B&W Series
Communication Protocol SDK
Development Handbook

Date: July, 2012

If there is some change of information in this handbook,Granding won't inform specially.

- GRANDING, is our company logo. Biokey is core technology logo of Granding. Both logos have been registered in China and America.
- Other trademarks and product names mentioned in this handbook are produced by other companies. Granding has no property of them.
- Please solve use problems (instead of development kit problem) during development by yourself.
- Granding has no responsibility for data loss caused by users or programs.
- This handbook is designed on the basis of SDK 6.2.4.1.

Contents

1 SDK Description

The offline communication SDK is an interface for data communication with offline fingerprint devices, access control devices, and RFID card devices. It can be used to conveniently manage user information and fingerprints, download attendance records, operation records, user information, and fingerprint templates, set devices, and configure access control. The SDK is used to:

1. Download attendance records.
2. Upload and download user information, card information, and fingerprints information.
3. Set access control rules of access control devices.
4. Set device time, match thresholds, etc.
5. Trigger various events of devices in real time, for example, fingerprint verification.
6. Directly enroll users online.
7. Set SMS and work code (available only on devices that support this function) of users.
8. Set personalized prompt tones, function keys, etc.

2 Quick Start

2.1 Terms

1. Real-time event

   After the SDK and the device communicate with each other successfully, some operations on the device (for example, connecting to the device, verifying a user, and enrolling a user)

trigger corresponding events in real time, and data is transmitted to the PC (host computer). The triggered events are called real-time events. Users can monitor device states and user operations in real time through real-time events.

2. FP

   Shortened form of "fingerprint".

3. Fingerprint algorithm

   A fingerprint algorithm refers to the algorithm used to generate and verify fingerprint templates. At present, Finger 9.0 is the latest fingerprint algorithm used by Granding black & white devices.

4. High-speed buffer

   A high-speed buffer refers to the memory requested by the SDK on a PC during usage. In the data upload or download process, data is first saved in the buffer before being processed.

5. Time slot, group, open door combination

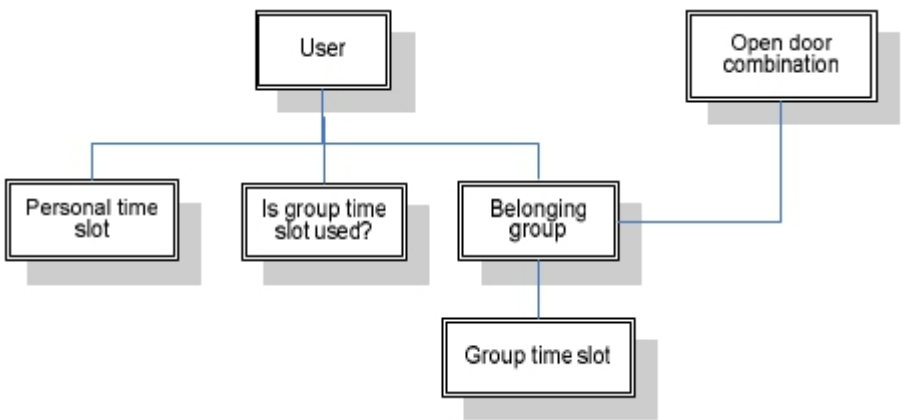
   These three terms are the most important concepts of access control.

   A time slot is a time range. A time slot includes the time information of one week, and a time range is specified for each day of this week. For example, the following expression indicates a time range from 00:00 to 22:11 in each day of one week: 00002211000022110000221100002211000022110000221100002211. Generally, 50 time slots can be set in the device.

   A group is a collection. When many users have the same access control privileges, these users can be added to the same group and use the group time slot. Then, time slots can be set for the group.

   An open door combination refers to the groups that are required for unlock. If the open door combination contains only one group, it indicates that the door is opened when any of the users in this group passes verification. If the open door combination contains two or more groups, the door is opened only after all groups pass verification. For example, an open door combination contains groups A and B, the door is opened only after a member of group A and a member of group B pass verification.

   The following figure shows the relationship of the three concepts:



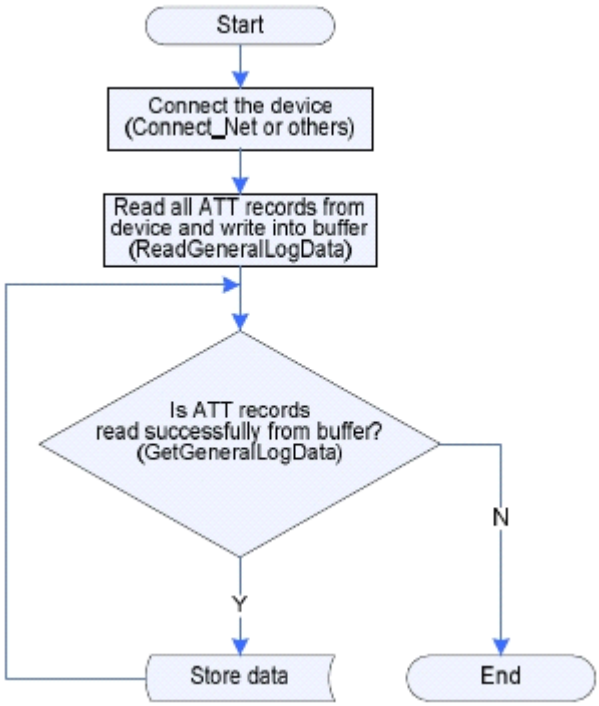
6. Operation record

   An operation record, also called management record, is a record generated when users or administrators operate on the device, for example, powering on/off the device and enrolling a user.

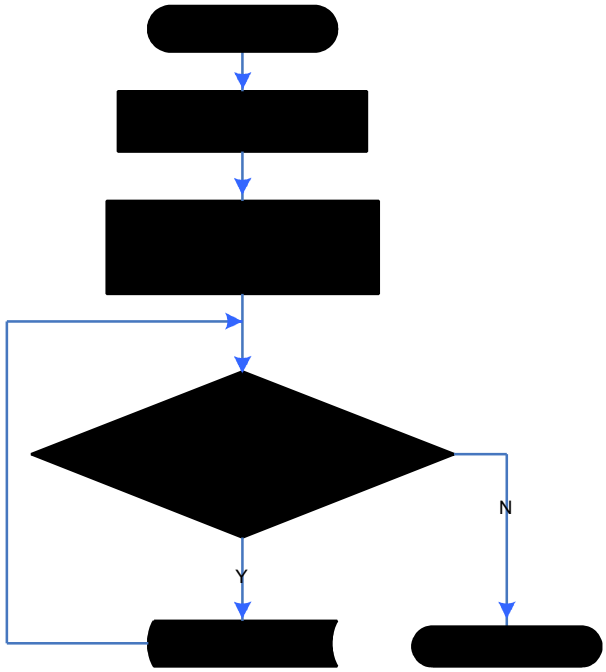
2.2 Common Processes

For details, see the descriptions of the demo program.

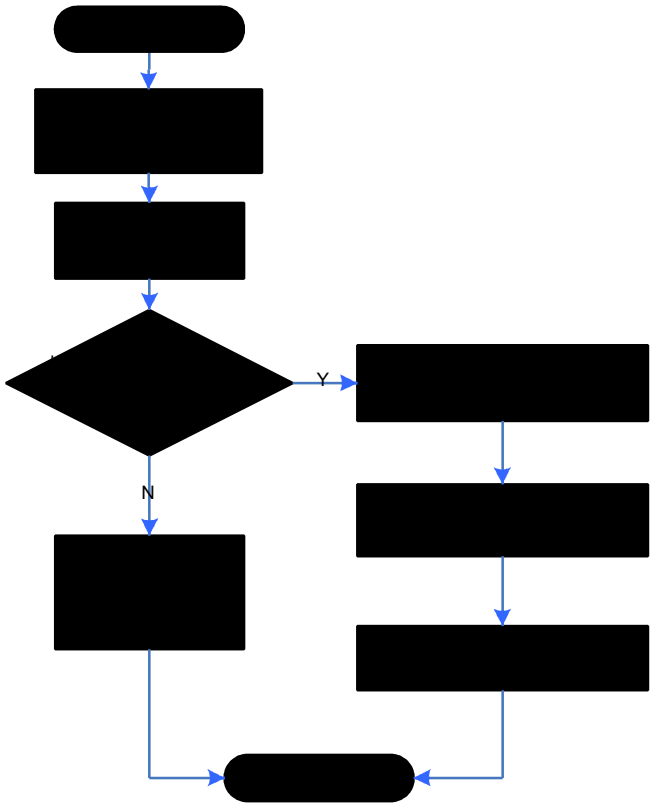
2.2.1 Downloading Attendance Records



2.2.2 Downloading Operation Records

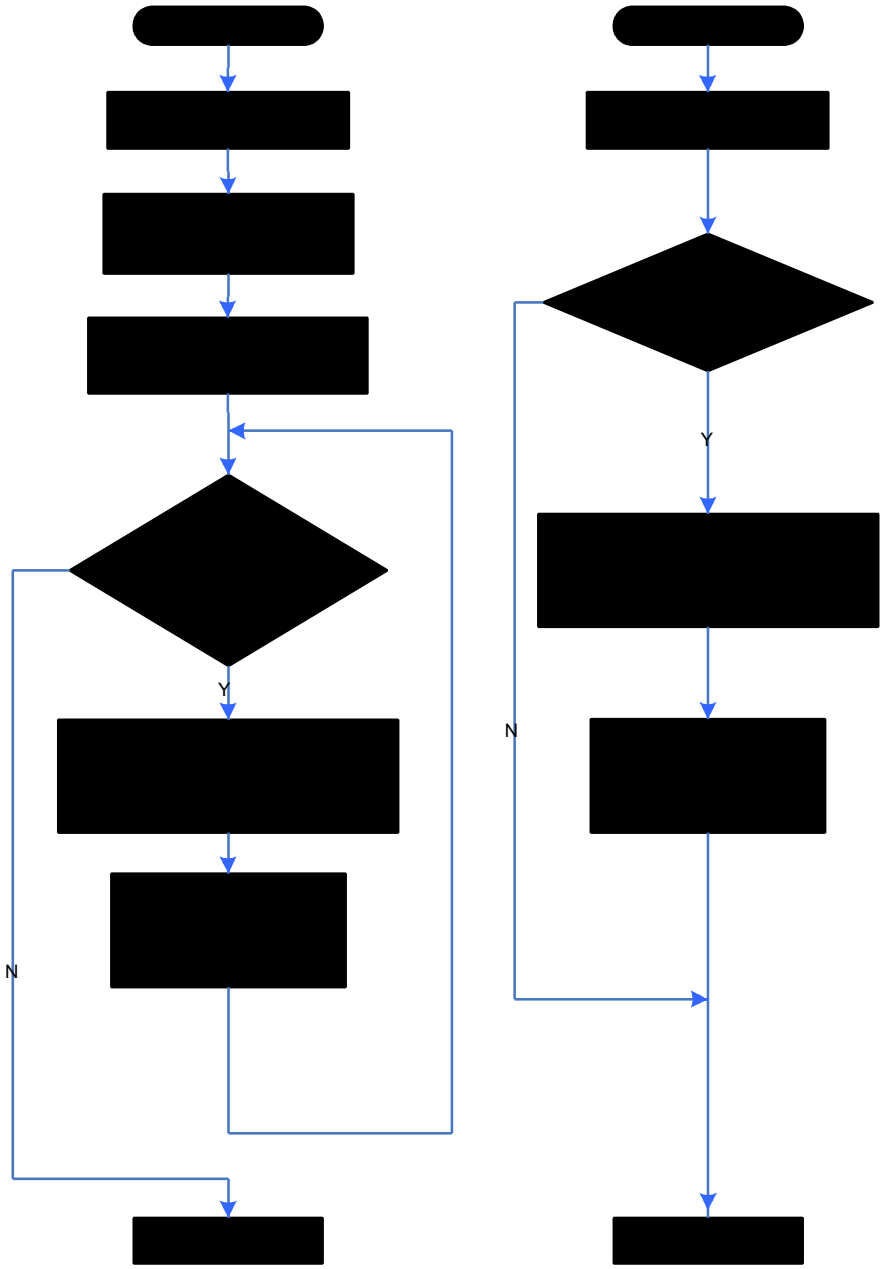


2.2.3 Setting Access Control



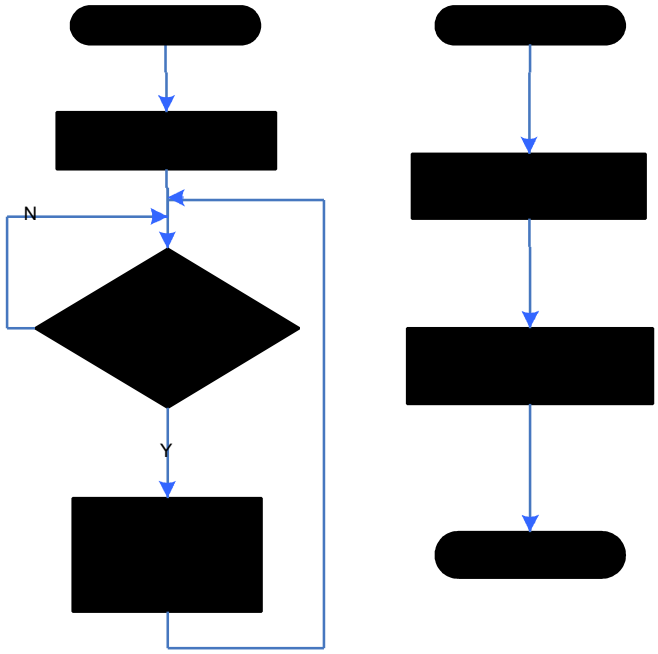
2.2.4 Downloading User Information, Fingerprint Templates, or Face Templates

The left diagram shows the process of downloading the information of all users. The right diagram shows the process of downloading the information of a specified user.



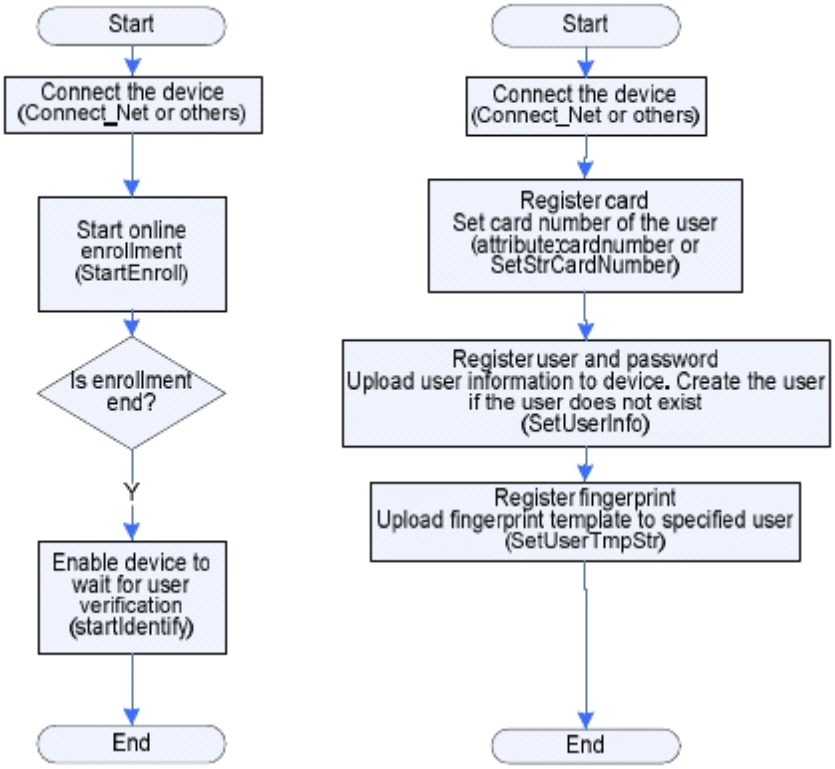
2.2.5 Receiving Real-time Events

Real-time events can be received in two modes. The second mode is recommended.

