Advanced Card Systems Ltd.
Card & Reader Technologies

eH880 and ACR880

Application Programming Interface

info@acs.com.hk
www.acs.com.hk

Table of Contents

1.0. Introduction

1.1. Purpose

This manual describes the API (Application Programming Interface) calls developed specifically for both the eH880 and ACR880 products. Application software developers can make use of these APIs to develop their smart-card related applications.

1.2. Scope

Both the eH880 and the ACR880 share similar hardware platform. Both use the ARM9 processor and run on Linux 2.6.18 OS. They also support contacted and contactless smart-cards as well as fingerprint modules. However, each reader has its own hardware extension.

This document is organized as follows:

API Calls which are common to both the eH880 and ACR880. These include mainly the Man-Machine Interface (MMI) like the LCD / LED and keypad controls; and other peripheral controls which are common to both systems.

As both systems run on the Linux OS, many shareware packages can be installed and made available to system developers. These include, but not limited to, PC/SC standard for smart-card support, SSL for security control, etc. Application developers are advised to refer to the respective API manuals of these packages for reference.

1.3. Other Referenced Documents

Both the eH880 and ACR880 adopt the PC/SC standard to access the contact and contactless smart card. Refer to the following documentations for the PC/SC API:

http://pcsclite.alioth.debian.org/

If OpenSSL is installed in the device, refer to the following web-links for the documentation related to this shareware:

http://www.openssl.org/docs/

This book is also a good reference for OpenSSL: “Network Security with OpenSSL” by Pravir Chandra, Matt Messier, and John Viega, published by O’Reilly, June 2002; ISBN: 0-596-00270-X,

Moreover, the Linux system call can be found in:

www.tldp.org

and a concise description of its system calls can be found in:

www.chinalinuxpub.com/doc/pro/syscalls_toc.html

The following site also provides helpful information on arm-based Linux kernel:

www.arm.linux.org.hk

2.0. Overview of EH880/ACR880 API Function Calls

The API is compiled using the gcc compiler and is intended to run on the Linux O.S. version 2.6.12.

Section 3.0 describes the APIs that are common to both the eH880 and ACR880.

3.0. eH880/ ACR880 API Functions

3.1. Keypad API

3.1.1. A880_KPD_Clean

The **APAC_KPD_Clean** clears the keypad buffer.

int A880_KPD_Clean(

);

Parameters

None.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is not equal to zero.

Remarks

Example Code

Requirements

Header: Declared in A880_KPD.h
Library: Use libA880_KPD.a

See Also
A880_KPD_Read
A880_KPD_Check

3.1.2. A880_KPD_Read

The **A880_KPD_Read** catches a keypad input.

int **A880_KPD_Read** (

int *timeout*

);

Parameters

timeout

timeout [I]: Timeout to wait for key in unit of 100ms, -ve for infinite

Return Values

If a key was pressed, the return value is 0 or greater than 0.
If no key has been pressed or a key failed, the return value is less than 0.

Remarks

Example Code

Requirements

Header: Declared in A880_KPD.h
Library: Use libA880_KPD.a

See Also
A880_KPD_Clean
A880_KPD_Check

3.1.2. A880_KPD_Check

The **A880_KPD_check** listens for a key press.

int **A880_KPD_check** (

int *key*

);

Parameters

key

key {I]: key code of the key to check

Return Values

If a key was pressed, the return value is 1.
If a key was not pressed, the return value is 0.
If a key fails, the return value is neither 0 nor 1.

Requirements

Header: Declared in A880_KPD.h
Library: Use libA880_KPD.a

See Also
A880_KPD_Clean
A880_KPD_Read

3.2. LCD API

3.2.1. A880_LCD_Pixel

The **A880_LCD_Pixel** draws a pixel on the LCD screen.

int **A880_LCD_Pixel** (

int *x,*
int *y*,
int *color*

);

Parameters

x

x coordinate of the pixel

y

y coordinate of the pixel

color

color of the pixel

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is not equal to zero.

Requirements

Header: Declared in A880_LCD.h
Library: Use libA880_LCD.a

See Also
A880_LCD_Line
A880_LCD_Flood
A880_LCD_Draw
A880_LCD_Textout

3.2.2. A880_LCD_Line

The **A880_LCD_Line** draws a line on the LCD screen.

int **A880_LCD_Line** (

int *x1,*
int *y1*,
int *x2,*
int *y2,*
long *color*

);

Parameters

x1

x1 coordinate of stating point

y1

y1 coordinate of stating point

x2

x2 coordinate of end point

y2

y2 coordinate of end point

color

Color of the line

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is not equal to zero.

Requirements

Header: Declared in A880_LCD.h
Library: Use libA880_LCD.a

See Also
A880_LCD_Pixel
A880_LCD_Flood
A880_LCD_Draw
A880_LCD_Textout

3.2.3. A880_LCD_Flood

The **A880_LCD_Flood** fills the whole LCD with a specified color.

int **A880_LCD_Flood** (

long *color*

);

Parameters

color

Color of the line

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is not equal to zero.

Requirements

Header: Declared in A880_LCD.h
Library: Use libA880_LCD.a

See Also
A880_LCD_Pixel
A880_LCD_Line
A880_LCD_Draw
A880_LCD_Textout

3.2.4. A880_LCD_Draw

The **A880_LCD_Draw** draws on LCD with a specified data.

int **A880_LCD_Draw** (

int *x,*
int *y*,
int *width,*
int *height,*
const long **data*

);

Parameters

x

x coordinate of top-left corner of the drawing region

y

y coordinate of top-left corner of the drawing region

width

width (from left to right) of drawing region

height

height (from top to bottom) of drawing region

**data*

pointer to the buffer containing RGB values

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is not equal to zero.

