User Software Development Manual

catalogue

Sound.....1

Contactformula2

VersionHistory3

Contents4

I.Software communication protocolinterface6

(i)Communicationsagreements.....6

1. Communication MessageFormat.....6

2. Face lock algorithm module message reception(H>>M).....7

3. Face lock algorithm module message sending(M>>H).....16

(ii)Functional implementationexample.....26

1. Master receives messageflow.....26

2. General message processingflow.....26

3. Up and down currentrange.....27

4. Face entryprocess.....27

5. EntryDescription.....28

6. Single framerecording.....29

7. Face verificationflow.....30

8. Encrypted communicationflow.....30

9. Photo Delivery RegistrationExample.....31

MID_ ENROLL_WITH_PHOTO.....31

(3)Supplementaryexplanation 34

1. MID_REPLY 34

2. MID_NOTE 35

3. MID_RESET 36

4. MID_GETSTATUS 36

5. MID_VERIFY 36

6. MID_ENROLL/MID_ENROLL_SINGLE 37

7. MID_ENROLL_ITG 38

8. MID_DELUSER 39

9. MID_DELALL 39

10. MID_GETUSERINFO 39

11. MID_ ENROLL_WITH_PHOTO 39

12. MID_DEMOMODE 40

13. MID_GET_ALL_USERID 40

I. Software communication protocol docking

(i) Communication protocols

Face Recognition module is in a subordinate position, and the main control needs to complete various functions through different instructions.

1. communication message format

The basic message format of communication between master control and binocular Face Recognition module is shown in the table below

Basic message format

SyncWord	MsgID	Size	Data	ParityCheck
2 bytes	1 byte	2 bytes	N bytes	1 byte

The following table provides a detailed description of each field.

Basic Message Format Field Details

field	length	explain
SyncWord	2bytes	Fixed message start sync word 0xEF0xAA
MsgID	1byte	MessageID(e.g.RESET)
Size	2bytes	Data size, unitbyte
Data	N bytes	Data corresponding to the message,parameters corresponding to the command message. 65535>=N>=0, N=0 indicates that this message has no parameters.
Parity Check	1 byte	Parity code of protocol, calculated as the entire protocol excluding SyncWord After that, the rest of the bytes areXORed bitwise.

2. Face lock algorithm module message reception (H>>M)

The complete protocol format received by the module is shown in the table below, and the master control shall send commands to the module according to this format.

H>>MProtocol Format Description

name	SyncWor	MsgID	Size		Data	ParityCheck
number of bytes	2 bytes	1 byte	2 bytes		N bytes	1 byte
content	0xEFAA	command	upper eight bits	lower eight bits	data	checksum

Command and data are defined in the following table.

H>>Mprotocol format data detailed description

command(* indicates carryingdata)	**code**	**data**		**explain**
		structural body	**content**	
RESET	0x10	not have		not have
GET STATUS	0x11	not have		not have
VERIFY	0x12	s_msg_verify_ data	pd_rightaway (1byte)	retain (Default:0)
			Timeout (1byte)	Unlock timeout (ins)
AUTO_VERIFY* (AI-10module applicable)	0x12	s_msg_Auto verify_data	at_verify (1byte)	Initiate low power auto-detection (yes:1 no:0) 1= Automatically recognize and output the result after judging the face or QR code;(Note: Palm vein recognition is only started after detecting the face) 0= Exit the mode.

				You can also exit this mode directly by issuing the 0x10 RESET command.
			Timeout (1byte)	single identification timeout (ins) Face detection, automatic recognition, At this time, if the person has walked away, xS exits recognition, andthe module continues to perform low-power face detection.

VERIFY*(FM226T-XF Series and AI-10module applicable)	0x12	s_msg_verify_data	pd_rightaway (1byte)	retain (Default:0)
			Timeout (1byte)	Unlock timeout (ins)
			Max_Recognition_times (1byte) (Optional)	The number of repeated recognition is limited,ranging from 0-255. The smaller this number, the fewer attempts to identify, and the faster the time to return to unknown users! The specific time is obtained by usertesting. This parameter is an extended parameter. If this parameter is not sent when the following is sent, the module defaultsto 30 attempts to identify.
ENROLL	0x13	s_msg_enroll_data	admin	Set as Administrator (yes:1no:0)
			user_name (32bytes)	Enter user name
			s_face_dir (1byte)	The direction that the user needs to enter isshown in thefollowing table "Face Direction Definition" for the physical parameters.
			timeout (1 byte)	Enter timeout time (ins)
		s_msg_enroll_	admin	Set as Administrator (yes:1no:0)
			user_name (32bytes)	Enter user name

ENROLL_SINGLE	0x1D	data	s_face_dir (1byte)	nonuse
			timeout (1 byte)	Enter timeout time (ins)
DELETE USER*	0x20	s_msg_delus er_data	user_id_heb (1byte) user_id_leb (1byte)	IDof the user to be deleted
DELETE ALL	0x21	not have		delete all users
GET USER INFO*	0x22	s_msg_getuser info_data	user_id_heb (1byte)	IDof registered user
			user_id_leb	

			(1byte)	
FACE RESET	0x23	not have		Clear Entry Status
MID_GET_ALL_USERID	0x24	00(1byte)		Get thenumber and IDof all registered users
MID_ENROLL_ITG	0x26	s_msg_enroll_itg	admin	Set as Administrator (yes:1no:0)
			user_name (32bytes)	Enter user name
			face_dir (1byte)	nonuse
			enroll_type (1 byte)	Registration type: interactiveentry or single frame entry
			enable_duplicate (1 byte)	0
			timeout (1 byte)	Enter timeout time (ins)
			Reserved (3 byte)	reserved bit
GET VERSION	0x30	not have		Get software version
INIT ENCRYPTIO*	0x50	s_msg_init_encryption_data	seed[4](4 bytes)	Control sends a random number
			mode	
MID_SET_RELEASE_ENC_KEY	0x52	s_msg_enc_key_number_data	enc_key_number [16]	encryption sequence
MID_SET_DEBUG_ENC_KEY	0x53	s_msg_enc_key_number_data	enc_key_number[16]	encryption sequence

MID_GET_SN	0x93	not have		Equipment unique serial number
READ_USB_UVC_PARAMETERS	0xB0	not have		USBtransfer parameter settings
SET_USB_UVC_PARAMETERS	0xB1	s_msg_usb_uvc_data	USB type (1byte)	USB1.1 : 0x11 USB2.0 : 0x20

(FM223SeriesFM888 Series applies)			Image rotation180degrees andimage mirroring (1byte)	BIT0: 0-no rotation, 1-rotation 180 degrees BIT1: 0-no mirror, 1-mirror flip
			JPEGImage QualityPercentage (1byte)	10 ~ 99
SET_USB_UVC_PARAMETERS FM226 Series, AI-10 module applicable)	0xB1	s_msg_usb_uvc_data	USB Type (1byte)	USB1.1: 0x11 USB2.0: 0x20 BIT7:0-Bulk 1-ISOC (Note:0xa0-USB2.0 ISOC 0x20-USB2.0&Bulk 0x11-USB1.1&Bulk 0x91-USB1.1&ISOC)
			UVC bit rate(1byte)	Uvc bit rate, unit Mbps,example: 0x18 = 24Mbps
			Quality of MJPG(1byte)	10-100, e.g. 0x4 b=75 This means 75% quality, andthe closer the value is to 100, the clearer the picture, but subject to rate control. Therefore,high-quality pictures need to be set to a high bitrate!
			image attribute (1byte)	BIT0:1Enable Mirrors BIT1:1Enable Reverse 180degrees
MID_SCAN_QR_CODE	0x70	s_msg_scan_QR	00 (1 byte)	Read QR code information
			timeout (1 byte)	timeout (Unit: s)
MID_SNAP&UPLOAD_IMAGE_1	0x71	s_msg_snapimage	00(1byte)	Capture photos and upload

MID_SNAP&UPLOAD_IMAGE_2	0x71	s_msg_snapimage	Seq(1byte)	upload photos (240*320pixel)Seq accumulates 1 (Seqreceives photo data from 1), and each packet fixedly transmits 1024 bytes of photo data. When the last packet is less than 1024 bytes, it is actually remaining. Number of bytes transmitted.
MID_SNAP&UPLOAD_FACEIMAGE_1	0x72	s_msg_snapfaceimage	00(1byte)	After capturing the face photo, compare it locally and complete the upload.

