XPOS
Secondary development interface document

Document magament

Version history

Date	Version	Modify record	Author
20181109	1.0	Basic interface	yangjy
20190219	1.1	Add some interface and Modify some details	LinZhu
20190719	1.2	Add some interface and Modify some details	luoxizhu
20200522	1.3	Add some interface and Modify some details	linbingxing

Content

Content **3**

1 profile **13**

1.1 overall structure 13

1.1 modular design 13

2 System module(libapi_system) **14**

2.1 interface list 14

2.2 API interface 16

2.2.1 Sys_GetModuleVer *16*

2.2.2 System initialization(Sys_Init) *16*

2.2.3 Vendor personality parameter setting(Sys_Config) *17*

2.2.4 Get terminal info(Sys_GetTerminalInfo) *18*

2.2.5 Get system time(Sys_GetDateTime) *18*

2.2.6 Set systemn time(Sys_SetDateTime) *19*

2.2.7 Scan button(Sys_CheckKey) *20*

2.2.8 Clear button cache(Sys_ClrKey) *20*

2.2.9 Open timer(Sys_TimerOpen) *21*

2.2.10 Test timer (Sys_TimerCheck) *21*

2.2.11 Turn off timer(Sys_TimerClose) *22*

2.2.12 delay(Sys_Delay) *23*

2.2.13 Terminal sleep (Sys_Sleep) *23*

2.2.14 Terminal reboot(Sys_Reboot) *24*

2.2.15 Open Scan (Sys_scaner_open) *24*

2.2.16 Start scanning (Sys_scaner_start) *25*

2.2.17 Stop scanning (Sys_scaner_stop) *26*

2.2.18 Close Scan (Sys_scaner_close) *26*

2.2.19 Sys_GetTermSn *27*

2.2.20 Sys_SetAppVer *27*

2.2.21 Sys_driverlib_init........28
2.2.22 Sys_get_sublcd_probe........28
2.2.23 Sys_lcd_set_index........29
2.2.24 Sys_power_key_set_light........29
2.2.25 Sys_lcd_PowerDownTime........30
2.2.26 Sys_lcd_SetPowerDownTime........30
2.2.27 Sys_lcd_BackLightTime........31
2.2.28 Sys_lcd_SetBackLightTime........31
2.2.29 Sys_GetAppVer........32
3 Tool module (libapi_util)........32
3.1 interface list........32
3.2 API interface........34
3.2.1 Util_GetModuleVer........34
3.2.2 ASCII code change to BCD code (Util_Asc2Bcd)........34
3.2.3 BCD code convert to ASCII code (Util_Bcd2Asc)........35
3.2.4 Int type data convert to BCD code (Util_Int2Bcd)........36
3.2.5 BCD code convert to int type(Util_Bcd2Int)........36
3.2.6 Caculate LRC(Util_GenLrc)........37
3.2.7 DES encryption and decryption (Util_Des)........38
3.2.8 Chinese character copy(Util_StrCopy)........39
3.2.9 Waiting button(Util_WaitKey)........39
3.2.10 Input method input(Util_InputMethod)........40
3.2.11 String input (Util_InputText)........41
3.2.12 Amount input(Util_InputAmount)........42
3.2.13 IP input (Util_InputIp)........43
3.2.14 beep(Util_Beep)........45
3.2.15 Voice play (Play_Voice)........45
3.2.16 Generate random numbers(Util_Rand)........46
3.2.17 Play_Voice........46

3.2.18 Util_GeneCodePic *47*

3.2.19 Util_Led *48*

3.2.20 Util_Malloc *48*

3.2.21 Util_Free *49*

4 File module(libapi_file) 49

4.1 Interface list 49

4.2 API interface 51

4.2.1 UFile_GetModuleVer *51*

4.2.2 Check if the file exists (UFile_Check) *51*

4.2.3 File open / create(UFile_OpenCreate) *52*

4.2.4 File read(UFile_Read) *53*

4.2.5 Write file (UFile_Write) *54*

4.2.6 Positioning file pointer(UFile_Lseek) *55*

4.2.7 Delete file record (UFile_Delete) *56*

4.2.8 Close file (UFile_Close) *57*

4.2.9 Delete file (UFile_Remove) *58*

4.2.10 Rename file (UFile_Rename) *58*

4.2.11 Empty file(UFile_Clear) *59*

4.2.12 Get the number of file records (UFile_GetNumberOfRecords) *60*

4.2.13 Append file record(UFile_AppendRecord) *61*

4.2.14 Query records based on index number(UFile_GetRecordByIndex) *62*

4.2.15 Check record(UFile_GetRecord) *63*

4.2.16 Update record(UFile_UpdateRecord) *64*

4.2.17 Update records based on index number(UFile_UpdateRecordByIndex) *65*

4.2.18 Delete record (UFile_DeleteRecord) *67*

4.2.19 Delete records based on index number(UFile_DeleteRecordByIndex) *68*

4.2.20 Read one line text(UFile_ReadLine) *69*

4.2.21 Read non-fixed length records (UFile_ReadTLV) *69*

4.2.22 Write non-fixed data(UFile_WriteTLV) *70*

4.2.23 *Delete non-fixed length record (UFile_DeleteTLV)* *71*

5 IC card module (libapi_iccard) **73**

5.1 interface list 73

5.2 API interface 73

5.2.1 *Icc_GetModuleVer* *73*

5.2.2 *Turn on IC card device （Icc_Open)* *74*

5.2.3 *Turn off IC card device (Icc_Close)* *75*

5.2.4 *Turn off IC card device (Icc_Close)* *75*

5.2.5 *Test card(Icc_GetCardStatus)* *77*

5.2.6 *Contact card power on(Icc_PowerUp)* *77*

5.2.7 *Contact card power off (Icc_PowerDown)* *78*

5.2.8 *Contact card communication (Icc_ICComm)* *79*

5.2.9 *NFC card searching card (Icc_CTLSPowerUpAndSeek)* *81*

5.2.10 *NFC card power off(Icc_CTLSPowerDown)* *82*

5.2.11 *Use APDU to communicate with NFC card(Icc_CTLSComm)* *82*

5.2.12 *Icc_CTLSPowerUpAndSeek* *84*

5.2.13 *Icc_GetCardATR* *84*

6 communication（ libapi_comm） **85**

6.1 interface list 85

6.2 API interface 86

6.2.1 *comm_net_link* *86*

6.2.2 *comm_net_link_ex* *87*

6.2.3 *comm_net_unlink* *87*

6.2.4 *comm_sock_create* *88*

6.2.5 *comm_sock_connect* *89*

6.2.6 *comm_sock_recv* *89*

6.2.7 *comm_sock_send* *90*

6.2.8 *comm_sock_close* *91*

6.2.9 comm_ssl_init 91

6.2.10 comm_ssl_connect 92

6.2.11 comm_ssl_connect2 93

6.2.12 comm_ssl_send 93

6.2.13 comm_ssl_recv 94

6.2.14 comm_ssl_close 95

6.2.15 comm_wifi_list_ap 96

6.2.16 comm_wifi_link_ap 96

6.2.17 comm_wifi_unlink_ap 97

6.2.18 comm_wifi_get_link_state 98

6.2.19 comm_wifi_get_signal 98

6.2.20 wifi_get_ssid 99

6.2.21 wifi_get_ap_mac 99

6.2.22 wifi_get_rssi 100

6.2.23 wifi_get_channel 100

6.2.24 wifi_get_local_mac 101

6.2.25 wifi_get_local_ip 101

6.2.26 comm_atc_imei 102

6.2.27 comm_atc_cpin 102

6.2.28 comm_atc_imsi 103

6.2.29 comm_atc_signal 103

6.2.30 comm_atc_cell 104

6.2.31 comm_atc_lac 104

6.2.32 comm_atc_iccid 105

7 security（ libapi_security） 105

7.1 interface list 105

7.2 API interface 106

7.2.1 mksk_save_plaintext_key 106

7.2.2 mksk_save_encrypted_key 107

7.2.3 mksk_3des_run 108
7.2.4 dukpt_3des_run_ex 108
7.2.5 dukpt_load_key 109
7.2.6 dukpt_get_ksn 109
7.2.7 dukpt_prepare_key 110
7.2.8 dukpt_3des_run 110
7.2.9 dukpt_3des_run_ex 111
7.2.10 dukpt_load_key 112
7.2.11 dukpt_init_ipek 112
7.2.12 dukpt_init_ciphertext_ipek 113
7.2.13 dukpt_init_bdk 114
7.2.14 dukpt_init_key 114
7.2.15 sec_mac_proc 115
7.2.16 sec_encrypt_pin_proc 116
7.2.17 sec_set_pin_mode 116
7.2.18 sec_save_rsa_pri_key 117
7.2.19 sec_save_rsa_puk_key 118
7.2.20 sec_rsa_block 118
7.2.21 sec_get_hw_ver 119
7.2.22 sec_get_fw_ver 120
7.2.23 dukpt_get_ksn 120

8 Gui （libapi_gui） 121

8.1 interface list 121
8.2 API interface 123
8.2.1 gui_bar_rc 123
8.2.2 gui_set_bar_color 123
8.2.3 gui_get_bar_color 124
8.2.4 gui_set_font 125
8.2.5 gui_get_font 125

8.2.6 gui_set_text_color 126
8.2.7 gui_get_text_color 127
8.2.8 gui_set_text_bg_color 127
8.2.9 gui_get_text_bg_color 128
8.2.10 gui_set_color 129
8.2.11 gui_get_color 129
8.2.12 gui_set_bg_color 130
8.2.13 gui_get_bg_color 130
8.2.14 gui_clear_dc 131
8.2.15 gui_set_pixel 131
8.2.16 gui_pixel 132
8.2.17 gui_get_pixel 133
8.2.18 gui_text_out 133
8.2.19 gui_text_out_ex 134
8.2.20 gui_get_text_width 135
8.2.21 gui_get_text_height 135
8.2.22 gui_cline 136
8.2.23 gui_line_to 137
8.2.24 gui_get_width 137
8.2.25 gui_get_height 138
8.2.26 gui_page_op_paint 139
8.2.27 gui_ime_set_mode 139
8.2.28 gui_ime_start_input 140
8.2.29 gui_main_menu_func_add 141
8.2.30 gui_main_menu_item_add 142
8.2.31 gui_main_menu_show 142
8.2.32 gui_post_message 143
8.2.33 gui_proc_default_msg 144
8.2.34 gui_messagebox_show 144

8.2.35 gui_load_bmp 145

8.2.36 gui_out_bits 146

8.2.37 gui_out_bits_ex 147

8.2.38 gui_text_width_ex 147

8.2.39 gui_settextstyle 148

8.2.40 gui_begin_batch_paint 149

8.2.41 gui_end_batch_paint 149

8.2.42 gui_set_full_screen 150

8.2.43 gui_bmp_free 150

8.2.44 gui_out_bits_zoom 151

8.2.45 gui_select_page_ex 151

8.2.46 gui_titlecolorback 152

8.2.47 gui_titlecolorfore 152

8.2.48 gui_menuhightlinecolor 153

8.2.49 gui_textout_line_center 153

8.2.50 gui_clear_rect 154

9 EMV(libapi_emv) 154

9.1 interface list 154

9.2 API interface 155

9.2.1 emv_read_card 155

9.2.2 EMV_iKernelInit 156

9.2.3 EMV_TermConfigInit 156

9.2.4 EMV_GetKernelVersion 157

9.2.5 EMV_GetKernelData 157

9.2.6 EMV_PrmSetAIDPrm 158

9.2.7 EMV_PrmGetAIDPrm 159

9.2.8 EMV_PrmDelAIDPrm 159

9.2.9 EMV_PrmClearAIDPrmFile 160

9.2.10 EMV_PrmSetCAPK 160

9.2.11 EMV_PrmGetCAPK *161*
9.2.12 EMV_PrmDelCAPK *161*
9.2.13 EMV_PrmClearCAPKFile *162*
9.2.14 EMV_GetDataByTag *163*
9.2.15 EMV_PackTLVData *163*
9.2.16 EMV_GetVersion *164*
9.2.17 EMV_SetReadingCardDisp *164*
9.2.18 EMV_SetInputPin *165*
9.2.19 EMV_SetDispOffPin *165*
9.2.20 EMV_ShowAID_Prm *166*
9.2.21 EMV_ShowCAPK_Prm *166*

10 Print (libapi_print) 167

10.1 interface list 167
10.2 API interface 167
10.2.1 UPrint_GetModuleVer *167*
10.2.2 UPrint_Init *168*
10.2.3 UPrint_Str *169*
10.2.4 UPrint_BitMap *169*
10.2.5 UPrint_Start *170*
10.2.6 UPrint_StrBold *171*
10.2.7 UPrint_Feed *172*
10.2.8 UPrint_MatrixCode *172*
10.2.9 UPrint_SetFont *173*
10.2.10 UPrint_SetDensity *174*

11 EMV_API(lib_emvapi) 174

11.1 interface list 174
11.2 API interface 175
11.2.1 emv_online_resp_proc *175*

11.2.2 EMV_online_cardemv_free 176
11.2.3 emv_onlineresp_proc_pack 176
11.2.4 emv_card_begin 177
11.2.5 emv_card_loop 177
11.2.6 emv_card_end 178
11.2.7 Emvapi_Version 178
11.2.8 emvapi_onlinpin_proc_page 179
11.2.9 EMV_SetRuPayServiceList 180
11.2.10 EMV_GetRuPayServiceList 180
11.2.11 EMV_SetRuPayPRMacqKeyList 181
11.2.12 EMV_GetRuPayPRMacqKeyList 181
11.2.13 EMV_ShowRuPayPRMacqKey 182
11.2.14 EMV_ShowRuPayService 182
11.2.15 EMV_ClearRuPayServiceFile 183
11.2.16 EMV_ClearRuPayPRMacqKeyFile 183

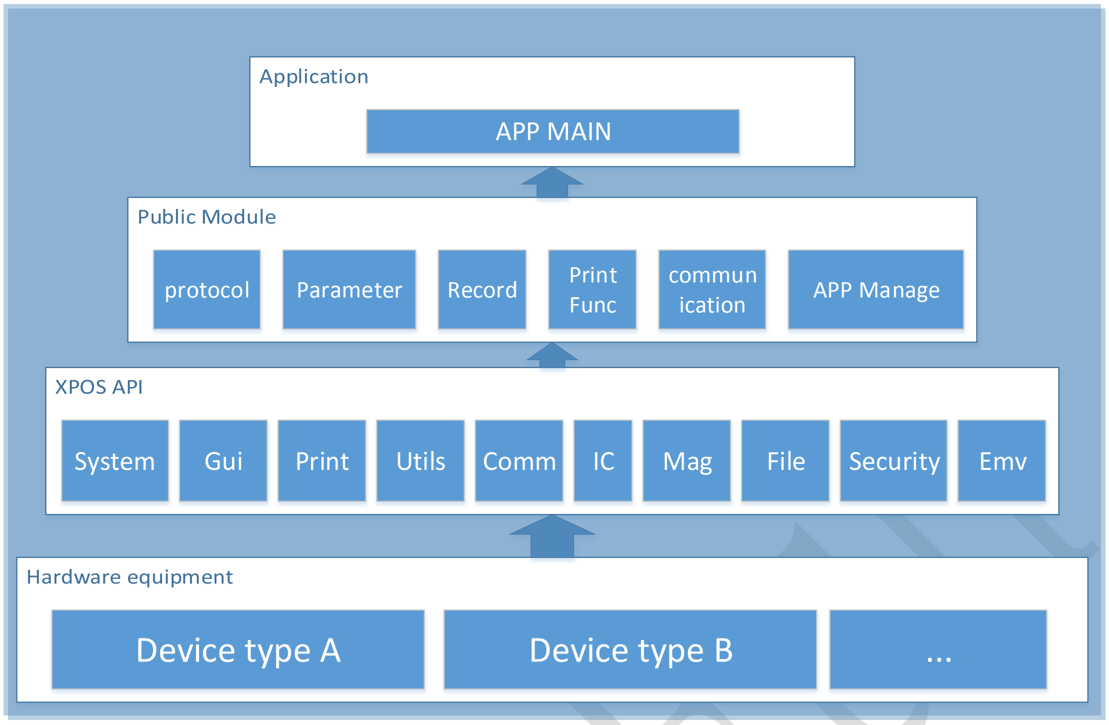
1 profile

1.1 overall structure

This document provides a comprehensive introduction to the application development interface to assist application developers in better secondary development.