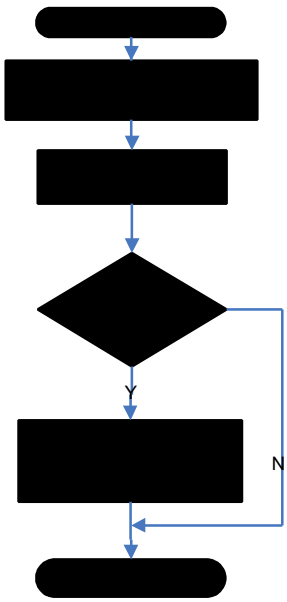


2.2.6 Enrolling Users Online (Uploading Information, and Fingerprint Templates of Users)

There are two online user enrollment modes. The left diagram shows the process in which the device accesses the enrollment interface to enroll a user after being connected. The right diagram shows the process of creating a user on the device and uploading the card number, password, and fingerprint information for the user (that is, enrolling a card user, a password user, and a fingerprint user).



2.2.7 Uploading Short Messages



3 Related Attributes

3.1 AccGroup

Usage: Set or obtain the group to which a user belongs.

If this attribute is set before user information is uploaded, the user group is set when SetUserInfo is used to upload user information. Otherwise, AccGroup is 1 by default.

Type: LONG, writable

3.2 AccTimeZones

Usage: Set the time slot of a user.

If this attribute is set before user information is uploaded, the time slot of a user is set when SetUserInfo is used to upload user information.

Type: LONG *. It can be regarded as a long one-dimensional array with subscript 3. Readable/writable

3.3 BASE64

Usage: When the value is set to True, the SDK outputs the character string template in base64 codes. Otherwise, the SDK outputs the character string template in hexadecimal codes.

Type: LONG, readable/writable

3.4 CardNumber

Usage: Set or read the card number of a user. If this attribute is unavailable, use GetStrCardnumber and SetStrCardnumber instead.

Type: LONG, readable/writable

3.5 CommPort

Usage: Set the number of the serial port or the port for RS485 communication.

Type: LONG, readable/writable

3.6 ConvertBIG5

Usage: When the value is set to True, the SDK automatically converts the font from simplified Chinese to traditional Chinese for offline development. This function is invalid for series products of Multilanguage versions. Do not set this function in this case.

Type: LONG, readable/writable

Caution: This attribute is invalid for Multilanguage versions. In addition, you do not need to modify this attribute for ZEM100 5.22, ZEM200 5.30, and later versions.

3.7 PINWidth

Usage: Indicate the maximum length of the user ID (Arabic numeral).

Type: LONG, read only

3.8 GetStrCardNumber

[Definition]

VARIANT_BOOL GetStrCardNumber([out] BSTR* ACardNumber)

[Usage]

Obtain the value of cardnumber of the SDK. Generally, this function can be used to obtain the card number of a user immediately after the user information is obtained.

[Parameter]

AcardNumber

Card number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetUserInfo

3.9 SetStrCardNumber

[Definition]

VARIANT_BOOL SetStrCardNumber([in] BSTR ACardNumber)

[Usage]

Set the value of cardnumber of the SDK. Generally, this function can be used to set the card number of a user before the user information is set.

[Parameter]

AcardNumber

Card number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetUserInfo

4 Real-time Event Functions

4.1 Obtaining Real-Time Events

4.1.1 RegEvent

[Definition]

VARIANT_BOOL RegEvent([in] LONG dwMachineNumber, [in] LONG EventMask)

[Usage]

Register required real-time events.

[Parameter]

dwMachineNumber:

Device number

EventMask:

Code of an event. Values are as follows:

1	OnAttTransaction, OnAttTransactionEx
2 (1<<1)	OnFinger
4 (1<<2)	OnNewUser
8 (1<<3)	OnEnrollFinger
16 (1<<4)	OnKeyPress
256 (1<<7)	OnVerify
512 (1<<8)	OnFingerFeature
1024 (1<<9)	OnDoor, OnAlarm
2048 (1<<10)	OnHIDNum
4096 (1<<11)	OnWriteCard
8192 (1<<12)	OnEmptyCard
16384 (1<<13)	OnDeleteTemplate

To register multiple real-time events, perform the XOR operation between binary codes of related events. For example, to register all real-time events, the value of EventMask is 65535.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ReadRTLog, GetRTLog

4.1.2 ReadRTLog

[Definition]

VARIANT_BOOL ReadRTLog([in] LONG dwMachineNumber)

[Usage]

Read real-time events and write them to the buffer of the PC. This function can be used with GetRTLog to actively obtain real-time events from the device after the PC connects to the device successfully.

[Parameter]

dwMachineNumber:

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetRTLog

4.1.3 GetRTLog

[Definition]

VARIANT_BOOL GetRTLog(LONG dwMachineNumber)

[Usage]

Obtain an event from the buffer of the PC and trigger the event. This function can be used with ReadRTLog to actively obtain real-time events from the device after the PC connects to the device successfully.

[Parameter]

dwMachineNumber:

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ReadRTLog

4.2 Real-Time Events

4.2.1 OnConnected

This event is triggered when the PC connects to the device successfully. No value is returned.

4.2.2 OnDisConnected

This event is triggered when the PC disconnects from the device successfully. No value is returned.

4.2.3 OnAlarm

OnAlarm (LONG AlarmType,LONG EnrollNumber,LONG Verified)

This event is triggered when the device reports an alarm.

[Return Value]

Alarm Type: Type of an alarm.	55: Tamper alarm. 58: False alarm. 32: Threatened alarm. 34: Anti-pass back alarm.
EnrollNumber: User ID.	The value is 0 when a tamper alarm, false alarm, or threatened alarm is given. The value is the user ID when other threatened alarm or anti-pass back alarm is given.
Verified: Whether to verify	The value is 0 when a tamper alarm, false alarm, or threatened alarm is given. The value is 1 when other alarms are given.

4.2.4 OnDoor

OnDoor (LONG EventType)

This event is triggered when the device opens the door.

[Return Value]

EventType: Open door type

4: The door is not closed. 53: Exit button. 5: The door is closed. 1: The door is opened unexpectedly.

4.2.5 OnAttTransaction

OnAttTransaction (LONG EnrollNumber,LONG IsInValid,LONG AttState,LONG VerifyMethod,LONG Year,LONG Month,LONG Day,LONG Hour,LONG Minute,LONG Second)

This event is triggered after verification succeeds.

[Return Value]

EnrollNumber: UserID of a user

IsInValid: Whether a record is valid. 1: Not valid. 0: Valid

AttState: Attendance state. 0—Check-In (default value) 1—Check-Out 2—Break-Out 3—Break-In 4—OT-In 5—OT-Out

VerifyMethod: Verification mode. Generally, 0: password verification, 1: fingerprint verification, 2: card verification.

In multi-verification mode:

FP_OR_PW_OR_RF	0
FP	1
PIN	2
PW	3

RF	4
FP_OR_PW	5
FP_OR_RF	6
PW_OR_RF	7
PIN_AND_FP	8
FP_AND_PW	9
FP_AND_RF	10
PW_AND_RF	11
FP_AND_PW_AND_RF	12
PIN_AND_FP_AND_PW	13
FP_AND_RF_OR_PIN	14

Year/Month/Day/Hour/Minute/ Second indicates the time when verification succeeds.

4.2.6 OnAttTransactionEx

OnAttTransactionEx (BSTR EnrollNumber,LONG IsInValid,LONG AttState, LONG VerifyMethod,LONG Year, LONG Month, LONG Day, LONG Hour, LONG Minute, LONG Second, LONG WorkCode)

This event is triggered after verification succeeds.

[Return Value]

EnrollNumber: UserID of a user.

IsInValid: Whether a record is valid. 1: Not valid. 0: Valid.

AttState: Attendance state. 0—Check-In (default value) 1—Check-Out 2—Break-Out 3—Break-In 4—OT-In 5—OT-Out

VerifyMethod: Verification mode. Generally, 0: password verification, 1: fingerprint verification, 2: card verification.

In multi-verification mode:

FP_OR_PW_OR_RF	0
FP	1
PIN	2
PW	3
RF	4
FP_OR_PW	5
FP_OR_RF	6
PW_OR_RF	7
PIN_AND_FP	8
FP_AND_PW	9
FP_AND_RF	10

PW_AND_RF	11
FP_AND_PW_AND_RF	12
PIN_AND_FP_AND_PW	13
FP_AND_RF_OR_PIN	14

Year/Month/Day/Hour/Minute/ Second indicates the time when verification succeeds.

WorkCode: work code returned during verification. Return 0 when the device does not support work code.

4.2.7 OnDeleteTemplate

OnDeleteTemplate (LONG EnrollNumber, LONG FingerIndex)

This event is triggered when FP templates on the device are deleted.

[Return Value]

EnrollNumber: ID of the user whose FP template is deleted.

FingerIndex: Index of the deleted FP template.

4.2.8 OnEnrollFinger

OnEnrollFinger (LONG EnrollNumber, LONG FingerIndex, LONG ActionResult, LONG TemplateLength)

This event is triggered when a fingerprint is registered.

[Return Value]

EnrollNumber: User ID of the fingerprint being registered

FingerIndex: Index of the current fingerprint

ActionResult: Operation result. Return 0 if the operation is successful, or return a value greater than 0.

TemplateLength: Length of the fingerprint template

4.2.9 OnFinger

This event is triggered when a fingerprint is placed on the fingerprint sensor of the device. No value is returned.

4.2.10 OnFingerFeature

OnFingerFeature (LONG Score)

This event is triggered when a user places a finger and the device registers the fingerprint.

[Return Value]

Score: Quality score of a fingerprint

4.2.11 OnHIDNum

OnHIDNum (LONG CardNumber)

This event is triggered when a card is swiped.

[Return Value]

CardNumber: Number of a card that can be an ID card or an HID card. If the card is a Mifare card, the event is triggered only when the card is used as an ID card.

4.2.12 OnKeyPress

OnKeyPress (LONG Key)

This event is triggered when a key is pressed.

[Return Value]
Key: The key being pressed

[Supported Device**]**

Only black & white devices support this function.

4.2.13 OnNewUser

OnNewUser (LONG EnrollNumber)

This event is triggered when a new user is successfully enrolled.

[Return Value]

EnrollNumber: UserID of the newly enrolled user.

4.2.14 OnVerify

OnVerify (LONG UserID)

This event is triggered when a user is verified.

[Return Value]

When verification succeeds, UserID indicates the ID of the user. Return the card number if card verification is successful, or return -1.

4.2.15 OnWriteCard

OnWriteCard (LONG EnrollNumber, LONG ActionResult, LONG Length)

This event is triggered when the device writes a card.

[Return Value]

EnrollNumber: ID of the user to be written into a card

ActionResult: Result of writing user information into a card. 0: Success. Other values: Failure.

Length: Size of the data to be written into a card

4.2.16 OnEmptyCard

OnEmptyCard (LONG ActionResult)

This event is triggered when a Mifare card is emptied.

[Return Value]

ActionResult: Result of emptying a card. 0: Success. Other values: Failure.

4.2.17 OnEMData

OnEMData (LONG DataType, LONG DataLen, CHAR* DataBuffer)

This event is triggered when the device sends an unknown event to SDK.

[Return Value]

DataType: Type of the returned event

DataLen: Data length

DataBuffer: Actual data

5 Common Functions

5.1 Device Connection Functions

5.1.1 Connect_Net

[Definition]

VARIANT_BOOL Connect_Net([in] BSTR IPAdd, [in] long Portl)

[Usage]

Connect to the device via the IP address and set up a network connection with the device.

[Parameter]

IPAdd:

IP address of the device

Port:

Port number used for connecting to the device. The default value is 4370.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

Disconnect, Connect_Com, Connect_USB

5.1.2 Connect_Com

[Definition]

VARIANT_BOOL Connect_Com([in] long ComPort, [in] long MachineNumber, [in] long BaudRate)

[Usage]

Connect to the device via a serial port, that is, via RS232 or RS485 port.

Note: This function can be also used for some devices that use USB Client to communicate with the PC. However, the USB client driver must be first installed to simulate a serial port. After the installation succeeds, you can view the serial port number via the device manager on the PC or find the virtual serial port number via the program. For details, see "USBClient" of the demo program.

[Parameter]

ComPort:

Serial port number of the PC for connecting to the device

MachineNumber:

Device number

BaudRate:

Baud rate

[Return Value]

Return True if it is successful, or return False.

[Related Function]

Disconnect, Connect_Net, Connect_USB

5.1.3 Connect_USB

[Definition]

VARIANT_BOOL Connect_USB([in] long MachineNumber)

[Usage]

Connect to the device via USB ports. This function is used for communicating with H-series devices via USB ports instead of USB client. For communication via USB client, see Connect_Com.

[Parameter]

MachineNumber:

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

Disconnect, Connect_Net, Connect_Com

5.1.4 Disconnect

[Definition]

Disconnect(void)

[Usage]

Disconnect from the device and release related resources.

[Parameter]

None

[Return Value]

None

[Related Function]

Connect_Net, Connect_Com, Connect_USB

5.2 Data Management Functions

5.2.1 Attendance Record Data

5.2.1.1 ReadGeneralLogData

[Definition]

VARIANT_BOOL ReadGeneralLogData([in] long dwMachineNumber)

[Usage]

Read attendance records and write them into the internal buffer of the PC. This function is the same as ReadAllGLogData.

[Parameter]

dwMachineNumber:

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ReadAllGLogData, GetAllGLogData, GetGeneralLogData, GetGeneralLogDataStr, ClearGLog, GetGeneralExtLogData

5.2.1.2 ReadAllGLogData

[Definition]

VARIANT_BOOL ReadAllGLogData ([in] long dwMachineNumber)

[Usage]

Read attendance records and write them into the internal buffer of the PC. This function is the same as ReadGeneralLogData.

[Parameter]

dwMachineNumber:

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ReadGeneralLogData, GetAllGLogData, GetGeneralLogData, GetGeneralLogDataStr, ClearGLog, GetGeneralExtLogData

5.2.1.3 GetGeneralLogData

[Definition]

VARIANT_BOOL GetGeneralLogData([in] long dwMachineNumber,[out] long* dwTMachineNumber, [out] long* dwEnrollNumber, [out] long* dwEMachineNumber, [out] long* dwVerifyMode, [out] long* dwInOutMode, [out] long* dwYear, [out] long* dwMonth, [out] long* dwDay, [out] long* dwHour, [out] long* dwMinute)

[Usage]

Read attendance records one by one from the internal buffer. Before using this function, you can use ReadAllGLogData or ReadGeneralLogData to read attendance records from the device and write them into the internal buffer of the PC. Each time this function is executed, the pointer points to the next attendance record. This function is the same as GetAllGLogData except that interface names are different to achieve compatibility. This function is used on only black & white devices.

[Parameter]

dwMachineNumber

Device number

dwTMachineNumber

Pointer pointing to a long variable. The value is the device number of the received attendance record.

dwEnrollNumber

Pointer pointing to a long variable. The value is the user ID of the received attendance record.

dwEMachineNumber

Pointer pointing to a long variable. The value is the device number of the received attendance record.

dwVerifyMode

Pointer pointing to a long variable. The value is the verification mode of the received attendance record. The specific values are as follows:

Generally, 0: password verification, 1: fingerprint verification, 2: card verification.