				After judging that the photos are qualified to the front face, they are automatically captured, compared locallyand uploaded. Face photo for 320*320pixel。
MID_SNAP&UPLOAD_FACEIMAGE_2	0x72	s_msg_snapfaceimage	Seq(1byte)	Upload face photos Seq accumulates 1 (Seqreceives photo data from 1), and each packet transmits 1024 bytes of photo data. When the last packet is less than 1024 bytes, it is transmitted according to the actual remaining number of bytes.
ENROLL_SNAPFACEIMAGE	0x73	s_msg_enroll_data	admin	Set as Administrator (yes:1no:0)
			user_name (32bytes)	Enter user name
			s_face_dir(1byte)	reserved
			timeout (1 byte)	reserved
MID_SNAP&UPLOAD_IMAGE_1	0x74	s_msg_snapimage_b	Seq (1byte)	Take a full-view photo, DPI isactive whenSeq=0
			DPI (1byte)	picture resolution 0 - 480x640 1 - 600x800 2- 1200 x 1600(only AI-10 effective)

MID_SNAP&UPLOAD_IMAGE_2	0x74	s_msg_snapimage_b	Seq(1byte)	When Seq>0, upload photo Seq accumulated 1 (Seqreceived photo data from 1), each packet fixed transmission of 1024 bytes of photo data, the last packet less than 1024 bytes as the actual residual Number of bytes transmitted.
MID_ENROLL_WITH	0xD7	s_msg_enroll_photo&id	00(1byte)	Send photos or feature registration (First order)
			00(1byte)	
			Phpoto len(4bytes)	Photo or Feature Length

	_PHOTO&ID_1 **(0xF7 is recommended for preference)** **Command)**			**BioType**(1byte) 0for normal photo mode, 1is encrypted photo mode, **2 Normal signature mode** (2048 bytes [color] or 4096[color + infrared]**),** **3 compressed signature mode** (1024 bytes**)**	The signature file is 4100 or 2052 or 1028 bytes, of which 4096 or 2048 or 1024 bytes are thesignature,4bytes ofcrc32 The initial vector value of CRC32is 0xffffff.
				UID(1byte) **1-100 (FM226)** **orUID**(2byte) **1-1000 (AI-10)**	Developer specified user ID. UID=0represents ID generated by module, FM226 is set to 1-100 is the user-specified ID, other values return failure.
				Duplicate_CkFg (1byte)	Repeat entry of inspection flags. If it is set to 1, the check is started,and the query will be made in the existing database. If it is related to the currently issued user ID (existing user), If the comparison is successful, it is allowed to issue coverage. Failure is returned if it is duplicated with another userid. Set to 0 to not check for duplicate entries.

			NameLen(1byte)	Name length, not greater than20,when 0or no such field default to none Name [Optional]
			Name String	User Name Content
MID_ENROLL_WITH_PHOTO&ID_2	0xD7	s_msg_enroll_photo&id	Seq heb (1byte)	When Seq>0, photo or feature data transmission Seq accumulates 1 (Seqsends photo data from 1) hoto datafor photo data , photo data packet size
			Seq leb (1byte)	
			Phpoto data[n] (n bytes)	

				MTU≤246
MID_UPGRADE_FW	0xF6	not have		Start USBflash drive Upgrade module firmware
MID_ENROLL_WITH_PHOTO_1	**0xF7**	s_msg_enroll_photo	00(1byte)	Send photos or feature registration (First order) where the signature file is4100or 2052 or 1028bytes, where 4096or 2048 or 1024words section is the signature,4 bytes crc32 (big endian) CRC32 initial vector value is 0xffffff
			00(1byte)	
			Phpoto len(4bytes)	
			BioType(1byte) 0for normal photo mode, 1for encrypted photo mode, **2 Normal signature mode (**2048 bytes [color] or4096[color + infrared]**),** **3 is the compressed signaturepatter n (**1024 bytes**)**	
		Extended parameter area: If you don't need to set the name, you don't need to send extended parameters. When the total length of parameters received by the module exceeds 7, it is considered	Name Len(1byte) (Optional)	Name string length 1-20over 20 willbe savedas 20
			Name String[n] (Optional)	name string n<20

		that there are extended parameters!		
MID_ENROLL_WITH _PHOTO_2	**0xF7**	s_msg_enroll_ photo	Seq heb (1byte)	Photo or feature data transmission Seq accumulation 1 (Seq sends photo data from 1), hotodata is photo data, photo data packet size MTU≤246
			Seq leb (1byte)	
			Phpoto data[n] (n bytes)	
MID_READ_FEATURE_1 (AI-10 module applicable)	0xFA	s_msg_read_feature_data_1	seq_id_heb (1byte)	packet sequence number (00 00)
			seq _id_leb (1byte)	
			user_id_heb (1byte)	UserIDto read (1~1000are face users,

			user_id_leb (1byte)	1001~2000 is palmprintvein)
			feature_type (1byte)	1: Color Face Template 2: Color + outer red face template 3: Palmprint Palm Vein Template
MID_READ_FEATURE_2 (AI-10 module applicable)	0xFA	s_msg_read_feature_data_2	seq_id_heb (1byte)	packet sequence number (Each packet is 246 bytes, calculate the total package number, and then send it from 1to accumulate in turn until the last packet is received)
			seq _id_leb (1byte)	
			user_id_heb (1byte)	UserIDto read (1~1000 for face users,1001~2000 for palmvein)
			user_id_leb (1byte)	
			Feature data[n] (nbyte)	Receive template data (Each packet is 246 bytes, and the last packet is the remaining actual length data)
MID_WRITE_FEATURE_1 (AI-10 module applicable)	0xFB	s_msg_write_feature_data_1	seq_id_heb (1byte)	packet sequence number (00 00)
			seq _id_leb (1byte)	
			feature_type (1byte)	1: Color Face Template 2: Color + outer red face template 3: Palmprint Palm Vein Template

		Extended parameter area:If you do not need to set the name, you can not send extended parameters. When the total length of parameters received by the module exceeds 3, it is considered that Extended parameters exist!	Name Len(1byte) (Optional)	Name string length 1-20over 20 willbe savedas 20
			Name String[n] (Optional)	name string n<20
MID_WRITE_FEATURE_2 (AI-10 module applicable)	0xFB	s_msg_write_feature_data_2	seq_id_heb (1byte)	packet sequence number (Each packet is 246 bytes, calculate thetotal package number, and thensend it from 1. Add up to the last packet.
			seq _id_leb	

			(1byte)	End of data transmission)
			feature_type (1byte)	1: Color Face Template 2: Color + outer red face template 3: Palmprint Palm Vein Template
			Feature data[n] (n byte)	template data (246 bytes per packet, last one) packet is the remaining actual length data)
MID_DUPLICATE_CHECK	0xFC	s_msg_duplicate_check	checkflag	0: No duplicate check 1: Duplicate check (Photo registration and distribution Whether to check duplication in case of characteristic mode)
DEMO MODE*	0xFE	s_msg_demo mode_data	enable (1byte)	enable:1 disable: 0

Face orientation is defined as shown in the table below.