1.1 modular design

The terminal software is divided into modules to face relatively independent devices or functions, to meet the goal of rapid development of terminal software and frequent update of requirements.



2 System module(libapi_system)

2.1 interface list

Function name	Function prototype	Function function
Syytem get version	Sys_GetModuleVer	Get version number of system class module
System initialization	Sys_Init	System initialization, independent initialization with application layer
manufacturer personality parameter setting	Sys_Config	Manufacturer personality parameter setting, call each vendor personalization function
Get terminal info	Sys_GetTerminalInfo	Get terminal info
Obtain system time	Sys_GetDateTime	Obtain system time

Set system time	Sys_SetDateTime	Set system time
Scanning buttun	Sys_CheckKey	Scan button, non-blocking
Clear button cache	Sys_ClrKey	Clear button cache
oepn timer	Sys_TimerOpen	Turn on the timer and set the timing
test timer	Sys_TimerCheck	Check if the timing time is up
Close timer	Sys_TimerClose	Close timer
delay	Sys_Delay	Delay, block
Display battery power in real time	Sys_GetBatter	Display battery power in real time
terminal sleep	Sys_Sleep	Terminal enter into sleep status
terminal reboot	Sys_Reboot	terminal reboot
Get network license for terminal	Sys_GetNetworkLicense	Get network license for terminal
Set screen backlight	Sys_SetScrBackLight	Set screen backlight
Get terminal fireware info	Sys_GetFirmwareInfo	Get terminal fireware info
Open Scan	Sys_scaner_open	Open Scan
Start scanning	Sys_scaner_start	Start scanning
Stop scanning	Sys_scaner_stop	Stop scanning
Close Scan	Sys_scaner_close	Close Scan
Get term sn	Sys_GetTermSn	Get term sn
Set app ver	Sys_SetAppVer	Set app ver
driverlib_init	Sys_driverlib_init	driverlib_init
get_sublcd_probe	Sys_get_sublcd_probe	get_sublcd_probe
lcd_set_index	Sys_lcd_set_index	lcd_set_index
power_key_set_light	Sys_power_key_set_light	power_key_set_light
lcd_PowerDownTime	Sys_lcd_PowerDownTime	lcd_PowerDownTime

lcd_SetPowerDownTime	Sys_lcd_SetPowerDownTime	lcd_SetPowerDownTime
lcd_BackLightTime	Sys_lcd_BackLightTime	lcd_BackLightTime
lcd_SetBackLightTime	Sys_lcd_SetBackLightTime	lcd_SetBackLightTime
GetAppVer	Sys_GetAppVer	GetAppVer
GetDeviceType	Sys_GetDeviceType	GetDeviceType

2.2 API interface

2.2.1 Sys_GetModuleVer

Function prototype	int Sys_GetModuleVer(char *pszVer);	
Function function	Get version number of system class module	
Parameter description	In parameter	none
	Out parameter	pszVer
Return value	0-succ -1 fail	
Supplementary explanation		

2.2.2 System initialization(Sys_Init)

Function	int Sys_Init(int Argc， char **Argv， char *AppName);

prototype		
Function function	System initialization, independent initialization with application layer	
Parameter Description	In parameter	Argc：Reuse the main function parameter Argc Argv：Reuse the main function parameter Argv AppName：application name
	Out parameter	none
Return value	USYS_FAIL = -1, // failure USYS_FIRST = 1,//First run after the program is updated USYS_NOFIRST = 2//The program is not the first time running	
Supplementary explanation	no process for not using in parameter Function internal call vendor private API	
	The API only returns to the first run when the program is newly installed. The program update is not the first run.	

2.2.3 Vendor personality parameter setting**(Sys_Config)**

Function prototype	void Sys_Config(void);	
Function function	Manufacturer personality parameter setting, call each vendor personalization function	
Parameter	In parameter	none

description	Out parameter	none
Return value		
Supplementary explanation		

2.2.4 Get terminal info(**Sys_GetTerminalInfo)**

Function prototype	int Sys_GetTerminalInfo(TERMINALINFO *terminal);	
Function function	Get terminal info	
Parameter description	In parameter	none
	Out parameter	terminal referance TERMINALINFO

Return value	USYS_FAIL = -1, // failure
	
	USYS_SUCCES= 0，// success	

Supplementary explanation		

2.2.5 Get system time(**Sys_GetDateTime)**

Function prototype	`int Sys_GetDateTime(byte *DateTime);`

Function function	Get system time	
Parameter description	In parameter	None
	Out parameter	DateTime："YYYYMMDDHHMMSS" 14 byte

Return value
Supplementary explanation
	USYS_FAIL = -1, // failure
	USYS_SUCCES = 0，// success	

Function prototype		

2.2.6 Set systemn time**(Sys_SetDateTime)**

Function prototype	int Sys_SetDateTime(byte *DateTime);	
Function function	Set system time	
Parameter description	In parameter	DateTime："YYYYMMDDHHMMSS" 14 byte
	Out parameter	None

Return value	USYS_PARAERROR = -2,// parameter wrong
	USYS_FAIL = -1, // failure
	USYS_SUCCES= 0，// success	

Supplementary	API internal judge whether time format is correct	

explanation	

2.2.7 Scan button(**Sys_CheckKey)**

Function prototype	`int Sys_CheckKey(void);`	
Function function	Scan button, non-block	
Parameter description	In parameter	none
	Out parameter	none
Return value	Success return key value KEY_VALUE No button return 0 USYS_FAIL = -1, // failure	
Supplementary explanation	None enum KEY_VALUE defined ket, unified return 0	

2.2.8 Clear button cache(**Sys_ClrKey)**

Function prototype	`void Sys_ClrKey(void);`
Function function	Clear button cache

Parameter description	In parameter	none
	Out parameter	none
Return value		
Supplementary explanation		

2.2.9 Open timer**(Sys_TimerOpen)**

Function prototype	int Sys_TimerOpen(uint TimerMs);	
Function function	turn on timer, set timer timing	
Parameter description	In parameter	TimerMs： in millisecond
	Out parameter	none
Return value	success return timer handle USYS_FAIL = -1, // failure	
Supplementary explanation		

2.2.10 Test timer **(Sys_TimerCheck)**

Function	int Sys_TimerCheck(int iHandle);

prototype		
Function function	test whether timer time is up	
Parameter description	In parameter	iHandle：timer handle
	Out parameter	none
Return value	Successful return, remaining time, in milliseconds (0 means the time is up) USYS_FAIL = -1, // failure	
Supplementary explanation		

2.2.11 Turn off timer**(Sys_TimerClose)**

Function prototype	`int Sys_TimerClose(int iHandle);`	
Function function	turn off timer	
Parameter description	In parameter	iHandle：timer handle
	Out parameter	none

Return value	USYS_FAIL = -1, // failure
	USYS_SUCCESS = 0 // success	

Supplementary		

explanation	

2.2.12 delay**(Sys_Delay)**

Function prototype	`void Sys_Delay(uint uiMs);`	
Function function	Postpone, block	
Parameter description	In parameter	uiMs: delay time in ms
	Out parameter	none
Return value		
Supplementary explanation		

2.2.13 Terminal sleep **(Sys_Sleep)**

Function prototype	`int Sys_Sleep(uint Time);`	
Function function	terminal enter into sleep status	
Parameter description	In parameter	Time: Enter sleep time, in seconds (unsupported manufacturers, considered invalid)
	Out parameter	none

Return value	USYS_FAIL = -1, // failure
	USYS_SUCCESS = 0 // success

Supplementary explanation	The application needs to detect the wireless network registration status after waking up.

2.2.14 Terminal reboot**(Sys_Reboot)**

Function prototype	int Sys_Reboot(void);	
Function function	Terminal reboot	
Parameter description	In parameter	none
	Out parameter	none

Return value	USYS_FAIL = -1, // failure
	USYS_SUCCESS = 0 // success	

Supplementary explanation	Each vendor implements according to its own OS and for unsupport, then direct return. (Considering unsupported vendors, the application layer needs to prompt a forced restart after calling the API)	

2.2.15 Open Scan **(Sys_scaner_open)**

Function	int Sys_scaner_open()

prototype		
Function function	Open Scan	
Parameter description	In parameter	none
	Out parameter	none
Return value	USYS_SUCCESS = 0 Success	
Supplementary explanation		

2.2.16 Start scanning **(Sys_scaner_start)**

Function prototype	int Sys_scaner_start()	
Function function	Start scanning	
Parameter description	In parameter	none
	Out parameter	none
Return value	USYS_SUCCESS = 0 Success	
Supplementary explanation		

2.2.17 Stop scanning **(Sys_scaner_stop)**

Function prototype	int Sys_scaner_stop()	
Function function	stop scanning	
Parameter description	In parameter	none
	Out parameter	none
Return value	USYS_SUCCESS = 0 Success	
Supplementary explanation		

2.2.18 Close Scan **(Sys_scaner_close)**

Function prototype	int Sys_scaner_close ()	
Function function	Close Scan	
Parameter description	In parameter	none
	Out parameter	none
Return value	USYS_SUCCESS = 0 Success	

Supplementary explanation	

2.2.19 Sys_GetTermSn

Function prototype	int Sys_GetTermSn(char *Sn);	
Function function	Get terminal serial number	
Parameter description	In parameter	none
	Out parameter	sn
Return value	USYS_SUCCESS = 0 Success	
Supplementary explanation		

2.2.20 Sys_SetAppVer

Function prototype	int Sys_SetAppVer(char *pszVer);	
Function function	Setting Application Version Number	
Parameter description	In parameter	pszVer
	Out parameter	none

Return value	USYS_SUCCESS = 0 Success	
Supplementary explanation		

2.2.21 Sys_driverlib_init

Function prototype	void Sys_driverlib_init();	
Function function	Initialization driver	
Parameter description	In parameter	none
	Out parameter	none
Return value	USYS_SUCCESS = 0 Success	
Supplementary explanation		

2.2.22 Sys_get_sublcd_probe

Function prototype	int Sys_get_sublcd_probe();	
Function function	Judging whether there is a secondary liquid crystal	
Parameter	In parameter	none

description	Out parameter	none
Return value	Return 1 has a secondary liquid crystal	
Supplementary explanation		

2.2.23 **Sys_lcd_set_index**

Function prototype	void Sys_lcd_set_index(int index);	
Function function	Switching liquid crystal	
Parameter description	In parameter	index=0 Main liquid crystal ,index =1 Paraliquid crystal
	Out parameter	none
Return value	Return 1 has a secondary liquid crystal	
Supplementary explanation		

2.2.24 **Sys_power_key_set_light**

Function prototype	void Sys_power_key_set_light();

Function function	Brighten the backlight by pressing the key	
Parameter description	In parameter	none
	Out parameter	none
Return value		
Supplementary explanation		

2.2.25 Sys_lcd_PowerDownTime

Function prototype	int Sys_lcd_PowerDownTime();	
Function function	Get the shutdown time	
Parameter description	In parameter	none
	Out parameter	none
Return value	Return shutdown time	

2.2.26 Sys_lcd_SetPowerDownTime

Function prototype	void Sys_lcd_SetPowerDownTime(int ntime);

Function function	Set the shutdown time	
Parameter description	In parameter	ntime shutdown time
	Out parameter	none
Return value		

2.2.27 Sys_lcd_BackLightTime

Function prototype	int Sys_lcd_BackLightTime();	
Function function	get Backlight time	
Parameter description	In parameter	
	Out parameter	none
Return value	Return Backlight time	

2.2.28 Sys_lcd_SetBackLightTime

Function prototype	void Sys_lcd_SetBackLightTime(int ntime);
Function function	Set Backlight time

Parameter description	In parameter	ntime Backlight time
	Out parameter	none
Return value		

2.2.29 Sys_GetAppVer

Function prototype	const char * Sys_GetAppVer();	
Function function	Get app version	
Parameter description	In parameter	
	Out parameter	none
Return value	the app version	

3 Tool module (**libapi_util)**

3.1 interface list

Function name	Function prototype	Function function
GetModuleVer	Util_GetModuleVer	GetModuleVer
ASCII code change to BCD code	Util_Asc2Bcd	ASCII code change to BCD code

BCD code change to ASCII code	Util_Bcd2Asc	BCD code change to ASCII code
Int type data change to BCD code	Util_Int2Bcd	Int type data change to BCD code
BCD code change to int type	Util_Bcd2Int	BCD code data change to int type
Caculate LRC	Util_GenLrc	Calculate and generate LRC check digits (bitwise XOR)
DES encryption and decryption	Util_Des	DES encryption and decryption of data or 3DES encryption and decryption
Chinese character copy	int Util_StrCopy	Chinese intelligent truncation function, solves the problem of displaying half a Chinese character in a line of Chinese
waiting key	Util_WaitKey	Wait for the button within the set time, wait for the timeout without the button
input method	Util_InputMethod	Support data input for input method switching
string input	Util_InputText	Number, letter, password type in
amount input	Util_InputAmount	Input amount
IP input	Util_InputIp	Input IP address
beep	Util_Beep	beep, non-block
Voice play	Play_Voice	voice play, non-block
Production random number	Util_Rand	generate random number
GeneCodePic	Util_GeneCodePic	GeneCodePic
LED light control	Util_Led	LED light control
equest memory	Util_Malloc	equest memory
Release memory	Util_Free	Release memory

3.2 API interface

3.2.1 Util_GetModuleVer

Function prototype	int Util_GetModuleVer(char *pszVer);	
Function function	Get util module version(Util_GetModuleVer)	
Parameter description	In paramter	
	Out parameter	pszVer:module version
Return value	0-succ -1 fail	

3.2.2 ASCII code change to BCD code (Util_Asc2Bcd)

Function prototype	int Util_Asc2Bcd(char *AscBuf, char *BcdBuf, int AscLen)	
Function function	ASCII code change to BCD code	
Parameter description	In parameter	AscBuf: ASCII code data to be converted AscLen: Importing ASCII code data length
	Out	BcdBuf: Convert output BCD code data

	parameter	

Return value	UUTIL_FAIL = -1, // failure
	
	UUTIL_SUCCESS = 0 // success	

Supplementary explanation	'F' Left on the BCD code, after the number of digits is insufficient, make up 'F'	