In multi-verification mode:

FP_OR_PW_OR_RF 0

FP	1
PIN	2
PW	3
RF	4
FP_OR_PW	5
FP_OR_RF	6
PW_OR_RF	7
PIN_AND_FP	8
FP_AND_PW	9
FP_AND_RF	10
PW_AND_RF	11
FP_AND_PW_AND_RF	12
PIN_AND_FP_AND_PW	13
FP_AND_RF_OR_PIN	14

dwInOutMode

Pointer pointing to a long variable. The value is the attendance state of the received record. The specific values are as follows:

0—Check-In (default value) 1—Check-Out 2—Break-Out

3—Break-In 4—OT-In 5—OT-Out

dwYear, dwMonth, dwDay, dwHour, dwMinute

Pointers pointing to long variables. The values are the date and time of the received attendance record.

[Return Value]

Return 1 if it is successful, return 0 if all records are read, or return a negative number if an error occurs.

[Related Function]

GetAllGLogData, GetGeneralLogDataStr

5.2.1.4 GetAllGLogData

[Definition]

VARIANT_BOOL GetAllGLogData ([in]longdwMachineNumber,[out] long* dwTMachineNumber, [out] long* dwEnrollNumber, [out] long* dwEMachineNumber, [out] long* dwVerifyMode, [out] long* dwInOutMode, [out] long* dwYear, [out] long* dwMonth, [out] long* dwDay, [out] long* dwHour, [out] long* dwMinute)

[Usage]

Read attendance records one by one from the internal buffer. Before using this function, you can use ReadAllGLogData or ReadGeneralLogData to read attendance records from the device and write them into the internal buffer of the PC. Each time this function is executed,

the pointer points to the next attendance record. This function is the same as GetGeneralLogData except that interface names are different to achieve compatibility.

[Parameter]

Same as GetGeneralLogData

[Return Value]

Return 1 if it is successful, return 0 if all records are read, or return a negative number if an error occurs.

[Related Function]

GetGeneralLogData, GetGeneralLogDataStr

5.2.1.5 GetGeneralLogDataStr

[Definition]

VARIANT_BOOL GetGeneralLogDataStr([in] long dwMachineNumber, [out] long* dwEnrollNumber, [out] long* dwVerifyMode, [out] long* dwInOutMode, [out] BSTR *TimeStr)

[Usage]

Read attendance records one by one from the internal buffer. Before using this function, you can use ReadAllGLogData or ReadGeneralLogData to read attendance records from the device and write them into the internal buffer of the PC. Each time this function is executed, the pointer points to the next attendance record. The difference between this function and GetGeneralLogData is the returned time type.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

Pointer pointing to a long variable. The value is the user ID of the received attendance record.

dwVerifyMode

Pointer pointing to a long variable. The value is the verification mode of the received attendance record. The specific values are the same as those of GetGeneralLogData.

dwInOutMode

Pointer pointing to a long variable. The value is the attendance state of the reserved attendance record. The specific values are the same as those of GetGeneralLogData.

TimeStr

Pointer pointing to a BSTR variable. The value is the attendance time of the received attendance record.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetGeneralLogData

5.2.1.6 GetGeneralExtLogData

[Definition]

VARIANT_BOOL GetGeneralExtLogData([in] LONG dwMachineNumber, [out] LONG* dwEnrollNumber, [out] LONG* dwVerifyMode, [out] LONG* dwInOutMode, [out] LONG* dwYear, [out] LONG* dwMonth, [out] LONG* dwDay, [out] LONG* dwHour, [out] LONG* dwMinute, [out] LONG* dwSecond, [out] LONG* dwWorkCode, [out] LONG* dwReserved)

[Usage]

Read attendance records one by one from the internal buffer. Before using this function, you can use ReadAllGLogData or ReadGeneralLogData to read attendance records from the device and write them into the internal buffer of the PC. Each time this function is executed, the pointer points to the next attendance record. This function is an enhancement of GetGeneralLogData and is compatible with GetGeneralLogData.

[Parameter]

wWorkCode

Pointer pointing to a long variable. The value is the work code of the received attendance record.

dwReserved

Reserved parameter without specific meanings.

For other parameters, see GetGeneralLogData.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetGeneralLogData

5.2.1.7 ClearGLog

[Definition]

VARIANT_BOOL ClearGLog([in] long dwMachineNumber)

[Usage]

Clear all attendance records from the device.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ClearSLog, ClearKeeperData

5.2.2 Operation Record Data

5.2.2.1 ReadSuperLogData

[Definition]

VARIANT_BOOL ReadSuperLogData([in]long dwMachineNumber)

[Usage]

Read operation records and write them into the internal buffer of the PC. This function is the same as ReadAllSLogData.

[Parameter]

dwMachineNumber:

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ReadAllSLogData, GetAllSLogData, GetSuperLogData, ClearSLog

5.2.2.2 ReadAllSLogData

[Definition]

VARIANT_BOOL ReadAllSLogData([in] long dwMachineNumber)

[Usage]

Read operation records and write them into the internal buffer of the PC. This function is the same as ReadSuperLogData.

[Parameter]

dwMachineNumber:

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ReadSuperLogData, GetAllSLogData, GetSuperLogData, ClearSLog

5.2.2.3 GetSuperLogData

[Definition]

VARIANT_BOOL GetSuperLogData([in] long dwMachineNumber, [out] long* dwTMachineNumber, [out] long* dwSEnrollNumber, [out] long* Params4, [out] long* Params1, [out] long* Params2, [out] long* dwManipulation, [out] long* Params3, [out] long* dwYear, [out] long* dwMonth, [out] long* dwDay, [out] long* dwHour, [out] long* dwMinute)

[Usage]

Read operation records one by one from the internal buffer. Before using this function, you can use ReadAllSLogData or ReadSuperLogData to read operation records from the device and write them into the internal buffer of the PC. Each time this function is executed, the pointer points to the next operation record. This function is the similar to GetSuperLogData2 except that GetSuperLogData2 can be used to obtain more accurate operation record time (in seconds).

[Parameter]

dwMachineNumber

Device number

dwTMachineNumber

Pointer pointing to a long variable. The value is the device number of the received operation record.

dwSEnrollNumber

Pointer pointing to a long variable. The value is the administrator ID of the received operation record.

Params4

Pointer pointing to a long variable. The value varies with dwManipulation.

Params1

Pointer pointing to along variable. The value varies with dwManipulation.

Params2

Pointer pointing to a long variable. The value varies with dwManipulation.

dwManipulation

Pointer pointing to a long variable. The value is the operation type. The specific values are as follows:

Value of dwManipulation	Meaning of dwManipulation	Params1	Params2	Params3	Params4
0	Power on				
1	Power off				
3	Alarm	Alarm type. 58: False alarm. 54: Door entry alarm, 53: Exit button alarm, 55: Tamper alarm, 65535: Alarm off			
4	Enter menu				

Value of dwManipulation	Meaning of dwManipulation	Params1	Params2	Params3	Params4
5	Change setting	Number of the set option			
6	Register a fingerprint	ID of the operated user	Operation result. 0: Success. Other values: Failure.	Index of the registered fingerprint	Length of the fingerprint template (2: Threatened fingerprint)
7	Register a password				
8	Register an ID card				
9	Delete a user			Index of the fingerprint	
10	Delete a fingerprint			Index of the fingerprint	
11	Delete a password				
12	Delete an ID card				
13	Clear data			Operation result	
14	Create an MF card	ID of the operated user		Number of fingerprints written into the MF card	Size of fingerprint data written into the MF card
15	Register an MF card			Number of fingerprints written into the MF card	
16	Register an MF card				
17	Unregister an MF card				

Value of dwManipulation	Meaning of dwManipulation	Params1	Params2	Params3	Params4
18	Clear MF card data				
19	Move registration data from the device to the MF card	ID of the operated user	Operation result	Number of fingerprints written into the MF card	
20	Copy data from the MF card to the device			Number of fingerprints read out from the MF card	
21	Setting time	Year	Month	Day	Hour*Minute
22	Restore factory setting				
23	Delete all attendance records from the device		Operation result		
24	Clear administrator privileges				
25	Modify access control group settings	Group number		Number of group time slot 1	Number of group time slot 2
26	Modify access control settings of a user	Internal user ID		0: The group number of the user is modified. 1: The option of using group time slot is modified. 3: Other settings are modified.	When Params3 is 0, the value indicates the group number.

Value of dwManipulation	Meaning of dwManipulation	Params1	Params2	Params3	Params4
27	Modify access control time slot	Time slot flag			
28	Modify open door combination				
29	Open door operation				
30	Enroll a new user	ID of the operated user	Operation result		
31	Change fingerprint attribute	ID of the operated user	Operation result	Index of the fingerprint	Whether the fingerprint is a threatened fingerprint. 1: Non-threatened fingerprint. 2: Threatened fingerprint.
32	Threatened alarm	Whether to verify. 0: Key alarm. 1: Verify alarm	Note: When the value is Verify alarm, dwSEnrollNumber returns the ID of the threatened user.		
33	Verification failure	Verification mode. 20: password, 21: fingerprint, 22: ID card, 23: PIN, 24: Mifare card, 25: iclass card			
34	Anti-pass back	Whether the alarm is an anti-pass back alarm.			

Params3

Pointer pointing to a long variable. The value varies with dwManipulation.

dwYear, dwMonth, dwDay, dwHour, dwMinute

Pointers pointing to long variables. The values are the date and time of the received operation record.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetSuperLogData2, GetAllSLogData

5.2.2.4 GetAllSLogData

[Definition]

GetAllSLogData([in] long dwMachineNumber, [out] long* dwTMachineNumber, [out] long* dwSEnrollNumber, [out] long* dwSMachineNumber, [out] long* dwGEnrollNumber, [out] long* dwGMachineNumber, [out] long* dwManipulation, [out] long* dwBackupNumber, [out] long* dwYear, [out] long* dwMonth, [out] long* dwDay, [out] long* dwHour, [out] long* dwMinute)

[Usage]

Read operation records one by one from the internal buffer. Before using this function, you can use ReadAllSLogData or ReadSuperLogData to read operation records from the device and write them into the internal buffer of the PC. Each time this function is executed, the pointer points to the next operation record. This function is the same as GetSuperLogData except that interface names are different to achieve compatibility.

[Parameter]

Same as GetSuperLogData

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetSuperLogData

5.2.2.5 ClearSLog

[Definition]

VARIANT_BOOL ClearSLog([in] long dwMachineNumber)

[Usage]

Clear all operation records from the device.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ClearGLog, ClearKeeperData

5.2.2.6 GetSuperLogData2

[Definition]

VARIANT_BOOL GetSuperLogData2([in] LONG dwMachineNumber, [out] LONG* dwTMachineNumber, [out] LONG* dwSEnrollNumber, [out] LONG* Params4, [out] LONG*

Params1, [out] LONG* Params2, [out] LONG* dwManipulation, [out] LONG* Params3, [out] LONG* dwYear, [out] LONG* dwMonth, [out] LONG* dwDay, [out] LONG* dwHour,[out] LONG* dwMinute, [out] LONG* dwSecs)

[Usage]

Read operation records one by one from the internal buffer. Before using this function, you can use ReadAllSLogData or ReadSuperLogData to read operation records from the device and write them into the internal buffer of the PC. Each time this function is executed, the pointer points to the next operation record. This function is the similar to GetSuperLogData except that GetSuperLogData can be used to obtain more accurate operation record time (in seconds).

[Parameter]

dwYear, dwMonth, dwDay, dwHour, dwMinute, dwSecs

Pointers pointing to long variables. The values are the date and time of the received operation record.

For other parameters, see GetSuperLogData.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ReadAllSLogData, GetSuperLogData

5.2.3 User Information Functions

5.2.3.1 ReadAllUserID

[Definition]

VARIANT_BOOL ReadAllUserID([in] long dwMachineNumber)

[Usage]

Read all user information except fingerprint templates, such as user ID, password, name, and card number, and write the user information into the memory of the PC. After this function is executed, you can use GetAllUserID to take out user information.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetAllUserID

5.2.3.2 GetAllUserID

[Definition]

VARIANT_BOOL GetAllUserID([in] long dwMachineNumber,[out] long* dwEnrollNumber, [out] long* dwEMachineNumber, [out] long* dwBackupNumber, [out] long* dwMachinePrivilege, [out] long* dwEnable)

[Usage]

Obtain all user information. Before executing this function, you can use ReadAllUserID to read out all user information and write it into the memory. Each time GetAllUserID is executed, the pointer points to the next user information. After all user information is read, False is returned. The difference between this function and GetAllUserInfo is that GetAllUserInfo can obtain also user name and password.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

dwEMachineNumber

Invalid parameter. Return 0.

dwMachinePrivilege

User privilege. 0: common user, 1: enroller, 2: administrator, 3: super administrator

dwEnable

Whether the user is enabled. 1: Enabled. 0: Disabled.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ReadAllUserID, GetAllUserInfo

5.2.3.3 GetAllUserInfo

[Definition]

VARIANT_BOOL GetAllUserInfo([out] long dwMachineNumber, [out] long* dwEnrollNumber, [out] BSTR * Name, [out] BSTR * Password, [out] long * Privilege, [out] VARIANT_BOOL * Enabled)

[Usage]

Obtain all user information. Before executing this function, you can use ReadAllUserID to read out all user information and write it into the memory. Each time GetAllUserInfo is executed, the pointer points to the next user information. After all user information is read, False is returned. The difference between this function and GetAllUserID is that this function can obtain more information.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

Name

User name

Password

User password

Privilege

User privilege. 0: common user, 1: enroller, 2: administrator, 3: super administrator

dwEnable

Whether the user is enabled. 1: Enabled. 0: Disabled.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ReadAllUserID, GetAllUserID

5.2.3.4 EnableUser

[Definition]

VARIANT_BOOL EnableUser([in] long dwMachineNumber, [in] long dwEnrollNumber, [in] long dwEMachineNumber, [in] long dwBackupNumber, [in] VARIANT_BOOL bFlag)

[Usage]

Set whether a user is enabled.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

dwEMachineNumber, dwBackupNumber

Invalid parameter without specific meanings

bFlag

User enable flag. True: Enabled. False: Disabled.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.2.3.5 ModifyPrivilege

[Definition]

VARIANT_BOOL ModifyPrivilege([in] long dwMachineNumber, [in] long dwEnrollNumber, [in] long dwEMachineNumber, [in] long dwBackupNumber, [in] long dwMachinePrivilege)

[Usage]

Modify user privileges.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

dwEMachineNumber, dwBackupNumber

Invalid parameter without specific meanings

dwMachinePrivilege

User privilege. 0: common user, 1: enroller, 2: administrator, 3: super administrator

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetUserInfo, GetUserInfo

5.2.3.6 SetUserInfo

[Definition]

VARIANT_BOOL SetUserInfo([in] long dwMachineNumber, [in] long dwEnrollNumber, [in] BSTR Name, [in] BSTR Password, [in] long Privilege, [in] VARIANT_BOOL Enabled)

[Usage]

Set user information. If the user is unavailable, the device automatically creates the user.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

Name

User name to be set

Password

User password to be set. If the value is null, the user password on the device is cleared.

Privilege

User privilege. 0: common user, 1: enroller, 2: administrator, 3: super administrator

Enabled

User enable flag. 1: Enabled. 0: Disabled

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetUserInfo

5.2.3.7 GetUserInfo

[Definition]

VARIANT_BOOL GetUserInfo([in] long dwMachineNumber, [in] long dwEnrollNumber, [out] BSTR * Name, [out] BSTR * Password, [out] long * Privilege, [out] VARIANT_BOOL * Enabled)

[Usage]

Obtain the information of the specified user.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

Name

User name corresponding to the user ID described by dwEnrollNumber

Password

User password. If the value is null, the user does not use a password.