Face Orientation Definition Description

st_face_dir(face orientation definition)	code	explain
FACE_DIRECTION_UP	0x10	Enter faces facing upward
FACE_DIRECTION_DOWN	0x08	Enter faces facing down
FACE_DIRECTION_LEFT	0x04	Enter faces facing left
FACE_DIRECTION_RIGHT	0x02	Enter faces facing right
FACE_DIRECTION_MIDDLE	0x01	Enter positive faces
FACE_DIRECTION_UNDEFINE	0x00	Undefined, default is positive

The definition of face entry command type is shown in the table below.

Face Entry Command Type Definition Description

enroll_type(face entry command type)	Code	explain
FACE_ENROLL_TYPE_INTERA CTIVE	0x0	interactive entry
FACE_ENROLL_TYPE_SINGLE	0x1	Single frame entry

Example: 0xEF 0xAA 0x10 0x00 0x000x10

H>>MProtocol Example

name	SyncWord	MsgID	Size		Data	ParityCheck
content	0xEFAA	0x10 (MID_RESET)	0x00	0x00	not have	0x10

This message is a RESET message sent by the master to the module, and the data length is0.

3. Face lock algorithm module message sending (M>>H)

The module sends three types of messages: REPLY, NOTE and IMAGE.

1) Transmission ofa REPLY message

The complete protocol of the REPLY message sent by the module to the master is shown in the table below.

M>>H REPLYMessage Protocol Format Description

name	SyncWord	MsgID	Size		Data			ParityCheck
number of bytes	2 bytes	1 byte	2 bytes		N bytes			1 byte
content	0xEFAA	MID_REPLY (0x00)	upper eight bits	lower eight bits	s_msg_reply_data			checksum
					mid (1byte	result (1byte)	data[0] (n-byte)	

midindicates the task currently being processed by the module. For example,whenmid=MID_ENROLL(0x13), it indicates that the message is a message replied by the module after processing the enroll task. mid details are shown in the table below.

M>>H REPLYmessage ormiddefinition descriptioninImageOrTemplatemessage

mid (*indicates carrying data)	code	data[0]		explain
		structural body	content	
MID_RESET	0x10	not have		not have
				The status of the module includes:

MID_GETSTATUS *	0x11	s_msg_repy_getstatus_data	status (1byte)	IDLE(0),BUSY(1),ERROR(2),INVALID(3)
MID_VERIFY* (Only if result resultsin MR_SUCCESS user_id)	0x12	s_msg_reply_verify_data	user_id_heb (1byte)	IDof authenticated user
			user_id_leb (1byte)	
			user_name (32 bytes)	user name
			admin (1byte)	Is it an administrator?
			unlockStatus (1 byte)	not have
MID_ENROLL* (Only iftheresult MR_SUCCESS user_id)	0x13	s_msg_reply_enroll_data	user_id_heb (1byte)	IDof registered user
			user_id_leb (1byte)	
			face_direction(1 byte)	Face registration in all directions in status
MID_ENROLL_SINGLE * (Only if result resultsin MR_SUCCESS user_id)	0x1D	s_msg_reply_enroll_single_data	user_id_heb (1byte)	IDof registered user
			user_id_leb (1byte)	
			face_direction(1 byte)	01 (indicates face entry)
MID_DELUSER	0x20	not have		not have
MID_DELALL	0x21	not have		not have
MID_GETUSERINFO*	0x22	s_msg_reply_getuserinfo_data	user_id_heb (1byte)	IDof registered user
			user_id_leb (1byte)	
			user_name (32bytes)	Registered user's name
			admin(1byte)	Is it an administrator?

MID_FACERESET	0x23	not have		not have
MID_GET_ALL_US ERID*	0x24	s_msg_reply_all_u serid_data	user_counts (1 byte)	Number of registered users,single A maximum of100per package

				End when the number returned is less than 100.
			users_id[MAX_USER_COUNTS*2] (≤100*2 bytes)	All registered user IDs,use two consecutive bytes to store an ID, save the uppereight bitsfirst
MID_ENROLL_ITG *(only if resultis MR_SUCCESS user_id)	0x26	s_msg_reply_enroll_data	user_id_heb (1byte) user_id_leb (1byte)	IDof registered user
MID_GET_VERSION*	0x30	s_msg_reply_version_data		version information
MID_INIT_ENCRYPTION	0x50	s_msg_reply_init_encryption_data	device_id[20](20 bytes)	deviceIDinformation
MID_SET_RELEASE_ENC_KEY	0x52	not have		not have
MID_SET_DEBUG_ENC_KEY	0x53	not have		not have
MID_GET_SN	0x93	Device_SN[32] (32bytes)		Equipment unique serial numberinformation, the first 8 bytes are valid
READ_USB_UVC_PARAMETERS (FM223SeriesFM888 Series applies)	0xB0	s_msg_usb_uvc_data	USB type (1byte)	USB1.1 : 0x11 USB2.0 : 0x20
			Image rotation 180 degreesand image mirroring (1 byte)	BIT0: 0-No rotation, 1-rotated 180degrees BIT1: 0-No Mirrors, 1-Mirror Flip
			JPEG image quality percentage(1	10 ~ 99

			byte)	
READ_USB_UVC_PARAMETERS (FM226 seriesapplicable)	0xB0	s_msg_usb_uvc_data	USB Type (1byte)	USB1.1: 0x11 USB2.0: 0x20 BIT7:0-Bulk 1-ISOC (0xa0-USB2.0&ISOC 0x20-USB2.0&Bulk 0x11-USB1.1&Bulk 0x91-USB1.1&ISOC)
			UVCbit rate	Code rate ofUVC, unit

			(1byte)	Mbps, for example: 0x18=24Mbps
			Quality of MJPG(1byte)	10-100 Current imagequality
			image attribute (1byte)	BIT0:1Enable Mirrors BIT1:1Enable Reverse 180degrees
SET_USB_UVC_PARAMETERS	0xB1	not have		Usbimage transfer parameter settingresults
MID_SNAP&UPLOAD_IMAGE_1	0x71	s_msg_reply_snapimage1	result(1byte)	0= success, others = failure
			Phpoto len(4bytes)	For example, 00 00 28 a3 representsthe length of the picture, and the large module is equal to 0x28 a3.
MID_SNAP&UPLOAD_IMAGE_2	0x71	s_msg_reply_snapimage2 (Note: This responseis returnedaccordingto ImageOrTemplate message protocol format MsgID Size Data	Phpoto data[n] (n bytes)	Size is the length of photo data in this packet. Each packet is fixed totransmit 1024 bytes of photodata. When the last packet is lessthan 1024 bytes, it is transmitted according to the actual number of remaining bytes. Photo data is photodata