Requirements

Header: Declared in A880_LCD.h

Library: Use libA880_LCD.a

See Also
A880_LCD_Pixel
A880_LCD_Line
A880_LCD_Flood
A880_LCD_Textout

3.2.5. A880_LCD_TextOut

The **A880_LCD_TextOut** places a string on the LCD.

int **A880_LCD_ TextOut** (

int **x,*
int **y*,
long *FGColor,*
long *BGColor,*
const char **FontFile,*
int *scale,*
const char **Str,*
*const char *Encoding*
bool *TxtWrap*

);

Parameters

**x*

x[I/O]: On input, pointer to the x coordinate of starting point; On output, pointer to the x coordinate of next starting point

**y*

y[I/O]: On input, pointer to the y coordinate of starting point; On output, pointer to the y coordinate of next starting point

FGColor

FGColor [I]: Specify the Foreground color

BGColor

BGColor [I]: Specify the Background color

scale

scale [I]: Specify the scale factor to the font

Str

Str [I]: Specify the string to put

Encoding

Encoding [I]: Specify the character encoding of the string. Valid encoding includes "ASCII", "ISO646-DE", "BIG5", "UTF8"

TxtWrap

TxtWrap [I]: Specify if text wrap is enabled at LCD boundary

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is not equal to zero.

Requirements

Header: Declared in A880_LCD.h
Library: Use libA880_LCD.a

See Also
A880_LCD_Pixel
A880_LCD_Line
A880_LCD_Flood
A880_LCD_Draw

3.3. LED API

3.3.1. A880_LedCtl

The **A880_LedCtl** controls the LED.

int **A880_LedCtl** (

int *Idx,*
int *Status,*
int *Duration*

);

Parameters

Idx

Specifies which led to control

Idx	Description
0	1st LED, Dual color
1	2nd LED, Dual Color
2	3rd LED, Red
3	4th LED, Green
4	LCD and Keypad Backlight
5	Contactless backlight
6	Fingerprint backlight

Status

Color and blinking speed (color | speed)
*refer to the A880.h

Duration

Duration (in 0.1s) of the Led Status before OFF. 0: Infinite, Ignore if Status is OFF.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is not equal to zero.

Remarks

Example Code

Requirements

Header: Declared in A880.h
Library: Use libA880.a

See Also

3.4. Audio API

3.4.1. A880_Beep

The **A880_Beep** turn on (with specific Frequency and Duration) or off the speaker.

int **A880_Beep** (

int *freq,*
int *Duration*

);

Parameters

freq

frequency (Hz), 0 to turn off

duration

ON duration (ms), 0 to play forever

Requirements

Header: Declared in A880.h
Library: Use libA880.a

3.5. Firmware-Upgrade API

3.5.1. A880_FwWrite

The **A880_FwWrite** used to Write/Update the device Firmware.

int **A880_FwWrite** (

const char *FwFile,*
const char **DiverseKey,*
Int *(*ReportFunc)* (int *Status)*

);

Parameters

FwFile

Name of Firmware Image File

**DiverseKey*

String that is used to diversify the firmware image. Set it to NULL unless a diversified key is used

ReportFunc

Callback function that is used to report the execution status of the API. Set it to NULL unless there is a need to access the execution status of the API. The execution status ranges from â€“1 (checking FwFile) to 0 (0%done) and then to 100 (100% done)

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is -1.

Requirements

Header: Declared in A880.h
Library: Use libA880.a

3.6. Real Time Clock API

3.6.1. A880_RtcRead

The **A880_RtcRead** is used to read the system time.

time_t **A880_ RtcRead** (

);

Parameters

None

Return Values

The function returns system time.

Requirements

Header: Declared in A880.h
Library: Use libA880.a

3.6.2. A880_RtcWrite

The **A880_RtcWrite** used to modify the system time.

time_t **A880_RtcWrite** (

);

Parameters

Time_t t

Current time value which the user wants to set as the system time

Return Values

The function returns system time.

Requirements

Header: Declared in A880.h
Library: Use libA880.a

3.7. Contactless Reader API

This section describes the APIs for contactless reader, Mifare Classic and DesFire card.

3.7.1. A880_PCD_Open

The **A880_PCD_Open** opens a new connection to the Contactless Card Reader (PCD).

int **A880_PCD_Open** (

);

Return Values

If the function succeeds, the handle to the PCD is returned.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_PICC.h
Library: Use libA880_CL.a

3.7.2. A880_PCD_Close

The **A880_PCD_Close** closes the connection to the Contactless Card Reader opened by **A880_PCD_Open**.

int **A880_PCD_Close** (

int *hpcd*

);

Parameters

hpcd
[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_PICC.h
Library: Use libA880_CL.a

3.7.3. A880_PCD_ReadEEPROM

The **A880_PCD_ReadEEPROM** reads the internal EEPROM of the contactless reader chip.

int **A880_PCD_ReadEEPROM**(

int *hpcd,*
int *addr,*
int *len,*
int **DataContent*

);

Parameters

hpcd
[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

addr
[in] Specifies the address of the EEPROM to read. The readable range is from location 0x00 to 0x7F. The read-only location (0x0-0x03), which contains a 4 byte unique serial number of the reader chip, can also be read by this operation.

len
[in] Specifies the length of data in bytes to read from EEPROM, up to a maximum of 48 bytes.

DataContent
[out] Specifies a pointer to the location where the 1 byte requested data will be stored.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_PICC.h
Library: Use libA880_CL.a

See Also

A880_PCD_WriteEEPROM
A880_PCD_StoreKey

3.7.4. A880_PCD_WriteEEPROM

The **A880_PCD_WriteEEPROM** writes the internal EEPROM of the contactless reader chip with data.

int **A880_PCD_WriteEEPROM**(

int *hpcd*,
int *addr,*
int *len,*
const int **DataContent*

);

Parameters

hpcd
[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

addr
[in] Specifies the address of the EEPROM to write. The

writable range is from location 0x10 to 0x1FF.

len

[in] Specifies the length of data in bytes to be written to the EEPROM which is up to a maximum of 61 bytes.

