Aidil Door Lock System Interface Function (SDK) Description

# (MAINDLL .DLL Function Description)

## Applicable door locking system: V6.51 and above. 1 . Door lock function (the door lock system must be installed and set up)

Is Init

Initialize. Function Form: int Init (int software, char \*server, char \*username, int port, int Encoder, int TMEncoder);

Parameter:

software [in]: Specify the door lock system, see Note 2. server[in]: A character pointer that specifies the name of the server (SQL Server) where the lock system database is installed. username [in]: Character pointer, username, used to record the operator of the door lock system. Port [in]: Serial port number, 0-USB, I.COM 1, 2-

COM2, 3-COM3, 4-COM4 and so on. Encoder[in]: Distributor Type, 0 - Manual Distributor, 1 - Automatic Distributor, 2 - MSR206 (Magnetic Card). TMEncoder [in]: TM Publisher Type, I-DS9097E,5-DS90971J Return Value: See Note 1.

## 2 s EndSession

End of the work period. Function Form: int EndSession(void); Parameters: No return value: See Note 1.

### ChangeUser

Change operators. Function Form:

void ChangeUser(char \*username);

username [in]: Character pointer, username, used to record the operator of the door lock system. Return value: None.

### 4 s NewKey

Issue a new guest card, this function cancels the current guest card, and the original guest card will not be usable after the newly issued guest card opens. Function Form:

### In Nuke (four \*room, four \*gate, four \*stime, four \*guestname, four \*guestname, int overflag, long \*cardno, four \* trackl, four \* track2);

Parameter:

room [in]: The room number, a 6-byte string, must be the room number set by the door lock system. gate [in]: public channel, string parameter, "00" means to authorize the channel by default, "99" means to authorize all public channels, and the other is the specified channel code. For example: "010203" means that three channels are authorized 01 , 02, and 03.

Stime [in]: Start and end time, 24-byte string, format yyyymmddhhn, for example:

"200012311230200101011230" means 12:30 December 31, 2000 to 12:30 January 1, 2001.

Lock9200: The start and end time cannot be within the same day and cannot exceed 1 month.

Guestname [in]: Guest name, up to 30 bytes, can be

NULL. Guestid [in]: Guest ID, up to 30 bytes, can be NULL.

Overfla [in]: Shaping, whether to overwrite what's on the current card. 1 - Override, the current card is not a new card and will be automatically canceled (must be a guest card). 0 - No coverage, not a new card will return. The magnetic card (manual issuer) ignores this parameter and overwrites it directly, and does not automatically log out.

Cardno [out]: Long shaped pointer that receives the card number (the unique card code assigned after the card is issued). can be NULL. trackl [in]: String pointer, magnetic card track 1 data, if this track is not written, it can be NULL. track2 [in]: String pointer, magnetic card track 2 data, if this track is not written, it can be NULL. Return value: See Note 1.

### DupKey

"Duplicate" the guest card, the new guest card can be used at the same time as the original guest card and the start time is the same. Function Form:

### Int Dupke (four \*room, four \*gate, four \*stime, four \*guestname, four \*guestname, int overflag, long \*cardno, four \* trackl, four \* track2);

Parameter:

room [in]: The room number, a 6-byte string, must be the room number set by the door lock system. gate [in]: public channel, string parameter, "00" means to authorize the channel by default, "99" means to authorize all public channels, and the other is the specified channel code. For example: "010203" means that three channels are authorized 01 , 02, and 03.

Stime [in]: Start and end time, 24-byte string, format yyyymmddhhn, for example:

"200012311230200101011230" means 12:30 December 31, 2000 to 12:30 January 1, 2001. The start time is the same as the active guest card.

Guestname [in]: Guest name, up to 30 bytes, can be NULL. Guestid [in]: Guest ID, up to 30 bytes, can be NULL. Overfla [in]: Shaping, whether to overwrite what's on the current card. 1 - Override, the current card is not a new card and will be automatically canceled (must be a guest card). 0 - No coverage, not a new card will return. The magnetic card (manual issuer) ignores this parameter and overwrites it directly, and does not automatically log out.