MID_SNAP&UPLOAD_FACEIMAGE_1	0x72	s_msg_reply_snapfaceimage1	SnapFaceresult (1byte)	0= success, others = failure
			Phpoto len(4bytes)	For example, 00 00 28 a3 representsthe length of the picture, and the large module is equal to 0x28 a3.
			UserID（2bytes） (Big End Mode)	Search for local user ID. UserID = 0000, which means there is no corresponding user in the module. If it is not 0, it means there is a corresponding userLarge-endmode id_high id_low

MID_SNAP&UPLOAD_FACEIMAGE_2	0x72	s_msg_reply_snapfaceimage2(Note: This replyis returnedaccordingto ImageOrTemplate message protocol format MsgID Size Data)	Phpoto data[n] (n bytes)	Size is the length of photo data in this packet. Each packet is fixed totransmit 1024 bytes of photodata. When the last packet is less than 1024 bytes, it is transmitted according to the actualnumber of remaining bytes. Photo data is photodata
ENROLL_SNAPFACEIMAGE* (user_idonly exists if SnapFaceresolution result is 0	0x73	s_msg_reply_enrollsnapfaceimage_data	user_id_heb (1byte)	ID of registered userNote: Used in conjunction with 0x72 when no local user is found. This entry command does not check duplicates.
			user_id_leb (1byte)	
			face_direction(1 byte)	Default is0
MID_SNAP&UPLOAD_IMAGE_1	0x74	s_msg_reply_snapimage1	result(1byte)	0= success, others = failure
			Phpoto len(4bytes)	For example, 00 00 28 a3 representsthe length of the picture, and the large module is equal to 0x28 a3.

MID_SNAP&UPLOAD_IMAGE_2	0x74	s_msg_reply_snapimage2 (Note: This responseis returnedaccordingto ImageOrTemplate message protocol format MsgID Size Data)	Phpoto data[n] (n bytes)	Size is the length of photo data in this packet. Each packet is fixed totransmit 1024 bytes of photodata. When the last packet is less than 1024 bytes, it is transmitted according to the actualnumber of remaining bytes. Photo data is photodata
MID_UPGRADE_FW	0xF6	s_msg_reply_upgradefw	Progress (1byte)	Percentage of upgrade progress
MID_ENROLL_WITH_PHOTO_1	0xF7	s_msg_reply_enroll_photo_1	Seq heb(1 byte) Seq leb(1 byte)	packet sequence number
			UserID heb(1 byte) UserID leb(1 byte)	retain (User ID to be returned,default is 00 00)
MID_ENROLL_WITH_PHOTO_2	0xF7	s_msg_reply_enroll_photo_2	Seq heb(1 byte) Seq leb(1 byte)	packet sequence number

			UserID heb(1 byte)	UserIDwhen registration is successful (Only after the photo is sent, the corresponding UserID will be registered successfully, otherwise0)
			UserID leb(1 byte)	
MID_ENROLL_WITH_PHOTO_F **(Recommended for priority use) 0xF7 Command)**	0xD7	s_msg_reply_enroll_photo_f	Seq heb(1 byte)	packet sequence number
			Seq leb(1 byte)	
			UserID heb(1 byte)	Registered UserID (Only after the photo is sent, the corresponding UserID will be registered successfully, otherwise0)
			UserID leb(1 byte)	
MID_READ_FEATURE_1	0xFA	s_msg_reply_read_feature_data	result(1byte)	0= success, others = failure
			Feature len(4bytes)	Such as 00 00 16 D4 generation Table Palmprint Palmprint Vein Feature Length, Big End Mode equals 0x16 DE
MID_READ_FEATURE_2	0xFA	s_msg_reply_read_feature_data(Note: This reply followsthe ImageOrTemplate message protocol format MsgID Size Data Return)	Feature data[n] (n bytes)	Size is the characteristic data length of the packet. Each packet transmits246 bytes of characteristicdata. When the last packet is lessthan

				246 bytes, it is transmitted according to the actual number ofremaining bytes. Feature data ischaracteristic data
MID_WRITE_FEATURE_1	0xFB	s_msg_ reply_write_feature_data_1	Seq heb(1 byte)	packet sequence number
			Seq leb(1 byte)	
			UserID heb (1 byte)	retain (User ID to be returned,default is 00 00)
			UserID leb (1 byte)	
MID_WRITE_FEATURE_2	0xFB	s_msg_ reply_write_feature_data_2	Seq heb(1 byte)	packet sequence number
			Seq leb(1 byte)	

			UserID heb (1 byte)	UserIDwhen registration is successful (Only after the photo is sent, the corresponding UserID will be registered successfully, otherwise0)
			UserID leb (1 byte)	
MID_DUPLICATE_CHECK	0xFC	not have		not have
MID_DEMOMODE	0xFE	not have		not have

result indicates the execution result of this command, as shown in the following table.

M>>H REPLYresultdefinition description

result	code	explain
MR_SUCCESS	0	succeed
MR_REJECTED	1	module rejects the command
MR_ABORTED	2	Entry/validation algorithm terminated
MR_FAILED4_CAMERA	4	Camera failed to open
MR_FAILED4_UNKNOWNREASON	5	unknown error
MR_FAILED4_INVALIDPARAM	6	invalid parameter
MR_FAILED4_NOMEMORY	7	run out of memory
MR_FAILED4_UNKNOWNUSER	8	No user has entered.
MR_FAILED4_MAXUSER	9	Entry exceeds maximum number of users
MR_FAILED4_FACEENROLLED	10	Face entered
MR_FAILED4_LIVENESSCHECK	12	Vivo test failed
MR_FAILED4_TIMEOUT	13	Entry or unlock timeout
MR_FAILED4_AUTHORIZATION	14	Encryption chip authorization failed.

MR_FAILED4_READ_FILE	19	Failed to read file
MR_FAILED4_WRITE_FILE	20	Failed to write file
MR_FAILED4_NO_ENCRYPT	21	Communication protocol unencrypted
MR_FAILED4_NO_RGBIMAGE	23	RGBimages are not ready
MR_FAILED4_JPGPHOTO_LARGE	24	Photos are too large (photo registration)

MR_FAILED4_JPGPHOTO_SMALL	25	Photos are too small (photo registration)

Example: 0xEF 0xAA 0x00 0x00 0x05 0x13 0x00 0x00 0x03 0x1F0x0A

M>>H REPLYMessage Protocol Example

name	Sync Word	MsgID	Size		Data					Parity Check
					s_msg_reply_data					
content	0xEFAA	0x00 (MID_REPLY)	0x00	0x05	0x13 (MID_ENROLL)	0x00 (MR_SUCCESS)	0x00 (user_id_heb)	0x03 (user_id_leb)	0x1F (face_direction)	0x0A

This message indicates that this is a REPLY message returned by the module to the master control. The data part occupies 5 bytes. The entry is successful and the entered user ID is 3.

2) Send NOTE message

NOTE message is mainly used to actively return some information to the master control. Currently, NOTEmessage is mainly sent in three cases:

a) the module sends NID_READY to the master control during startup; b) the module sends face information to the master control during entry; c) the module sends face information to the master control during unlocking.

The complete protocol for NOTE messages sent by modules to the master is shown in the table below.