DataContent

[in] Specifies a pointer to the EEPROM location of the contactless reader chip where the 1 byte data to write is stored.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_PICC.h
Library: Use libA880_CL.a

See Also

A880_PCD_ReadEEPROM
A880_PCD_StoreKey

3.7.5. A880_PCD_StoreKey

The **A880_PCD_StoreKey** writes the internal non-volatile key storage of the contactless reader chip with supplied key data, to be used in the Mifare Classic card log in operation.

int **A880_PCD_StoreKey**(

int *hpcd*,
int *KeyIndex,*
const uchar **KeyContent*

);

Parameters

hpcd

[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

KeyIndex

[in] Specifies the index of the key storage. It ranges from 0x00 to 0x20.

DataContent

[in] Specifies a pointer to the 6 bytes key data to be written into the contactless reader chip.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_PICC.h
Library: Use libA880_CL.a

See Also

A880_PCD_ReadEEPROM
A880_PCD_WriteEEPROM

3.7.6. A880_PCD_ReadRegister

The **A880_PCD_ReadRegister** reads data from the internal registers of the contactless reader chip.

int **A880_PCD_ReadRegister**(

int *hpcd*,
int *addr,*
uchar **DataContent*

);

Parameters

hpcd
[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

addr
[in] Specifies the register address of the contactless reader chip to read.

DataContent
[out] Specifies a pointer to the location where the 1 byte requested data wll be stored.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_PICC.h
Library: Use libA880_CL.a

See Also

A880_PCD_WriteRegister

3.7.7. A880_PCD_WriteRegister

The **A880_PCD_WriteRegister** writes the internal registers of the contactless reader chip with data.

int **A880_PCD_WriteRegister**(

int *hpcd,*
int *addr,*
int *DataContent*

);

Parameters

hpcd
[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

addr
[in] Specifies the register address of the contactless reader chip to write through the operation.

DataContent
[in] Specifies the register value to be set

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_PICC.h
Library: Use libA880_CL.a

See Also

A880_PCD_ReadRegister

3.7.8. A880_PCD_RFPower

The **A880_PCD_RFPower** controls the RF power emission of the contactless reader chip.

int **A880_PCD_RFPower**(

int *hpcd*,
int *control,*

);

Parameters

hpcd
[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

control

[in] Specifies the new RF power status: zero to turn RF off and non-zero to turn RF on.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_PICC.h
Library: Use libA880_CL.a

3.7.1 A880_PICC_TxDataTelegram

The **A880_PICC_TxDataTelegram** transfers a data frame to the ISO 14443 compliant card and collects the response from the card.

int **A880_PICC_TxDataTelegram**(

int *hpcd*,
int *ComType*,
int *XmtLength*,
int *BitLength,*
int *ParityControl,*
int *CRCControl,*
int *CryptoControl,*
const uchar **TxDataContent*
uchar **pRcvLength,*
uchar **RxDataContent*

);

Parameters

hpcd
[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

ComType
[in] Specifies the communication type to transfer and receive data.

Value	Meaning
A880_PCD_ComType_None (0x00)	Don’t Specify, use default
A880_PCD_ComType_106A (0x01)	Type A, 106kbps baud rate
A880_PCD_ComType_212A (0x03)	Type A, 212kbps baud rate
A880_PCD_ComType_424A (0x05)	Type A, 424kbps baud rate
A880_PCD_ComType_848A (0x07)	Type A , 848kbps baud rate

XmtLength

[in] Specifies the length of the whole data frame in bytes. Incomplete octet will be treated as a complete byte.

BitLength

[in] Specifies the number of bits in the last octet. The value will be zero for a complete octet.

ParityControl

[in] Specifies the parity setting to be adopted throughout the PCD - PICC communication.

Value	Parity setting
0x00	Parity disabled
0x01	Parity disabled
0x02	Parity enabled, parity even
0x03	Parity enabled, parity odd

CRCControl

[in] Specifies the CRC formula to be adopted throughout the PCD - PICC communication

Value	CRC setting
0xX0	CRC transmission disabled
0xX1	CRC transmission disabled
0xX2	CRC transmission enabled, use CRC16
0xX3	CRC transmission disabled, use CRC-CCITT

Value	CRC setting
0x0X	CRC reception disabled
0x1X	CRC reception disabled
0x2X	CRC reception enabled, use CRC16
0x3X	CRC reception disabled, use CRC-CCITT

* The initial value of the CRC register should be set separately through the **A880_PCD_WriteRegister** command.

CryptoControl

[in] Specifies if the internal Mifare Classic crypto unit inside the contactless card reader chip is to be enabled after logging in.

TxDataContent

[in] Specifies a pointer to the location which the data will be sent to the card.

pRcvLength

[out] Specifies a pointer to the location which stores the length

of the data frame received.

RxDataContent
[out] Specifies a pointer to the location where the data responded will be stored.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_PICC.h
Library: Use libA880_CL.a

3.7.9. A880_PICC_List

The **A880_PICC_List** obtains card information of all cards in the field.

int **A880_PICC_List**(

int *hpcd,*
int **picc_nr,*
A880_PICC **picc,*
int *CardScope*

);

Parameters

hpcd
[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

picc_nr
[in/out] Specifies the maximum number of card information to return on input. The number of card in the field is returned on output.

picc
[out] Specifies a pointer to the array of A880_PICC structures for the returned card information. (see A880_PICC)

CardScope
[in] Specifies the scope of the detection, either type A only, type B only or not specific

The value of the scope could be

0x00	All types of cards would be detected
0x01	Only type A cards will be detected
0x02	Only type B cards will be detected

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Remarks

The definition of **A880_PICC** is:

```
typedef struct {

        int     hpcd;
        uchar   Type;    /* Card Type */
        uchar   SNByteNr /* Number of Byte of Card SN */
        uchar   SN[10];/* Card Serial number, MSB First*/
        uchar   CID;     /* Card ID, for internal use
        uchar   BlkNo;/* Block number, for internal use
        uchar   ATS[8];/*Extra information, for internal use */

} A880_PICC;
```