3.2.3 BCD code convert to ASCII code (Util_Bcd2Asc)

Function prototype	int Util_Bcd2Asc(char *BcdBuf, char *AscBuf, int AscLen)	
Function function	BCD code convert to ASCII code	
Parameter description	In parameter	BcdBuf: BCD code data that need to be converted AscLen: ASCII code data length, which is double the length of BCD code data
	Out parameter	AscBuf: Convert output ASCII code data

Return value	UUTIL_FAIL = -1, // failure
	
	UUTIL_SUCCESS = 0 // success	

Supplementary explanation		

3.2.4 Int type data convert to BCD code **(Util_Int2Bcd)**

Function prototype	int Util_Int2Bcd(uint IntData, char *BcdBuf, int BcdLen)	
Function function	Int type data convert to BCD code	
Parameter description	In parameter	IntData: Int data to be converted BcdLen：BCD code data length after conversion
	Out parameter	BcdBuf：BCD data after conversion

Return value	UUTIL_FAIL = -1, // failure
	
	UUTIL_SUCCESS = 0 // success	

Supplementary explanation	Right by BCD code, the number of digits is less then add 0 in the left side	

3.2.5 BCD code convert to int type**(Util_Bcd2Int)**

Function	int Util_Bcd2Int(char *BcdBuf, uint *IntData, int BcdLen)

prototype		
Function function	BCD code data convert to int type	
Parameter description	In parameter	BcdBuf:BCD data to be converted BcdLen: BCD code data length
	Out parameter	IntData: int type data after conversion

Return value	UUTIL_FAIL = -1, // failure
	
	UUTIL_SUCCESS = 0 // success	

Supplementary explanation		

3.2.6 **Caculate LRC(Util_GenLrc)**

Function prototype	Byte Util_GenLrc(char *Data, int DataLen)	
Function function	Calculate and generate LRC check digits (bitwise XOR)	
Parameter description	In parameter	Data: Data of the LRC check digit to be calculated DataLen: data length

	Out parameter	
Return value	Calculate the generated LRC check value	
Supplementary explanation		

3.2.7 DES encryption and decryption (Util_Des)

Function prototype	intUtil_Des(bytebDesType,char*Key,char*InData,char*OutData)	
Function function	DES encryption and decryption of data or 3DES encryption and decryption	
Parameter description	In parameter	bDesType：DES encryption and decryption algorithm: 0 means DES encryption, 1 means DES decryption, 2 means 3DES encryption, 3 means 3DES decryption Key：The transport key used for encryption and decryption must be a multiple of 8. InData: The ciphertext data to be encrypted and decrypted must be 8 bytes.
	Out parameter	OutData：The encrypted and decrypted key must be 8 bytes.

Return value	UUTIL_FAIL = -1, // failure
	
	UUTIL_SUCCESS = 0 // success

Supplementary explanation	

3.2.8 Chinese character copy(Util_StrCopy)

Function prototype	int Util_StrCopy(char *dst, cchar *src, int len)	
Function function	Chinese intelligent truncation function, solves the problem of displaying half a Chinese character in a line of Chinese	
Parameter description	In parameter	scr: Source data string len: Source data length
	Out parameter	Dst: target data string
Return value	Returns the length of the copied string	
Supplementary explanation		

3.2.9 Waiting button(Util_WaitKey)

Function prototype	int Util_WaitKey(int TimeOut)	
Function function	Wait for the button within the set time, without button then waiting timeout	
Parameter description	In parameter	TimeOut: Waiting timeout (seconds), 0 means blocking
	Out parameter	
Return value	UUTIL_TIMEOUT Or return button value	
Supplementary explanation		

3.2.10 Input method input(Util_InputMethod)

Function prototype	int Util_InputMethod(int disp_line, char * msgPrompt, int input_line, char *str, int min, int max, byte disp_pattern, int timeout)	
Function function	data input that support input method switching	
Parameter description	In paramater	disp_line: Prompt message shows the number of lines msgPrompt: Prompt message (left alignment) input_line： Input data display line number

		min: Minimum input length max:Maximum input length disp_pattern: Input data display position, 0 left aligned; 1 centered; 2 right alignment timeout: Waiting for input timeout (seconds)
	Out parameter	str: input data

Return value	Success: return the input data str bytes
	
	UUTIL_TIMEOUT = -3, // input timeout
	
	UUTIL_CANCEL = -2, // input cancel
	
	UUTIL_FAIL = -1, // failure	

Supplementary explanation		

3.2.11 String input (Util_InputText)

Function prototype	int Util_InputText(int disp_line, char＊msgPrompt, int input_line, char *str, int min, int max, int disp_pattern, byte disp_mode , int timeout)	
Function function	number, letter, password input	
Parameter	In	disp_line: Prompt message shows the number of

description	paramater	lines msgPrompt: Prompt message input_line: Input data display line number min: Minimum input length max: Maximum input length disp_pattern: Display position, 0 left aligned; 1 centered; 2, right aligned disp_mode: Input mode, 0 digital input; 1 number, letter input
		password input timeout: Timeout (seconds)
	Out parameter	str: Input data

Return value	Success: return the input data str bytes
	UUTIL_TIMEOUT = -3, // input timeout
	UUTIL_CANCEL = -2, // input cancel
	UUTIL_FAIL = -1, // failure	

Supplementary explanation	When the input mode is numeric or letter input, switching between a certain number, uppercase and lowercase, and lowercase is performed by pressing a button continuously.	

3.2.12 Amount input(Util_InputAmount)

Function prototype	int Util_InputAmount(int disp_line, char * msgPrompt, int input_line, char *amount, byte disp_ pattern，int timeout)		
Function function	Input amount		
Parameter description	In parameter	disp_line:	Prompt message shows the number of lines
		msgPrompt:	Prompt message
		input_line:	Input data display line number
		disp_pattern:	Display mode, 0 left aligned; 1 centered; 2, right aligned
		timeout:	Timeout (second)
	Out parameter	amount：	Output amount
Return value	Success: return the output amount bytes UUTIL_TIMEOUT = -3, // input timeout		

	UUTIL_CANCEL = -2, // input cancel
	
	UUTIL_FAIL = -1, // failure		

Supplementary description	the amount input is accurate to cent (two decimal places are reserved) and stored in the Amount variable in 12-bit ASCII code.		

3.2.13 IP input (Util_InputIp)

Function prototype	int Util_InputIp(int disp_line, char * msgPrompt, int input_line, char *ip, byte disp_pattern, int timeout)	
Function function	Enter IP address	
Parameter description	In parameter	disp_line: Prompt message shows the number of lines msgPrompt: prompt message input_line: Input data display line number disp_pattern: Display position, 0 left aligned; 1 centered; 2, right pair Timeout: timeout (seconds)
	Out parameter	ip: enter IP address

Return value	Success: return input IP address bytes
	UUTIL_TIMEOUT = -3, // input timeout
	UUTIL_CANCEL= -2, // input cancel
	UUTIL_FAIL = -1, // failure	

Supplementary explanation	API internal with IP address format judgment	

3.2.14 beep(Util_Beep)

Function prototype	Void Util_Beep(int num)	
Function function	Buzzer, non-blocking	
Parameter description	In paramter	num: Beep times
	Out parameter	
Return value		
Supplementary explanation		

3.2.15 Voice play (Play_Voice)

Function prototype	void Play_Voice(char *msg)	
Function function	Specified line display	
Parameter description	In parameter	Msg: Information that requires voice play
	Out parameter	None

Return value	None
Supplementary explanation	Voice playback is non-blocking

3.2.16 Generate random numbers(Util_Rand)

Function prototype	`int Util_Rand(byte *psRandom)`	
Function function	Generate random numbers	
Parameter description	In parameter	
	Out parameter	psRandom8byte Binary random number

Return value	UUTIL_FAIL = -1, // failure
	
	UUTIL_SUCCESS = 0 // success	

Supplementary explanation	Priority use true random numbers	

3.2.17 Play_Voice

Function prototype	`void Play_Voice(char *msg);`
Function function	Voice Play

Parameter description	In parameter	msg Audio file
	Out parameter	
Return value		
Supplementary explanation	Voice playback is non blocking.	

3.2.18 Util_GeneCodePic

Function prototype	int Util_GeneCodePic(char * chData, int iLen, Util_QR_INFO *qrparam , char * bitmap);	
Function function	QR code generation	
Parameter description	In parameter	chData:QR code data, iLen: data length Qrparam: QR code parameters
	Out parameter	Bitmap:Generated two-dimensional code dot matrix data
Return value	0-succ -1-fail	

3.2.19 Util_Led

Function prototype	`void Util_Led(int num, int type);`	
Function function	LED light control	
Parameter description	In parameter	num LED light number(0 red, 1 blue, 2 yellow, 3 green) type LED light switch(0 close, 1 open)
	Out parameter	
Return value		

3.2.20 Util_Malloc

Function prototype	`void * Util_Malloc(int size);`	
Function function	Request memory	
Parameter description	In parameter	size memory size
	Out parameter	
Return value	Application Memory Pointer	

3.2.21 Util_Free

Function prototype	`void Util_Free( void * p);`	
Function function	Release memory	
Parameter description	In parameter	A pointer to release memory
	Out parameter	
Return value	Application Memory Pointer	

4 File module(libapi_file)

4.1 Interface list

Function name	Function prototype	function function
GetModuleVer	UFile_GetModuleVer	GetModuleVer
Check if the file exists	UFile_Check	Check if the file exists
File open/create	UFile_OpenCreate	File open/create
File reading	UFile_Read	File reading
File writing	UFile_Write	File writing
Positioning file pointer	UFile_Lseek	Positioning file pointer
Delete file record	UFile_Delete	Delete file record

Close file	UFile_Close	Close file
Delete file	UFile_Remove	Delete file
Rename file	UFile_Rename	Rename file
Empty file	UFile_Clear	Empty file content
Take the number of file records	UFile_GetNumberOfRecords	Take the number of file records
Append file record	UFile_AppendRecord	At the end of the file, add a fixed length record file. Automatically create a file when it does not exist
check records based on index number	UFile_GetRecordByIndex	Find any record by record index
Check record	UFile_GetRecord	Find any record by condition
Update record	UFile_UpdateRecord	Update any record by condition
Update records based on index number	UFile_UpdateRecordByIndex	Update any records based on index number
Delete record	UFile_DeleteRecord	Delete any record by condition
Delete records based on index number	UFile_DeleteRecordByIndex	Delete any record by recording the index number
Read a line of text	UFile_ReadLine	Read a line of text and support \r \n newline
Read non-fixed length records	UFile_ReadTLV	Read non-fixed length record TLV, consistent with IC card TVL format
Write non-fixed data	UFile_WriteTLV	Write non-fixed data TLV
Delete non-fixed record	UFile_DeleteTLV	Delete non-fixed record TLV

4.2 API interface

4.2.1 UFile_GetModuleVer

Function prototype	int UFile_GetModuleVer(char *pszVer);	
Function function	Get File module version (UFile_GetModuleVer)	
Parameter description	In parameter	pszVer module version
	Out parameter	None
Return value	0-succ -1fail	

4.2.2 Check if the file exists (UFile_Check)

Function prototype	int UFile_Check(cchar *FileName, int iFileLocation);
Function function	Check if the file exists

Parameter description	In parameter	FileName: File name, ending with NULL, up to 16 bytes iFileLocation: Storage location, see enum FILELOCATION
	Out parameter	None

Return value	UFILE_NO_EXIST = -12, // The specified file does not exist
	
	UFILE_PARAERROR = -11, // parameter wrong
	
	UFILE_SUCCESS = 0 //File operation succeeded	

Supplementary explanation		

4.2.3 **File open / create(UFile_OpenCreate)**

Function prototype	intUFile_OpenCreate(cchar*FileName, intiFileLocation, intFlag, FILE_HANDLE *fh, int RecSize);
Function function	File open / create

Parameter description	In parameter	FileName：open file name, end with NULL, maximum 16 bytes in length iFileLocation: storage place, see enum FILELOCATION Flag：Open file mode, the value refer to FileFlags define RecSize：File record size RecSize = 0,Create Open Stream File, Text File RecSize = 1，Create open non-fixed length record (TLV) file 8<=RecSize<=4090 Create open fixed length record file
	Out parameter	Fh：File handle

Return value	UFILE_NO_EXIST = -12, //The specified file does not exist
	
	UFILE_PARAERROR = -11， //Parameter error
	
	UFILE_OPEN_FAIL = -2, //Open error
	
	UFILE_FAIL = -1, //File operation failed
	
	UFILE_SUCCESS = 0 //File operation succeeded	

Supplementary explanation	RecSize only works for the newly created file For opening a created file, the RecSize parameter should be ignored.	

4.2.4 File read(UFile_Read)

Function prototype	`int UFile_Read(FILE_HANDLE handle, char *buffer, int size);`	
Function function	File reading	
Parameter description	In parameter	handle：Read file handle size：Read data size
	Out parameter	buffer：read data

Return value	The file was read successfully: the return value is equal to the number of bytes actually read.
	
	UFILE_PARAERROR = -11， // paramter error
	
	UFILE_READ_FAIL = -5, //reading error
	
	UFILE_FAIL = -1, //file operation failed	

Supplementary explanation		

4.2.5 **Write file (UFile_Write)**

Function prototype	`int UFile_Write(FILE_HANDLE handle, char *buffer, int size);`

Function function	File writing	
Parameter description	In parameter	handle：Read file handle size：The size of the data to be written buffer：Data to be written
	Out parameter	None

Return value	File write succeeded: the return value is equal to the number of bytes actually written
	
	UFILE_PARAERROR = -11， // parameter erro
	
	UFILE_WRITE_FAIL = -4, //write error
	
	UFILE_FAIL = -1, //file operation failed	

Supplementary explanation		

4.2.6 Positioning file pointer(UFile_Lseek)

Function prototype	long UFile_Lseek(FILE_HANDLE handle, long offset, int origin);
Function function	Positioning file pointer

Parameter description	In parameter	Handle: file handle, offset: offset, origin: starting position, see FileSeekFlags type
	Out parameter	None

Return value	UFILE_PARAERROR = -11，　　//parameter error
	
	UFILE_SEEK_FAIL = -6,　　//Positioning file pointer error
	
	UFILE_SUCCESS = 0	

Supplementary explanation		

4.2.7 **Delete file record (UFile_Delete)**

Function prototype	`int UFile_Delete(FILE_HANDLE handle, uint size);`	
Function function	delete file record	
Parameter description	In parameter	handle：file handle size：Number of deleted files
	Out parameter	None

Return value	UFILE_PARAERROR = -11, //parameter error
	UFILE_DELETE_FAIL = -7, //Delete file record error
	UFILE_SUCCESS = 0

Supplementary explanation	The specific location of the deletion is determined by the File_Lseek() function.