M>>HNOTEMessage Protocol Format Description

name	SyncWord	MsgID	Size		Data		ParityCheck
number of bytes	2 bytes	1 byte	2 bytes		N bytes		1 byte
content	0xEFAA	MID_NOTE (0x01)	upper	lowerei	s_msg_note_data		checksum
					nid (1byte)	data[0] (n-bytes)	

			eight bits	ght bits			

nidrepresentsthe execution result of the algorithm in the enroll process, as follows:

M>>H NOTEDescription of nid definitionin message

nid(*indicates carrying data)	code	explain
NID_READY	0	Module ready.
NID_FACE_STATE*	1	The algorithm executes successfully and returns face information_note_data_face

NID_UNKNOWNERROR	2	unknown error
NID_OTA_DONE*	3	not have
NID_EYE_STATE	4	not have

The data carried by NID_FACE_STATEmainly stores face information, and the details are as follows:

M>> Datadescriptioncorrespondingto NID_FACE_STATE in H NOTE

construction	s_note_data_face							
content	state	left	top	Right	bottom	yaw	pitch	roll
type	int16_t	int16_t	int16_t	int16_t	int16_t	int16_t	int16_t	int16_t
byte	2bytes	2bytes	2bytes	2bytes	2bytes	2bytes	2bytes	2bytes

state: Indicates the current state of the face, mainly including the following situations:

M>>H NOTE Statedescriptionof Datacorresponding to NID_FACE_STATEin message

Face state	**cod**	**explain**
FACE_STATE_NORMAL	0	Face normal
FACE_STATE_NOFACE	1	No faces detected.
FACE_STATE_TOOUP	2	The face is too close to the top edge of the picture, so it cannot be entered.
FACE_STATE_TOODOWN	3	The face is too close to the bottom edge of the picture, so it cannot be entered.
FACE_STATE_TOOLEFT	4	The face is too close to the left edge of the picture, so it cannot be entered.
FACE_STATE_TOORIGHT	5	The face is too close to the right edge of the picture, so it cannot be entered.
FACE_STATE_FAR	6	Face distance too far, failed to enter

FACE_STATE_CLOSE	7	The face distance is too close to enter.
FACE_STATE_EYEBROW_OCCLUSION	8	Eyebrow occlusion
FACE_STATE_EYE_OCCLUSION	9	Eye occlusion
FACE_STATE_FACE_OCCLUSION	10	Face occlusion
FACE_STATE_DIRECTION_ERROR	11	Wrong face direction entered
FACE_STATE_EYE_CLOSE_STATUS_OPEN_EYE	12	Eyes open state detected in eyes closed mode

FACE_STATE_EYE_CLOSE_STATUS	13	eyes-closed state
FACE_STATE_EYE_CLOSE_ UNKNOW_STATUS	14	Closed eyes cannot be determined in closed eyes mode detection

The other members ofs_note_data_face have the following meanings:
left: Distance from the face frame to the leftmost side of the picture (negative number means the face frame is beyond the leftmost side of the picture) top: Distance from the face frame to the top of the picture (negative number means the face frame is beyond the top of the picture) right: Distance from the face frame to the rightmost side of the picture (negative number means the face frame is beyond the rightmost side of the picture)

bottom: Distance of the face box from the bottom of the picture (negative number means the face box is beyond the bottom of the picture)

yaw,pitch,rollindicates the direction of rotation. Yawis negative for left head,yawis positive for right head;pitchis negative for upward head, pitch is positive for downward head;roll is negative for right tilt, roll is positive for left tilt.

Example: 0xEF 0xAA 0x01 0x00 0x01 0x000x00

M>>H NOTEMessage Protocol Example

name	SyncWord	MsgID	Size		Data		ParityCheck
content	0xEFAA	0x01(MID_NOTE)	0x00	0x01	s_msg_note_data		0x00
					0x00(NID_READY)	not have	

This message indicates that this is a NOTE message sent by the module to the master, the data length is 1 byte, and the module is ready to receive other commands.

3) ImageOrTemplateMessage Sending

The complete protocol for sending Image or Template messages from the module to the master is shown in the table below.

M>>H ImageOrTemplate Message Protocol Format Description

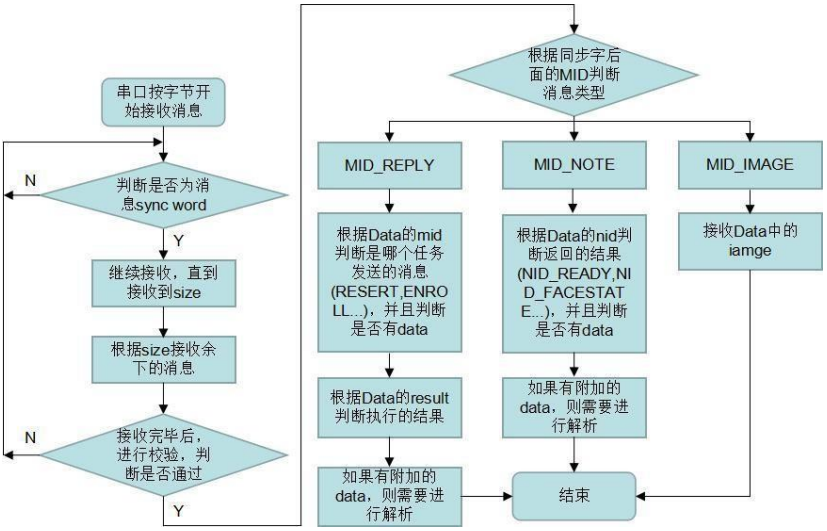
name	SyncWord	MsgID	Size		Data	ParityCheck
number of bytes	2 bytes	1 byte	2 bytes		N bytes	1 byte
content	0xEFAA	mid	high height	low height	image data	checksum

mid indicates the task currently being processed by the module. For example, when mid = MID_SNAP& UPLOAD_FACEIMAGE_2(0x72), it indicates that the message is a message replied by the module after processing the capture task. For detailsof mid, see "**Description of mid definition inREPLYmessage orImageOrTemplate**message".

(ii) Examples of functional implementation

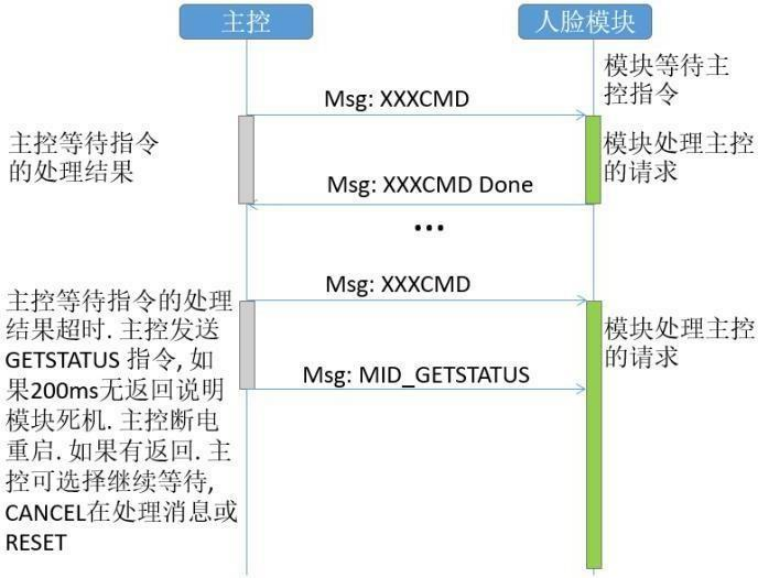
1. Master receives message flow

The message sent by the master control receiving module may follow the flow shown in the figure below.