Requirements

Header: Declared in A880_PICC.h
Library: Use libA880_CL.a

See Also

A880_PICC_Sel

3.7.10. A880_PICC_SelAny

The **A880_PICC_SelAny** selects one of the cards (PICC) in the field for further operation.

int **A880_PICC_SelAny**(

int *hpcd*,
A880_PICC **picc*

);

Parameters

hpcd
[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

picc
[out] Specifies a pointer to the A880_PICC structure for the returned card information. (see A880_PICC)

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_PICC.h
Library: Use libA880_CL.a

See Also

A880_PICC_SelSpecific
A880_PICC_LIST

3.7.11. A880_PICC_SelSpecific

The **A880_PICC_SelSpecific** selects a specific card from a pile of cards (PICC) in the field for further operation.

```
int A880_PICC_SelSpecific(

        const A880_PICC *pPicc

);
```

Parameters

pPicc
[in] Specifies a pointer to the A880_PICC structure which contains the card details for card selection.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_PICC.h
Library: Use libA880_CL.a

See Also

A880_PICC_SelAny
A880_PICC_LIST

3.7.12. Mifare Classic

3.7.12.1. A880_MF_Login

The **A880_MF_Login** selects a sector to login for Mifare Classic specific operations.

```
int A880_MF_Login(

        int hpcd,
        A880_PICC *p_picc,
        int sector,
        int keyType,
        int KeyIndex,
        const uchar *KeyContent

);
```

Parameters

hpcd
[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

p_picc

[in] Specifies a pointer to the A880_PICC structure of the card to be logged into.

sector

[in] Specifies the sector number for logging in.

keyType

[in] Specifies the key source used for logging in the selected Mifare Classic card (PICC).

Value	Meaning
0x00	Default key A: A0A1A2A3A4A5
0x01	Default key B: B0B1B2B3B4B5
0x02	Default Transport Key: FFFFFFFFFFFF
0x03	Stored Key, log in as Key A
0x04	Stored Key, log in as Key B
0x05	Supplied Key, log in as Key A
0x06	Supplied Key, log in as Key B

keyIndex

[in] Specifies the index to the key location within the contactless card reader chip for card login when keyType is 0x03 or 0x04.

keyContent

[in] Specifies a pointer to the 6 bytes long array of containing the key content used for logging in.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_MF.h
Library: Use libA880_CL.a

3.7.12.2. A880_MF_Read

The **A880_MF_Read** reads a block of the logged in Mifare Classic card sector.

int **A880_MF_Read** (

int *hpcd,*
int *block,*
uchar **DataContent*

);

Parameters

hpcd

[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

block

[in] Specifies the absolute block number for reading.

DataContent

[out] Specifies a pointer to a 16-byte array where the read data will be stored.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_MF.h
Library: Use libA880_CL.a

See Also

A880_MF_ReadVal
A880_MF_Write
A880_MF_WriteVal
A880_MF_IncVal
A880_MF_DecVal
A880_MF_CopyVal

3.7.12.3. A880_MF_ReadVal

The **A880_MF_ReadVal** reads the value stored in a block of the logged in MifareClassic card sector in Mifare Value Block Format.

int **A880_MF_ReadVal** (

int *hpcd,*
int *block,*
long **Value*

);

Parameters

hpcd

[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

block

[in] Specifies the absolute block number for reading through the operation.

Value

[out] Specifies a pointer to the location where the read value is stored.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_MF.h
Library: Use libA880_CL.a

See Also

A880_MF_Read
A880_MF_Write
A880_MF_WriteVal
A880_MF_IncVal
A880_MF_DecVal
A880_MF_CopyVal

3.7.12.4. A880_MF_Write

The **A880_MF_Write** writes a block in the logged in Mifare Classic card sector.

int **A880_MF_Write** (

int *hpcd*,
int *block,*
const uchar **DataContent*

);

Parameters

hpcd
[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

block
[in] Specifies the absolute block number for writing.

DataContent
[in] Specifies a pointer to a 16-byte array where the data will be written to the card.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_MF.h
Library: Use libA880_CL.a

See Also

A880_MF_Read
A880_MF_ReadVal
A880_MF_WriteVal
A880_MF_IncVal
A880_MF_DecVal
A880_MF_CopyVal

3.7.12.5. A880_MF_WriteVal

The **A880_MF_WriteVal** writes a block of the logged in Mifare Classic card sector in Mifare Value Block Format.

int **A880_MF_WriteVal** (

int *hpcd,*
int *block,*
const long **Value*

);

Parameters

hpcd
[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

block
[in] Specifies the absolute block number for writing.

Value
[in] Specifies a pointer to the location of the value to be written to the card.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_MF.h
Library: Use libA880_CL.a

See Also

A880_MF_Read
A880_MF_ReadVal
A880_MF_Write
A880_MF_IncVal
A880_MF_DecVal
A880_MF_CopyVal

3.7.12.6. A880_MF_IncVal

The **A880_MF_IncVal** increments a block of the logged in Mifare Classic card sector in Mifare Value Block Format by a specific value.

int **A880_MF_IncVal** (

int *hpcd,*
int *block,*
long *Value,*
long **pNewValue*

);

Parameters

hpcd
[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

block
[in] Specifies the absolute block number for the increment operation.

Value
[in] Specifies the value to be incremented on the card.

pNewValue
[out] Specifies a pointer to a location of the resultant value.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Remarks

Example Code

Requirements

Header: Declared in A880_MF.h
Library: Use libA880_CL.a

See Also

A880_MF_Read
A880_MF_ReadVal
A880_MF_Write
A880_MF_WriteVal
A880_MF_DecVal
A880_MF_CopyVal

3.7.12.7. A880_MF_DecVal

The **A880_MF_DecVal** decrements a block of the logged in Mifare Classic card sector in Mifare Value Block Format by a specified value.

int **A880_MF_DecVal** (

int *hpcd,*
int *block,*
long *Value,*
long **pNewValue*

);

Parameters

hpcd
[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open.**

block

[in] Specifies the absolute block number for the decrement operation.