Privilege

User privilege. 0: common user, 1: enroller, 2: administrator, 3: super administrator

Enabled

User enable flag. 1: Enabled. 0: Disabled

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetUserInfo

5.2.3.8 SetUserInfoEx

[Definition]

VARIANT_BOOL SetUserInfoEx([in] LONG dwMachineNumber, [in] LONG dwEnrollNumber, [in] LONG VerifyStyle, [in] BYTE* Reserved)

[Usage]

Upload the user verification mode or group verification mode. Only the devices supporting multiple verification modes support this function.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

VerifyStyle

Verification mode
0 (group verification mode),
128(FP/PW/RF), 129(FP), 130(PIN), 131(PW), 132(RF), 133(FP&RF), 134(FP/PW), 135(FP/RF), 136(PW/RF), 137(PIN&FP), 138(FP&PW), 139(PW&RF), 140(FP&PW&RF), 141(PIN&FP&PW), 142(FP&RF/PIN).

Reserved

Reserved parameter temporarily without specific meanings.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetUserInfoEx, DeleteUserInfoEx

5.2.3.9 GetUserInfoEx

[Definition]

VARIANT_BOOL GetUserInfoEx([in] LONG dwMachineNumber, [in] LONG dwEnrollNumber, [out] LONG* VerifyStyle, [out] BYTE* Reserved)

[Usage]

Obtain the user verification mode. Only the devices support multiple verification modes support this function.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

VerifyStyle

Verification mode of the user described by dwEnrollNumber. The values are as Follows:
0 (group verification mode),
128(FP/PW/RF), 129(FP), 130(PIN), 131(PW), 132(RF), 133(FP&RF), 134(FP/PW), 135(FP/RF), 136(PW/RF), 137(PIN&FP), 138(FP&PW), 139(PW&RF), 140(FP&PW&RF), 141(PIN&FP&PW), 142(FP&RF/PIN).

Reserved

Reserved parameter without specific meanings.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetUserInfoEx, DeleteUserInfoEx

5.2.3.10 DeleteUserInfoEx

[Definition]

VARIANT_BOOL DeleteUserInfoEx([in] LONG dwMachineNumber, [in] LONG dwEnrollNumber)

[Usage]

Delete the multiple verification modes of the specified user. Only the devices supporting multiple verification modes support this function.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetUserInfoEx, GetUserInfoEx

5.2.3.11 GetUserInfoByPIN2

[Definition]

VARIANT_BOOL GetUserInfoByPIN2(([in] long dwMachineNumber, ([out] BSTR * Name, [out] BSTR * Password, [out] long * Privilege, [out] VARIANT_BOOL * Enabled)

[Usage]

Obtain user information via the current pin2 value.

[Parameter]

dwMachineNumber

Device number

Name

User name

Password

User password

Privilege

User privilege. 0: common user, 1: enroller, 2: administrator, 3: super administrator

Enabled

User enable flag. 1: Enabled. 0: Disabled

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetUserInfo

5.2.3.12 GetUserInfoByCard

[Definition]

VARIANT_BOOL GetUserInfoByCard([in] long dwMachineNumber, [out] BSTR * Name, [out] BSTR * Password, [out] long * Privilege, [out] VARIANT_BOOL * Enabled)

[Usage]

Obtain user information via the current CardNumber value.

[Parameter]

dwMachineNumber

Device number

Name

User name

Password

User password

Privilege

User privilege. 0: common user, 1: enroller, 2: administrator, 3: super administrator

Enabled

User enable flag. 1: Enabled. 0: Disabled

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetUserInfo

5.2.3.13 GetUserIDByPIN2

[Definition]

VARIANT_BOOL GetUserIDByPIN2([in] LONG PIN2, [out] LONG* UserID)

[Usage]

Obtain user ID via pin2.

[Parameter]

PIN2

Value of pin2

UserID

User ID

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetPIN2

5.2.3.14 GetPIN2

[Definition]

VARIANT_BOOL GetPIN2([in] LONG UserID, [out] LONG* PIN2)

[Usage]

Obtain pin2 of the user via user ID.

[Parameter]

UserID

User ID

PIN2

Value of pin2

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetUserIDByPIN2

5.2.4 Registration Data Functions (Including Both User Information and Fingerprint)

5.2.4.1 GetEnrollData

[Definition]

VARIANT_BOOL GetEnrollData([in] long dwMachineNumber, [in] long dwEnrollNumber, [in] long dwEMachineNumber, [in] long dwBackupNumber, [out] long* dwMachinePrivilege, [out] long* dwEnrollData, [out] long* dwPassWord)

[Usage]

Obtain registration data (user fingerprint template and some user information) by user ID and corresponding index.

[Parameter]

dwMachineNumber, dwEMachineNumber

Device number. Both values must be equal.

dwEnrollNumber

User ID

dwBackupNumber

Index of the fingerprint. The value ranges from 0 to 9. If the operation succeeds, the fingerprint template and password can be obtained. If the value is 10, only user password is obtained.

dwMachinePrivilege

User privilege. 0: common user, 1: enroller, 2: administrator, 3: super administrator

dwEnrollData

Fingerprint template

dwPassWord

Password

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetEnrollData, DeleteEnrollData, GetEnrollDataStr, GetUserInfo, GetUserTmp

5.2.4.2 SetEnrollData

[Definition]

VARIANT_BOOL SetEnrollData([in] long dwMachineNumber, [in] long dwEnrollNumber, [in] long dwEMachineNumber, [in] long dwBackupNumber, [in] long dwMachinePrivilege, [in] long* dwEnrollData, [in] long dwPassWord)

[Usage]

Set registration data (user fingerprint template and some user information).

[Parameter]

dwMachineNumber, dwEMachineNumber

Device number. Both values must be equal.

dwEnrollNumber

User ID

dwBackupNumber

Index of the fingerprint. The value ranges from 0 to 9. When the value is 10, it indicates the user password is set.

dwMachinePrivilege

User privilege. 0: common user, 1: enroller, 2: administrator, 3: super administrator

dwEnrollData

Fingerprint template to be uploaded

dwPassWord

User password

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetEnrollData, DeleteEnrollData, SetEnrollDataStr, SetUserInfo, SetUserTmp

5.2.4.3 DeleteEnrollData

[Definition]

VARIANT_BOOL DeleteEnrollData([in] long dwMachineNumber, [in] long dwEnrollNumber, [in] long dwEMachineNumber, [in] long dwBackupNumber)

[Usage]

Delete registration data.

[Parameter]

dwMachineNumber, dwEMachineNumber

Device number

dwEnrollNumber

User ID

dwBackupNumber

Index of the fingerprint

The value ranges from 0 to 9. It this case, the device also checks whether the user has other fingerprints or passwords. If no, the device deletes the user.

When the value is 10, the device deletes the password. The device also checks whether the user has fingerprint data. If no, the device deletes the user.

When the value is 11 or 13, the device deletes all fingerprint data of the user. When the value is 12, the device deletes the user (including all fingerprints, card numbers and passwords of the user).

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetEnrollData, GetEnrollData

5.2.4.4 GetEnrollDataStr

[Definition]

VARIANT_BOOL GetEnrollDataStr([in] long dwMachineNumber, [in] long dwEnrollNumber, [in] long dwEMachineNumber, [in] long dwBackupNumber, [out] long* dwMachinePrivilege, [out] BSTR* dwEnrollData, [out] long* dwPassWord)

[Usage]

Obtain registration data (user fingerprint template and some user information) by user ID and corresponding index. The only difference between this function and GetEnrollData is the fingerprint template format.

[Parameter]

dwMachineNumber, dwEMachineNumber

Device number. Both values must be equal.

dwEnrollNumber

User ID

dwBackupNumber

Index of the fingerprint. The value ranges from 0 to 9. If the operation succeeds, the fingerprint template and password can be obtained. If the value is 10, only user password is obtained.

dwMachinePrivilege

User privilege. 0: common user, 1: enroller, 2: administrator, 3: super administrator

dwEnrollData

Fingerprint template to be uploaded

dwPassWord

User password

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetEnrollData , SetEnrollData, DeleteEnrollData, GetUserInfo, GetUserTmp

5.2.4.5 SetEnrollDataStr

[Definition]

VARIANT_BOOL SetEnrollDataStr([in] long dwMachineNumber, [in] long dwEnrollNumber, [in] long dwEMachineNumber, [in] long dwBackupNumber, [in] long dwMachinePrivilege, [in] BSTR dwEnrollData, [in] long dwPassWord)

[Usage]

Set registration data (user fingerprint template and some user information).

[Parameter]

dwMachineNumber, dwEMachineNumber

Device number. Both values must be equal.

dwEnrollNumber

User ID

dwBackupNumber

Index of the fingerprint. The value ranges from 0 to 9. When the value is 10, it indicates the user password is set.

dwMachinePrivilege

User privilege. 0: common user, 1: enroller, 2: administrator, 3: super administrator

dwEnrollData

Fingerprint template to be uploaded

dwPassWord

User password

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetEnrollData, GetEnrollData, DeleteEnrollData, SetUserInfo, SetUserTmp

5.2.5 Fingerprint Template Functions

5.2.5.1 ReadAllTemplate

[Definition]

VARIANT_BOOL ReadAllTemplate([in] LONG dwMachineNumber)

[Usage]

Read out all fingerprint templates from the device and write them into the memory of the PC. This function is used to read out and write all the fingerprints into the memory at a time. This function achieves a higher speed in comparison with the way of obtaining fingerprints one by one.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.2.5.2 DelUserTmp

[Definition]

VARIANT_BOOL DelUserTmp([in] long dwMachineNumber, [in] long dwEnrollNumber, [in] long dwFingerIndex)

[Usage]

Delete the fingerprint template of a specified user.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

dwFingerIndex

Index of the fingerprint. The value ranges from 0 to 9. When the device uses Finger10.0, the index can be specified as 15 to delete all fingerprint templates of the user at a time.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetUserTmp, SetUserTmp

5.2.5.3 GetUserTmp

[Definition]

VARIANT_BOOL GetUserTmp([in] long dwMachineNumber, [in] long dwEnrollNumber, [in] long dwFingerIndex, [out] BYTE* TmpData, [out] long * TmpLength)

[Usage]

Obtain the Finger 9.0 fingerprint templates of a user in binary mode. The only difference between this function and GetUserTmpStr is the fingerprint template format.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

dwFingerIndex

Index of the fingerprint. The value ranges from 0 to 9. When the device uses Finger10.0, the index can be specified as 15 to obtain all fingerprint templates of the user at a time.

TmpData

Fingerprint template data

TmpLength

Fingerprint template length

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetUserTmp

5.2.5.4 SetUserTmp

[Definition]

VARIANT_BOOL SetUserTmp([in] long dwMachineNumber, [in] long dwEnrollNumber, [in] long dwFingerIndex, [in] BYTE* TmpData)

[Usage]

Upload the Finger 9.0 fingerprint templates of a user in binary mode. The only difference between this function and SetUserTmpStr is the fingerprint template format. Caution: The user must have been created on the device. If the same index is already registered by the user, the fingerprint template will be overwritten.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

dwFingerIndex

Index of the fingerprint. The value ranges from 0 to 9. When the device uses Finger10.0, the index can be specified as 15 to obtain all fingerprint templates of the user at a time.

TmpData

Fingerprint template data

TmpLength

Fingerprint template length

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetUserTmp

5.2.5.5 GetUserTmpStr

[Definition]

VARIANT_BOOL GetUserTmpStr([in] long dwMachineNumber,[in] long dwEnrollNumber, [in] long dwFingerIndex, [out] BSTR* TmpData, [out] long * TmpLength)

[Usage]

Obtain the Finger 9.0 fingerprint templates of a user in string form. The only difference between this function and GetUserTmp is the fingerprint template format.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

dwFingerIndex

Index of the fingerprint. The value ranges from 0 to 9. When the device uses Finger10.0, the index can be specified as 15 to obtain all fingerprint templates of the user at a time.

TmpData

Fingerprint template data

TmpLength

Fingerprint template length

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetUserTmpStr

5.2.5.6 SetUserTmpStr

[Definition]

VARIANT_BOOL SetUserTmpStr([in] long dwMachineNumber, [in]long dwEnrollNumber, [in] long dwFingerIndex, [in]BSTR TmpData)

[Usage]

Set the Finger 9.0 fingerprint templates of a user in string form. The only difference between this function and SetUserTmp is the fingerprint template format. Caution: The user must have been created on the device. If the same index is already registered by the same user, the fingerprint template will be overwritten.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

dwFingerIndex

Index of the fingerprint. The value ranges from 0 to 9. When the device uses Finger10.0, the index can be specified as 15 to set all fingerprint templates of the user at a time.

TmpData

Fingerprint template

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetUserTmpStr

5.2.5.7 GetUserTmpEx

[Definition]

VARIANT_BOOL GetUserTmpEx([in] long dwMachineNumber, [in] BSTR dwEnrollNumber, [in] long dwFingerIndex,[out] long* Flag, [out] BYTE* TmpData, [out] long* TmpLength)

[Usage]

Obtain the Finger 10.0 fingerprint templates of a user in binary mode. The only difference between this function and GetUserTmpStr is the fingerprint template format.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

dwFingerIndex

Index of the fingerprint

Flag

Flag used to indicate whether the fingerprint template is valid or is a duress fingerprint template. 0: Invalid; 1: Valid; 3: duress fingerprint template

TmpData

Fingerprint template data

TmpLength

Fingerprint template length

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetUserTmpEx

5.2.5.8 SetUserTmpEx

[Definition]

VARIANT_BOOL SetUserTmp([in] long dwMachineNumber, [in] long dwEnrollNumber, [in] long dwFingerIndex, [in] long Flag, [in] BYTE* TmpData)

[Usage]

Upload the Finger 10.0 fingerprint templates of a user in binary mode.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

dwFingerIndex

Index of the fingerprint. The value ranges from 0 to 9.

Flag

Flag used to indicate whether the fingerprint template is valid or is a duress fingerprint template. 0: Invalid; 1: Valid; 3: duress fingerprint template

TmpData

Fingerprint template data

TmpLength

Fingerprint template length

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetUserTmpEx

5.2.5.9 GetUserTmpExStr

[Definition]

VARIANT_BOOL GetUserTmpExStr([in] long dwMachineNumber, [in] BSTR dwEnrollNumber, [in] long dwFingerIndex,[out] long* Flag, [out] BSTR* TmpData, [out] long* TmpLength)

[Usage]

Obtain the Finger 10.0 fingerprint templates of a user in string form.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

dwFingerIndex

Index of the fingerprint

Flag

Flag used to indicate whether the fingerprint template is valid or is a duress fingerprint template. 0: Invalid; 1: Valid; 3: duress fingerprint template

TmpData

Fingerprint template data

TmpLength

Fingerprint template length

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetUserTmpExStr

5.2.5.10 SetUserTmpExStr

[Definition]

VARIANT_BOOL SetUserTmpStr([in] long dwMachineNumber, [in] long dwEnrollNumber, [in] long dwFingerIndex, [in] long Flag, [in] BYTE* TmpData)

[Usage]

Upload the Finger 10.0 fingerprint templates of a user in string form.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

dwFingerIndex

Index of the fingerprint. The value ranges from 0 to 9.

Flag

Flag used to indicate whether the fingerprint template is valid or is a duress fingerprint template. 0: Invalid; 1: Valid; 3: duress fingerprint template

TmpData

Fingerprint template data

TmpLength

Fingerprint template length

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetUserTmpExStr

5.2.6 SMS Functions

5.2.6.1 SetSMS

[Definition]

VARIANT_BOOL SetSMS([in] LONG dwMachineNumber, [in] LONG ID, [in] LONG Tag, [in] LONG ValidMinutes, [in] BSTR StartTime, [in] BSTR Content)

[Usage]

Add a short message to the device. To set a short message of a user, use SetUserSMS to allocate the short message to the user.