Message flow chart of main control receiving module

2. General message processing flow



General Message Processing Flowchart

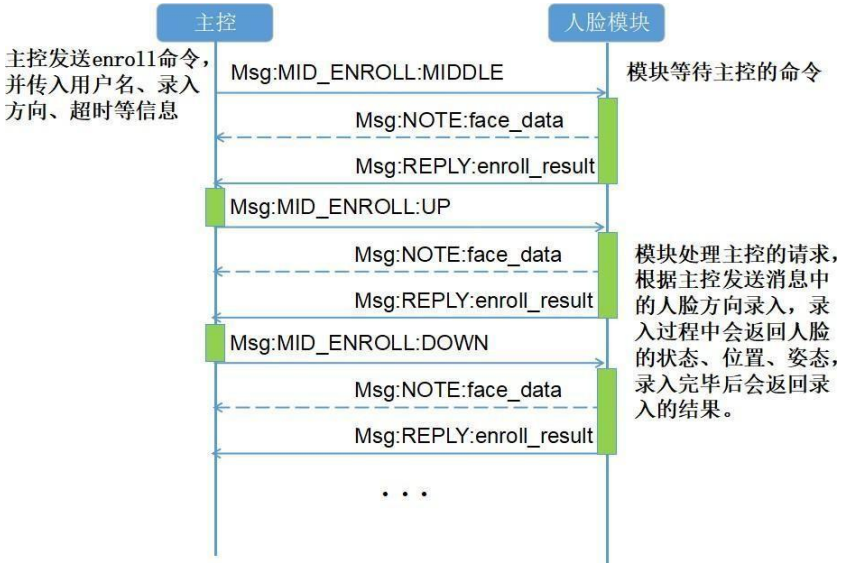
3. up-down flow

The power-on and power-off of the main control active control module are shown in the figure below. When the module is powered on, it enters the startup process. When the module enters thestandbystate,it notifies the master controlMSG:NOTE:READY. The master control starts to send

Flow chart of

command messages, and the module processes and returns the results. Module no message sent or timeout, master control can power off.

4. Face Entry Process



Input face flow chart

In the input process, the face lock algorithm module does not limit the order of the input direction, and the user can arbitrarily combine the order of the input direction. The above figure takes one of the orders as an example to illustrate.

When the master control sends an instruction to input a face in a certain direction, the module starts to work and returns the state, position and posture of the face in real time.

Information (returned in the form of NOTE message) The user can remind the entry person to adjust the posture position according to this information. When the entry is successful, themodule willreturn the entry result in the form of REPLY message. If the entry is unsuccessful, the module will return the reason for the failure accordingly.

It is worth noting that when the first frame image collected by the module is successfully entered, the result will be returned directly inthe form of a REPLY message. During the entry process, the entry can be terminated by the FACE RESET command, and the previous entry status will also be cleared.

If there is a sudden power failure during the entry process, the previously entered faces will not be saved.

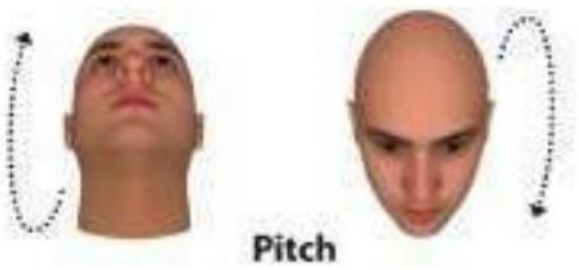
5. entry instructions

The angle of face input in each direction is described as follows:

Face Input Angle Description

Enter direction	deflection angle
positive face	Facing the camera, no deflection angle
up	5~55degreesupward deflection of the front face
downward	5 - 55degrees downward
towards the right	Turn your face 8~60degrees to theright.
towards the left	Turn your face 8~60degreesto the left.

The upward and downward deflections are shown below.



Face input up and down schematic diagram

Left and right deflections are shown below.

Face input left and right schematic diagram

Please do not use the method shown below.

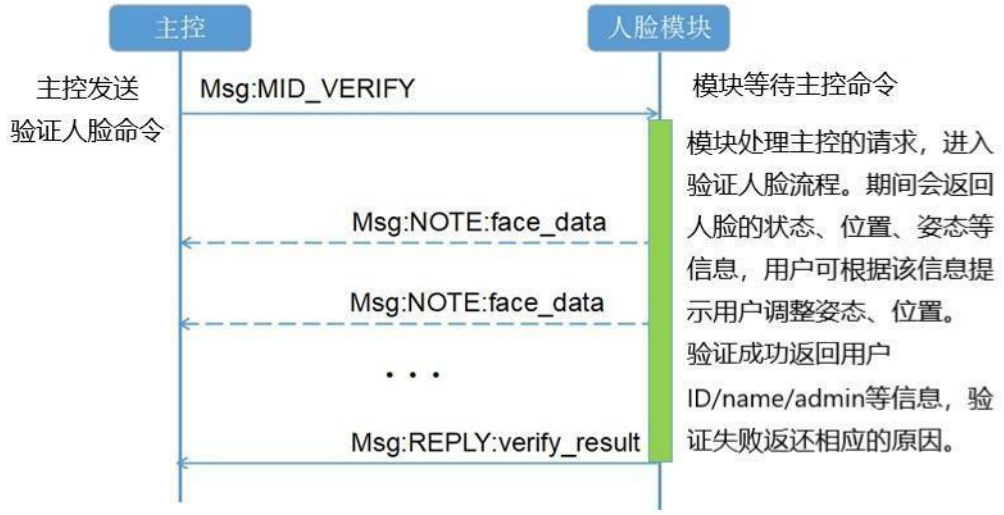
Schematic diagram of face entry error

We suggest that users slowly turn their heads in a certain direction during the actual entry process, and keep turning, please do not turn too fast or immediately return to the face state.

6. Single frame entry

The difference between single-frame entry and entry process is that single-frame entry only needs one face to enter successfully. without having to interact five times to complete registration.

7. Face verification process



Face verification process

8. Encrypted communication flow

The encrypted communication flow is shown in the figure below.

① The main control powerson FM21xseries Face Recognition module

②FM21xseries Face Recognition module sendsREADY (plaintext)to master control

③ The module needs to call the MID_SET_RELEASE_ENC_KEYcommand to set the 16bytes sequence required by the private protocolfor the first time.

④ The master control uses random numbers to generate random sequences and sends them to FM21x series Face Recognition module(4bytes)

④ FM21xseries Face Recognition module and master control adopt customized private protocol(generate 16bytes password)according to this random sequence. Both parties use the same algorithm to generate the same password. After that, both parties used this password to encrypt and decrypt this session.