Value

[in] Specifies the value to be decremented on the card.

pNewValue

[out] Specifies a pointer to a location of the resultant value.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_MF.h
Library: Use libA880_CL.a

See Also

A880_MF_Read
A880_MF_ReadVal
A880_MF_Write
A880_MF_WriteVal
A880_MF_IncVal
A880_MF_CopyVal

3.7.12.8. A880_MF_CopyVal

The **A880_MF_CopyVal** copies one block of the logged in Mifare® Classic card sector in Mifare Value Block Format to another block in the same sector.

int **A880_MF_CopyVal** (

int *hpcd*,
int *sblock,*
int *tblock,*
long **pNewValue*

);

Parameters

hpcd

[in] Specifies the handle to the Contactless Card Reader returned by **A880_PCD_Open**

sblock

[in] Specifies the absolute block number of the source.

tblock

[in] Specifies the absolute block number of the destination.

pNewValue

[out] Specifies a pointer to the location of the resultant value after copying.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_MF.h
Library: Use libA880_CL.a

See Also

A880_MF_Read, A880_MF_ReadVal, A880_MF_Write, A880_MF_WriteVal, A880_MF_IncVal, A880_MF_DecVal

3.7.13. DesFire

3.7.13.1. A880_DF_Start

The **A880_DF_Start** initializes the resource for accessing a selected Mifare DesFire card. A previous call of **A880_PICC_Sel** is required.

int **A880_DF_Start** (

const A880_PICC **picc,*
uchar *Dr,*
uchar *Ds*

);

Parameters

picc
[in] Specifies a selected card.

Dr
[in] Specifies the desired baud rate for the direction from PCD to PICC.

Ds
[in] Specifies the desired baud rate for the direction from PICC to PCD.

Return Values

If the function succeeds, the Card ID is returned.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.2. A880_DF_End

The **A880_DF_End** releases the resource for accessing a selected Mifare DesFire card. A previous call of **A880_DF_Start** is required.

int **A880_DF_End** (

int *Cid*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.3. A880_DF_SelAID

The **A880_DF_SelAID** selects an application on the DesFire card. A previous call of **A880_DF_Start** is required.

int **A880_DF_SelAID** (

int *Cid,*
ulong *Aid*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Aid
[in] Specifies the ID of the application would like to select.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.4. A880_DF_Format

The **A880_DF_Format** releases all allocated user memory such that all files and applications on the DesFire card are deleted. A previous call of **A880_DF_Auth** with the card master key is required.

int **A880_DF_Format** (

int *Cid*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.5. A880_DF_MkAID

The **A880_DF_MkAID** creates a new application on the DesFire card. Depending on card master key setting, a previous call of **A880_DF_Auth** with the card master key may be required.

int **A880_DF_MkAID** (

int *Cid*,
ulong *Aid,*
uchar *KeySet,*
uchar *Nr_Key*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Aid
[in] Specifies the ID of the application to be created. The AID 0x000000 is reserved as a reference to the DesFire card itself and should not be used for creating an application.

KeySet
[in] Specifies the application master key setting.

Nr_Key
[in] Specifies the maximum number of keys stored within the application. The maximum allowable is 14 and all keys are initialized with sixteen 0x00 bytes.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.6. A880_DF_DelAID

The **A880_DF_DelAID** deletes an application on the DesFire card. Depending on card master key setting, a previous call of **A880_DF_Auth** with the card master key may be required.

int **A880_DF_DelAID** (

int *Cid,*
ulong *Aid*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Aid
[in] Specifies the ID of the application to be deleted. The AID 0x000000 is reserved as a reference to the DesFire card itself and should not be used for deleting an application.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.7. A880_DF_Auth

The **A880_DF_Auth** authenticates the access to the selected application.

int **A880_DF_Auth** (

int *Cid,*
uchar *KeyNo*,
const uchar **KeyDat*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

KeyNo
[in] Specifies the key number that authenticate as.

KeyDat
[in] Specifies the 16-byte key data used for authentication.

Return Values

If the function succeeds, the return value is 0.

If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.8. A880_DF_GetAID

The **A880_DF_GetAID** returns the ID of all application on the DesFire card. Depending on master key setting, a previous call of **A880_DF_Auth** with the card master key may be required.

int **A880_DF_GetAID** (

int *Cid,*
uchar **Nr_Aid,*
ulong **AidTbl*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Nr_Aid
[out] Specifies the pointer the number of AID returned.

AidTbl
[out] Specifies the table that stores the AID returned.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.9. A880_DF_GetFID

The **A880_DF_GetFID** returns the ID of all files of the selected application on the DesFire card. Depending on application master key setting, a previous call of **A880_DF_Auth** with the application master key may be required.

int **A880_DF_GetFID** (

int *Cid,*
uchar **Nr_Fid,*
uchar **FidTbl*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Nr_Fid
[out] Specifies the pointer to the number of FID returned.

FidTbl
[out] Specifies the table to store the FID returned.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.10. A880_DF_MkStd

The **A880_DF_MkStd** creates an unformatted data file under the current selected application. Depending on application master key setting, a previous call of **A880_DF_Auth** with the application master key may be required.

int **A880_DF_MkStd** (

int *Cid,*
uchar *Fid,*
uchar *ComSet,*
uchar *RKeyNo*,
uchar *WKeyNo*,
uchar *RWKeyNo*,
uchar *CfgKeyNo*,
ulong *FSize*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Fid
[in] Specifies the ID of the file to be created.

ComSet
[in] Specifies the communication setting to access the file.

RKeyNo
[in] Specifies the access right/key number for read access of the file.

WKeyNo
[in] Specifies the access right/key number for write access of the file.

RWKeyNo
[in] Specifies the access right/key number for read-write access of the file.

CfgKeyNo
[in] Specifies the access right/key number for making change of access right of the file.

