassWord1, BYTE _bPassWord2);

// parameter:

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	20/33

```
// _bPassWord1
//     first byte of the password
// _bPassWord2
//     second byte of the password
// return value:
//     success: 0.
//     failure: not 0.
```

//

// Read password on SLE4428 card.

10. int APIENTRY SLE4428_ReadPWD(HANDLE ComHandle, BYTE *_bPWD);

// parameter:

// _bPWD

// character pointer, which point to password read from card and return to application. Under
HEXADECIMAL mode, application should have given at least 2 byte room to _bPWD pointer; Under
ASCIISTRING mode, at least 4 byte room..

// return value:

// success: 0.

// failure: not 0.

//

//Modify the password in SLE4428 card.

**9. int APIENTRY SLE4428_WritePWD(HANDLE ComHandle, BYTE _bPassWord1, BYTE
_bPassWord2);**

// parameter:

// _bPassWord1

// first byte of the password

// _bPassWord2

// second byte of the password

// return value:

// success: 0.


// failure: not 0.

//

// Read password error counter value on SLE4428 card.

10. int APIENTRY SLE4428_ReadPAC(HANDLE ComHandle, BYTE *_bPAC);

// parameter:

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	21/33

// _bPAC


// character pointer, point to counter value read from card and return to application, which should have given at least 1 byte room to _bPAC pointer before use it.

// return value:

// success: 0.

// failure: not 0.

CREATOR

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	22/33

//////////////////////////////////// SLE4442 Card Function //////////////////////////////////////

SEL4442 Reset.

1. int APIENTRY SLE4442_Reset(HANDLE ComHandle);

//Parameter:

// ComHandle: Handle of COM.

//Return Value:

// Success: 0.

// Failure: not 0.

SEL4442 read data.

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

//Parameter:

// ComHandle: Handle of COM.

// _Address: Address.

// _dataLen: Data length.

// _BlockData: Data.

//Return Value:

// Success: 0.

// Failure: not 0.

SEL4442 read protect data.

3. int APIENTRY SLE4442_ReadP(HANDLE ComHandle, BYTE _BlockData[32]);

//Parameter:

// ComHandle: Handle of COM.

// _BlockData: Data.

//Return Value:


// Success: 0.

// Failure: not 0.

SEL4442 read safety data.

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

//Parameter:

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	23/33

// ComHandle: Handle of COM.

// _BlockData: Data.

//Return Value:

// Success: 0.

// Failure: not 0.

//

SEL4442 password parity.

5. int APIENTRY SLE4442_VerifyPWD(HANDLE ComHandle, BYTE _PWData[3]);

//Parameter:

// ComHandle: Handle of COM.

// _PWData: Password.

//Return Value:

// Success: 0.

// Failure: not 0.

//

SEL4442 write data.

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

//Parameter:

// ComHandle: Handle of COM. .

// _Address: Address.

// _dataLen: Data length.

// _BlockData: Data.

//Return Value:

// Success: 0.

// Failure: not 0.

//

SEL4442 write protect data.

7. int APIENTRY SLE4442_WriteP(HANDLE ComHandle, BYTE _BlockData[32]);


//Parameter:

// ComHandle: Handle of COM. .

// _BlockData: Data.

//Return Value:

// Success: 0.

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	24/33

// Failure: not 0.

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

SEL4442 modify password.

8. int APIENTRY SLE4442_WritePWD(HANDLE ComHandle, BYTE _PWData[3]);

//Parameter:

// ComHandle: Handle of COM.


// _PWData: Password.

//Return Value:

// Success: 0.

// Failure: not 0.

CREATOR

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	25/33

//////////////////////////////////// 24Cxx Series Card Function //////////////////////////////////////

Test the type of 24CXX card.

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

//Parameter:

// ComHandle: Handle of COM.

// _CardType: Card Type.

//Return Value:

// Success: 0.

// Failure: not 0.

Card power on.

2. int APIENTRY IC_CardOpen(HANDLE ComHandle);

//Parameter:

// ComHandle: Handle of COM.

//Return Value:

// Success: 0.

// Failure: not 0.

Card power off.

3. int APIENTRY IC_CardClose(HANDLE ComHandle);

//Parameter:

// ComHandle: Handle of COM.

//Return Value:

// Success: 0.

// Failure: not 0.

Read data from appointed address.


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

//Parameter:

// ComHandle: Handle of COM.

// _Address: Starting address.

// _dataLen: Data length.

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	26/33

// _BlockData: Data.

//Return Value:

// Success: 0.

// Failure: not 0.

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

Write data to appointed address.

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

//Parameter:

// ComHandle: Handle of COM.

// _Address: Starting address.

// _dataLen: Data length.


// _BlockData: Data.

//Return Value:

// Success: 0.

// Failure: not 0.

CREATOR

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	27/33

//////////////////////////////////// **AT88SC102 Card Function** //////////////////////////////////////

Reset.

1. int APIENTRY AT88SC102_Reset(HANDLE ComHandle);

//Parameter:

// ComHandle: Handle of CCOM.

//Return:

// Success: 0

// Failure:-1

Password parity.

2. int APIENTRY AT88SC102_VerifyPWD(HANDLE ComHandle, BYTE _PWIndex, BYTE _PWDData[]);

//Parameter:

// ComHandle: Handle of COM.

// PWIndex: Password Type Index. 0=main password; 1=Erase password 1; 2=Erase password 2

// _PWDData: Password data.

//Return:

// Success: 0

// Failure:-1

Modify password.

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

//Parameter:

// ComHandle: Handle of COM.