[Parameter]

dwMachineNumber

Device number

ID

Short message number

Tag

Short message type. 253: public short message, 254: personal short message, 255: reserved short message

ValidMinutes

Valid duration of a short message. The value ranges from 0 to 65535. That is, the short message becomes valid at StartTime and keeps valid for ValidMinutes.

StartTime

Start time when a short message becomes valid, in a string format of yyyy-mm-dd hh:mm:ss

Content

Content of a short message

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetSMS, DeleteSMS, ClearSMS, SetUserSMS

5.2.6.2 SetUserSMS

[Definition]

VARIANT_BOO SetUserSMS([in] LONG dwMachineNumber, [in] LONG dwEnrollNumber, [in] LONG SMSID)

[Usage]

Set a short message of a user, that is, allocate the short message with the specified number in the device to a specific user.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

SMSID

Short message number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetSMS, GetSMS, DeleteSMS, ClearSMS

[Supporting Device**]**

Only TFT devices support this function. For black & white devices, see ssr_SetUserSMS.

5.2.6.3 GetSMS

[Definition]

VARIANT_BOOL GetSMS([in] LONG dwMachineNumber, [in] LONG ID, [out] LONG* Tag, [out] LONG* ValidMinutes, [out] BSTR* StartTime, [out] BSTR *Content)

[Usage]

Obtain details of a short message (including content, start time, short message type, and valid duration) by short message number.

[Parameter]

dwMachineNumber

Device number

ID

Short message number

Tag

Short message type. 253: public short message, 254: personal short message, 255: reserved short message

ValidMinutes

Valid duration of a short message. The value ranges from 0 to 65535. That is, the short message becomes valid at StartTime and keeps valid for ValidMinutes.

StartTime

Start time when a short message becomes valid

Content

Content of a short message

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetSMS, SetUserSMS, GetSMS, DeleteSMS, ClearSMS

5.2.6.4 DeleteSMS

[Definition]

VARIANT_BOOL DeleteSMS([in] LONG dwMachineNumber, [in] LONG ID)

[Usage]

Delete a short message with the specified number from the device.

[Parameter]

dwMachineNumber

Device number

ID

Short message number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetSMS, ClearSMS

5.2.6.5 DeleteUserSMS

[Definition]

VARIANT_BOOL DeleteUserSMS([in]LONG dwMachineNumber, [in]LONG dwEnrollNumber, [in] LONG SMSID)

[Usage]

Delete the specified short message of a specified user. In this case, only the mapping between the user and short message, instead of the short message, is deleted.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

SMSID

Short message number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetUserSMS

5.2.6.6 ClearUserSMS

[Definition]

VARIANT_BOOL ClearUserSMS([in] LONG dwMachineNumber)

[Usage]

Clear only all mappings between a user and short messages, instead of short messages.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ClearSMS, SetSMS

5.2.6.7 ClearSMS

[Definition]

VARIANT_BOOL ClearSMS([in] LONG dwMachineNumber)

[Usage]

Clear all the short messages from the device.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetSMS

5.2.7 Work Code Functions

5.2.7.1 SetWorkCode

[Definition]

VARIANT_BOOL SetWorkCode([in] LONG WorkCodeID, [in] LONG AWorkCode)

[Usage]

Define the work code with the specified number. Note: Black & white devices support work codes in all ranges. However, after this function is used to define work codes, only the work codes within the defined range can be input. For example:

SetWorkCode (1, 345)

SetWorkCode (2, 567)

In the preceding example, only the work codes 345 and 567 can be input.

[Parameter]

WorkCodeID

ID of a work code

AWorkCode

ID of the work code described by WorkCodeID

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetWorkCode

5.2.7.2 GetWorkCode

[Definition]

VARIANT_BOOL GetWorkCode([in] LONG WorkCodeID, [out] LONG* AWorkCode)

[Usage]

Obtain the ID of the specified work code. For details, see SetWorkCode.

[Parameter]

WorkCodeID

ID of the work code

AWorkCode

ID of the work code described by WorkCodeID

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetWorkCode

5.2.7.3 DeleteWorkCode

[Definition]

VARIANT_BOOL DeleteWorkCode([in] LONG WorkCodeID)

[Usage]

Delete the work code with specified ID. For details, see SetWorkCode.

[Parameter]

WorkCodeID

ID of the work code

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetWorkCode, GetWorkCode, ClearWorkCode

5.2.7.4 ClearWorkCode

[Definition]

VARIANT_BOOL ClearWorkCode(void)

[Usage]

Clear all work codes defined in the device. For details, see SetWorkCode.

[Parameter]

None

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetWorkCode, GetWorkCode, DeleteWorkCode

5.2.8 Holiday Functions

5.2.8.1 SetHoliday

[Definition]

VARIANT_BOOL SetHoliday([in] LONG dwMachineNumber, [in] BSTR Holiday)

[Usage]

Set holidays.

[Parameter]

dwMachineNumber

Device number

Holiday

Holiday to be set, in format of mmddmmdd. For example, 04140511 indicates that holidays start from April 14 to May 11.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetHoliday, SSR_GetHoliday

5.2.8.2 GetHoliday

[Definition]

VARIANT_BOOL GetHoliday([in] LONG dwMachineNumber, [out] BSTR* Holiday)

[Usage]

Obtain holidays set on the device.

[Parameter]

dwMachineNumber

Device number

Holiday

Holidays set on the device

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetHoliday

5.2.9 DST Functions

5.2.9.1 SetDaylight

[Definition]

VARIANT_BOOL SetDaylight([in] LONG dwMachineNumber, [in] LONG Support, [in] BSTR BeginTime, [in] BSTR EndTime)

[Usage]

Set whether to use daylight saving time (DST), start time and end time of DST.

[Parameter]

dwMachineNumber

Device number

Support

Whether to use DST. 1: use, 0: not use

BeginTime

Start time of DST, in format of mmdd hh:mm

EndTime

End time of DST, in format of mmdd hh:mm

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetDaylight

5.2.9.2 GetDaylight

[Definition]

VARIANT_BOOL GetDaylight ([in] LONG dwMachineNumber, [out] LONG* Support, [out] BSTR* BeginTime, [out] BSTR* EndTime)

[Usage]

Obtain DST settings of the current device.

[Parameter]

dwMachineNumber

Device number

Support

Whether to use DST. 1: use, 0: not use

BeginTime

Start time of DST, in format of mmdd hh:mm

EndTime

End time of DST, in format of mmdd hh:mm

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetDaylight

5.2.10 Customization Functions (Voice and Attendance State)

5.2.10.1 EnableCustomizeAttState

[Definition]

VARIANT_BOOL EnableCustomizeAttState([in] LONG dwMachineNumber, [in] LONG StateID, [in] LONG Enable)

[Usage]

Enable the customized attendance state value. Note: This function is a customized function. To use this function, the device must enable the expansion function and support customized attendance state values.

Customized attendance state value: The mappings between attendance state values and states of the current device are as follows: If this function is enabled, the value of a specific state can be changed by using SetCustomizeAttState.

0—Check-In 1—Check-Out 2—Break-Out

3—Break-In 4—OT-In 5—OT-Out

For example, EnableCustomizeAttState (1, 0, 1)//Enable the customized state of value 0 (that is, check-in).

EnableCustomizeAttState (1, 0, 8)//Set the customized state of value 0 (that is, check-in) to 8.

After the user selects check-in on the device and passes verification, the saved attendance record state value is 8.

[Parameter]

dwMachineNumber

Device number

StateID

Attendance state value to be customized

Enable

Whether the customized attendance state is used for the state described by StateID

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetCustomizeAttState, DelCustomizeAttState

5.2.10.2 SetCustomizeAttState

[Definition]

VARIANT_BOOL SetCustomizeAttState([in] LONG dwMachineNumber, [in] LONG StateID, [in] LONG NewStateI)

[Usage]

Set the customized attendance state value according to the original attendance state value. Note: This function is a customized function. To use this function, the device must enable the expansion function and support customized attendance state values (see descriptions of EnableCustomizeAttState).

[Parameter]

dwMachineNumber

Device number

StateID

Original state value

NewStateI

New state value to be set

[Return Value]

Return True if it is successful, or return False.

[Related Function]

EnableCustomizeAttState, DelCustomizeAttState

5.2.10.3 DelCustomizeAttState

[Definition]

VARIANT_BOOL DelCustomizeAttState([in] LONG dwMachineNumber, [in] LONG StateID)

[Usage]

Delete the customized attendance state specified by to the original attendance state value. Note: This function is a customized function. To use this function, the device must enable the expansion function and support customized attendance state values (see EnableCustomizeAttState).

[Parameter]

dwMachineNumber

Device number

StateID

Original attendance state value of the customized attendance state value to be deleted

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetCustomizeAttState, EnableCustomizeAttState

5.2.10.4 SetCustomizeVoice

[Definition]

VARIANT_BOOL SetCustomizeVoice([in] LONG dwMachineNumber, [in] LONG VoiceID, [in] BSTR FileName)

[Usage]

Set customized voice according to the original voice number. Note: This function is a customized function. To use this function, the device must support customized voice. For details, see EnableCustomizeVoice.

[Parameter]

dwMachineNumber

Device number

VoiceID

Original voice number

FileName

Name of customized voice file (including the path)

[Return Value]

Return True if it is successful, or return False.

[Related Function]

DelCustomizeVoice, EnableCustomizeVoice

5.2.10.5 DelCustomizeVoice

[Definition]

VARIANT_BOOL DelCustomizeVoice([in] LONG dwMachineNumber, [in] LONG VoiceID)

[Usage]

Delete customized voice according to the original voice number. Note: This function is a customized function. To use this function, the device must support customized voice. For details, see EnableCustomizeVoice.

[Parameter]

dwMachineNumber

Device number

VoiceID

Original voice number of the customized voice to be deleted

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetCustomizeVoice, EnableCustomizeVoice

5.2.10.6 EnableCustomizeVoice

[Definition]

VARIANT_BOOL EnableCustomizeVoice([in] LONG dwMachineNumber, [in] LONG VoiceID, [in] LONG EnableI)

[Usage]

Enable the customized voice of the specified number. Note: This function is a customized function. To use this function, the device must support voice customization.

Customized voice: The device plays corresponding voice tones when performing some operations. With this function, the device can be used to play customized voice. For example, after verification succeeds, the device plays voice 1, that is, "Thank you". After this function is used, the device can play customized after verification succeeds, for example:

EnableCustomizeVoice (1, 1, 1)//Enable customization function of voice 1

SetCustomizeVoice (1, 1, 'C:\test.wav')//Set customized voice of voice 1

[Parameter]

dwMachineNumber

Device number

VoiceID

Voice number

EnableI

Whether to enable customized voice of the specified number. 1: enable, 0: disable

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetCustomizeVoice, DelCustomizeVoice

5.2.11 Fingerprint Template Conversion Functions

5.2.11.1 GetFPTempLength

[Definition]

GetFPTempLength([in] BYTE* dwEnrollData, [out] long * Len)

[Usage]

Calculate the length of a specified fingerprint template.

[Parameter]

dwEnrollData

Pointer pointing at the fingerprint template

Len

Length of the fingerprint template described by dwEnrollData

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetFPTempLengthStr

5.2.11.2 GetFPTempLengthStr

[Definition]

GetFPTempLengthStr([in] BSTR dwEnrollData, [out] long * Len)

[Usage]

Calculate the length of a specified fingerprint template.

[Parameter]

dwEnrollData

Fingerprint template in string form

Len

Length of the fingerprint template described by dwEnrollData

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetFPTempLength

5.2.11.3 FPTempConvert

[Definition]

VARIANT_BOOL FPTempConvert([in] BYTE* TmpData1, [out] BYTE* TmpData2, [out] long *Size)

[Usage]

Convert an offline fingerprint template into a Biokey fingerprint template. The only difference between this function and FPTempConvertStr is the data format.

[Parameter]

TmpData1

Offline fingerprint template to be converted

TmpData2

Converted Biokey fingerprint template

Size

Size of the converted Biokey fingerprint template

[Return Value]

Return True if it is successful, or return False.

[Related Function]

FPTempConvertStr, FPTempConvertNew, FPTempConvertNewStr

5.2.11.4 FPTempConvertStr

[Definition]

VARIANT_BOOL FPTempConvertStr([in] BSTR TmpData1,[out]BSTR* TmpData2, [out] long *Size)

[Usage]

Convert an offline fingerprint template into a Biokey fingerprint template in string form. The only difference between this function and FPTempConvert is the data format.

[Parameter]

TmpData1

Offline fingerprint template to be converted

TmpData2

Converted Biokey fingerprint template

Size

Size of the converted Biokey fingerprint template

[Return Value]

Return True if it is successful, or return False.

[Related Function]

FPTempConvert, FPTempConvertNew, FPTempConvertNewStr

5.2.11.5 FPTempConvertNew

[Definition]

VARIANT_BOOL FPTempConvertNew([in] BYTE* TmpData1, [out] BYTE* TmpData2, [out] long *Size)

[Usage]

Convert a Biokey fingerprint template into an offline fingerprint template. The only difference between this function and FPTempConvertNewStr is the data format.

[Parameter]

TmpData1

Offline fingerprint template to be converted into

TmpData2

Converted offline fingerprint template

Size

Size of the converted offline fingerprint template

[Return Value]

Return True if it is successful, or return False.

[Related Function]

FPTempConvertNewStr, FPTempConvert, FPTempConvertStr

5.2.11.6 FPTempConvertNewStr

[Definition]

VARIANT_BOOL FPTempConvertNewStr([in] BSTR TmpData1, [out] BSTR* TmpData2, [out] long *Size)

[Usage]

Convert a Biokey fingerprint template into an offline fingerprint template in string form. The only difference between this function and FPTempConvertNew is the data format.

[Parameter]

TmpData1

Offline fingerprint template to be converted into

TmpData2

Converted offline fingerprint template

Size

Size of the converted offline fingerprint template

[Return Value]

Return True if it is successful, or return False.

[Related Function]

FPTempConvertNew, FPTempConvert, FPTempConvertStr

5.2.12 System Data Management Functions

5.2.12.1 ClearKeeperData

[Definition]

VARIANT_BOOL ClearKeeperData(**[**in**]** long dwMachineNumber)

[Usage]

Clear all data in the device.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ClearGLog, ClearSLog

5.2.12.2 ClearData

[Definition]

VARIANT_BOOL ClearData([in] LONG dwMachineNumber, [in] LONG DataFlag)

[Usage]

Clear the record specified by DataFlag from the device.

[Parameter]

dwMachineNumber

Device number

DataFlag

Type of the records to be cleared. The value ranges from 1 to 5. The meanings are as follows:

1. Attendance record
2. Fingerprint template data
3. None
4. Operation record
5. User information

When the value of this parameter is 5, all user data in the device is deleted. Note: All fingerprint templates are also deleted.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ClearKeeperData

5.2.12.3 GetDataFile

[Definition]

VARIANT_BOOL GetDataFile([in] LONG dwMachineNumber, [in] LONG DataFlag, [in] BSTR FileName)

[Usage]

Obtain the specified data file from the device.

[Parameter]

dwMachineNumber

Device number

DataFlag

Type of the data file to be obtained

1. Attendance record data file
2. Fingerprint template data file
3. None
4. Operation record data file
5. User information data file
6. SMS data file
7. SMS and user relationship data file
8. Extended user information data file
9. Work code data file

FileName

Name of the obtained data file

[Return Value]

Return True if it is successful, or return False.

[Related Function]

5.2.12.4 SendFile

[Definition]

VARIANT_BOOL SendFile([in] LONG dwMachineNumber,[in] BSTR FileName)

[Usage]

Send files to the device, usually to the **/mnt/mtdblock/** directory. For TFT devices, if user pictures or advertisement pictures are sent, the files should be named in the following formats and automatically moved to corresponding directories.