FSize

[in] Specifies the desired file size.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.11. A880_DF_MkBak

The **A880_DF_MkBak** creates an unformatted data file with integrated backup mechanism under the current selected application. Depending on the application master key setting, a previous call of **A880_DF_Auth** with the application master key may be required.

int **A880_DF_MkBak** (

int *Cid,*
uchar *Fid,*
uchar *ComSet*,
uchar *RKeyNo*,
uchar *WKeyNo*,
uchar *RWKeyNo*,
uchar *CfgKeyNo*,
ulong *FSize*

);

Parameters

Cid

[in] Specifies the card ID returned by A880_DF_Start.

Fid

[in] Specifies the ID of the file to be created.

ComSet

[in] Specifies the communication setting on access the file.

RKeyNo

[in] Specifies the access right/key number for read access of the file.

WKeyNo

[in] Specifies the access right/key number for write access of the file.

RWKeyNo

[in] Specifies the access right/key number for read-write access of the file.

CfgKeyNo

[in] Specifies the access right/key number for making changes to the access right of the file.

FSize

[in] Specifies the desired file size .

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.12. A880_DF_MkVal

The **A880_DF_MkVal** creates a file for storage and manipulation of 32-bit signed integer under the current selected application. Depending on the application master key setting, a previous call of **A880_DF_Auth** with the application master key may be required.

int **A880_DF_MkVal**(

int *Cid*,
uchar *Fid*,
uchar *ComSet*,
uchar *RKeyNo*,
uchar *WKeyNo*,
uchar *RWKeyNo*,
uchar *CfgKeyNo*,
long *LLim*,
long *ULim*,
long *Value*,
uchar *LimCreditEn*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Fid
[in] Specifies the ID of the file to be created.

ComSet
[in] Specifies the communication setting to access the file.

RKeyNo
[in] Specifies the access right/key number used for read access of the file.

WKeyNo
[in] Specifies the access right/key number used for write access of the file.

RWKeyNo
[in] Specifies the access right/key number used for read-write access of the file.

CfgKeyNo
[in] Specifies the access right/key number used for making changes to the access right of the file.

LLim
[in] Specifies the lower limit for debit operation.

ULim
[in] Specifies the upper limit for credit operation.

Value
[in] Specifies the initial value of the file.

LimCredit
[in] Specifies limited-credit feature enable if set to 0x01 and disable if set to 0x00.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.13. A880_DF_MkLRec

The **A880_DF_MkLRec** creates a linear record file for the storage of data record. If the file is full, new records cannot be added unless the file is cleared. Depending on the application master key setting, a previous call of **A880_DF_Auth** with the application master key may be required.

```
int A880_DF_MkLRec(

        int Cid,
        uchar Fid,
        uchar ComSet,
        uchar RKeyNo,
        uchar WKeyNo,
        uchar RWKeyNo,
        uchar CfgKeyNo,
        ulong Rsize,
        ulong Nr_Rec
);
```

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Fid
[in] Specifies the ID of the file to be created.

ComSet
[in] Specifies the communication setting to access the file.

RKeyNo
[in] Specifies the access right/key number used for read access of the file.

WKeyNo
[in] Specifies the access right/key number used for write access of the file.

RWKeyNo
[in] Specifies the access right/key number used for read-write access of the file.

CfgKeyNo
[in] Specifies the access right/key number used for making changes to the access right of the file.

RSize
[in] Specifies the record size of each record.

Nr_Rec
[in] Specifies the maximum number of record that can be stored in the file.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.14. A880_DF_MkCRec

The **A880_DF_MkCRec** creates a cyclic record file for the storage of data record. If the file is full, a new record will overwrite the oldest record. Depending on the application master key setting, a previous call of **A880_DF_Auth** with the application master key may be required.

int **A880_DF_MkCRec**(

int *Cid,*
uchar *Fid*,
uchar *ComSet*,
uchar *RKeyNo*,
uchar *WKeyNo*,
uchar *RWKeyNo*,
uchar *CfgKeyNo*,
ulong *Rsize*,
ulong *Nr_Rec*

);
Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Fid
[in] Specifies the ID of the file to be created.

ComSet
[in] Specifies the communication setting to access the file.

RKeyNo
[in] Specifies the access right/key number for read access of the file.

WKeyNo

[in] Specifies the access right/key number for write access of the file.

RWKeyNo
[in] Specifies the access right/key number for read-write access of the file.

CfgKeyNo
[in] Specifies the access right/key number for making changes to the access right of the file.

RSize
[in] Specifies the record size of each record.

Nr_Rec
[in] Specifies the maximum number of records that can be stored in the file.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.15. A880_DF_Del

The **A880_DF_Del** deletes a file in the current selected application. Depending on the application master key setting, a previous call of **A880_DF_Auth** with the application master key may be required.

int **A880_DF_Del**(

int *Cid*,
uchar *Fid*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Fid
[in] Specifies the ID of the file to be deleted.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.16. A880_DF_Read

The **A880_DF_Read** reads data from standard data file or backup data file. A previous call of **A880_DF_Auth** with the key specified for “Read” or “Read&Write” is required.

int **A880_DF_Read**(

int *Cid*,
uchar *Fid*,
uchar *ComSet*,
ulong *Ofs*,
ulong *Len*,
void **Data*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Fid
[in] Specifies the ID of the file to be accessed.

ComSet
[in] Specifies the communication setting to access the file.

Ofs
[in] Specifies the starting position for the read operation.

Len
[in] Specifies the number of byte to be read. If 0, data will read to end of file.

Data
[out] Specifies the pointer to the buffer for data return.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.17. A880_DF_Write

The **A880_DF_Write** writes data to Standard data file or Backup data file. A previous call of **A880_DF_Auth** with the key specified for “Write” or “Read&Write” is required.

int **A880_DF_Write**(

int *Cid*,
uchar *Fid*,
uchar *ComSet*,
ulong *Ofs*,
ulong *Len*,
const void **Data*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Fid
[in] Specifies the ID of the file to be accessed.

ComSet
[in] Specifies the communication setting to access the file.