4.2.8 **Close file (UFile_Close)**

Function prototype	int UFile_Close(FILE_HANDLE handle);	
Function function	Close file	
Parameter description	In parameter	handle: file handle
	Out parameter	None

Return value	UFILE_PARAERROR = -11, // parameter error
	UFILE_CLOSE_FAIL = -8, //Close file error
	UFILE_FAIL = -1, //File operation failed
	UFILE_SUCCESS = 0 //File operation succeeded	

Supplementary explanation		

4.2.9 Delete file (UFile_Remove)

Function prototype	`int UFile_Remove(cchar *filename, int iFileLocation);`	
Function function	Delete file	
Parameter description	In paramater	fileName：File name, ending with NULL, up to 16 bytes iFileLocation: storage location, see enum FILELOCATION
	Out parameter	None

Return value	UFILE_NO_EXIST = -12, //The specified file does not exist
	
	UFILE_PARAERROR = -11， //parameter error
	
	UFILE_FAIL = -1, //File operation failed
	
	UFILE_SUCCESS = 0 //File operation succeed	

Supplementary explanation		

4.2.10 Rename file (UFile_Rename)

Function prototype	int UFile_Rename(cchar *oldname, int iFileLocation, cchar *newname);	
Function function	Rename file	
Parameter description	In parameter	oldname： old file name iFileLocation: storage location, see enum FILELOCATION newname：new file name
	Out parameter	None
Return value	UFILE_NO_EXIST = -12, //The specified file does not exist	

	UFILE_PARAERROR = -11， //parameter error
	UFILE_FAIL = -1, //File operation failed
	UFILE_SUCCESS = 0 //File operation succeed	

Supplementary explanation		

4.2.11 Empty file(UFile_Clear)

Function prototype	int UFile_Clear(cchar *FileName, int iFileLocation);
Function function	Empty file content

Parameter description	In parameter	FileName：File name, ending with NULL, up to 16 bytes iFileLocation: storage location, see enum FILELOCATION
	Out parameter	None

Return value	UFILE_NO_EXIST = -12, //The specified file does not exist
	
	UFILE_PARAERROR = -11， //parameter error
	
	UFILE_FAIL = -1, //File operation failed
	
	UFILE_SUCCESS = 0 //File operation succeed	

Supplementary explanation		

4.2.12 Get the number of file records (UFile_GetNumberOfRecords)

Function prototype	`int UFile_GetNumberOfRecords(cchar *FileName, int iFileLocation, int Record_Len);`
Function function	Get the number of file records

Parameter description	In parameter	FileName： file name iFileLocation: storage location, see enum FILELOCATION Record_Len: Single record length
	Out parameter	None

Return value	Success: Returns the number of records
	failure: UFILE_PARAERROR = -11, //parameter error
	UFILE_FAIL = -1, //file operation failed	

Supplementary explanation		

4.2.13 Append file record(UFile_AppendRecord)

Function prototype	int UFile_AppendRecord(cchar *FileName, int iFileLocation, char *Record, int Record_Len);	
Function function	At the end of the file, add a fixed length record file. When the file does not exist, automatically create the file;	
Parameter description	In parameter	FileName：File name, ending with NULL, up to 16 bytes iFileLocation: storage location，see enum FILELOCATION Record：record data
		Record_Len：record the length of data
	Out parameter	None

Return value	UFILE_PARAERROR = -11， // parameter error
	
	UFILE_FAIL = -1, // file operation failed
	
	UFILE_SUCCESS = 0 //file operation succeed

Supplementary explanation	Power failure protection

4.2.14 Query records based on index number(UFile_GetRecordByIndex)

Function prototype	int UFile_GetRecordByIndex(cchar *FileName, int iFileLocation, void *Record, int Record_Len, uint Record_Index);	
Function function	Find any record by record index	
Parameter description	In parameter	FileName : file name iFileLocation: storage location, see enum FILELOCATION Record_Len: record length Record_Index : Record index (starting at 0)
	Out parameter	Record: record data

Return value	UFILE_NO_EXIST = -12, //The specified file does not exist
	UFILE_PARAERROR = -11, //parameter error
	UFILE_NO_RECORD = -10, //record not found
	UFILE_READ_FAIL = -5, //reading error
	UFILE_OPEN_FAIL = -2, //opening error
	UFILE_FAIL = -1, //File operation failed
	UFILE_SUCCESS = 0 //File operation succeed

Supplementary explanation	

4.2.15 Check record(UFile_GetRecord)

Function prototype	int UFile_GetRecord(cchar *FileName, int iFileLocation, void *Record, int Record_Len, DBSEARCOND *Condtion);	
Function function	Find any record by condition	
Parameter description	In parameter	FileName：file name iFileLocation: storage location，see enum FILELOCATION Record_Len： record length Condtion： query condition, see DBSearCond structure
	Out	Record： record data

	parameter	

Return value	UFILE_NO_EXIST = -12, //The specified file does not exist
	UFILE_PARAERROR = -11, //parameter error
	UFILE_NO_RECORD = -10, //record not found
	UFILE_READ_FAIL = -5, //reading error
	UFILE_OPEN_FAIL = -2, //opening error
	UFILE_FAIL = -1, //File operation failed
	UFILE_SUCCESS = 0 //File operation succeed	

Supplementary explanation		

4.2.16 Update record(UFile_UpdateRecord)

Function prototype	int UFile_UpdateRecord(cchar *FileName, int iFileLocation, void *Record, int Record_Len, DBSEARCOND *Condtion);	
Function function	Update any record by condition	
Parameter description	In parameter	FileName: file name
		iFileLocation: storage location, see enum FILELOCATION Record: record data

		Record_Len：record length Condtion：query condition, see DBSearCond structure
	Out parameter	Record

Return value	UFILE_NO_EXIST = -12, //The specified file does not exist
	
	UFILE_PARAERROR = -11， //parameter error
	
	UFILE_NO_RECORD = -10, //record not founded UFILE_READ_FAIL = -5, // read error
	
	UFILE_WRITE_FAIL = -4, //write error
	
	UFILE_OPEN_FAIL = -2, // opening error
	
	UFILE_FAIL = -1, //File operation failed
	
	UFILE_SUCCESS = 0 //File operation succeed	

Supplementary explanation	Power failure protection Record is both in parameter and out parameter In the case of a successful search, the Record is populated by the search results.	

4.2.17 Update records based on index number(UFile_UpdateRecordByIndex)

Function prototype	intUFile_UpdateRecordByIndex(cchar*FileName,intiFileLocation, void *Record, int Record_Len, uint Index);	
Function function	Update any record by index number	
Parameter description	In parameter	FileName：file name iFileLocation: storage location，see enum FILELOCATION Record： record data
		Record_Len： record length Index： Record index number
	Out parameter	Record

Return value	UFILE_NO_EXIST = -12, //The specified file does not exist
	UFILE_PARAERROR = -11， //parameter error
	UFILE_NO_RECORD = -10, //record not founded UFILE_READ_FAIL = -5, // reading error
	UFILE_WRITE_FAIL = -4, // writing error
	UFILE_OPEN_FAIL = -2, // opening error
	UFILE_FAIL = -1, //File operation failed
	UFILE_SUCCESS = 0 //File operation succeed	

Supplementary explanation	Power failure protection Record is both in parameter and out parameter In the case of a successful search, the Record is populated by the search results.	

4.2.18 Delete record (UFile_DeleteRecord)

Function prototype	int UFile_DeleteRecord(cchar *FileName, int iFileLocation, int Record_Len, DBSEARCOND *Condtion);		
Function function	Delete any record by condition		
Parameter description	In parameter	FileName：file name iFileLocation: storage location, see enum FILELOCATION Record_Len：record length Condtion：query condition，see DBSearCond structure	
	Out parameter		
Return value	UFILE_NO_EXI ST	= -12,	//The specified file does not exist
	UFILE_PARAE RR	)R = -11，	//parameter error
	UFILE_NO_RE CO	RD = -10,	//record not founded
	UFILE_DELET E_	FAIL = -7,	//Delete file record error
	UFILE_OPEN_ FA	I L = -2,	//opening error
	UFILE_FAIL	= -1,	//File operation failed
	UFILE_SUCCESS	= 0	//File operation succeed

Supplementary explanation	Power failure protection

4.2.19 Delete records based on index number(UFile_DeleteRecordByIndex)

Function prototype	intUFile_DeleteRecordByIndex(cchar*FileName, intiFileLocation, int Record_Len, uint Index);	
Function function	Delete any record by recording the index number	
Parameter description	In parameter	FileName：file name iFileLocation: storage location，see enum FILELOCATION Record_Len：record length Index： record index number
	Out parameter	

Return value	UFILE_NO_EXIST = -12, //specified file not existed
	UFILE_PARAERROR = -11, //parameter error
	UFILE_NO_RECORD = -10, //record not founded
	UFILE_DELETE_FAIL = -7, //Delete file record error	

	UFILE_OPEN_FAIL = -2, //opening error
	UFILE_FAIL = -1, //File operation failed
	UFILE_SUCCESS = 0 //File operation succeed	

Supplementary explanation	Power failure protection

4.2.20 Read one line text(UFile_ReadLine)

Function prototype	int UFile_ReadLine(FILE_HANDLE pFile, char *pLineBuff,uint LineBuffSize);	
Function function	Read a line of text, and support \r \n newline (data read out should not contain newline)	
Parameter description	In parameter	pFile：file handle LineBuffSize：Buffer size
	Out parameter	pLineBuff ：Read text data

Return value	Success: data length
	
	UFILE_PARAERROR = -11, // parameter error
	
	UFILE_READ_FAIL = -5, // reading error
	
	UFILE_FAIL = -1, //file operation failed	

Parameter description	For a text file, read a row of data from the current location and jump to the next row.	

4.2.21 Read non-fixed length records (UFile_ReadTLV)

Function prototype	int UFile_ReadTLV(char *FileName, int iFileLocation, uint FldID, char *Data, uint *DataLen);

Function function	Read non-fixed length record TLV, consistent with IC card TVL format	
Parameter description	In parameter	FileName: file name iFileLocation: storage location, see enum FILELOCATION
		FldID: tag (Tag)
	Out parameter	Data: data (Value) DataLen: length (length)

Return value	UFILE_NO_EXIST = -12, //The specified file does not exist
	UFILE_PARAERROR = -11， // parameter error
	UFILE_NO_RECORD = -10, //record not founded
	UFILE_READ_FAIL = -5, // reading error
	UFILE_OPEN_FAIL = -2, //opening error
	UFILE_FAIL = -1, //File operation failed
	UFILE_SUCCESS = 0 //File operation succeed	

Parameter description	Read the record in TLV format	

4.2.22 Write non-fixed data(UFile_WriteTLV)

Function prototype	int UFile_WriteTLV(char *FileName, int iFileLocation, uint FldID, char *Data, uint DataLen);
Function function	Write non-fixed length record TLV

Parameter description	In parameter	FileName: file name iFileLocation: storage location, see enum FILELOCATION FldID: tag (Tag) Data: data (Value) DataLen: length (length)
	Out parameter	none

Return value	UFILE_NO_EXIST = -12, //specified file does not existed
	UFILE_PARAERROR = -11, //parameter error	

	UFILE_WRITE_FAIL = -4, //writing error
	UFILE_OPEN_FAIL = -2, //opening error
	UFILE_FAIL = -1, //File operation failed
	UFILE_SUCCESS = 0 //File operation succeed	

Supplementary description		

4.2.23 Delete non-fixed length record (UFile_DeleteTLV)

Function prototype	`int UFile_DeleteTLV(char *FileName, int iFileLocation, uint FldID);`

Function function	Delete non-fixed length record TLV	
Parameter description	In parameter	FileName : file name iFileLocation: storage location, see enum FILELOCATION FldID: tag(Tag)
	Out parameter	None

Return value	UFILE_NO_EXIST = -12, //specified file does not exist
	UFILE_PARAERROR = -11, //parameter error
	UFILE_NO_RECORD = -10, //record not founded
	UFILE_DELETE_FAIL = -7, //Delete file record error
	UFILE_OPEN_FAIL = -2, //opening error
	UFILE_FAIL = -1, //File operation failed
	UFILE_SUCCESS = 0 //File operation succeed	

Supplementary description		

5 IC card module (libapi_iccard)

5.1 interface list

function name	function prototype	Function function
GetModuleVer	Icc_GetModuleVer	GetModuleVer
Turn on IC card device	Icc_Open	Turn on IC card device
Turn off IC card device	Icc_Close	Turn off IC card device
Check the card	Icc_GetCardStatus	Contact card: Check if the card is in the card slot
Contact card powering	Icc_PowerUp	Powering on contact IC card: setting IC card type, card slot category
Contact card power off	Icc_PowerDown	Contact card power off
Contact card communication	Icc_ICComm	Contact IC card communication function
NFC card card search	Icc_CTLSPowerUpAndSeek	NFC card reader search card
NFC card power off	Icc_CTLSPowerDown	NFC card power off
NFC card communication	Icc_CTLSComm	Use APDU to communicate with NFC card
RF Seek	Icc_CTLSPowerUpAndSeek	RF Seek
GET CARD ATR	Icc_GetCardATR	GET CARD ATR

5.2 API interface

5.2.1 Icc_GetModuleVer

Function prototype	int Icc_GetModuleVer(char *pszVer);

Function function	Get File module version (Icc_GetModuleVer)	
Parameter description	In paramate	pszVer Get File module version
	Out parameter	None

Return value	UICC_FAIL = -1,// operation failed
	
	UICC_OK = 0// operation succeed	

5.2.2 **Turn on IC card device（Icc_Open)**

Function prototype	int Icc_Open(int iSlotType);	
Function function	Turn on IC card device	
Parameter description	In paramate	iSlotType：card slot number，see enum SlotType
	Out parameter	None

Return value	UICC_FAIL = -1,// operation failed
	
	UICC_OK = 0// operation succeed	

Supplementary description	

5.2.3 Turn off IC card device (Icc_Close)

Function prototype	int Icc_Close(int iSlotType);	
Function function	Turn off IC card device	
Parameter description	In parameter	iSlotType： card slot number， see enum SlotType
	Out parameter	None

Return value	UICC_FAIL = -1, // operation failed
	UICC_OK = 0 // operation succeed	

Supplementary description		

5.2.4 Turn off IC card device (Icc_Close)

Function prototype	int Icc_CTLSComm(int iCardType,int iSlotType ， ICCAPDU *Apdu);

Function function	Use APDU to communicate with NFC card	
Parameter description	In parameter	iCardType：NFC card type, see enum IccType iSlotType：card slot, see enum SlotType Apdu：refer to ICCAPDU Structure description The various types of card operations are based on the type of OperType operation in the ICCAPDU structure. The data that needs to be passed in during various card operations and the way it is stored in the Apdu structure are discussed separately.
	Out parameter	Apdu：refer to ICCAPDU structure description The returned data is based on the type of OperType operation in the ICCAPDU structure, placed in R_Data

Return value	UICC_COMMAND_FAIL = -2,// Communication error with card
	
	UICC_FAIL = -1, // operation failed
	
	UICC_OK = 0 // operation succeed	

Supplementary description		

5.2.5 **Test card(Icc_GetCardStatus)**

Function prototype	int Icc_GetCardStatus(int iSlotType);	
Function function	Contact card: Check if the card is in the card slot	
Parameter description	In parameter	iSlotType：card slot number，refer to enum SlotType
	Out parameter	
Return value	UICC_EMPTY = -3,// no card in card slot UICC_FAIL = -1,// operation failed	
	UICC_OK = 0// operation succeed	
Supplementary description	Please call first to open the IC card device (Icc_Open)	

5.2.6 **Contact card power on(Icc_PowerUp)**

Function prototype	int Icc_PowerUp(int iCardType, int iSlotType);	
Function function	Powering on the contact IC card: Set the IC card type and card slot category.	
Parameter description	In parameter	iCardType：IC card type，see enum IccType iSlotType： card slot type， refer to enum SlotType
	Out parameter	None