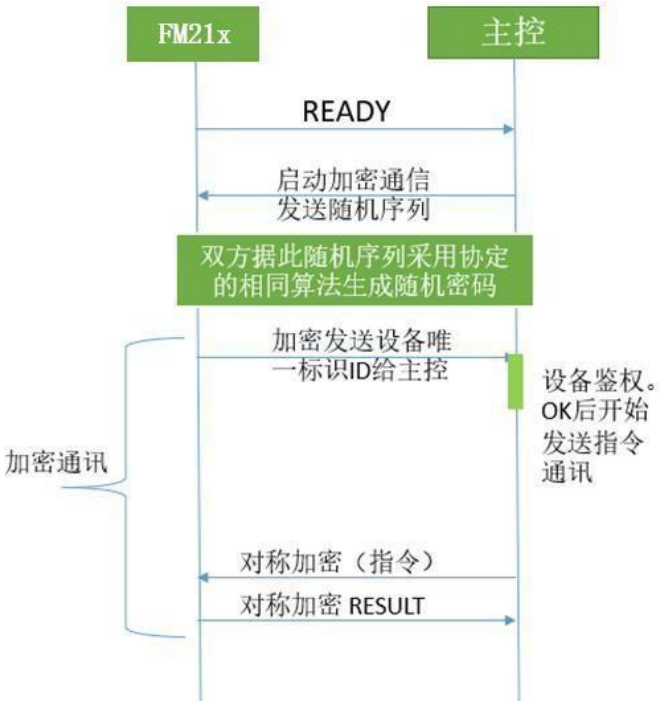
Face Recognition module of FM21x series returns to status andsends product serial number to main controller with random cipher AES/SMPL encryption.

7. The master control decrypts, identifies the equipment ID, confirms the identity, and then begins to process the demand instructions, input or verify, etc.

The master uses randomly generated password, AES/SMPL encryption method to encrypt instructions and data and send them to FM21x series face recognition module.

FM21xseries Face Recognition moduledecrypts the password decrypted in step 4, judges the legality of the instruction and processes the instruction.

FM21xseries Face Recognition module encrypts theinstruction processing result and data with the password decrypted in step 4, and sends it to the main control.



Encrypted communication flow

9. Photo Delivery Registration Example

MID_ ENROLL_WITH_PHOTO

Master (PC) initiates photo registration, Seqis set to 0, Photo datais 4 bytes (big end mode)Photo size, Photo type
(normal photo 0, encrypted photo 1)is 1 byte
[Seq+ photo
length]module
returnsResultandSeq

WhenResultisMR_SUCCESS, the master control accumulatesSeqby1 (Seqstarts to send photo datafrom1)Photodatais photodata, and the packet size of photo data is 246 bytes. When the last packet is less than 246 bytes, it is sent for its actual byte number.

When the photo is sent, the module finally returns the registration result. When the ResultisMR_SUCCESS, it also carries the userID.

Example: Send a small photo with a photo size of2763, without the user name (Send:PC to module, Recv:module toPC)

Send:ef aa f7 00 07 **00 00** 00 00 0a cb 01 30 《**Seq=0,photolen=0x00000acb,type=1**》

Recv:ef aa 00 00 06 f7 00 **00 00** 00 00 f1 **《result=0,Seq=0,userid=0》**

Send:ef aa f7 00 f8 **00 01** 5b b5 01 35 7b 8e 76 99 ed 2b fe d4 7a 9e 3c de a4 6c fe d5 84 45 3c 9c a4 65 f8 d3 7c 98 39 d7 a3 6a f9 dc 72 96 36 d3 b0 60 f2 de 70 92 25 cd b7 62 ea c8 61 81 22 c2

be 71 e2 f5 5f b0 1b ff 86 41 dd c9 67 b6 0b f6 88 5d cf e1 4f aa 23 f8 9d 50 c6 e7 47 b0 0f eb 96 92 25 d5 38 9f 35 d6 ad 61 f5 d9 63 93 31 c7 96 4c e2 f4 49 ac 0e ed 96 5f cc e7 49 ac 0e ed 96 5f cc e7 49 ac 0e ed 96 5f cc e7 49 ac 0e ed 96 5f cc e7 49 ac 0e ed 96 5f cc e7 49 ac c3 1f a4 7c f6 d5 03 9e 44 dc a5 4f fe d7 6a 9f 3f ce a5 92 3a d5 64 9e 3c de a1 6c ff d4 7a 9f 3d df a4 6d fe d5 7b 9e 3c de a6 6e fa d0 7d 99 34 d6 ae 66 01 11 7b 2b 2c df a6 6c fd d6 79 9a 3f da a1 69 fa d5 7b 9f 41 de a6 6e fe d1 6a 9b 2e fe 95 2c f8 c6 2a ff 92

Recv:ef aa 00 00 06 f7 00 **00 01** 00 00 f0

Send:ef aa f7 00 f8 **00 02** 3b fd d5 79 cc 54 ea 3f 34 fc e6 dc 3f c0 29 4f cc fb 97 0f 8c 57 72 94 2a c8 bc 74 e4 f0 5d b9 14 f6 8e 59 cb e3 4c a6 05 e5 e7 29 bb 93 3c d6 75 95 f7 39 ab 83 2c c6 65 85 c7 09 9b b3 1c f6 55 b5 d7 19 8b a3 0c e6 45 a5 27 e9 7b 53 fc 16 b5 55 36 fe 6a 40 ed 09 a4 46 3e cf 5d 71 de 38 9b 77 0d c7 4c 66 cf 2b 8a 68 1c d4 44 17 b8 5a f9 19 63 a5 37 1f a9 4d e8 0a 72 ba 26 0c a1 7f de 3c 40 88 18 32 93 77 d6 2e 56 9e 0a 20 8d 69 c4 26 5e 92 3a d5 64 9f 3c dc a5 6c ff d4 7a 9f 3d de a5 6d fe d5 7b 9e 3c de a6 6e fa d0 7d 99 34 d6 ae 66 01 11 7b 2b 2d df a6 6c fc d1 7f 9d 38 d8 a1 69 fa d5 7a 9c 4b df a5 6f fd c4 7f 9b 1d ee a2 7f bf 84 7c ff 4d cc 86 5f 7f dd 6f dc ad 7e 15 ac f7 f6 48 cc cc ca c6 1f 2f df 6d ba 08 3e 81 9c e9 cd 94

Recv:ef aa 00 00 06 f7 00 **00 02** 00 00 f3

Send:ef aa f7 00 f8 **00 03** 62 84 1a f8 8c 44 d4 e0 4d a9 04 e6 9e 2e ba 90 3d d9 74 96 ee 3e aa 80 2d c9 64 86 fe 0e 9a b0 1d f9 54 b6 ce 1e 8a a0 0d e9 44 a6 de ef 7d 51 fe 18 bb 57 2d e7 6c 46 ef 0b aa 48 3c f4 64 77 d8 3a 99 79 03 c5 57 7f c9 2d 88 6a 12 da 46 6c c1 5c ff 1b 61 ab 39 1d b2 54 ee 0c 70 b8 28 02 a3 47 e6 3d 47 89 1b 33 9c 76 d5 35 56 9e 0a 20 8d 69 c4 26 5e 92 24 d5 77 9d 3d df a6 7c fd c4 7b a1 3c 2b ff 02 0b 9e 8f 37 63 7e 0c 40 a5 e4 fa cd 03 95 50 a2 da a9 26 b4 5d 01 05 e6 5b 9e e7 cc 3c a1 31 38 14 bc 1e c6 df f3 48 6d 52 6e f0 6d ae ce 95 e3 13 2b 6e ad 95 c7 8b 14 6b cf 29 01 fe 54 e9 bc 54 4e 84 9e 90 23 e3 9e 54 e7 a5 d0 e3 2d 74 ed 83 ed d1 2c 60 94 3a 33 b2 2e 4a 30 75 bc 8d b1 14 5b 69 e8 e8 37 46 92 fe 83 52 51 cb b4 32 36 13