Ofs
[in] Specifies the starting position of the write operation.

Len
[in] Specifies the number of byte to be written. Zero is not allowed.

Data
[in] Specifies the data buffer to be written.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.18. A880_DF_GetVal

The **A880_DF_GetVal** reads the stored value from Value files. A previous call of **A880_DF_Auth** with the key specified for “Read”, “Write” or “Read&Write” is required.

int **A880_DF_GetVal**(

int *Cid*,
uchar *Fid*,
uchar *ComSet*,
long **Value*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Fid
[in] Specifies the ID of the file to be accessed.

ComSet
[in] Specifies the communication setting to access the file.

Value

[in] Specifies the buffer to store the returned value.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.19. A880_DF_Credit

The **A880_DF_Credit** increases the value stored in a Value files. A previous call of **A880_DF_Auth** with the key specified for "Read&Write" is required.

int **A880_DF_Credit**(

int *Cid*,
uchar *Fid*,
uchar *ComSet*,
long *Amount*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Fid
[in] Specifies the ID of the file to be accessed.

ComSet
[in] Specifies the communication setting to access the file.

Amount
[in] Specifies the amount to be credited.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.20. A880_DF_Debit

The **A880_DF_Debit** decreases the value stored in a Value files. A previous call of **A880_DF_Auth** with the key specified for "Read", "Write" or "Read&Write" is required.

int **A880_DF_Debit**(

int *Cid,*
uchar *Fid*,
uchar *ComSet*,
long *Amount*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Fid
[in] Specifies the ID of the file to be accessed.

ComSet
[in] Specifies the communication setting to access the file.

Amount
[in] Specifies the amount to be debited.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.21. A880_DF_LimCredit

The **A880_DF_LimCredit** increases the value stored in a Value file. The increase is limited by the sum of the Debit commands on this value file within the most recent transaction containing at least one Debit. The limit reset to 0 after this command. A previous call of **A880_DF_Auth** with the key specified for "Write" or "Read&Write" is required.

int **A880_DF_LimCredit**(

int *Cid*,
uchar *Fid*,
uchar *ComSet*,
long *Amount*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Fid
[in] Specifies the ID of the file to be created.

ComSet
[in] Specifies the communication setting used to access the file.

Amount
[in] Specifies the amount to be credited.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.22. A880_DF_WrRec

The **A880_DF_WrRec** adds a new record to a record file. A previous call of **A880_DF_Auth** with the key specified for "Write" or "Read&Write" is required.

int **A880_DF_WrRec**(

int *Cid*,
uchar *Fid*,
uchar *ComSet*,
ulong *Ofs*,
ulong *Len*,
const void **Data*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Fid
[in] Specifies the ID of the file to be accessed.

ComSet
[in] Specifies the communication setting to access the file.

Ofs
[in] Specifies the starting position in a record for the write operation.

Len
[in] Specifies the number of byte to be written to the record. Zero is not allowed.

Data
[in] Specifies the data to be written to the record.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.23. A880_DF_ReRec

The **A880_DF_ReRec** reads records from a record file. A previous call of **A880_DF_Auth** with the key specified for "Read" or "Read&Write" is required.

int **A880_DF_ReRec**(

int *Cid*,
uchar *Fid*,
uchar *ComSet*,
ulong *Ofs*,
ulong *LenRec*,
ulong *NumRec*,
void **Data*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

Fid
[in] Specifies the ID of the file to be accessed.

ComSet
[in] Specifies the communication setting to access the file.

Ofs
[in] Specifies the newest record to be read. Zero means the latest record in the record file.

LenRec
[in] Specifies the size of each record.

NumRec
[in] Specifies the number of records to be read counting from *Ofs* to the oldest entry in the record. Zero means read from *Ofs* to oldest record.

Data
[out] Specifies the pointer to the buffer that stores the returned records. Oldest entry goes first.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.24. A880_DF_ClrRec

The **A880_DF_ClrRec** resets the entire record file to empty state. A previous call of **A880_DF_Auth** with the key specified for “Read&Write” is required.

```
int A880_DF_ClrRec(

        int Cid,
        uchar Fid

);
```

Parameters

Cid
: [in] Specifies the card ID returned by A880_DF_Start.

Fid
: [in] Specifies the ID of the file to be accessed.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.25. A880_DF_ComTran

The **A880_DF_ComTran** validates all previous write access on Backup, Value, and records files within the current selected application.

```
int A880_DF_ComTran(

        int Cid

);
```

Parameters

Cid
: [in] Specifies the card ID returned by A880_DF_Start.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.26. A880_DF_AboTran

The **A880_DF_AboTran** invalidates all previous write access on Backup, Value, and records files within the current selected application.

int **A880_DF_AboTran**(

int *Cid*

);

Parameters

Cid

[in] Specifies the card ID returned by A880_DF_Start.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.27. A880_DF_GetFInfo

The **A880_DF_GetFInfo** gets the properties of a specific file. Depending on the application master key setting, a previous call of **A880_DF_Auth** with the application master key may be required.

int **A880_DF_GetFInfo**(

int *Cid*,
uchar *Fid*,
A880_DF_FInfo **FInfo*

);

Parameters

Cid

[in] Specifies the card ID returned by A880_DF_Start.

Fid

[in] Specifies the ID of the File to be accessed.file to be accessed.