Return value	UICC_EMPTY = -3,// no card in card slot
	
	UICC_FAIL = -1,// operation failure
	
	UICC_OK = 0// operation succeed	

Supplementary description	Contains the card reset operation, and subsequently obtains the card reset information through Icc_GetCardATR	

5.2.7 **Contact card power off (Icc_PowerDown)**

Function prototype	int Icc_PowerDown(int iCardType , int iSlotType);
Function function	contact card power off

Parameter description	In papameter	iCardType：IC card type，see enum IccType iSlotType： card slot type， see enum SlotType
	Out parameter	None

Return value	UICC_FAIL = -1,// operation failure
	
	UICC_OK = 0// operation succeed	

Supplementary description	Pay attention to call after power off. Close the IC card device (Icc_Close)	

5.2.8 Contact card communication (Icc_ICComm)

Function prototype	int Icc_ICComm (int iCardType,int iSlotType, ICCAPDU *Apdu);
Function function	Contact IC card communication function

Parameter description	In parameter	iCardType：IC card type，see enum IccType iSlotType： card slot type， see enum SlotType Apdu：refer to ICCAPDU structure The various types of card operations are based on the type of OperType operation in the ICCAPDU structure.
		The data that needs to be passed in during various card operations and the way it is stored in the Apdu structure are discussed separately.
	Out parameter	Apdu：refer to ICCAPDU structure The returned data is based on the type of OperType operation in the ICCAPDU structure, placed in R_Data

Return value	UICC_COMMAND_FAIL= -2,// Communication error with card
	
	UICC_FAIL = -1,// operation failure
	
	UICC_OK = 0// operation succeed	

Supplementary description	None	

5.2.9 NFC card searching card (Icc_CTLSPowerUpAndSeek)

Function prototype	int Icc_CTLSPowerUpAndSeek (int iCardType, int iSlotType, char *psUID);	
Function function	NFC card reader searching card	
Parameter description	In parameter	iCardType： NFC card type, see enum IccType iSlotType：card slot, see enum SlotType
	Out parameter	psUID：Card serial number, the first byte is the serial number length

Return value	UICC_NORF = -4,// no NFC card
	
	UICC_FAIL = -1,// operation failure
	
	UICC_OK = 0// operation succeed	

Supplementary description	Please call first to open the IC card device (Icc_Open) Contains card reset operation application layer loop call Get card reset information via Icc_GetCardATR	

5.2.10 NFC card power off(Icc_CTLSPowerDown)

Function prototype	int Icc_CTLSPowerDown (int iSlotType);	
Function function	NFC card power off	
Parameter description	In parameter	iSlotType：card slot number，see enum SlotType
	Out parameter	None

Return value	UICC_FAIL = -1,// operation failure
	
	UICC_OK = 0// operation succeed	

Supplementary description	Pay attention to call after power off. Close the IC card device (Icc_Close)	

5.2.11 Use APDU to communicate with NFC card(Icc_CTLSComm)

Function prototype	int Icc_CTLSComm(int iCardType,int iSlotType ，ICCAPDU *Apdu);

Function function	use APDU to communicate with NFC card	
Parameter description	In parameter	iCardType：NFC card type，see enum IccType iSlotType：card slot，see enum SlotType Apdu ： refer to ICCAPDU structure description The various types of card operations are based on the type of OperType operation in the ICCAPDU structure. The data that needs to be passed in during various card operations and the way it is stored in the Apdu structure are discussed separately.
	Out parameter	Apdu ： refer to ICCAPDU structure description The returned data is based on the type of OperType operation in the ICCAPDU structure, placed in R_Data

Return value	UICC_COMMAND_FAIL= -2,// communication error with card
	
	UICC_FAIL = -1,// operation failure
	
	UICC_OK = 0// operation succeed	

Supplementary description		

5.2.12 Icc_CTLSPowerUpAndSeek

Function prototype	int Icc_CTLSPowerUpAndSeek (int iCardType, char *psUID);	
Function function	RF Seek（Icc_CTLSPowerUpAndSeek）	
Parameter description	In paramate	ICardType: non card type, see enum IccType iSlotType：Card slot number, see enum SlotType
	Out parameter	Output : psUID：Card serial number. The first byte is the serial number length.

Return value	UICC_NORF = -4, // No Card
	UICC_FAIL = -1, // Fail
	UICC_OK = 0 // Success	

5.2.13 Icc_GetCardATR

Function prototype	int Icc_GetCardATR(int iCardType, int iSlotType, byte *psATR, int*pnATRLen);	
Function function	Get IC card reset information ATR（Answer To Reset）	
Parameter description	In paramate	iCardType：IC card type, see enum IccType iSlotType：Card slot number, see enum SlotType pnATRLen：psATR Cache size
	Out parameter	

Return value	UICC_FAIL = -1, // Fail
	UICC_OK = 0 // Success

6 communication（ libapi_comm）

6.1 interface list

Function prototype	Function function
comm_net_link	Connect Network
comm_net_link_ex	Tips for connecting to the network
comm_net_unlink	Disconnect from the network
comm_sock_create	create socket
comm_sock_connect	connect to the server
comm_sock_recv	Receive data
comm_sock_send	send data
comm_sock_close	Disconnect the server
comm_ssl_init	ssl initialization
comm_ssl_connect	ssl connect to the server
comm_ssl_connect2	ssl connect to the server
comm_ssl_send	ssl send data
comm_ssl_recv	ssl Receive data
comm_ssl_close	ssl Disconnect
comm_wifi_list_ap	Get the router list
comm_wifi_link_ap	Connecting router
comm_wifi_unlink_ap	unlink router
comm_wifi_get_link_state	Get connection status
comm_wifi_get_signal	get wifi signal
wifi_get_ssid	get wifi signal
wifi_get_ap_mac	get wifi ap mac address
wifi_get_rssi	get wifi rssi

wifi_get_channel	get wifi channel
wifi_get_local_mac	get wifi local mac address
wifi_get_local_ip	get wifi local ip
comm_atc_imei	get Module imei
comm_atc_cpin	get Module sim card status
comm_atc_imsi	get Module imsi
comm_atc_signal	get Module signal
comm_atc_cell	get net registered cell
comm_atc_lac	get net registered lac
comm_atc_iccid	get Module iccid

6.2 API interface

6.2.1 comm_net_link

Function prototype	`int comm_net_link(char * title, char * apn , int timeover);`	
Function function	Connect Network	
Parameter description	In parameter	title：Tips for connecting to the network apn：gprs apn timeover : Connection timeout
	Out parameter	
Return value	0, success Other, failure	

Supplementary description	

6.2.2 comm_net_link_ex

Function prototype	int comm_net_link_ex(char * title, char * apn, int timeover, char *user, char *pwd, int auth);	
Function function	Connect Network	
Parameter description	In parameter	title: Tips for connecting to the network apn: gprs apn timeover : Connection timeout user: gprs apn user id pwd: gprs apn user password auth:Authentication parameter
	Out parameter	
Return value	0, success Other, failure	
Supplementary description		

6.2.3 comm_net_unlink

Function prototype	int comm_net_unlink();

Function function	Disconnect from the network	
Parameter description	In parameter	
	Out parameter	
Return value	0, success Other, failure	
Supplementary description		

6.2.4 comm_sock_create

Function prototype	int comm_sock_create(int index);	
Function function	create socket	
Parameter description	In parameter	index(0/1)
	Out parameter	
Return value	0, success Other, failure	

6.2.5 **comm_sock_connect**

Function prototype	int comm_sock_connect(int index, char * ip, int port);	
Function function	Connect to the server	
Parameter description	In parameter	index sock index ip server ip port server port
	Out parameter	
Return value	0, success Other, failure	
Supplementary description		

6.2.6 **comm_sock_recv**

Function prototype	int comm_sock_recv(int index, unsigned char * buff, int len, unsigned int timeover);	
Function function	Receive data	
Parameter description	In parameter	index sock index buff Receive buffer len Receiving length

		timeover overtime time
	Out parameter	
Return value	0, success Other, failure	
Supplementary description		

6.2.7 **comm_sock_send**

Function prototype	int comm_sock_send(int index, unsigned char * buff , int size);	
Function function	send data	
Parameter description	In parameter	index sock index buff Send buffer len Send length
	Out parameter	
Return value	0, success Other, failure	

Supplementary description	

6.2.8 **comm_sock_close**

Function prototype	int comm_sock_close(int index);		
Function function	Disconnect the server		
Parameter description	In parameter	index	sock index
	Out parameter		
Return value	0, success Other, failure		
Supplementary description			

6.2.9 **comm_ssl_init**

Function prototype	int comm_ssl_init(int index, char * cacert, char * clientcert, char * clientkey,int level);
Function function	ssl initialization

Parameter description	In parameter	index sock index cacert Server certificate clientcert Client certificate clientkey Client key level Verification level 0=Not verified 1=Verify server certificate
	Out parameter	
Return value	0, success Other, failure	
Supplementary description		

6.2.10 **comm_ssl_connect**

Function prototype	int comm_ssl_connect(int index , char * ip , int port);	
Function function	ssl connect to the server	
Parameter description	In parameter	index sock index ip server ip port server port
	Out parameter	

Return value	0, success Other, failure
Supplementary description	

6.2.11 comm_ssl_connect2

Function prototype	int comm_ssl_connect2(int index , char * ip , int port, void *func);	
Function function	ssl connect to the server	
Parameter description	In parameter	index sock index ip server ip port server port func callback - Disconnect by callback
	Out parameter	
Return value	0, success Other, failure	
Supplementary description		

6.2.12 comm_ssl_send

Function prototype	int comm_ssl_send(int index, char * pdata, int size);	
Function function	ssl send data	
Parameter description	In parameter	index sock index data ssl data size Data size
	Out parameter	
Return value	0, success Other, failure	
Supplementary description		

6.2.13 comm_ssl_recv

Function prototype	int comm_ssl_recv(int index, char * pdata, int size);	
Function function	ssl Receive data	
Parameter description	In parameter	index sock index data ssl data size Data size
	Out parameter	

Return value	0, success Other, failure
Supplementary description	

6.2.14 comm_ssl_close

Function prototype	int comm_ssl_close(int index);		
Function function	ssl Disconnect		
Parameter description	In parameter	index	sock index
	Out parameter		
Return value	0, success Other, failure		
Supplementary description			

6.2.15 comm_wifi_list_ap

Function prototype	int comm_wifi_list_ap(st_wifi_ap_list * ap_list);	
Function function	Get the router list	
Parameter description	In parameter	
	Out parameter	ap_list Router list data, The ap_list space is allocated by the caller with an array size of 10
Return value	Number of routers	
Supplementary description		

6.2.16 comm_wifi_link_ap

Function prototype	int comm_wifi_link_ap(st_wifi_ap_list * ap_list , char * pwd);	
Function function	Connecting router	
Parameter description	In parameter	ap_list: Router data pwd: password
	Out parameter	

Return value	0, success Other, failure
Supplementary description	

6.2.17 comm_wifi_unlink_ap

Function prototype	int comm_wifi_unlink_ap();	
Function function	unlink router	
Parameter description	In parameter	
	Out parameter	ap_list Router list data, The ap_list space is allocated by the caller with an array size of 10
Return value	0, success Other, failure	
Supplementary description		

6.2.18 comm_wifi_get_link_state

Function prototype	int comm_wifi_get_link_state();	
Function function	Get connection status	
Parameter description	In parameter	
	Out parameter	
Return value	1, connection 0 , disconnect	
Supplementary description		

6.2.19 comm_wifi_get_signal

Function prototype	int comm_wifi_get_signal();	
Function function	get wifi signal	
Parameter description	In parameter	
	Out parameter	

Return value	wifi signal	

6.2.20 wifi_get_ssid

Function prototype	char * wifi_get_ssid();	
Function function	get ssid	
Parameter description	In parameter	
	Out parameter	
Return value	ssid	

6.2.21 wifi_get_ap_mac

Function prototype	char * wifi_get_ap_mac();	
Function function	get wifi ap mac address	
Parameter description	In parameter	
	Out parameter	
Return value	ap mac address	

6.2.22 wifi_get_rssi

Function prototype	int wifi_get_rssi();	
Function function	get wifi rssi	
Parameter description	In parameter	
	Out parameter	
Return value	rssi	

6.2.23 wifi_get_channel

Function prototype	int wifi_get_rssi();	
Function function	get wifi channel	
Parameter description	In parameter	
	Out parameter	
Return value	channel	

6.2.24 wifi_get_local_mac

Function prototype	char * wifi_get_local_mac();	
Function function	get wifi local mac address	
Parameter description	In parameter	
	Out parameter	
Return value	local mac address	

6.2.25 wifi_get_local_ip

Function prototype	char * wifi_get_local_ip();	
Function function	get wifi local ip	
Parameter description	In parameter	
	Out parameter	
Return value	wifi signal	

6.2.26 comm_atc_imei

Function prototype	const char * comm_atc_imei();	
Function function	get Module imei	
Parameter description	In parameter	
	Out parameter	
Return value	imei	

6.2.27 comm_atc_cpin

Function prototype	int comm_atc_cpin();	
Function function	get Module sim card status	
Parameter description	In parameter	
	Out parameter	
Return value	sim card status	

6.2.28 comm_atc_imsi

Function prototype	const char * comm_atc_imsi();	
Function function	get Module imsi	
Parameter description	In parameter	
	Out parameter	
Return value	imsi	

6.2.29 comm_atc_signal

Function prototype	int comm_atc_signal();	
Function function	get Module signal	
Parameter description	In parameter	
	Out parameter	
Return value	signal	

6.2.30 comm_atc_cell

Function prototype	int comm_atc_cell();	
Function function	get net registered cell	
Parameter description	In parameter	
	Out parameter	
Return value	cell	

6.2.31 comm_atc_lac

Function prototype	int comm_atc_cell();	
Function function	get net registered lac	
Parameter description	In parameter	
	Out parameter	
Return value	lac	

6.2.32 comm_atc_iccid

Function prototype	const char * comm_atc_iccid();	
Function function	get Module iccid	
Parameter description	In parameter	
	Out parameter	
Return value	iccid	

7 security（ libapi_security）

7.1 interface list

Function prototype	Function function
mksk_save_plaintext_key	Save key plaintext
mksk_save_encrypted_key	Save key ciphertext
mksk_3des_run	Use key 3des operation
mksk_3des_run_ex	Use key 3des operation
dukpt_get_ksn	Get a set of dukpt keys
dukpt_prepare_key	Get a set of dukpt keys
dukpt_3des_run	Use the previously obtained key 3des operation
dukpt_3des_run_ex	Use the previously obtained key 3des operation
dukpt_load_key	Load key
dukpt_init_ipek	Initialize the dukpt key use IPEK

dukpt_init_key	Initialize the dukpt key
dukpt_get_ksn	Get ksn
dukpt_init_ciphertext_ipek	Initialize the dukpt key use IPEK Ciphertext
dukpt_init_bdk	Initialize the dukpt key use BDK
sec_mac_proc	Computing mac
sec_encrypt_pin_proc	Read pin ciphertext from the security keyboard
sec_set_pin_mode	Set enable/disable pin input mode
sec_save_rsa_pri_key	Save the private key to the security module
sec_save_rsa_puk_key	Save the public key to the security module
sec_rsa_block	RSA block calculation
sec_get_hw_ver	get pci hardware version
sec_get_fw_ver	get pci firmware version