Naming of advertisement pictures: prefix "ad" +a numeral ranging from 1 to 20 + suffix ".jpg", for example, **ad_4.jpg**

Naming of user pictures: user ID + ".jpg", for example, **1.jpg**

[Parameter]

dwMachineNumber

Device number

FileName

Name of the file to be sent

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ReadFile

5.2.12.5 ReadFile

[Definition]

VARIANT_BOOL ReadFile([in] LONG dwMachineNumber, [in] BSTR FileName, [in] BSTR FilePath)

[Usage]

Read the file with the specified name from the device. The file path is usually /mnt/mtdblock/.

[Parameter]

dwMachineNumber

Device number

FileName

Name of the file to be read from the device

FilePath

Path of the file to be read

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SendFile

5.2.12.6 RefreshData

[Definition]

VARIANT_BOOL RefreshData([in] long dwMachineNumber)

[Usage]

Refresh the data in the device. This function is usually called after user information or fingerprints are uploaded. In this way, the modifications take effect immediately.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.3 Access Control Functions (Time Slot, Group, Open Door Combination)

5.3.1 GetUserGroup

[Definition]

VARIANT_BOOL GetUserGroup([in] long dwMachineNumber, [in] long dwEnrollNumber, [out] long *UserGrp)

[Usage]

Obtain the number of the group to which a specified user belongs.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

UserGrp

Group number of the user described by dwEnrollNumber. The value ranges from 1 to 5.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetUserGroup

5.3.2 SetUserGroup

[Definition]

VARIANT_BOOL SetUserGroup([in] long dwMachineNumber, [in] long dwEnrollNumber, [in] long UserGrpl)

[Usage]

Set the group to which a specified user belongs.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

UserGrpI

Group number. The value ranges from 1 to 5.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetUserGroup

5.3.3 GetTZInfo

[Definition]

VARIANT_BOOL GetTZInfo([in] long dwMachineNumber, [in] long TZIndex, [out] BSTR *TZ)

[Usage]

Obtain the information of the time slot with the specified number.

[Parameter]

dwMachineNumber

Device number

TZIndex

Time slot index

TZ

Time slot with the index described by TZIndex. Every eight bits represent a time slot, in format of hhmmhhmm. For example, 1011122300002359000023590000235900002359000023590000235900002359 indicates a time slot covering the whole day from Monday to Saturday and from 10:11 to 12:23 of Sunday.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetTZInfo

5.3.4 SetTZInfo

[Definition]

VARIANT_BOOL SetTZInfo([in] long dwMachineNumber, [in] long TZIndex, [in] BSTR TZ)

[Usage]

Set the information of the time slot with the specified number.

[Parameter]

dwMachineNumber

Device number

TZIndex

Time slot index

TZ

Time slot to be set. Every eight bits represent a time slot, in format of hhmmhhmm. For example, 10111223000023590000235900002359000023590000235900002359 indicates a time slot covering the whole day from Monday to Saturday and from 10:11 to 12:23 of Sunday.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetTZInfo

5.3.5 GetUnlockGroups

[Definition]

VARIANT_BOOL GetUnlockGroups([in] long dwMachineNumber, [out] BSTR *Grps)

[Usage]

Obtain the open door combination of the device.

[Parameter]

dwMachineNumber

Device number

Grps

Open door combination of the current device. The return values include 10 combinations separated by ":". For example, "12:23:14:15::::::." represent four valid combinations, that is, combination 1 (12 represents groups 1 and 2), combination 2 (23 represent groups 2 and 3), combination 3 (14 represent groups 1 and 4), and combination 4 (15 represent groups 1 and 5).

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetUnlockGroups

5.3.6 SetUnlockGroups

[Definition]

VARIANT_BOOL SetUnlockGroups([in] long dwMachineNumber, [in] BSTR Grps)

[Usage]

Set the open door combination.

[Parameter]

dwMachineNumber

Device number

Grps

Open door combination. Ten combinations need to be set and separated by ":". For example, "12:23:14:15::::::" represent four valid combinations, that is, combination 1 (12 represents groups 1 and 2), combination 2 (23 represent groups 2 and 3), combination 3 (14 represent groups 1 and 4), and combination 4 (15 represent groups 1 and 5).

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetUnlockGroups

5.3.7 GetGroupTZs

[Definition]

VARIANT_BOOL GetGroupTZs([in] long dwMachineNumber, [in] long GroupIndex, [out] long *TZs)

[Usage]

Obtain the time slot setting of a specified group. The difference from GetGroupTZStr is the return value format.

[Parameter]

dwMachineNumber

Device number

GroupIndex

Group index. The value ranges from 1 to 5.

TZs

Pointer of long type. Indexes of three time slots used by the group described by GroupIndex are returned respectively by TZs[0], TZs[1], and TZs[2].

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetGroupTZs, GetGroupTZStr

5.3.8 SetGroupTZs

[Definition]

VARIANT_BOOL SetGroupTZs([in] long dwMachineNumber, [in] long GroupIndex, [in] long *TZs)

[Usage]

Set the three time slot setting of a specified group. The difference from SetGroupTZStr is the return value format.

[Parameter]

dwMachineNumber

Device number

GroupIndex

Group index. The value ranges 1 to 5.

TZs

Time slot index, which is a pointer of long type. Indexes of the three time slots are imported respectively by TZs[0], TZs[1], and TZs[2].

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetGroupTZs, SetGroupTZStr

5.3.9 GetGroupTZStr

[Definition]

VARIANT_BOOL GetGroupTZStr([in] long dwMachineNumber, [in] long GroupIndex, [out] BSTR *TZs)

[Usage]

Obtain the time slot setting of a specified group. The difference from GetGroupTZs is the return value format.

[Parameter]

dwMachineNumber

Device number

GroupIndex

Group index. The value ranges from 1 to 5.

TZs

Index of the time slot used by the group described by GroupIndex. Each group contains three time slots separated by ":". For example, the return value "1:23:13" means that the indexes of the three time slots of the group are 1, 23, and 13.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetGroupTZStr, GetGroupTZs

5.3.10 SetGroupTZStr

[Definition]

VARIANT_BOOL SetGroupTZStr([in] long dwMachineNumber, [in] long GroupIndex, [in]BSTR TZs)

[Usage]

Set the time slot of a specified group. The only difference between this function and SetGroupTZs is the return value format.

[Parameter]

dwMachineNumber

Device number

GroupIndex

Group index. The value ranges 1 to 5.

TZs

Index of the time slot. Each group contains three time slots separated by ":". For example, the return value "1:23:13" means that the indexes of the three time slots of the group are 1, 23, and 13.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetGroupTZs, SetGroupTZs

5.3.11 GetUserTZs

[Definition]

VARIANT_BOOL GetUserTZs([in] long dwMachineNumber, [in] long UserID, [out] long *TZs)

[Usage]

Obtain the time slot setting of a user. Each user has three time slots. The only difference between this function and GetUserTZStr is the format of returned time slot index.

[Parameter]

dwMachineNumber

Device number

UserID

User ID

TZs

Open door time slot of a user. The TZs pointers have three values that store indexes of three time slots respectively. The indexes of three time slots can be obtained by using TZs[0], TZs[1], and TZs[2] respectively.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetUserTZs, SetUserTZStr

5.3.12 SetUserTZs

[Definition]

VARIANT_BOOL SetUserTZs([in] long dwMachineNumber, [in] long UserID, [in] long *TZs)

[Usage]

Set the time slot of a user. A maximum of three time slots can be set for each user. The difference between this function and SetUserTZStr is the format of imported time slot index.

[Parameter]

dwMachineNumber

Device number

UserID

User ID

TZs

Time slot index. When TZs[0] is 0, the group setting is used. When TZs[0] is not 0, a time slot number set by the user can be specified. This parameter is a long pointer. Indexes of three time slots can be imported respectively by TZs[0], TZs[1], and TZs[2].

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetUserTZs, GetUserTZStr

5.3.13 GetUserTZStr

[Definition]

VARIANT_BOOL GetUserTZStr([in] long dwMachineNumber, [in] long UserID, [out] BSTR *TZs)

[Usage]

Obtain the time slots of a user. The only difference between this function and GetUserTZs is the format of returned time slot index.

[Parameter]

dwMachineNumber

Device number

UserID

User ID

TZs

Unlock time slot of a user. The formats are as follows:

Black & white access control devices: X1:X2:X3. X1, X2, and X3 represent the indexes of the customized time slots used by the user. To determine whether the user uses the group time slot, check the return value of the UseGroupTimeZone function. For example, if user A uses customized time slots 1, 2, and 3, the fingerprint device returns "1:2:3".

TFT access control devices: X1:X2:X3:X4. X4 represents whether to use the group time slot. If the value is 0, the group time slot is used; if the value is **1**, the group time slot is not used, that is, the personal time slot is used. X1, X2, and X3 represent the indexes of the used time slots. For example, if user A uses customized time slots 1 and 2, the fingerprint device returns "1:2:0:1".

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetUserTZStr, GetUserTZs

5.3.14 SetUserTZStr

[Definition]

VARIANT_BOOL SetUserTZStr([in] long dwMachineNumber, [in] long UserID, [in] BSTR TZs)

[Usage]

Set the time slots of a user. Time slots are separated by ":". The only difference between this function and SetUserTZs is the format of imported time slot index.

[Parameter]

dwMachineNumber

Device number

UserID

User ID

TZs

See GetUserTZStr

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetUserTZStr, SetUserTZs

5.3.15 ACUnlock

[Definition]

VARIANT_BOOL ACUnlock([in] long dwMachineNumber, [in] long Delay)

[Usage]

Open the door, enable the open door controller to output a unlock signal, and close the door after 10-second delay.

[Parameter]

dwMachineNumber

Device number

Delay

Open door delay

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.3.16 GetACFun

[Definition]

VARIANT_BOOL GetACFun([out] long* ACFun)

[Usage]

Obtain whether the device has the access control function.

[Parameter]

ACFun

Flag of the access control function of the device. 0: no access control, 1: simple access control, 2: middle-level access control, 6: high-level access control, 14: high-level access control + always open, 15: access control available

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.3.17 GetDoorState

[Definition]

VARIANT_BOOL GetDoorState([in] LONG MachineNumber, [out] LONG* State)

[Usage]

Obtain the current door state. 1: opened, 0: closed. It is Deprecated, please refer to OnDoor Event.

[Parameter]

MachineNumber

Device number

State

Door state

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.3.18 UseGroupTimeZone

[Definition]

VARIANT_BOOL UseGroupTimeZone(void)

[Usage]

Determine whether a user uses the group time slot setting. This function must be used together with GetUserTZs or GetUserTZStr to ensure the correct return values. That is, use GetUserTZs or GetUserTZStr to obtain the time slot settings of the specified user, and then use UseGroupTimeZone to determine whether the user uses the group time slot setting.

[Parameter]

None

[Return Value]

Return True if the group time slot is used, or return False.

[Related Function]

GetUserTZs, GetUserTZStr

5.4 Device Management Functions

5.4.1 IsTFTMachine

[Definition]

VARIANT_BOOL IsTFTMachine([in] LONG dwMachineNumber)

[Usage]

Determine whether the current device is a TFT device.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if the current device is a TFT device. Return False if the current device is a black & white device.

[Related Function]

None

5.4.2 GetDeviceStatus

[Definition]

VARIANT_BOOL GetDeviceStatus([in] long dwMachineNumber, [in] long dwStatus, [out] long* dwValue)

[Usage]

Obtain the data storage status of the device, for example, number of administrators and number of current users.

[Parameter]

dwMachineNumber

Device number

dwStatus

Data to be obtained. The value ranges from 1 to 22. Values:

1 Number of administrators

2 Number of registered users

3 Number of fingerprint templates in the device

4 Number of passwords

5 Number of operation records

6 Number of attendance records

7 Fingerprint template capacity

8 User capacity

9 Attendance record capacity

10 Residual fingerprint template capacity

11 Residual user capacity

12 Residual attendance record capacity

21 Number of faces

22 Face capacity

Returned 0 in other cases.

dwValue

Value of dwStatus

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.4.3 GetDeviceInfo

[Definition]

VARIANT_BOOL GetDeviceInfo([in] long dwMachineNumber, [in] long dwInfo, [out] long* dwValue)

[Usage]

Obtain related information of the device, such as language and baud rate.

[Parameter]

dwMachineNumber

Device number

dwInfo

Type of the information to be obtained. The value ranges from 1 to 68 (except for 65). Values:

1 Maximum number of administrators. Generally, return 500.

2 Device number

3 Language

The return values received by dwValue are as follows:

0 Language with suffix "E", usually representing English

1 Others

2 Language with suffix "T", usually representing traditional Chinese

3 Language with suffix "L", usually representing Thai language

4 Idle duration (minutes). That is, the device enters standby state or is powered off after keeping idle for a period specified by this value.

5 Lock duration, namely, driver lock duration

6 Number of attendance record alarms. That is, the device reports an alarm when the number of attendance records reaches this value.

7 Number of operation record alarms. That is, the device reports an alarm when the number of operation records reaches this value.

8 Repeated record time, that is, the minimum interval of opening the same attendance record by the same user

9 Baud rate in RS232/RS485 communication

The return values received by dwValue are as follows:

0 1200bps

1 2400

2 4800

3 9600

4 19200

5 38400

6 57600

Others: 115200

10 Parity check. Generally, return 0.

11 Stop bit. Generally, return 0.

12 Date separator. Generally, return 1.

13 Whether network function is enabled. Values: 1: enabled, 0: disabled

14 Whether RS232 is enabled

15 Whether RS485 is enabled

16 Whether voice function is supported

17 Whether high-speed comparison is performed

18 Idle mode, that is, the state that the device enters after idle period. Values: 87: power-off, 88: hibernate

19 Automatic power-off time point. The default value is 255, that is, the device does not power off automatically.

20 Automatic power-on time point. The default value is 255, that is, the device does not power on automatically.

21 Automatic hibernation time point. The default value is 255, that is, the device does not hibernate automatically.

22 Automatic ring time point. The default value is 65535, that is, the device does not ring automatically.

23 1:N match threshold

24 Match threshold during registration

25 1:1 match threshold

26 Whether to display match score during verification

27 Number of concurrent unlock users

82 Verify only card number

29 Network speed

The return values received by dwValue are as follows:

1 100M-H

4 10M-F

5 100M-F

8 AUTO

Others: 10M-H

30 Whether the card must be registered

31 Waiting time before the device automatically returns when there is temporarily no operation

32 Waiting time before the device automatically returns when no response is returned after the PIN is input

33 Waiting time before the device automatically returns when there is response after entering the menu

34 Time format

35 Whether 1:1 match must be used

36—40 Automatic ring time points 2, 3, 4, 5, and 6. The default values are **65535**, that is, the device does not ring automatically.

41—56 Automatic state change time points 1 to 16. The default values are **-1**, that is, the device does not change state automatically.

57 Wiegand failure ID

58 Wiegand threaten ID

59 Wiegand region-position code

60 Wiegand output pulse width

61 Wiegand output pulse interval

62 Start sector of Mifare card for storing fingerprints

63 Total sectors of Mifare card for storing fingerprints

64 Number of fingerprints stored on Mifare card

66 Whether to display attendance state

67 Not available temporarily

68 Not supported

8999 In this case, dwValue is used as both input and output parameters. As the input parameter, dwValue represent the name of another option to be obtained. As the output parameter, dwValue represents the value of the option (in this case, dwValue is similar to GetSysOption).