Cardno [out]: Long shaped pointer that receives the card number (the unique card code assigned after the card is issued). can be NULL. trackl [in]: String pointer, magnetic card track 1 data, if this track is not written, it can be NULL. track2 [in]: String pointer, magnetic card track 2 data, if this track is not written, it can be NULL. Return value: See Note 1.

6s ReadCard

Read the card. Function Form:

### Int readcard (four \*room, four \*gate, four \*stime, four \*guestname, four \*guestname, four \*trackl, four \*track2, long \*cardno, int \*st);

Parameter:

room [out]: A string pointer that receives the returned room number, suggested 10 bytes. gate [out]: A string pointer that receives the returned authorized public channel, which can be NULL. Guestname [out]: String pointer to receive the returned guest name, which can be NULL. Guestid [out]: A string pointer that receives the returned guest ID, which can be NULL. trackl [out]: Receives track 1 data from the magnetic card, which can be NULL. track2 [out]: Receives track 2 data from the magnetic card, which can be NULL. Cardno [out]: A long shaped pointer that receives the returned card number, which can be NULL. St [out]: Shape the pointer, receive the returned card status, 1 - normal use, 3 - normal logout, 4 - lost note pin, 5 - damage cancellation, 6 - automatic cancellation. can be NULL.

Return value: See Note 1.

## EraseCard

Log out of the card while updating the database data. Function Form: int EraseCard (long cardno,char \* trackl,char \* track2);

cardno [in]: The card number, which can be 0. When this parameter is O: IC card, RF card, TM card, magnetic card (automatic issuer) automatically read the card number and log out, and update the database at the same time; If the magnetic card (manual issuer) cancels the current card and does not update the database, it is recommended to call the CheckOut function to update the database, or read the card to obtain the card number before calling

trackl [in]: String pointer, magnetic card track 1 data, if this track is not written, it can be NULL. track2 [in]: String pointer, magnetic card track 2 data, if this track is not written, it can be NULL. Return value: See Note 1.

 CheckOut

Check out, update only database data, do not cancel the card. Function Form:

### int CheckOut (char \*room, long cardno);

room [in]: room number. Cardno [in]: Card number, which can be 0. When this parameter is 0, mark all guest cards in the room as normal logouts.

Return value: See Note 1.

## 9s LostCard

If you lose the logout, only the database data will be updated, and the card will not be canceled. Function Form:

### int Lostcard (char \*room, long cardno);

room [in]: room number. Cardno [in]: Card number, marked as the number of the lost guest card. Return value: See Note 1.

### 2. Common functions (door lock interface and independent use are acceptable) PopCard

Pop-up card. Function prototype: int PopCard(void); Parameters: No return value: See Note 1, returning 20 when used independently means 'no SetPort called'.

# 11 s SetPort

Set up a serial port, and you don't have to call the Init function after calling it. Function Form: int SetPort(int software, int port, int encoder, int tmencoder); software [in]: Specify the door lock system, see Note 2. Port [in]: serial slogan, shape, I-COMI, 2-COM2, 3-COM3, 4-COM4 and so on. Encoder[in]:

Distributor type: 0 - manual distributor, 1 - automatic distributor.

TMEncoder[in]: TM Distributor Type: I-DS9097E, 5-DS9097U. Return value:

See Note 1 .

## 12 s ReadCardId

Read the Mifare card ID, the function in its original form:

### int ReadCardId(unsigned long \*pid);

pid [out]: 4 bytes of unsigned pointer that receives the card ID.

Return value: See Note 1 

## 13s CheckSC (Lock3200K, A30

Reconcile the 4442 card password. Function Form: int CheckSC(unsigned char \*sc);

sc[in]: IC card password, 3-byte non-token pointer. Return value: See Note 1.

## 14s ReadlC (Lock3200K, Lock4200D, Lock7200D, A30, A90

Card reading data. Function Form:

int ReadlC(unsigned int start, unsigned int len, unsigned char

# \*str);

start[in]: The start address, specifying where the data is written. For Mifare cards (Ade19200)i the starting address consists of two bytes: the high byte is the block number and the low byte is the area code. For example, means reading 2 zones and I block.