7.2 API interface

7.2.1 **mksk_save_plaintext_key**

Function prototype	int mksk_save_plaintext_key(int type, int gid, unsigned char * key, unsigned char *kvc);	
Function function	Save key plaintext	
Parameter description	In parameter	type： Key type(0x00-0x04) gid ： Key grouping(0-9) key : Key plaintext

	Out parameter	kvc Key kvc(Key plaintext encryption 8 0x00)
Return value	0, success Other, failure	
Supplementary description		

7.2.2 **mksk_save_encrypted_key**

Function prototype	int mksk_save_encrypted_key(int type, int gid, unsigned char * key, unsigned char *kvc);	
Function function	Save key ciphertext	
Parameter description	In parameter	type: Key type(0x00-0x04) gid : Key grouping(0-9) key : Key plaintext
	Out parameter	kvc Key kvc(Key plaintext encryption 8 0x00)
Return value	0, success Other, failure	
Supplementary description		

7.2.3 **mksk_3des_run**

Function prototype	int mksk_3des_run(int type, int gid, int mode, unsigned char *ind, int size, unsigned char *outd);	
Function function	Use key 3des operation	
Parameter description	In parameter	type： Key type(0x00-0x04) gid ： Key grouping(0-9) mode: Operation type (encryption/decryption) ind: Raw data size: Data length (8-byte multiple)
	Out parameter	outd: Calculation results
Return value	0, success Other, failure	
Supplementary description		

7.2.4 **dukpt_3des_run_ex**

Function prototype	int dukpt_load_key(int mode, int type, int gid, unsigned char* init_ksn, unsigned char* init_key, char * kvc);
Function function	Use the previously obtained key 3des operation

Parameter description	In parameter	mode:Operationtype(encryption/decryption) ind: Raw data size: Data length (8-byte multiple) des_mode:Data padding(DES_MODE_ECB/DES_MODE_CBC) key_tpye:DUKPT_DES_KEY_PIN/DUKPT_DES_KEY_MAC1/DUKPT_DES_KEY_MAC2/DUKPT_DES_KEY_DATA1/DUKPT_DES_KEY_DATA2
	Out parameter	outd: Calculation results
Return value	0, success Other, failure	

7.2.5 **dukpt_load_key**

7.2.6 **dukpt_get_ksn**

Function prototype	int dukpt_get_ksn(unsigned char gid, unsigned char * ksn);	
Function function	Get a set of dukpt keys	
Parameter description	In parameter	gid ： Key grouping,0
	Out parameter	ksn: Key corresponds to ksn

Return value	0, success Other, failure
Supplementary description	

7.2.7 **dukpt_prepare_key**

Function prototype	int dukpt_prepare_key(unsigned char gid, unsigned char * ksn);		
Function function	Get a set of dukpt keys		
Parameter description	In parameter	gid ：	Key grouping(0)
	Out parameter	ksn:	Key corresponds to ksn
Return value	0, success Other, failure		

7.2.8 **dukpt_3des_run**

Function prototype	int dukpt_3des_run(int mode, char *ind, int size, char *outd);
Function function	Use the previously obtained key 3des operation

Parameter description	In parameter	mode: Operation type (encryption/decryption) ind: Raw data size: Data length (8-byte multiple)
	Out parameter	outd: Calculation results
Return value	0, success Other, failure	
Supplementary description		

7.2.9 **dukpt_3des_run_ex**

Function prototype	int dukpt_3des_run_ex(int mode, char *ind, int size, char *outd, int des_mode, int key_tpye);	
Function function	Use the previously obtained key 3des operation	
Parameter description	In parameter	mode: Operation type (encryption/decryption) ind: Raw data size: Data length (8-byte multiple) des_mode:Data padding(DES_MODE_ECB/DES_MODE_CBC) key_tpye: DUKPT_DES_KEY_PIN/DUKPT_DES_KEY_MAC1/ DUKPT_DES_KEY_MAC2/DUKPT_DES_KEY_DATA1/DUKPT_DES_KEY_DATA2
	Out parameter	outd: Calculation results

Return value	0, success Other, failure

7.2.10 dukpt_load_key

Function prototype	int dukpt_load_key(int mode, int type, int gid, unsigned char* init_ksn, unsigned char* init_key, char * kvc);	
Function function	Initialize the dukpt key	
Parameter description	In parameter	type:Initial key type0 = ipek 1= bdk mode: Encryption method of initial key 0=Plaintext 1= tmk encryption 2= kek encryption gid: Key grouping，0 init_ksn: Initial ksn init_key: Initial key kvc : Key kvc(Key plaintext encryption 8 0x00)
	Out parameter	
Return value	0, success Other, failure	

7.2.11 dukpt_init_ipek

Function prototype	int dukpt_init_ipek(unsigned char gid, unsigned char* init_ksn, unsigned char* init_key);

Function function	Initialize the dukpt key use IPEK	
Parameter description	In parameter	gid: Key grouping，0 init_ksn: Initial ksn init_key: Initial key
	Out parameter	
Return value	0, success Other, failure	

7.2.12 **dukpt_init_ciphertext_ipek**

Function prototype	int dukpt_init_ciphertext_ipek(unsigned char gid, unsigned char* key, char * kvc);	
Function function	Initialize the dukpt key use IPEK Ciphertext	
Parameter description	In parameter	gid: Key grouping，0 key: ipek Ciphertext kvc: Key kvc(Key plaintext encryption 8 0x00)
	Out parameter	
Return value	0, success Other, failure	

7.2.13 dukpt_init_bdk

Function prototype	int dukpt_init_bdk(unsigned char gid, unsigned char* init_ksn, unsigned char* init_key);	
Function function	Initialize the dukpt key use BDK	
Parameter description	In parameter	gid: Key grouping，0 init_ksn: Initial ksn init_key: BDK
	Out parameter	
Return value	0, success Other, failure	

7.2.14 dukpt_init_key

Function prototype	int dukpt_init_key(unsigned char gid, unsigned char* init_ksn, unsigned char* init_key);	
Function function	Initialize the dukpt key	
Parameter description	In parameter	gid: Key grouping init_ksn: Initial ksn init_key: Initial key
	Out parameter	

Return value	0, success Other, failure
Supplementary description	

7.2.15 **sec_mac_proc**

Function prototype	int sec_mac_proc(int fid, int gid, int format, char *data, int len, char *mac, char * ksn);	
Function function	Computing mac	
Parameter description	In parameter	fid:SEC_MKSK_FIELD/SEC_DUKPT_FIELD gid: Key grouping, 0-9 format: SEC_MAC_UPAY_FORMAT... data: mac source data len: data length ksn: dukpt ksn
	Out parameter	mac: result
Return value	0, success Other, failure	
Supplementary description		

7.2.16 sec_encrypt_pin_proc

Function prototype	int sec_encrypt_pin_proc(int fid, int format, int gid, char * pan, char *pinblock, char * ksn);	
Function function	Read pin ciphertext from the security keyboard	
Parameter description	In parameter	fid: SEC_MKSK_FIELD/SEC_DUKPT_FIELD gid: Key grouping, 0-9 format:SEC_PIN_FORMAT0-SEC_PIN_FORMAT4 pan: card number
	Out parameter	ksn: dukpt ksn
Return value	0, success Other, failure	
Supplementary description		

7.2.17 sec_set_pin_mode

Function prototype	void sec_set_pin_mode(int mode, int length);	
Function function	Set enable/disable pin input mode	
Parameter description	In parameter	mode: 1 open 0 close length: Pin input length

	Out parameter	
Return value	0, success Other, failure	
Supplementary description		

7.2.18 sec_save_rsa_pri_key

Function prototype	int sec_save_rsa_pri_key(int index, int length, char * p, char * q);	
Function function	Save the private key to the security module	
Parameter description	In parameter	Parameter description
	Out parameter	index:key index(0-9) length:rsa byte size(128/256) p:Private key P component q:Private key Q component
Return value	0, success Other, failure	
Supplementary description		

7.2.19 sec_save_rsa_puk_key

Function prototype	int sec_save_rsa_puk_key(int index, int length, char * n);	
Function function	Save the public key to the security module	
Parameter description	In parameter	index: key index(0-9) length: rsa key byte size(128/256) n: public key N component
	Out parameter	
Return value	0, success Other, failure	
Supplementary description		

7.2.20 sec_rsa_block

Function prototype	int sec_rsa_block(int index, char * ind, char *outd, int length);	
Function function	RSA block calculation	
Parameter description	In parameter	index: key index(0-9) length: rsa key byte size(128/256) ind: in data

	Out parameter	outd: out data
Return value	0, success Other, failure	
Supplementary description		

7.2.21 sec_get_hw_ver

Function prototype	char * sec_get_hw_ver();	
Function function	get pci hardware version	
Parameter description	In parameter	
	Out parameter	
Return value	hardware version	
Supplementary description		

7.2.22 sec_get_fw_ver

Function prototype	char * sec_get_fw_ver();	
Function function	get pci firmware version	
Parameter description	In parameter	
	Out parameter	
Return value	firmware version	
Supplementary description		

7.2.23 dukpt_get_ksn

Function prototype	int dukpt_get_ksn(unsigned char gid, unsigned char * ksn);	
Function function	Get ksn	
Parameter description	In parameter	gid: Key grouping
	Out parameter	ksn

Return value	0, success Other, failure

8 Gui （libapi_gui）

8.1 interface list

Function prototype	Function function
gui_bar_rc	Gui filled area
gui_set_bar_color	Set the fill color
gui_get_bar_color	Get the fill color
gui_set_font	Set display font
gui_get_font	Get display font
gui_set_text_color	Set text color
gui_get_text_color	Get text color
gui_set_text_bg_color	Set the text background color
gui_get_text_bg_color	Get the text background color
gui_clear_dc	Clear screen display
gui_set_color	Set the foreground color
gui_get_color	Get the foreground color
gui_set_bg_color	Set the background color
gui_get_bg_color	Get the background color
gui_set_pixel	Draw on the screen
gui_get_pixel	The color of the point on the screen
gui_pixel	Draw a point
gui_text_out	Display text on the screen

gui_text_out_ex	Display text on the screen
gui_get_text_width	Get the display width of the text
gui_get_text_height	Get the display height of the text
gui_cline	Draw line
gui_line_to	Draw line
gui_get_width	Get screen width
gui_get_height	Get screen height
gui_page_op_paint	Display characters at the bottom left and bottom of the screen
gui_ime_set_mode	Set input method parameters
gui_ime_start_input	Open the input method page
gui_main_menu_func_add	Add menu handler
gui_main_menu_item_add	Add menu item
gui_main_menu_show	Add menu handler
gui_post_message	Send a message
gui_get_message	Recv a message
gui_proc_default_msg	Let the system process the default message
gui_messagebox_show	Display dialog
gui_load_bmp	Load bmp into memory
gui_out_bits	display image
gui_out_bits_ex	display image
gui_settextstyle	Setting Text Styles
gui_text_width_ex	get text width
gui_begin_batch_paint	Batch refresh starts
ui_end_batch_paint()	End of batch refresh
gui_set_full_screen	Set to full screen display
gui_bmp_free	Free memory
gui_out_bits_zoom	display image
gui_select_page_ex	select page
gui_titlecolorback	gui_titlecolorback
gui_titlecolorfore	gui_titlecolorfore
gui_menuhightlinecolor	gui_menuhightlinecolor
gui_textout_line_center	Display text on the screen ,Show

	only English
gui_clear_rect	Refresh the specified area

8.2 API interface

8.2.1 **gui_bar_rc**

Function prototype	void gui_bar_rc(int left, int top, int right, int bottom);	
Function function	Gui filled area	
Parameter description	In parameter	left Left border top Upper boundary right Right border bottom Lower boundary
	Out parameter	
Return value	0, success Other, failure	
Supplementary description		

8.2.2 **gui_set_bar_color**

Function prototype	void gui_set_bar_color(int color);

Function function	Set the fill color		
Parameter description	In parameter	color	Color format 0x00RRGGBB
	Out parameter		
Return value	0, success Other, failure		
Supplementary description			

8.2.3 gui_get_bar_color

Function prototype	int gui_get_bar_color();	
Function function	Get the fill color	
Parameter description	In parameter	
	Out parameter	
Return value	Fill color	

Supplementary description	

8.2.4 gui_set_font

Function prototype	Set display font	
Function function	void gui_set_font(int font);	
Parameter description	In parameter	font 0=12 lattice 1=16 lattice
	Out parameter	
Return value		
Supplementary description		

8.2.5 gui_get_font

Function prototype	int gui_get_font(void);
Function function	Get display font

Parameter description	In parameter	
	Out parameter	
Return value	Font index	
Supplementary description		

8.2.6 gui_set_text_color

Function prototype	void gui_set_text_color(int color);	
Function function	Set text color	
Parameter description	In parameter	color text color
	Out parameter	
Return value		
Supplementary description		

8.2.7 gui_get_text_color

Function prototype	int gui_get_text_color(void);	
Function function	Get text color	
Parameter description	In parameter	
	Out parameter	
Return value	Text color	
Supplementary description		

8.2.8 gui_set_text_bg_color

Function prototype	void gui_set_text_bg_color(int color) ;	
Function function	Set the text background color	
Parameter description	In parameter	cloro text color

	Out parameter	
Return value		
Supplementary description		

8.2.9 gui_get_text_bg_color

Function prototype	int gui_get_text_bg_color(void);	
Function function	Get the text background color	
Parameter description	In parameter	
	Out parameter	
Return value	Text background color	
Supplementary description		

8.2.10 gui_set_color

Function prototype	void gui_set_color(int color);	
Function function	Set the foreground color	
Parameter description	In parameter	color :the foreground color
	Out parameter	
Return value		
Supplementary description		

8.2.11 gui_get_color

Function prototype	void gui_get_color();	
Function function	Get the foreground color	
Parameter description	In parameter	
	Out parameter	
Return value	the foreground color	

Supplementary description	

8.2.12 gui_set_bg_color

Function prototype	void gui_set_bg_color(int color);	
Function function	Set the background color	
Parameter description	In parameter	color : the background color
	Out parameter	
Return value		
Supplementary description		

8.2.13 gui_get_bg_color

Function prototype	void gui_get_bg_color();	
Function function	Get the background color	
Parameter description	In parameter	
	Out parameter	

Return value	the background color
Supplementary description	

8.2.14 gui_clear_dc

Function prototype	void gui_clear_dc(void);	
Function function	Clear screen display	
Parameter description	In parameter	
	Out parameter	
Return value		
Supplementary description		

8.2.15 gui_set_pixel

Function prototype	int gui_set_pixel(int x, int y, int color);		
Function function	Draw on the screen		
Parameter description	In parameter	x y color	x coordinate y coordinate Point color
	Out parameter		
Return value	0	success	
Supplementary description			

8.2.16 gui_pixel

Function prototype	int gui_pixel(int x, int y);		
Function function	Draw a point		
Parameter description	In parameter	x y	x coordinate y coordinate
	Out parameter		

Return value	0, success Other, failure
Supplementary description	

8.2.17 gui_get_pixel

Function prototype	int gui_get_pixel(int x, int y);		
Function function	The color of the point on the screen		
Parameter description	In parameter	x y	x coordinate y coordinate
	Out parameter		
Return value	Point color		
Supplementary description			