Note: The return values of the preceding time points are numerals. To convert the numeral into the time point, convert the value into a binary numeral where the lowest eight bits represent minute and the highest bits represent hour. For example, if the return value is 2860, it can be converted into 101100101100 in binary, of which the lowest eight bits 00101100 (that is, 44) and the highest eight bits are 00001011 (that is, 11), that is, the actual time point is 11:44.

dwValue

Value described by dwInfo

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetDeviceInfo

5.4.4 SetDeviceInfo

[Definition]

VARIANT_BOOL SetDeviceInfo([in] long dwMachineNumber, [in] long dwInfo, [in] long dwValue)

[Usage]

Set the related information of the device, such as language and repeated record time.

[Parameter]

dwMachineNumber

Device number

dwInfo

Type of the information to be set. The value ranges from 1 to 20. For the meanings of values, see GetDeviceInfo.

dwValue

Value of the information described by dwInfo

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetDeviceInfo

5.4.5 SetDeviceTime

[Definition]

VARIANT_BOOL SetDeviceTime([in] long dwMachineNumber)

[Usage]

Set the local computer time to the device time. To set the specified time, see SetDeviceTime2.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetDeviceTime2, GetDeviceTime

5.4.6 SetDeviceTime2

[Definition]

VARIANT_BOOL SetDeviceTime2([in] LONG dwMachineNumber, [in] LONG dwYear, [in] LONG dwMonth, [in] LONG dwDay, [in] LONG dwHour, [in] LONG dwMinute, [in] LONG dwSecond)

[Usage]

Set the device time (or specify the time).

[Parameter]

dwMachineNumber

Device number

dwYear, dwMonth, dwDay, dwHour, dwMinute, dwSecond

Date and time to be set

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetDeviceTime, GetDeviceTime

5.4.7 GetDeviceTime

[Definition]

VARIANT_BOOL GetDeviceTime([in] long dwMachineNumber, [out] long* dwYear, [out] long* dwMonth, [out] long* dwDay, [out] long* dwHour, [out] long* dwMinute, [out] long* dwSecond)

[Usage]

Obtain the device time.

[Parameter]

dwMachineNumber

Device number

dwYear, dwMonth, dwDay, dwHour, dwMinute, dwSecond

Long pointers pointing to variables. The values are the date and time of the device.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetDeviceTime, SetDeviceTime2

5.4.8 GetSerialNumber

[Definition]

VARIANT_BOOL GetSerialNumber([in] long dwMachineNumber, [out] BSTR* dwSerialNumber)

[Usage]

Obtain the serial number of the device.

[Parameter]

dwMachineNumber

Device number

dwSerialNumber

Serial number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.4.9 GetProductCode

[Definition]

VARIANT_BOOL GetProductCode([in] long dwMachineNumber, [out] BSTR* lpszProductCode)

[Usage]

Obtain device name.

[Parameter]

dwMachineNumber

Device number

lpszProductCode

Device name

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.4.10 GetFirmwareVersion

[Definition]

VARIANT_BOOL GetFirmwareVersion([in] long dwMachineNumber, [out] BSTR* strVersion)

[Usage]

Obtain the firmware version of the device.

[Parameter]

dwMachineNumber

Device number

strVersion

Firmware version

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.4.11 GetSDKVersion

[Definition]

VARIANT_BOOL GetSDKVersion([out] BSTR* strVersion)

[Usage]

Obtain the SDK version.

[Parameter]

strVersion

SDK version

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.4.12 GetDeviceIP

[Definition]

VARIANT_BOOL GetDeviceIP([in] long dwMachineNumber, [out] BSTR *IPAddr)

[Usage]

Obtain the IP address of the device.

[Parameter]

dwMachineNumber

Device number

IPAddr

IP address

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetDeviceIP

5.4.13 SetDeviceIP

[Definition]

VARIANT_BOOL SetDeviceIP([in] long dwMachineNumber, [in] BSTR IPAddr)

[Usage]

Set the IP address of the device.

[Parameter]

dwMachineNumber

Device number

IPAddr

IP address

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetDeviceIP

5.4.14 GetDeviceMAC

[Definition]

VARIANT_BOOL GetDeviceMAC([in] LONG dwMachineNumber, [out] BSTR *sMAC)

[Usage]

Obtain the MAC address of the device.

[Parameter]

dwMachineNumber

Device number

sMAC

MAC address

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetDeviceMAC

[Supporting Device**]**

Black & white devices, TFT devices

5.4.15 SetDeviceMAC

[Definition]

VARIANT_BOOL SetDeviceMAC([in] LONG dwMachineNumber, [in] BSTR sMAC)

[Usage]

Set the MAC address of the device.

[Parameter]

dwMachineNumber

Device number

sMAC

MAC address

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetDeviceMAC

5.4.16 GetWiegandFmt

[Definition]

VARIANT_BOOL GetWiegandFmt([in] LONG dwMachineNumber, [out] BSTR *sWiegandFmt)

[Usage]

Obtain Wiegand format of the device.

[Parameter]

dwMachineNumber

Device number

sWiegandFmt

Wiegand format

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetWiegandFmt

5.4.17 SetWiegandFmt

[Definition]

VARIANT_BOOL SetWiegandFmt([in] LONG dwMachineNumber, [in] BSTR sWiegandFmt)

[Usage]

Set Wiegand format of the device.

[Parameter]

dwMachineNumber

Device number

sWiegandFmt

Wiegand format

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetWiegandFmt

5.4.18 GetCardFun

[Definition]

VARIANT_BOOL GetCardFun([in] LONG dwMachineNumber, [in] LONG* CardFun)

[Usage]

Obtain whether the device supports the RF card.

[Parameter]

dwMachineNumber

Device number

CardFun

Values: 1: The device supports only RF card verification. 2: The device supports both RF card verification and fingerprint verification. 0: The device does not support RF card verification.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.4.19 SetDeviceCommPwd

[Definition]

VARIANT_BOOL SetDeviceCommPwd([in] LONG dwMachineNumber, [in] LONG CommKey)

[Usage]

Set the communication password of the device. The communication password is stored in the device.

[Parameter]

dwMachineNumber

Device number

CommKey

Communication password

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetCommPassword

5.4.20 SetCommPassword

[Definition]

VARIANT_BOOL SetCommPassword([in] long CommKey)

[Usage]

Set the communication password of the PC. A connection can be set up only when the PC and the device use the same communication password.

[Parameter]

CommKey

Communication password

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetDeviceCommPwd

5.4.21 QueryState

[Definition]

VARIANT_BOOL QueryState([out] LONG *State)

[Usage]

Query the current state of the device.

[Parameter]

State

Current state of the device. Values are as follows:

0 Waiting

1 Registering a fingerprint

2 Identifying a fingerprint

3 Accessing menu

4 Busy (doing other tasks)

5 Waiting for writing data into card

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.4.22 GetVendor

[Definition]

VARIANT_BOOL GetVendor([in] BSTR* strVendor)

[Usage]

Obtain the vendor name of the device.

[Parameter]

strVendor

Vendor name of the device

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.4.23 GetDeviceStrInfo

[Definition]

VARIANT_BOOL GetDeviceStrInfo([in] LONG dwMachineNumber, [in] LONG dwInfo, [out] BSTR* Value)

[Usage]

Obtain the manufacturing time of the device.

[Parameter]

dwMachineNumber

Device number

dwInfo

This parameter can be set to 1 only.

Value

Manufacturing time of the device

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetDeviceInfo

5.4.24 GetPlatform

[Definition]

VARIANT_BOOL GetPlatform([in] LONG dwMachineNumber, [out] BSTR* Platform)

[Usage]

Obtain the platform name of the device.

[Parameter]

dwMachineNumber

Device number

Platform

Platform name

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.4.25 ReadAOptions

[Definition]

VARIANT_BOOL ReadAOptions([in] BSTR AOption, [out] BSTR* AValue)

[Usage]

Read the values of specified configuration parameters from the device. The parameters beginning with "~" are skipped.

[Parameter]

Aoption

Parameter name

Avalue

Value of the parameter described by Aoption

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetSysOption

5.4.26 GetSysOption

[Definition]

VARIANT_BOOL GetSysOption([in] LONG dwMachineNumber, [in] BSTR Option, [out] BSTR* Value)

[Usage]

Obtain the parameters from the device. Note: This function can be used to obtain the algorithm version used by the device.

[Parameter]

dwMachineNumber

Device number

Option

Parameter name. When the parameter is the character string "~ZKFPVersion", if the returned Value is 10, the current device uses Finger10.0; if the returned Value is null or 9, the current device uses Finger9.0.

Value

Value of the parameter described by Option

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SetSysOption

5.4.27 SetSysOption

[Definition]

VARIANT_BOOL SetSysOption([in] LONG dwMachineNumber, [in] BSTR Option, [in] BSTR Value)

[Usage]

Configure the parameters in the device.

[Parameter]

dwMachineNumber

Device number

Option

Name of the parameter to be set

Value

Value of the parameter described by Option

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetSysOption

5.5 Device Control Functions

5.5.1 ClearAdministrators

[Definition]

VARIANT_BOOL ClearAdministrators([in] long dwMachineNumber)

[Usage]

Clear all administrator privileges from the device.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.5.2 EnableDevice

[Definition]

VARIANT_BOOL EnableDevice([in] long dwMachineNumber, [in] VARIANT_BOOL bFlag)

[Usage]

Enable or disable the device. If the device is disabled, the fingerprint sensor, keypad, card modules, etc. are disabled.

[Parameter]

dwMachineNumber

Device number

bFlag

User enable flag. 1: Enabled. 0: Disabled.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.5.3 EnableClock

[Definition]

VARIANT_BOOL EnableClock([in] LONG Enabled)

[Usage]

Enable or disable the clock display with colon ":". If enabled, the device clock is displayed with a colon and synchronized to the main interface. If disabled, the clock is displayed without a colon.

[Parameter]

Enabled

Display control. 1: Enabled. 0: Disabled.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.5.4 DisableDeviceWithTimeOut

[Definition]

VARIANT_BOOL DisableDeviceWithTimeOut([in] LONG dwMachineNumber, [in] LONG TimeOutSec)

[Usage]

Disable the device for a period of time.

[Parameter]

dwMachineNumber

Device number

TimeOutSec

Duration of disabling the device

[Return Value]

Return True if it is successful, or return False.

[Related Function]

EnableDevice

5.5.5 PowerOffDevice

[Definition]

VARIANT_BOOL PowerOffDevice([in] long dwMachineNumber)

[Usage]

Power off the device.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.5.6 RestartDevice

[Definition]

VARIANT_BOOL RestartDevice([in] LONG dwMachineNumber)

[Usage]

Restart the device.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

PowerOffDevice

5.5.7 SleepDevice

[Definition]

VARIANT_BOOL SleepDevice([in] long dwMachineNumber)

[Usage]

Enable the device to enter hibernation mode.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

RestartDevice, PowerOffDevice, ResumeDevice, SuspendDevice

5.6 Online Registration Functions

5.6.1 StartEnroll

[Definition]

VARIANT_BOOL StartEnroll([in] LONG UserID, [in] LONG FingerID)

[Usage]

Enroll a user, and enable the device to enter enrollment state and wait until the user place a finger. Note: After this function is used, and a user enrolls the same finger three times to complete enrollment, the device may make no response. In this case, use StartIdentify to force the device to enter waiting state.

[Parameter]

UserID

ID of the user to be enrolled

FingerID

Index of the fingerprint of the user to be enrolled. The value ranges from 0 to 9.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

CancelOperation, StartVerify, StartIdentify

5.6.2 StartVerify

[Definition]

VARIANT_BOOL StartVerify([in] LONG UserID, [in] LONG FingerID)

[Usage]

Start 1:1 verification.

[Parameter]

UserID

ID of the user to be verified

FingerID

Index of the fingerprint of the user to be verified. The value ranges from 0 to 9.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

StartIdentify, CancelOperation

5.6.3 StartIdentify

[Definition]

VARIANT_BOOL StartIdentify(void)

[Usage]

Start 1: N comparison and enable the device to enter 1:N verification state.

[Parameter]

None

[Return Value]

Return True if it is successful, or return False.

[Related Function]

CancelOperation, StartVerify

5.6.4 CancelOperation

[Definition]

VARIANT_BOOL CancelOperation(void)

[Usage]

Cancel the current fingerprint enrollment state of the device.

[Parameter]

None

[Return Value]

Return True if it is successful, or return False.

[Related Function]

StartEnroll, StartEnrollEx

5.7 LCD and Card Operation Functions

5.7.1 WriteLCD

[Definition]

VARIANT_BOOL WriteLCD([in] LONG Row, [in] LONG Col, [in] BSTR Text)

[Usage]

Write LCD, that is, write character strings on the specified row and column on the device.

[Parameter]

Row

Start row

Col

Start column

Text

Text to be written on the LCD of the device

[Return Value]

Return True if it is successful, or return False.

[Related Function]

ClearLCD

5.7.2 ClearLCD

[Definition]

VARIANT_BOOL ClearLCD(void)

[Usage]

Clear all displays on the LCD of the device.

[Parameter]

None

[Return Value]

Return True if it is successful, or return False.

[Related Function]

WriteLCD

5.7.3 WriteCard

[Definition]

VARIANT_BOOL WriteCard([in] LONG dwMachineNumber, [in] LONG dwEnrollNumber, [in] LONG dwFingerIndex1, [in] BYTE* TmpData1, [in] LONG dwFingerIndex2, [in] BYTE* TmpData2, [in] LONG dwFingerIndex3, [in] BYTE* TmpData3, [in] LONG dwFingerIndex4, [in] BYTE* TmpData4,)

[Usage]

Write the information and fingerprint template of a specified user into the MF card. After this function is called, the MF card must be verified by the device.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

dwFingerIndex1, dwFingerIndex2, dwFingerIndex3, dwFingerIndex4

Index of fingerprint (0-3)

TmpData1, TmpData2, TmpData3, TmpData4

Fingerprint templates corresponding to fingerprints. TmpData1 cannot be null.

[Return Value]

Return True if it is successful, or return False.

[Related Function]

EmptyCard

5.7.4 EmptyCard

[Definition]

VARIANT_BOOL EmptyCard([in] LONG dwMachineNumber)

[Usage]

Clear the data from the MF card.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

WriteCard

5.8 Others

5.8.1 GetLastError

[Definition]

GetLastError([out] long* dwErrorCode)

[Usage]

Obtain the last error information.

[Parameter]

dwErrorCode

Error code returned. Values are as follows:

-100	Operation failed or data not exist
-10	Transmitted data length is incorrect
-5	Data already exists
-4	Space is not enough
-3	Error size
-2	Error in file read/write
-1	SDK is not initialized and needs to be reconnected
0	Data not found or data repeated
1	Operation is correct
4	Parameter is incorrect
101	Error in allocating buffer

[Return Value]

None

[Related Function]

Return True if it is successful, or return False.

5.8.2 GetHIDEventCardNumAsStr

[Definition]

VARIANT_BOOL GetHIDEventCardNumAsStr([out] BSTR* strHIDEventCardNum)

[Usage]

Obtain the number of the card that is latest used.

[Parameter]

strHIDEventCardNum

Number of the card lately used

[Return Value]

Return True if it is successful, or return False.

[Related Function]

OnHIDNum

5.8.3 CaptureImage

[Definition]

VARIANT_BOOL CaptureImage([in] VARIANT_BOOL FullImage, [in] LONG *Width, [in] LONG *Height, [in] BYTE *Image, [in] BSTR ImageFile)

[Usage]

Capture the image of the fingerprint on the fingerprint sensor.

[Parameter]

FullImage

Whether to capture the entire image. Return True if the device captures the whole image. Return False if the device captures only the fingerprint.

Width

Width of the image to be captured

Height

Height of the image to be captured

Image

Binary fingerprint image

ImageFile

Storage name of the specified fingerprint image to be captured (including the storage path)

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.8.4 UpdateFirmware

[Definition]

VARIANT_BOOL UpdateFirmware([in] BSTR FirmwareFile)

[Usage]

Upgrade the firmware. To use this function, you need to obtain corresponding firmware from technical engineers of Granding.