Len[in]: The length of the data to be read. For Mifare cards (Ade19200) the length is an integer multiple of 16. str [in,out]: No symbolic character pointer, receives the read data. For the Mifare card (A90), the authorization password that is passed into the read area as an input parameter at the same time consists of 7 bytes: the first byte is the authorization code: if it is 0, the A password must be checked, if it is 1, the B password must be checked, and the last 6 bytes are the password. Return value: See Note 1.

## 15s WritelC (Lock3200K, Lock4200D, Lock7200D, Lock9200Ts A30, A90

Write data to the IC card. (Lock3200K: Recommended user data stored in bytes Ox80 and later;

Lock9200T: It is recommended that user data be stored at Ox18 and later 8 bytes). Function Form:

int WriteIC(unsigned int start, unsigned int len, unsigned char

# \*str);

start [in]: the start address, specifying the location where the data is written; For Mifare cards (Ade19200), the starting address consists of two bytes: the high byte is the block number and the low byte is the area code. For example, Ox0102 means reading 2 zones and block.

Len [in]: The length of the data written. For Mifare cards (Ade19200) the length is an integer multiple of 16. Str [in]: No symbolic character pointer to write data to the IC card. For the authorization password of the first 7 bytes of the Mifare card (A90) into the read area: the first byte is the authorization code: if it is O, the A password must be checked, if it is 1, the B password must be checked, and the last 6 bytes are the password. Data written from the 8th byte.

Return value: See Note 1 

## 16s WriteMagCard

Write magnetic card, it is recommended that user data be stored in rail 1 and 2, and the door lock system uses track 3. Function Form:

int WriteMagCard(char \*trackl, char \* track2, char \* track3);

trackl [in]: String pointer, magnetic card track 1 data, if this track is not written, it can be NULL. track2 [in]: String pointer, magnetic card track 2 data, if this track is not written, it can be NULL. track3 [in]: String pointer, magnetic card track 3

data, if this track is not written, it can be NULL. Return value: See Note 1.

## 17 ReadMagCard

Read the magnetic card. Function Form:

int ReadMagCard(char \* trackl, char \* track2, char \* track3); trackl [out]: A string pointer that receives data from track 1 of the magnetic card, if this track is not read, it is NULL.

track2 [out]: A string pointer that receives data from track 2 of the magnetic card, if this track is not read, it is NULL. track3 [out]: A string pointer that receives data from track 3 of the magnetic card, if this track is not read, it is NULL. Return value: See Note 1 

18 s Reader\_Beep Buzzer. Function Form:

int Reader\_Beep (int Sound);

Sound [in]: Specifies the sound and LED status. 11 - Green light for 1 second, long scream; 12-Red light for 1 second, long scream; 15-Red light for 1 second, short call twice; 16- A short scream.

Return value: See Note 1 

3. Network power-saving switch function 19s GetBuff1nfo Function Form:

## int GetBuffInfo (unsigned char \*buff, int len, int \*cardtype,int char \*roomno, char \*username, char \*stime);

Parameter:

buff [in]: No symbolic character pointer, stuck on data. len [in]: data length in bytes. cardtype[out]: integer pointer, receiving card type: I-president card, 2-villa card, 3management card, 4-general control card, 5-clock card, 6-foreman card, 7-floor card, 8cleaning card, 9-guest card, 10-termination card, Il-conference card, 12-emergency card, 13-check-out card, 14-spare card; can be NULL.

cardst[out]: integer pointer, receiving card status: I-normal, 3-logged out, 4-lost, 5damaged, 6-expired; can be NULL.

cardno[out]: integer pointer, receive card number, can be NULL. roomno[out]: Character pointer, receive guest card/backup card room number, foreman card/floor card/cleaning card area. It is no less than 20 bytes and can be NULL.

username[out]: The character pointer that receives the name of the guest, which is not less than 20 bytes and can be NULL. Return value: See Note 1.

stime[out]: Character pointer, the validity period of the receiving guest/cleaning card, not less than 28 bytes, can be NULL. The format yyyymmddhhnnyyyymmddhhnn, for example:

"200012311230200101011230" indicates 12:30 on 31 December 2000 to 12:30 on 1 January 2001.