8.2.18 gui_text_out

Function prototype	int gui_text_out(int x, int y, char * text);	
Function function	Display text on the screen， Show only English	
Parameter description	In parameter	x x coordinate y y coordinate text Text content
	Out parameter	
Return value	0 success	
Supplementary description		

8.2.19 **gui_text_out_ex**

Function prototype	int gui_text_out_ex(int x, int y, char * text);	
Function function	Display text on the screen， Display different languages	
Parameter description	In parameter	Parameter description
	Out parameter	

Return value	0	success
Supplementary description		

8.2.20 gui_get_text_width

Function prototype	int gui_get_text_width(char *text);		
Function function	Get the display width of the text		
Parameter description	In parameter	text	Text content
	Out parameter		
Return value			
Supplementary description			

8.2.21 gui_get_text_height

Function prototype	int gui_get_text_height(char *text);	
Function function	Get the display height of the text	
Parameter description	In parameter	text Text content
	Out parameter	
Return value	Text height	
Supplementary description		

8.2.22 gui_cline

Function prototype	void gui_cline(int x1, int y1, int x2, int y2, int color);	
Function function	Draw line	
Parameter description	In parameter	x1 Point 1 X coordinate x2 Point 2 X coordinate y1 Point 1 Y coordinate y2 Point 2 Y coordinate color Line color
	Out parameter	

Return value	
Supplementary description	

8.2.23 gui_line_to

Function prototype	void gui_line_to(int x, int y);	
Function function	Draw line	
Parameter description	In parameter	x :x coordinate y: y coordinate
	Out parameter	
Return value		
Supplementary description		

8.2.24 gui_get_width

Function prototype	int gui_get_width(void);	
Function function	Get screen width	
Parameter description	In parameter	
	Out parameter	
Return value	Screen width	
Supplementary description		

8.2.25 gui_get_height

Function prototype	xxx	
Function function	int gui_get_height(void);	
Parameter description	In parameter	
	Out parameter	

Return value	Screen height
Supplementary description	

8.2.26 gui_page_op_paint

Function prototype	void gui_page_op_paint(char * left_str, char * right_str);	
Function function	Display characters at the bottom left and bottom of the screen	
Parameter description	In parameter	left_str The character displayed in the lower left corner right_str The character displayed in the lower right corner
	Out parameter	
Return value		
Supplementary description		

8.2.27 gui_ime_set_mode

Function prototype	int gui_ime_set_mode(int def_mode, int allow_mode, int password);	
Function function	Set input method parameters	
Parameter description	In parameter	def_mode Default input method allow_mode Support input method password enter password
	Out parameter	
Return value		
Supplementary description		

8.2.28 gui_ime_start_input

Function prototype	int gui_ime_start_input(char * buffer, int max, int * position, char * help);	
Function function	Open the input method page	
Parameter description	In parameter	buffer Input buffer max Maximum input character position Cursor position help Enter page title

	Out parameter	
Return value	Input length	
Supplementary description		

8.2.29 gui_main_menu_func_add

Function prototype	int gui_main_menu_func_add(void * pfunc);		
Function function	Add menu handler		
Parameter description	In parameter	pfunc	Menu handler
	Out parameter		
Return value	0 success		
Supplementary description			

8.2.30 gui_main_menu_item_add

Function prototype	int gui_main_menu_item_add(st_gui_menu_item_def * menu_item);		
Function function	Add menu item		
Parameter description	In parameter	menu_item	Menu data
	Out parameter		
Return value	0 success		
Supplementary description			

8.2.31 gui_main_menu_show

Function prototype	void gui_main_menu_show(char *id , int timeover);
Function function	Display menu

Parameter description	In parameter	id menu id timeover overtime time
	Out parameter	
Return value		
Supplementary description		

8.2.32 gui_post_message

Function prototype	unsigned int gui_post_message(unsigned int msg_id, unsigned int wparam, unsigned int lparam);	
Function function	Send a message	
Parameter description	In parameter	msg_id Message id wparam parameter 1 lparam parameter 2
	Out parameter	
Return value	0 success	

Supplementary description	

8.2.33 gui_proc_default_msg

Function prototype	int gui_proc_default_msg(st_gui_message * pmsg);		
Function function	Let the system process the default message		
Parameter description	In parameter	pmsg	Message structure
	Out parameter		
Return value	0 success		
Supplementary description			

8.2.34 gui_messagebox_show

Function prototype	int gui_messagebox_show(char *title, char *msg , char* pszLeftOp, char* pszRightOp , int timeover);
Function function	Display dialog

Parameter description	In parameter	title Message title msg Message content pszLeftOp Bottom left corner pszRightOp Tip in the lower right corner timeover overtime time
	Out parameter	
Return value	1 Confirm return 2 Cancel back 3 Timeout	
Supplementary description		

8.2.35 gui_load_bmp

Function prototype	char * gui_load_bmp(char * filename , int *width , int *height);	
Function function	Load bmp into memory	
Parameter description	In parameter	filename Image name
	Out parameter	width Image width height Picture height

Return value	Image content array, which needs to be released after use	
Supplementary description		

8.2.36 gui_out_bits

Function prototype	void gui_out_bits(int x, int y, unsigned char *pbits, int width , int height, int mode);	
Function function	display image	
Parameter description	In parameter	x X coordinate y Y coordinate pbits Image data width Image width height Picture height
	Out parameter	
Return value		
Supplementary description	Show attention to release pbits	

8.2.37 gui_out_bits_ex

Function prototype	void void gui_out_bits_ex(int x, int y, unsigned char *pbits, int width , int height, int mode , int color);	
Function function	display image	
Parameter description	In parameter	x X coordinate y Y coordinate pbits Image data width Image width height Picture height mode Positive display, 1 Reverse display color Bit color of the picture(1,4,24)
	Out parameter	
Return value		
Supplementary description	Show attention to release pbits	

8.2.38 gui_text_width_ex

Function prototype	int gui_text_width_ex(char * str);
Function function	get text width

Parameter description	In parameter	str: text
	Out parameter	
Return value	text width	
Supplementary description		

8.2.39 gui_settextstyle

Function prototype	void gui_settextstyle(int textStyle);		
Function function	Setting Text Styles		

Parameter description	In parameter	textStyles	textStyle = 0 opaque,
			textStyle = 1 transparent

	Out parameter		
Return value			
Supplementary description			

8.2.40 gui_begin_batch_paint

Function prototype	void void gui_begin_batch_paint();	
Function function	Batch refresh starts	
Parameter description	In parameter	
	Out parameter	
Return value		
Supplementary description		

8.2.41 gui_end_batch_paint

Function prototype	void gui_end_batch_paint();	
Function function	End of batch refresh	
Parameter description	In parameter	
	Out parameter	
Return value		
Supplementary description		

	Out parameter	
Return value	0, success Other, failure	
Supplementary description		

8.2.42 gui_set_full_screen

Function prototype	void gui_set_full_screen(int full);	
Function function	void gui_set_full_screen(int full);	
Parameter description	In parameter	full:1set full screen display, cancel full screen display
	Out parameter	
Return value		

8.2.43 gui_bmp_free

Function prototype	void gui_bmp_free(char * pbmp);	
Function function	Free bmp memory	
Parameter	In	pbmp Image content array

description	parameter	
	Out parameter	
Return value		

8.2.44 gui_out_bits_zoom

Function prototype	void gui_out_bits_zoom(int x, int y, unsigned char *pbits, int width , int height, int mode, int zoom);	
Function function	display image	
Parameter description	In parameter	x X coordinate y Y coordinate pbits Image data width Image width height Picture height mode 0 Positive display, 1 Reverse display zoom Amplification factor
	Out parameter	
Return value		

8.2.45 gui_select_page_ex

Function prototype	int gui_select_page_ex(char *title , char *items[],int itemscount,int timeover, int select);
Function function	select page

Parameter description	In parameter	title: the title of the select page items: Menu items itemscount: Number of menu items timeout: Menu timeout select: Default menu item
	Out parameter	
Return value		

8.2.46 gui_titlecolorback

Function prototype	void gui_titlecolorback(int color);	
Function function	gui_titlecolorback	
Parameter description	In parameter	color: the color of title background
	Out parameter	
Return value		

8.2.47 gui_titlecolorfore

Function prototype	void gui_titlecolorfore(int color);	
Function function	gui_titlecolorfore	
Parameter	In	color: the color of title foreground

description	parameter	
	Out parameter	
Return value		

8.2.48 gui_menuhightlinecolor

Function prototype	void gui_menuhightlinecolor(int color);	
Function function	gui_menuhightlinecolor	
Parameter description	In parameter	color: the color of menu hightline color
	Out parameter	
Return value		

8.2.49 gui_textout_line_center

Function prototype	void gui_textout_line_center(char *pMsg , int top);	
Function function	Display text on the screen ,Show only English	
Parameter description	In parameter	pMsg pmsg content top top coordinater
	Out parameter	

Return value	

8.2.50 gui_clear_rect

Function prototype	void gui_clear_rect(int left, int top, int right, int bottom, int color);	
Function function	Refresh the specified area	
Parameter description	In parameter	left Left border top Upper boundary right Right border bottom Lower boundary color Refresh with specified color
	Out parameter	
Return value		

9 EMV(libapi_emv)

9.1 interface list

Function prototype	Function function
emv_read_card	EMV card trans.
EMV_iKernelInit	emv kernel data init
EMV_TermConfigInit	Init terminal configure
EMV_GetKernelVersion	EMV kernel version
EMV_GetKernelData	TLV from EMV buffer.
EMV_PrmSetAIDPrm	Save AID buffer.

EMV_PrmGetAIDPrm	Get AID.
EMV_PrmDelAIDPrm	Delete specific AID
EMV_PrmClearAIDPrmFile	Clear all AID.
EMV_PrmSetCAPK	Save CAPK.
EMV_PrmGetCAPK	Get specific CAPK.
EMV_PrmDelCAPK	Delete specific CAPL.
EMV_PrmClearCAPKFile	Clear all CAPK.
EMV_GetDataByTag	Getting the specified tag value
EMV_PackTLVData	Process of pack emv tag
EMV_GetVersion	Get the EMV kernel version
EMV_SetReadingCardDisp	Set ReadingCard Tip CallBack Function
EMV_SetInputPin	Set the Offline PIN interface
EMV_SetDispOffPin	Set offline PIN prompt interface
EMV_ShowAID_Prm	Show AID
EMV_ShowCAPK_Prm	Show CAPK

9.2 API interface

9.2.1 **emv_read_card**

Function prototype	int emv_read_card(st_read_card_in *card_in, st_read_card_out *card_out);	
Function function	Process of emv card trans.	
Parameter description	In parameter	The parameter of EMV trans.
	Out parameter	Out buffer of EMV trans.

Return value	Result of emv trans.
Supplementary description	

9.2.2 **EMV_iKernelInit**

Function prototype	void EMV_iKernelInit(void)	
Function function	emv kernel data init	
Parameter description	In parameter	nothing
	Out parameter	nothing
Return value	nothing	
Supplementary description		

9.2.3 **EMV_TermConfigInit**

Function prototype	int EMV_TermConfigInit(const TERMCONFIG *ptermconfig);
Function function	Init terminal configure of emv.

Parameter description	In parameter	Terminal configure of emv.
	Out parameter	Null
Return value	Result of init terminal configure.	
Supplementary description		

9.2.4 **EMV_GetKernelVersion**

Function prototype	void EMV_GetKernelVersion(char *KernelVersion, int size);	
Function function	Get emv kernel version	
Parameter description	In parameter	Length of version buffer.
	Out parameter	Kernel Version
Return value	Null	
Supplementary description		

9.2.5 **EMV_GetKernelData**

Function prototype	int EMV_GetKernelData (char *Tag, int *Len, byte *Value);	
Function function	Get TLV from EMV buffer.	
Parameter description	In parameter	Tag
	Out parameter	Length Value
Return value	Result of get TLV data.	
Supplementary description		

9.2.6 **EMV_PrmSetAIDPrm**

Function prototype	int EMV_PrmSetAIDPrm(TERMINALAPPLIST *pTerminalApps);	
Function function	Set AID buffer into device.	
Parameter description	In parameter	Aid buffer.
	Out parameter	Null
Return value	Result of set aid.	
Supplementary description		

9.2.7 EMV_PrmGetAIDPrm

Function prototype	int EMV_PrmGetAIDPrm(TERMINALAPPLIST *pTerminalApps);	
Function function	Get all aid into memory.	
Parameter description	In parameter	Null
	Out parameter	The AID buffer
Return value	Result of get aid buffer.	
Supplementary description		

9.2.8 EMV_PrmDelAIDPrm

Function prototype	int EMV_PrmDelAIDPrm(byte *AID, byte AID_Len);	
Function function	Delete the specific AID.	
Parameter description	In parameter	AID Length of AID
	Out parameter	Null

Return value	Result of Delete.
Supplementary description	

9.2.9 EMV_PrmClearAIDPrmFile

Function prototype	int EMV_PrmClearAIDPrmFile(void);	
Function function	Clear all AID from device.	
Parameter description	In parameter	Null
	Out parameter	Null
Return value	Result of clear AID.	
Supplementary description		

9.2.10 EMV_PrmSetCAPK

Function prototype	int EMV_PrmSetCAPK(CAPUBLICKEY *ppkKey);
Function function	Save CAPK into device.

Parameter description	In parameter	CPAK
	Out parameter	Null
Return value	Result of save CAPK.	
Supplementary description		

9.2.11 EMV_PrmGetCAPK

Function prototype	int EMV_PrmGetCAPK(CAPUBLICKEY *ppkKey, byte *RID, byte PKIndex);	
Function function	Get the specific index of CAPK.	
Parameter description	In parameter	RID of CAPK Index of CAPK
	Out parameter	CAPK
Return value	Result of get CAPK.	
Supplementary description		

9.2.12 EMV_PrmDelCAPK

Function prototype	int EMV_PrmDelCAPK(byte *RID, byte PKIndex);	
Function function	Delete the specific index of CAPK.	
Parameter description	In parameter	RID of CAPK Index of CAPK
	Out parameter	Null
Return value	Result of delete.	
Supplementary description		

9.2.13 EMV_PrmClearCAPKFile

Function prototype	int EMV_PrmClearCAPKFile(void);	
Function function	Clear all CAPK from device.	
Parameter description	In parameter	Null
	Out parameter	Null
Return value	Result of clear.	
Supplementary description		

9.2.14 EMV_GetDataByTag

Function prototype	int EMV_GetDataByTag(byte *psTag, byte *psSrc, int nSrcLen,byte *psBuf, int *nBufLen);	
Function function	Getting the specified tag value from the resource data.	
Parameter description	In parameter	psTag:Afferent tag psSrc:Resource data nSrcLen:Resource data length
	Out parameter	psBuf:Returns the tag value nBufLen:Returns the length of the tag value
Return value	0 Success, 1 Failure.	
Supplementary description		

9.2.15 EMV_PackTLVData

Function prototype	int EMV_PackTLVData(byte *pTagName,byte *TagValue, int iTagValueLen, byte *psBuf,int *nBufLen)	
Function function	Process of pack emv tag	
Parameter description	In parameter	pTagName: tag name TagValue: tag value iTagValueLen : tag value length
	Out parameter	psBuf :Returns TLV packed data nBufLen:Returns length of packed data

Return value	0 Success, 1 Failure.