FInfo

[out] Specifies the pointer to structure to store the returned file information.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Remarks

The definition of **A880_DF_FInfo** is:

```
typedef struct {

        uchar FType,                /* File Type */
        uchar ComSet,               /* Communication Setting */
        uchar RKeyNo,               /* Access Right for "Read" */
        uchar WKeyNo,               /* Access Right for "Write" */
        uchar RWKeyNo,              /* Access Right for "Read&Write" */
        uchar CfgKeyNo,             /* Access Right for change config */

        union {

                ulong FSize;        /* File size for standard/backup file */

                struct {

                        long LLim;      /* Lower Limit for Debit */
                        long ULim;      /* Upper Limit for Credit */
                        long LimCredit; /* Limited Credit Value, -1: disable */

                } Val;              /* For value file type */

                struct {

                ulong Size;         /* Size per each Record */
                ulong Max;          /* Maximum number of record */
                ulong Num;          /* Current number of record */

                } Rec;              /* For Record file type */

        } Prop;                     /* File properties for diff file type */

} A880_DF_FInfo
```

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.28. A880_DF_ChgFSet

The **A880_DF_ChgFSet** changes the access parameter of an existing file. Depending on the access right setting, a previous call of **A880_DF_Auth** with the change access right key may be required.

int **A880_DF_ChgFSet**(

int *Cid*,
uchar *Fid*,
uchar *Secu_En*,
uchar *ComSet*,
uchar *RKeyNo*,
uchar *WKeyNo*,
uchar *RWKeyNo*,
uchar *CfgKeyNo*

);

Parameters

Cid

[in] Specifies the card ID returned by A880_DF_Start.

Fid

[in] Specifies the ID of the File to be accessed.

Secu_En

[in] Specifies whether security mechanism for transfer is enabled.

ComSet

[in] Specifies the communication setting to access the file.

RKeyNo

[in] Specifies the access right/key number for read access of the file.

WKeyNo

[in] Specifies the access right/key number for write access of the file.

RWKeyNo

[in] Specifies the access right/key number for read-write access of the file.

CfgKeyNo

[in] Specifies the access right/key number for making changes to the access right of the file.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.29. A880_DF_GetVer

The **A880_DF_GetVer** returns manufacturing related data of the DesFire card.

int **A880_DF_GetVer**(

int *Cid*,
void **Data*

);

Parameters

Cid

[in] Specifies the card ID returned by A880_DF_Start.

Data

[out] Specifies the pointer to the 28-byte buffer to store the returned data.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

See Also

3.7.13.30. A880_DF_GetKeyVer

The **A880_DF_GetKeyVer** returns the key version of specify key stored in the DesFire card.

int **A880_DF_GetKeyVer**(

int *Cid*,
uchar *KeyNo*,
uchar **KeyVer*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

KeyNo
[in] Specifies the key number.

KeyVer
[out] Specifies the pointer to the buffer to store the returned key version.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.31. A880_DF_ChgKey

The **A880_DF_ChgKey** changes any key in current selected application in the DesFire card. A previous call of **A880_DF_Auth** with the ChangeKey key is required.

```
int A880_DF_ChgKey(

    int Cid,
    uchar KeyNo,
    const uchar *OldKey,
    const uchar *NewKey

);
```

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

KeyNo
[in] Specifies the key number.

OldKey
[in] Specifies the 16-byte old key data.

NewKey
[in] Specifies the 16-byte new key data.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.32. A880_DF_GetKeySet

The **A880_DF_GetKeySet** gets the key setting of the current selected application. Depending on the master key setting, a previous call of **A880_DF_Auth** with the master key may be required.

int **A880_DF_GetKeySet**(

int *Cid*,
uchar **KeySet*,
uchar **MaxKey*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

KeySet
[out] Specifies the buffer to store the returned key setting.

MaxKey
[out] Specifies the buffer to store the returned maximum number of keys in the selected application.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.7.13.33. A880_DF_ChgKeySet

The **A880_DF_ChgKeySet** changes the key setting for the current selected application. A previous call of **A880_DF_Auth** with the master key is required.

int **A880_DF_ChgKeySet**(

int *Cid*,
uchar *NewKeySet*

);

Parameters

Cid
[in] Specifies the card ID returned by A880_DF_Start.

NewKeySet
[in] Specifies the new master key setting.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is –1.

Requirements

Header: Declared in A880_DF.h
Library: Use libA880_CL.a

3.8. Terminal Status API

3.8.1. A880_WaitFor

The **A880_WaitFor** is used to wait for new external Event/Status.

int **A880_WaitFor** (

int *event,*
int *timeout*

);

Parameters

event

Event to wait for (see A880_EVENT_xxx)

timeout

Timeout to wait for. In unit of ms. Negative for no timeout

Return Values

If there is an error or timeout, with err number, the return value is 0.
If no error, the return value is the current event.

Remarks

A pre-existing event/status will not trigger the API return.

Requirements

Header: Declared in A880.h
Library: Use libA880.a

3.8.2. A880_ReadStatus

The **A880_ReadStatus** is used to read External Status.

int **A880_WaitFor** (

);

Parameters
None

Return Values

If the function succeeds, the return value is the status read.
If the function fails, the return value is -1 with err number

Requirements

Header: Declared in A880.h
Library: Use libA880.a

See Also
A880_STATUS_xxx

3.9. Power Management API

3.9.1. A880_PowerOff

The **A880_PowerOff** is used to wait for new external Event/Status.

```
int A880_PowerOff (


);
```

Parameter

None

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is not equal to 0

Remarks

A pre-existing event/status will not trigger the API return.

Header: Declared in A880.h
Library: Use libA880.a

3.10. Tamper-Resettable Memory API

3.10.1. A880_TRM_Read

The **A880_TRM_Read** is used to read the tamper-resettable memory.

int **A880_TRM_Read** (

int *addr,*
int *count,*
void *buf

);

Parameters

addr

It is the memory address to read from.

count

The number of bytes to read.

**buf*

This is the data read from the memory.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is not equal to 0

Requirements

Header: Declared in A880_TRM.h
Library: Use libA880_TRM.a.a

3.10.2. A880_TRM_Write

The **A880_TRM_Write** is used to write the tamper-resettable memory.

int **A880_TRM_Write** (

int *addr,*
int *count,*
void *buf

);

Parameters

addr

It is the memory address to read from.

count

The number of bytes to read.

**buf*

This is the data to be written on the memory.

Return Values

If the function succeeds, the return value is 0.
If the function fails, the return value is not equal to 0

Requirements

Header: Declared in A880_TRM.h
Library: Use libA880_TRM.a.a