// PWIndex: Password Type Index. 0=main password; 1=Erase password 1; 2=Erase password 2

// _PWDData: Password data.

//Return:


// Success: 0

// Failure:-1

Read data from appointed address.

4. int APIENTRY AT88SC102_Read(HANDLE ComHandle, BYTE _Index, BYTE _Address, BYTE _dataLen, BYTE _BlockData[]);

//Parameter:

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	28/33

```
// ComHandle: Handle of COM.
// _Index: Data Zone select. 0=Control zone; 1=application zone 1; 2= application zone 2
// _Address: Starting address.
// _dataLen: Data length.
// _BlockData: data.
//Return:
// Success: 0
// Failure:-1
```

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

Erase data from appointed address.

5. int APIENTRY AT88SC102_Clear(HANDLE ComHandle, BYTE _Safelever, BYTE _Index, BYTE _Address, BYTE _dataLen);


```
//Parameter:
// ComHandle: Handle of COM.
// _Index: Data Zone select. 0= Control zone; 1= application zone 1; 2= application zone 2
// _Address: Starting address.
// _dataLen: Data length.
//Return:
// Success: 0
// Failure:-1
```

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


Write data to appointed address.

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

```
//Parameter:
// ComHandle: Handle of COM.
// _Index: Data Zone select. 0=Control zone; 1=application zone 1; 2= application zone 2
// _Address: Starting address.
// _dataLen: Data length.
// _BlockData: Data.
//Return:
// Success: 0
// Failure:-1
```

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	29/33

CREATOR

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	30/33

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

Reset.

1. int APIENTRY AT88SC1604_Reset(HANDLE ComHandle)

//parameter:

// ComHandle: Handle of Com.

//Return:

// Success: 0

// Failure:-1

Password parity.

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

//parameter:

// ComHandle: Handle of Com.

// PWIndex: Password type index.

Type of password: = 0 parity main password

=1 parity password of application zone 1

= 2 parity erase password of application zone 1

= 3 parity password of application zone 2

= 4 parity erase password of application zone 2

= 5 parity password of application zone 3

= 6 parity erase password of application zone 3

= 7 parity password of application zone 4

= 8 parity erase password of application zone 4

// _PWDData: Password

//Return:

// Success: 0

// Failure:-1

Modify password.


3. int APIENTRY AT88SC1604_WritePWD(HANDLE ComHandle, BYTE _PWIndex, BYTE _PWDData[])

//parameter:

// ComHandle: Handle of Com.

// PWIndex: Password type index.

Type of password: = 0 parity main password

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	31/33

- =1 parity password of application zone 1
- = 2 parity erase password of application zone 1
- = 3 parity password of application zone 2
- = 4 parity erase password of application zone 2
- = 5 parity password of application zone 3
- = 6 parity erase password of application zone 3
- = 7 parity password of application zone 4
- = 8 parity erase password of application zone 4

// _PWData: Password.

//Return:

// Success: 0

// Failure:-1

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

Read appointed address data.

4. int APIENTRY AT88SC1604_Read(HANDLE ComHandle, BYTE _Index, int _Address, BYTE _dataLen, BYTE _BlockData[])

//parameter:

// ComHandle: Handle of Com.

// _Index: Data zone index.

Zone code: = 0 zone 1 (0x020 --- 0x21A)
 = 1 zone 2 (0x21B --- 0x420)
 = 2 zone 3 (0x421 ---- 0x621)
 = 3 zone 4 (0x622 ---- 0x7F5)
 = 4 other zone (except zone 1, 2, 3)

// _Address: Starting address.

// _dataLen: Data length.

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

Write appointed address data.


5. int APIENTRY AT88SC1604_Write(HANDLE ComHandle, BYTE _Index, int _Address, BYTE _dataLen, BYTE _BlockData[])

//parameter:

// ComHandle: Handle of Com.

// _Index: Data zone index.

Zone code: = 0 zone 1 (0x020 --- 0x21A)
 = 1 zone 2 (0x21B --- 0x420)

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	32/33

- = 2 zone 3 (0x421 ---- 0x621)
- = 3 zone 4 (0x622 ---- 0x7F5)
- = 4 other zone (except zone 1, 2, 3)

// _Address: Starting address.

// _dataLen: Data length

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

Erase appointed address data.

6. int APIENTRY AT88SC1604_Clear(HANDLE ComHandle, BYTE _Index, int _Address, BYTE _dataLen)

//parameter:


// ComHandle: Handle of Com.

// _Index: Data zone index.

- Zone code: = 0 zone 1 (0x020 --- 0x21A)
- = 1 zone 2 (0x21B --- 0x420)
- = 2 zone 3 (0x421 ---- 0x621)
- = 3 zone 4 (0x622 ---- 0x7F5)
- = 4 other zone (except zone 1, 2, 3)

// _Address: Starting address.

// _dataLen: Data length

	SPECIFICATION	Model No.	CRT-286/287/288 /310/386
		Date	2005/9/1
	CRT Card Operation DLL Specification	Ver.	3.0
		Page	33/33

//////////////////////////////////// AT45D041 Card Function //////////////////////////////////////

Reset.

1. int APIENTRY AT45D041_Reset(HANDLE ComHandle)

//parameter:

// ComHandle: Handle of Com.

//Return:

// Success: 0

// Failure: -1

Read appointed address page data.

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

//parameter:

// ComHandle: Handle of Com.

// _Address: Starting address.

// _dataLen: Data length.

Write appointed address page data.

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

//parameter:

// ComHandle: Handle of Com.

// _Address: Starting address.

// _dataLen: Data length.