9.2.16 EMV_GetVersion

Function prototype	char * EMV_GetVersion(void)	
Function function	Get the EMV kernel version	
Parameter description	In parameter	Nothing
	Out parameter	Nothing
Return value	emv version	

9.2.17 EMV_SetReadingCardDisp

Function prototype	VoidEMV_SetReadingCardDisp(void (*ReadingCardDisp)(int));	
Function function	Set ReadingCard Tip CallBack Function	
Parameter description	In parameter	Callback function Funtion param: 1--Indicate contact transactions 2--Indicate contactless transactions
	Out parameter	Nothing
Return value	Nothing	

9.2.18 EMV_SetInputPin

Function prototype	EMV_SetInputPin(int (*InputPin)(char *,char *,char ,char *));	
Function function	Set the Offline PIN interface	
Parameter description	In parameter	Callback(InputPin):(char*psCardNo,char *psAmt,charcMsgType,char*psPin) Input: char *psCardNo (PAN) char *psAmt (Amount) char cMsgType: 1--PIN_LAST 2--PIN_AGAIN 3--PIN_NOMAL Output:char*psPi(Enteredpassword)
	Out parameter	Nothing
Return value	0 Success, 1 Failure.	

9.2.19 EMV_SetDispOffPin

Function prototype	void EMV_SetDispOffPin(void (*DispOffPin)(int));	
Function function	Set offline PIN prompt interface	
Parameter description	In parameter	Callback(DispOffPin): Input:int param value 0--PIN OK N--Number of re-verifications

	Out parameter	Nothing
Return value	Nothing.	

9.2.20 EMV_ShowAID_Prm

Function prototype	void EMV_ShowAID_Prm(void);	
Function function	Show AID	
Parameter description	In parameter	Nothing
	Out parameter	Nothing
Return value	Nothing.	

9.2.21 EMV_ShowCAPK_Prm

Function prototype	void EMV_ShowCAPK_Prm(void);	
Function function	Show CAPK	
Parameter description	In parameter	Nothing
	Out parameter	Nothing

Return value	Nothing.

10 Print (libapi_print)

10.1 interface list

Function prototype	Function function
UPrint_GetModuleVer	Get version number of print class module
UPrint_Init	Initialize, check the printer status (if it is out of paper), set the print font, use before printing
UPrint_Str	String printing with automatic line break function, support \r, \n newline
UPrint_BitMap	Picture printing
UPrint_Start	Start printing
UPrint_StrBold	String printing (UPrint_StrBold) with automatic line feed function, support \r, \n newline
UPrint_Feed	Printer paper feeding
UPrint_MatrixCode	Print QR code
UPrint_SetFont	Set print font
UPrint_SetDensity	Set print density

10.2 API interface

10.2.1 UPrint_GetModuleVer

Function prototype	int UPrint_GetModuleVer(char *pszVer);

Function function	Get version number of print class module	
Parameter description	In parameter	Nothing
	Out parameter	pszVer Module version number
Return value	> 0 Successfully returns module version number length USYS_FAIL = -1	
Supplementary description		

10.2.2 UPrint_Init

Function prototype	int UPrint_Init(void);	
Function function	Initialize, check the printer status (if it is out of paper), set the print font, use before printing.	
Parameter description	In parameter	Nothing
	Out parameter	Nothing
Return value	UPRN_FILE_FAIL Fail to open the file UPRN_OUTOF_PAPER The printer is out of paper UPRN_DEV_FAIL Printer device failure UPRN_FAIL Printer unknown fault UPRN_SUCCESS Success	

Supplementary description	

10.2.3 UPrint_Str

Function prototype	int UPrint_Str(char *str, byte attrib, int linegap, byte newline);	
Function function	String printing with automatic line break function, support \r, \n newline	
Parameter description	In parameter	str: Need to print string information attrib: Font size: 0 small, 1 medium, 2 large linegap: Line spacing: unit pixels, 0 is the default value (for Pin printing use) newline: 0 Does not support line breaks;1 support \r, \n newline
	Out parameter	Nothing
Return value	UPRN_CACHE_ERR Save cache failed UPRN_SUCCESS Success	
Supplementary description		

10.2.4 UPrint_BitMap

Function prototype	int UPrint_BitMap(char *BmpFile,byte pattern);

Function function	Picture printing	
Parameter description	In parameter	BmpFile: Image file name（XXX.bmp） pattern: Alignment: 0 left alignment, 1 center alignment, 2 right alignment
	Out parameter	Nothing
Return value	UPRN_CACHE_ERR Save cache failed UPRN_SUCCESS Success	
Supplementary description		

10.2.5 UPrint_Start

Function prototype	int UPrint_Start(void);	
Function function	Start printing	
Parameter description	In parameter	Nothing
	Out parameter	Nothing
Return value	UPRN_HANDLE_BACK Split machine handle is not put back UPRN_FILE_FAIL Fail to open the file UPRN_LOSE_COMMAND Print handle not obtained UPRN_OUTOF_PAPER The printer is out of paper UPRN_DEV_FAIL Printer device failure	

	UPRN_FAIL UPRN_SUCCESS	Printer unknown fault Success
Supplementary description		

10.2.6 UPrint_StrBold

Function prototype	int UPrint_StrBold(char *pszStr, byte cAttrib, byte cPattern,int nLinegap, byte newline);	
Function function	String printing with automatic line feed function, support \r, \n newline	
Parameter description	In parameter	pszStr: Need to print string information cAttrib: Font size: 0 small, 1 medium, 2 large cPattern: Print position: 0 left, 1 center, 2 right nlinegap: Line spacing, unit pixels, 0 is the default value (for Pin printing use) newline: 0 Does not support line breaks;1 support \r, \n newline
	Out parameter	Nothing
Return value	UPRN_CACHE_ERR UPRN_SUCCESS	Save cache failed Success
Supplementary description		

10.2.7 UPrint_Feed

Function prototype	int UPrint_Feed(int nFeedLines);		
Function function	Printer paper feeding		
Parameter description	In parameter	nFeedLines	Paper length (pixels)
	Out parameter	Nothing	
Return value	UPRN_CACHE_ERR UPRN_SUCCESS		Save cache failed Success
Supplementary description			

10.2.8 UPrint_MatrixCode

Function prototype	int UPrint_MatrixCode(const char *psMatrixCode, int nLen,byte cSize,byte cPattern);	
Function function	Print QR code （UPrint_MatrixCode） ,Convert incoming data to QR code and print	
Parameter description	In parameter	psMatrixCode: QR code data nLen: QR code data length cSize: QR code size, 0-small, 1-medium, 2-large cPattern: Alignment, 0 left alignment, 1 center alignment, 2 right alignment

	Out parameter	Nothing
Return value	UPRN_CACHE_ERR Save cache failed UPRN_SUCCESS Success	
Supplementary description		

10.2.9 UPrint_SetFont

Function prototype	int UPrint_SetFont(int size, int zoom_w, int zoom_h);	
Function function	Set print font	
Parameter description	In parameter	size: Set print English font size(0--8) zoom_w:Set the horizontal magnification of English(1--5) zoom_h: Set the vertical magnification of English(1--5)
	Out parameter	
Return value	> 0 Successfully returns module version number length USYS_FAIL = -7	

10.2.10 UPrint_SetDensity

Function prototype	int UPrint_SetDensity(int v);	
Function function	Set print density	
Parameter description	In parameter	Set print density (1--5, 3 is normal)
	Out parameter	
Return value	> 0 Successfully returns module version number length USYS_FAIL = -7	

11 EMV_API(lib_emvapi)

11.1interface list

Function prototype	Function function
emv_read_card	EMV card trans.
emv_online_resp_proc	Process of emv online resp proc
EMV_online_cardemv_free	Emv free
emv_onlineresp_proc_pack	Process of emv online resp proc and pack tlv data
emv_card_begin	Read card begin
emv_card_loop	Check card type
emv_card_end	Precess of emv card read
Emvapi_Version	Get EMV api version

EMV_iKernelInit	Kernel init
EMV_SetInputPin	Offline pin input
EMV_SetDispOffPin	Offpin display
EMV_SetReadingCardDisp	Read card display
EMV_GetVersion	Get kernel version
emvapi_onlinpin_proc_page	Process online pin entering and output encrypted PIN block
EMV_SetRuPayServiceList	Set rupay service data into device
EMV_GetRuPayServiceList	Get all rupay service data into memory
EMV_SetRuPayPRMacqKeyList	Set rupay PRMacqKey list into device
EMV_GetRuPayPRMacqKeyList	Get all rupay service PRMacqKey into memory
EMV_ShowRuPayPRMacqKey	Show RuPay PRMacq Key
EMV_ShowRuPayService	Show RuPay Service
EMV_ClearRuPayServiceFile	Clear all rupay service from device
EMV_ClearRuPayPRMacqKeyFile	Clear all rupay PRMacq Key from device

11.2API interface

11.2.1 emv_online_resp_proc

Function prototype	int emv_online_resp_proc(int nOnlineRes,char *sResp39,char *sField55,int nFieldLen);	
Function function	Process of emv online resp proc	
Parameter description	In parameter	nOnlineRes: 0--online success 1--online fail 2--Not online sResp39: Online Response Code sField55: contain 91/8A/71/72 Tag

		Data nFieldLen : sField55 Length
	Out parameter	Nothing
Return value	EMVAPI_RET_TC 0 //TC EMVAPI_RET_AAC -1 //AAC EMVAPI_RET_AAR -2 //Terminate	
Supplementary description	can call EMV_GetKernelData get tags,then must call EMV_online_cardemv_free() to free memory	

11.2.2 **EMV_online_cardemv_free**

Function prototype	void EMV_online_cardemv_free(void);	
Function function	EMV data free	
Parameter description	In parameter	Nothing
	Out parameter	Nothing
Return value	Nothig	

11.2.3 **emv_onlineresp_proc_pack**

Function prototype	int emv_onlineresp_proc_pack(int nOnlineRes,char *sResp39,char *sField55,char*emvtags, char*packvalue,int*packlen);
Function function	Process of emv online resp proc and pack tlv data

Parameter description	In parameter	nOnlineRes : 0--online success 1--online fail 2--Not online sResp39: Online Response Code sField55: ASCC code;contain 91/8A/71/72 Tag Data nFieldLen : sField55 Length emvtags: ASCC code;the tags want to get
	Out parameter	packvalue: HEX code,the tlv data match emvtags packlen:length of packvalue
Return value	EMVAPI_RET_TC 0 //TC EMVAPI_RET_AAC -1 //AAC EMVAPI_RET_AAR -2 //Terminate	
Supplementary description	Don't need to call EMV_online_cardemv_free()	

11.2.4 emv_card_begin

Function prototype	int emv_card_begin(st_read_card_in *card_in);	
Function function	EMV Read card begin	
Parameter description	In parameter	Card_in:The parameter of EMV trans
	Out parameter	Nothing
Return value	0--succ -1 --fail	

11.2.5 emv_card_loop

Function prototype	int emv_card_loop(int card_mode);

Function function	Check card type	
Parameter description	In parameter	Card_mode: 0x01:MAGTEK 0x02:ICC 0x03:RF
	Out parameter	Nothing
Return value	2:mag card 3:IC card 4:rf card 0:Nothing	

11.2.6 emv_card_end

Function prototype	int emv_card_end(int ret, st_read_card_in *card_in,st_read_card_out *card_out);	
Function function	Process of EMV read card	
Parameter description	In parameter	Ret:emv_card_loop api return value Card_in:The parameter of EMV trans.
	Out parameter	Card_out:Out buffer of EMV trans
Return value	Result of emv trans.	

11.2.7 Emvapi_Version

Function prototype	void Emvapi_Version(char *pszVersion);

Function function	Get emvapi version	
Parameter description	In parameter	nothing
	Out parameter	pszVersion:EMVAPI_VERSION
Return value	nothing	

11.2.8 emvapi_onlinpin_proc_page

Function prototype	int emvapi_onlinpin_proc_page(int bByPassPin,int key_pid,int pin_gid,char*amt,char *pan, int*pin_len,char*pin_ksn, char*pin_block);	
Function function	Process online pin entering and output encrypted PIN block	
Parameter description	In parameter	bByPassPin:0--not supportbypass 1--support bypass key_pid:SEC_MKSK_FIELD,SEC_DUKPT_FIELD;refer to libapi_security.h pin_gid: index of key;match with key_pid amt : amount pan: pan of this transaction
	Out parameter	pin_len: length of pin pin_ksn: ksn of KF_DUKPT mode; pin_block: encrypted PIN
Return value	EMVAPI_RET_SUCC EMVAPI_RET_TIMEOUT EMVAPI_RET_CANCEL	

11.2.9 EMV_SetRuPayServiceList

Function prototype	int EMV_SetRuPayServiceList(RUPAYSERVICELIST *pRuPayServiceList);	
Function function	Set rupay service data into device	
Parameter description	In parameter	pRuPayServiceList: rupay sevice data list buffer
	Out parameter	Nothing
Return value	0--succ -1 fail	

11.2.10 EMV_GetRuPayServiceList

Function prototype	int EMV_GetRuPayServiceList(RUPAYSERVICELIST *pRuPayServiceList);	
Function function	Get all rupay service data into memory	
Parameter description	In parameter	nothing
	Out parameter	pRuPayServiceList: rupay sevice data list buffer
Return value	0--succ -1 fail	

11.2.11 EMV_SetRuPayPRMacqKeyList

Function prototype	int EMV_SetRuPayPRMacqKeyList(RUPAYPRMACQKEYLIST *pRuPayPRMacqKeyList);	
Function function	Set rupay PRMacqKey list into device	
Parameter description	In parameter	pRuPayPRMacqKeyList: rupay PRMacqKeylist buffer
	Out parameter	nothing
Return value	0--succ -1 fail	

11.2.12 EMV_GetRuPayPRMacqKeyList

Function prototype	int EMV_GetRuPayPRMacqKeyList(RUPAYPRMACQKEYLIST *pRuPayPRMacqKeyList);	
Function function	Get all rupay service PRMacqKey into memory	
Parameter description	In parameter	nothing
	Out parameter	pRuPayServicePRMacqKeyList: rupay sevice PRMacqKey list buffer
Return value	0--succ -1 fail	

11.2.13 EMV_ShowRuPayPRMacqKey

Function prototype	void EMV_ShowRuPayPRMacqKey(void);	
Function function	Show RuPay PRMacq Key	
Parameter description	In parameter	nothing
	Out parameter	nothing
Return value	nothing	

11.2.14 EMV_ShowRuPayService

Function prototype	void EMV_ShowRuPayPRMacqKey(void);	
Function function	Show RuPay Service	
Parameter description	In parameter	nothing
	Out parameter	nothing
Return value	nothing	

11.2.15 EMV_ClearRuPayServiceFile

Function prototype	int EMV_ClearRuPayServiceFile(void);	
Function function	Clear all rupay service from device	
Parameter description	In parameter	nothing
	Out parameter	nothing
Return value	nothing	

11.2.16 EMV_ClearRuPayPRMacqKeyFile

Function prototype	int EMV_ClearRuPayPRMacqKeyFile(void);	
Function function	Clear all rupay PRMacq Key from device	
Parameter description	In parameter	nothing
	Out parameter	nothing
Return value	nothing	