Return value: See Note 1. 20s GetMagBuffInfo Function Form:

int GetMagBuffInfo (char \*buff, int \*cardtype, int \*cardst,int \*cardno, char\*roomno, char \*username, char \*stime);

Parameter:

buff [in]: string pointer, magnetic card data. cardtype[out]: integer pointer, receiving card type: I-president card, 2-villa card, 3-management card, 4-general control card, 5-clock card, 6-foreman card, 7-floor card, 8-cleaning card, 9-guest card, 10-termination card, 11conference card, 12-emergency card, 13-check-out card, 14-spare card; can be NULL.

cardst[out]: integer pointer, receiving card status: I-normal, 3-logged out, 4-lost, 5damaged, 6-expired; can be NULL.

cardno[out]: integer pointer, receive card number, can be NULL. roomno[out]: Character pointer, receive guest card/backup card room number, foreman card/floor card/cleaning card area. It is no less than 20 bytes and can be NULL.

username[out]: The character pointer that receives the name of the guest, which is not less than 20 bytes and can be NULL. Return value: See Note 1.

stime[out]: Character pointer, the validity period of the receiving guest/cleaning card, not less than 28 bytes, can be NULL. The format yyyymmddhhnnyyyymmddhhnn, for example:

"200012311230200101011230" indicates 12:30 on 31 December 2000 to 12:30 on 1 January 2001.

Return value: See Note 1 

21s GetCardInfo Function Form:

## int GetCardInfo(int cardno, int cardtype,int \*cardst, char \*roomno, char\*username, char \*stime);

Parameter:

cardno [in]: card number. cardtype[in]: The card type. cardst[out]: integer pointer, receiving card status: I-normal, 3-logged out, 4-lost, 5-damaged, 6-expired; can be

NULL.

roomno[out]: Character pointer, receive guest card/backup card room number, foreman card/floor card/cleaning card area. It is no less than 20 bytes and can be NULL.

username[out]: The character pointer that receives the name of the guest, which is not less than 20 bytes and can be NULL. stime[out]: Character pointer, the validity period of the receiving guest/cleaning card, not less than 28 bytes, can be NULL. The format yyyymmddhhnnyyyymmddhhnn, for example: "200012311230200101011230" indicates 12:30 on 31 December 2000 to 12:30 on 1 January 2001.

Return value: See Note 1.

Note: 1. Function return value:

|  |  |
| --- | --- |
| 0 - Operation is successful | 1- Read and write errors/data errors |
| 2- The card is damaged | 3- Cardless |

4 - Serial port error 5 - Card replaced 6 - Not a new card 7 - Card is a new card 8 - Not a system card 9 - Not a guest card 10 - Not a membership card 11 - Password error 12 - No door opening record 13 - Incorrect card type 14 - Parameter error 15 - User cancel operation <ESC>(press Key) 16 - Wait timeout 17 - Card insertion error 18 - Card is blank or card insertion error 19 - Reserved (for forward compatibility) 20 - Init function is not called 21 - The version of the door lock software is not supported 22 - Connection (door lock system database) error 23 - Door lock system parameters do not exist 24 Initialization failed 25 - No guest is staying/designated guest does not exist 26 - The room is full 27 - There is no record of this card 28 - SetPort is not called Function 29 - Invalid room number 30 - Incorrect time range 31 - Card number already exists and cannot be registered (Lock9200)

32- Calls are not supported 33- Invalid authorization code, wrong or expired authorization code

1. Door lock system constant:

3104200D-410c52ee-510k62ee6107200-7107200D-810c92ee-910k9200T-10

A3Ø-11 A5e-14 A9e-18 A92-22

1. Operating environment: 32-bit Windows operating system, MDAC needs to be installed to use the door lock function.
2. Calling method: Follow the Windows API calling standard, that is, stdcall.
3. Parameter description: [in] is the input parameter, [out] is the output parameter.