[Parameter]

FirmwareFile

File name of the firmware to be upgraded (including the path)

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.8.5 BeginBatchUpdate

[Definition]

VARIANT_BOOL BeginBatchUpdate([in] LONG dwMachineNumber, [in] LONG UpdateFlagl)

[Usage]

Start uploading data in batches. For example, if you call this function before uploading data such as user templates and user information, the SDK stores the data temporarily in the buffer during upload. Then, you can run BatchUpdate to import temporary data into the device.

[Parameter]

dwMachineNumber

Device number

UpdateFlagl

Fingerprint overwrite flag, that is, when a user fingerprint template is uploaded, if the fingerprint index has been specified for an existing fingerprint, the device prompts whether to overwrite the existing fingerprint template. 1: Forcibly overwrite, 0: Not overwrite

[Return Value]

Return True if it is successful, or return False.

[Related Function]

BatchUpdate, CancelBatchUpdate

5.8.6 BatchUpdate

[Definition]

VARIANT_BOOL BatchUpdate([in] LONG dwMachineNumber)

[Usage]

Start uploading data in batches. Generally, this function is used only after BeginBatchUpdate is used to upload related data.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

BeginBatchUpdate, CancelBatchUpdate

5.8.7 CancelBatchUpdate

[Definition]

VARIANT_BOOL CancelBatchUpdate([in] LONG dwMachineNumber)

[Usage]

Cancel uploading data in batches. Generally, this function can be used to release the buffer reserved for batch upload after BeginBatchUpdate and before BatchUpdate.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

BeginBatchUpdate, BatchUpdate

5.8.8 PlayVoice

[Definition]

VARIANT_BOOL PlayVoice([in] LONG Position, [in] LONG Length)

[Usage]

Play tones with the specified consecutive numbers. Tone numbers are determined by the device. You can view the tone numbers in Voice Test menu of the device. Generally, the values range from 0 to 11.

[Parameter]

Position

Start tone number

Length

End tone number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

PlayVoiceByIndex

5.8.9 PlayVoiceByIndex

[Definition]

VARIANT_BOOL PlayVoiceByIndex([in] LONG Index)

[Usage]

Play tones with the specified numbers. Tone numbers are determined by the device. You can view the tone numbers in Voice Test menu of the device. Generally, the values range from 0 to 11.

[Parameter]

Index

Number of the tone to be played

[Return Value]

Return True if it is successful, or return False.

[Related Function]

PlayVoice

5.9 OP1000 Functions

5.9.1 ReadAttRule

[Definition]

VARIANT_BOOL ReadAttRule([in] LONG dwMachineNumber)

[Usage]

Read attendance rules of the device. This function is applicable only to OP1000.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

5.9.2 SaveTheDataToFile

[Definition]

VARIANT_BOOL SaveTheDataToFile([in] LONG dwMachineNumber, [in] BSTR TheFilePath, [in] LONG FileFlag)

[Usage]

Save the data in the buffer as a file. This function is applicable only to OP1000.

[Parameter]

dwMachineNumber

Device number

TheFilePath

Path for saving the file

FileFlag

File type flag. Values are as follows:

1 Attendance record

2 User

3 Attendance rule

4 Department list

5 Shift

[Return Value]

Return True if it is successful, or return False.

[Related Function]

GetDataFile

5.9.3 ReadTurnInfo

[Definition]

VARIANT_BOOL ReadDPTInfo([in] LONG dwMachineNumber)

[Usage]

Read the shift information from the device. This function is applicable only to OP1000.

[Parameter]

dwMachineNumber

Device number

[Return Value]

Return True if it is successful, or return False.

[Related Function]

SaveTheDataToFile

5.9.4 SSR_OutPutHTMLRep

[Definition]

VARIANT_BOOL SSR_OutPutHTMLRep([in] LONG dwMachineNumber, [in] BSTR dwEnrollNumber, [in] BSTR AttFile, [in] BSTR UserFile, [in] BSTR DeptFile, [in] BSTR TimeClassFile, [in] BSTR AttruleFile, [in] LONG BYear, [in] LONG BMonth, [in] LONG BDay, [in] LONG BHour, [in] LONG BMinute, [in] LONG BSecond, [in] LONG EYear, [in] LONG EMonth, [in] LONG EDay, [in] LONG EHour, [in] LONG EMinute, [in] LONG ESecond, [in] BSTR TempPath, [in] BSTR OutFileName, [in] LONG HTMLFlag, [in] LONG resv1, [in] BSTR resv2)

[Usage]

Generate an .html ATT log of a specified user within the specified time range. This function is applicable only to OP1000.

[Parameter]

dwMachineNumber

Device number

dwEnrollNumber

User ID

AttFile

Attendance record file name

UserFile

User information file name

DeptFile

Department information file name

TimeClassFile

Shift information file name

AttruleFile

Attendance rule file name

Byear, Bmonth, Bday, Bhour, Bminute, Bsecond

Start time of the specified time range

Eyear, Emonth, Eday, Ehour, Eminute, Esecond

End time of the specified time range

TempPath

Path of other exception files to be processed

OutFileName

Output file name (including the path)

HTMLFlag

Log in HTML format. Values are as follows:

ATT log

Exception log

Statistics log

resv1

Reserved parameter

resv2

Name of ATT log

[Return Value]

Return True if it is successful, or return False.

[Related Function]

None

6 FAQs

6.1 How to Download Attendance Records?

First, use ReadGeneralLogData to read all attendance records and write them into the memory. Then, use GetGeneralLogData repeatedly to obtain attendance records. When GetGeneralLogData returns False, it means that all attendance records are obtained. Then, you can write the obtained records into database or display them in other forms to finish downloading. You can follow the same steps to down operation records.

6.2 How to Create a User Online?

First, use SetuserInfo to write user information (such as enrollment number, password, and name) into the device. Then, use SetUserTmpStr/SetUerTmp/SetEnrollDataStr/SetEnrollData to set fingerprint templates for the user. This method improves enrollment efficiency and is suitable when user information has been collected and stored in media such as database.

To upload user information and corresponding fingerprint templates in batches, use BeginBatchUpdata or SetUserInfo together with SetUserInfo, BarchUpdata, EnableDevice, or RefreshData. For details, see demo program.

6.3 How to Import or Download Data from USB Disk?

Many existing offline products support data download from USB disks. Many customers are concerned about data formats of USB disks. As the downloaded data formats are complex, Granding provides a tool for importing data from USB disks to database. Database is open for customers to download data. In addition, Granding also provides examples on how to process data files (*.dat, etc.) collected from USB disks and how to write the data into specified data files. All structs adopt 1 byte alignment mode.

Data in USB disks include user information, fingerprint templates, face templates, attendance records, and short messages. Detailed data structures are used in demo program. They are described briefly below:

User data structure:

```
typedef struct _User_{
    U16 PIN; //Internal number of a user
    U8 Privilege;
    char Password[5];
    char Name[8];
    U8 Card[5];         //ID No which used for store the relevant ID No
    U8 Group;           //the Group user belongs to
    U16 TimeZones;      //user can use time zone
    U32 PIN2;           //User ID
```

}

Data structure of 9.0 fingerprint template:

```
typedef struct _Template_{
U16 Size;               //fingerprint template length
U16 PIN;                //internal user ID, which corresponds to PIN2 in user table
BYTE FingerID;      // fingerprint backup index
BYTE Valid;
BYTE Template[MAXTEMPLATESIZE]; //maximize template length
}       //MAXTEMPLATESIZE        602 Bytes
```

Data structure of template.fp10:

```
typedef struct _Template_{
        U16 Size;                       //entire structure data size
        U16 PIN;                        //user ID
        BYTE FingerID;                  //fingerprint index
        BYTE Valid;                     //flag
        BYTE *Template; //template
}
```

Attendance record,

```
struct _AttLog_{
        U16 PIN;
        U8 PIN2[24];
        BYTE verified;
        time_t time_second;
        BYTE status;
        U32 workcode;
        BYTE reserved[4];
}
```

Exported as a text file:

attlog.dat format explanation:

segment:

BadgeNumber(employee number),

checktime,

DeviceID,

checktype(check status),

VerifyCode(verification ways:password or fingerprint or other)

Workcode

There is an Ascii code #9(Tab) between each segment. When development, move to the segment value you want to choose by "Tab".

If the device has the output data protection function, the serial number of the current device is displayed in first line and the hash value in the last line of the file to which attendance records are exported from USB disk.

Data structure of SMS

```
typedef struct _SMS_{
BYTE Tag;       //category
U16 ID;         //data content flag. 0 indicates that the record is invalid.
U16 ValidMinutes;       //valid minutes. 0 indicates that the record is permanently valid.
U16 Reserved;
U32 StartTime;      //start time
BYTE Content[MAX_SMS_CONTENT_SIZE+1];       //short message content
}       // MAX_SMS_CONTENT_SIZE        60 Bytes
```

Data structure between SMS and user pin//user->sms,udata.dat

```
typedef struct _UData_{
U16 PIN;            //0 indicates that the record is invalid
U16 SmsID;
}GCC_PACKED TUData, *PUData; //4Bytes
```

Data structure of face template:

```
typedef struct _FaceTmp_{
        U16 Size;//face template size
        U16 PIN;//user ID
        BYTE FaceID;//Face ID
        BYTE Valid;//flag
        U16 Reserve;//reserve
        U32 ActiveTime;
        U32 VfCount;//Verify Count
        BYTE FaceTmp[FACE_TMP_MAXSIZE]
}       //FACE_TMP_MAXSIZE=1024*2+512
```

6.4 How to Use Biokey to Write the Collected Fingerprint Templates Offline?

When a fingerprint is collected, Biokey usually obtains the fingerprint template during enrollment. For example, the currently enrolled fingerprint template can be obtained via OnEnroll. After

obtaining the fingerprint template, Biokey converts it into an offline fingerprint template. Then, the template can be written into the device.

6.5 How to Obtain All Information of All Users?

Use ReadAllUserID to read IDs of all users and write them into memory. Then, use GetAllUserID repeatedly to obtain EnrollNumber of users, and use GetUserInfo to obtain user information. If necessary, you can also use GetUserTmpStr to obtain the fingerprint templates in string form.

6.6 How to Connect to the Device?

During connection, the device can be regarded as an independent PC. However, the IP address of the device must match the IP address to be connected. Some devices, for example F4, support serial port connection and network connection. Therefore, during different connections, you need to set the device differently, modify communication mode, and set the controller switch to TCP/IP or RS232/485. Otherwise, the connection may fail. Sometimes, if the device fails to be connected due to busy serial ports, you can restart the program. If the application keeps connecting to the device without being manually disconnected, you can use DisableDeviceWithTimeOut to set the automatic disconnection time of the device. If some connections are used to download or modify data via serial ports or network, you can use EnableDevice to keep the device working and restore the connections after communication finishes, so as to maintain data consistency and avoid unknown errors.

6.7 Password Is Invalid After SetUserInfo Is Used.

After SetUserInfo is called, Password may be set to null. If so, password verification will fail. To keep the password unchanged when writing user information, use GetUserInfo to obtain user password and transmit the password value to the Password parameter of SetUserInfo before using SetUserInfo.

6.8 How to Convert an Online Template into an Offline Template?

Use FPTempConvertNew to convert the collected templates into offline fingerprint templates. See related descriptions of Biokey SDK for how to obtain the templates collected by Biokey. FPTempConvertNew is used to convert binary fingerprint templates. Parameters temp1 and temp2 are binary parameters. You can also use FPTempConvertNewStr to convert Biokey fingerprint templates of string type into offline fingerprint templates.

6.9 Demo Program Fails to Connect to the Device.

After the attendance management program is installed, users can connect to the device by using the attendance management program, but cannot connect to the device by using demo program. The reason is that DLL is copied to the directory of the attendance management program but registered in the installation directory during program installation. Generally, SDK loads controls from the system directory. Therefore, if the SDK version in the system directory is different from that in the attendance software directory, conflicts occur. (DLL function addresses of different

versions are different, but OCX functions are the same in programming. Therefore, the problem is found only during runtime.)

Caution: The common procedure for registering the SDK in the system is as follows:

1. If an SDK has been already registered in the system, run **regsvr32 /u zkemkeeper.dll** to unregister the original SDK.

2. Copy all DLLs to the system directory, for example, win2000 is located in winnt\system32.

3. Run regsvr32 "registration path\zkemkeeper.dll" to register the SDK.

4. Correctly load controls in development environment (learn the usage of development tool by yourself. Relevant details are omitted here).

5. Try to use the SDK of the same version in development or running environment.

6.10 Offline Fingerprint Device Keeps Working After Being Connected.

After connecting the SDK to the offline fingerprint device, use EnableDevice to keep the offline fingerprint device working (see EnableDevice), so as to maintain data consistency and avoid unknown errors. After the offline fingerprint device is working, the keypad and fingerprint sensor stop working. After communication finishes, disconnect the SDK from the device or use EnableDevice again to restore the offline fingerprint device to normal state.

DisableDeviceWithTimeOut is recommended.

6.11 Illegal Characters Are Displayed or Screen Display Is Abnormal After Non-English Names or Short Messages Are Uploaded to the Device.

First, check whether the device supports the specified language. For example, if the current language of the device is English, but an Arabic name is uploaded to the device, the name cannot be displayed normally. If the device already supports the language, but the name still cannot be displayed, use related functions to convert the user name into UTF-8 format (for example, use AnsiToUTF8() in Dephi), and then use SetUserInfo to upload the user name.

6.12 Card Management Problems

How to register a card in the device? How to obtain the user card?

The SDK has the cardnumber parameter. If this parameter is invisible in development environment, use GetStrCardNumber and SetStrCardNumber instead.

For a user enrolled in the device, the card number is a kind of user information. When SetUserInfo is used to set user information, cardnumber is automatically used as the card number and set for the user.

The procedure for registering a card is as follows:

Set cardnumber -> Upload user information

The procedure for obtaining the card number of a user is as follows:

Obtain information of the specified user -> Obtain cardnumber

Note: The card number is internally defined as four unsigned bytes of long type. If VB does not support four unsigned bytes, verification can be started after the last three bytes of the card number are input (if the last three bytes are different from each other).

6.13 Firewall or Router Traversal

In most cases, the device to be connected needs to traverse firewalls or routers, and UDP socket and port 4370 are used for SDK communication. Therefore, UDP and port 4370 must be enabled on firewalls or routers. If the device traverses gateways via port mapping, the device can be accessed via port number and IP address of routers. Generally, if UDP and port 4370 are enabled and can be pinged, the device can be connected. Certainly, in the case of data download, network connection must be considered. In addition, some devices that support SOAP ports can be accessed via embedded Web Server and SOAP.

Caution: The zem100 series products can traverse internet via port mapping. For zem200 products, as the devices run on Linux, they can be accessed after the gateway is configured if the local network environment supports gateway communication. Certainly, there are still some other methods for accessing the device, for example, VPN and IP mapping. The connection scheme should be selected according to specific network environments.

6.14 Uploading a Large Capacity of Fingerprints

Large capacity usually means over 1500 fingerprints. Some devices can hold 8000 fingerprints or more. Fingerprint templates must be uploaded in batches. In this mode, the upload is much faster. For how to upload fingerprint templates in batches, see descriptions of batch process function.

6.15 Differences between High-speed Upload and Ordinary Upload

In an ordinary upload, each time upload functions (such as SetUserinfo and SetUserTmpStr) are used, the SDK communicates with the device and uploads related data to the device.

In a high-speed upload, BeginBatchUpdata is used to create a temporary buffer to store the data to be uploaded in subsequent operations. Then, BatchUpdata can be used to upload all the data in the buffer to the device at a time. This mode greatly reduces communications between the SDK and the device, and raises the speed of large-capacity upload in particular.