Recv:ef aa 00 00 06 f7 00 **00 03** 00 00 f2

Send:ef aa f7 00 f8 **00 04** 25 4d 60 7a 84 a2 1a e1 25 6a e6 c4 06 3b 56 a9 9d cb 27 ec 2e ee 85 3f 3e d9 88 0b 38 a4 53 c0 97 b7 d2 55 7f e5 3c 07 6f 01 a4 d0 a8 2c 53 20 bb d9 f2 b2 5d 3c c4 a7 84 c4 29 01 86 b4 04 8b d5 6a 01 14 99 a1 51 64 15 99 19 10 88 8f 9e bc be c7 a1 81 ac 8a 36 15 54 e3 fc 7b 97 59 34 e4 9d f1 0a cb 92 aa a4 ff 65 ae 0e 8c 73 23 e7 df 83 aa b9 3c 83 64 cf 26 a2 12 f8 ad a1 6e 27 86 14 a8 fe d9 5f b3 84 e5 8f 42 47 2d 22 7b 3a 8b 4e 78 2c 0f 65 b1 d0 0c 16 26 c0 23 b7 a9 92 62 8b fb 92 3a 5b a9 6c a4 13 39 46 9b 11 29 3e db 1d 64 45 4b 5a c6 18 d3 62 ae 2f c9 4b 84 65 a2 18 be fd d1 a7 89 de 3c a2 6f 58 8c 54 9d 2a 6a c3 3f c0 5f aa 1e 9b e8 a5 a3 a7 5a b3 85 cf 9e a6 0d a3 36 ea 75 f0 9d d4 6e a1 34 ce d9 8e 9b 8c 98 2c 07 8c 6b 4e

Recv:ef aa 00 00 06 f7 00 **00 04** 00 00 f5

Send:ef aa f7 00 f8 **00 05** 80 4b b9 d8 bf 28 6f a6 34 f1 7d 22 ab 6a 53 2f 9a 98 2e 8f 26 59 45 b0 42 2b b1 76 52 74 9b 0f 73 47 2c 26 29 45 e3 44 ad a8 38 a4 74 d7 bb 75 5e 90 b9 8a b7 c9 41 c5 12 bc 62 cd f1 b7 92 bb d0 a2 20 74 46 f4 7b e3 70 4a 53 a3 d8 1e 4c 34 e1 48 ea 39 78 f2 e7 7a fb b0 6f 8c 4d 11 f4 72 94 13 a2 07 1f 3c b0 4e 3f e0 b2 46 f9 2f 14 8f ac 85 84 47 44 af c5 89 20 e6 1b 35 75 b4 37 da 8b 5e 62 c8 bd c8 d3 04 80 12 df 82 54 4e 9f a4 73 f2 a3 88 a7 6b 0c 25 8f 78 49 c6 bd e6 94 3a 29 f4 86 f5 29 ab f8 ce a6 33 8a d2 a8 ab 38 68 54 c5 da 1f fa e3 75 1a d7 70 a9 fe b2 26 09 99 d3 f3 63 3c f6 e6 af c3 0f 6f c3 7c 90 b6 6c d2 8f 78 8e e4 c9 10 89 91 47 58 98 be cf cf e8 ac fd ff a0 11 23 11 0d a2 16 04 5a f3 b3 06 dc 45 c7 da 36 16 ca 82 f9

Recv:ef aa 00 00 06 f7 00 **00 05** 00 00 f4

Send:ef aa f7 00 f8 **00 06** ee e3 2d f1 78 ca e6 ae a1 17 7c 8a 74 7d 93 56 46 7c 07 9d 03 14 c2 4e 91 86 2a 34 ea 13 09 e7 c5 96 fa 12 fe ff 96 fc 44 92 2d bf 10 d4 44 e8 d9 58 77 88 ad 8e 1f 7d 91 a5 74 1a b9 58 56 ba ec b9 2e 21 2f 81 36 54 4a 12 22 78 52 58 ef 8b bb dc c5 86 9c cb 18 56 d4 f5 c1 08 55 c1 68 f0 f4 6a 06 e2 64 a2 46 03 8d 20 a4 10 e4 70 99 f7 92 06 d2 b6 c6 97 01 00 c9 7e 00 6c 08 bc a7 ec 91 f6 65 84 00 af ad c0 89 77 e3 06 ea 46 b1 32 fd 98 ae c6 6c be fd 3a a9 10 f2 f9 a3 44 bb 40 d4 85 08 87 33 c0 88 bf 1b fb f2 4f 97 a2 8d 6d de 2d c2 01 89 09 36 f7 61 27 3b 3e 86 bb dc 14 f5 bf ec fa 8c 79 82 43 1c 30 24 5c 2d 78 08 6e 44 79 5f 3d 89 08 7f f3 8d 94 92 a1 70 28 c8 01 d5 87 f9 de 5f 90 56 50 87 93 1c 9f 78 e2 e9 af e2 69 a1 bc 26 9c 0c

Recv:ef aa 00 00 06 f7 00 **00 06** 00 00 f7

Send:ef aa f7 00 f8 **00 07** d5 e1 75 5d ff f0 6d 38 51 10 3d 1e bc c0 17 38 c7 26 b3 a5 6a 81 2c 1d b6 4a 53 9c 82 a1 60 be ec 24 ee f0 8a b2 cf ea a8 b8 89 70 27 36 5a 42 77 12 f6 43 45 2b b3 a6 8c c0 16 63 17 fa 9f 55 7e 7d 55 d2 28 9c 61 9f 0a f0 c6 cc 81 73 5b 14 d8 f4 4c 99 93 b3 4f 12 3e 73 67 a5 e0 ba db a9 c0 33 99 58 e6 56 28 9f 10 c9 57 f3 86 05 c9 12 59 98 11 c6 89 de c3 5a 79 29 91 46 b8 41 74 ad 46 b5 72 b1 42 03 c3 5b f4 5d ad c7 08 0a c3 4b 9e 46 42 5c 5e 3b 55 03 05 0d 4d d8 7d 3b 23 88 07 4b 10 95 43 f3 b7 ea ad 31 0d 39 19 45 cb 1f f4 bc 96 b5 ca 8f 19 c1 28 ac 54 74 97 d4 61 cb 3f 0a 56 b2 80 dd 27 90 70 0b 4f e9 d6 62 b2 b6 64 50 a6 13 65 e5 a0 32 d6 c1 ce 89 9f 45 9a f5 1e 82 0d e1 63 ee a7 b7 e8 ea a1 52 54 7f b4 6f c2 4f 55 2e 43 21 0f

Recv:ef aa 00 00 06 f7 00 **00 07** 00 00 f6

Send:ef aa f7 00 f8 **00 08** bd 13 35 d7 fb 09 0c b0 bc 56 0b eb 30 d0 bc 4e 53 bd 92 47 b3 91 2d 50 ed 7a d6 f8 e3 6e cd e3 d5 1a e5 56 c8 08 60 a3 85 87 f1 b9 38 ab 52 ad 52 ed 8f 46 99 40 b5 cd 38 ac a1 ae 17 7a 3a bd b7 49 e7 78 ae e7 06 da e6 cd 7d 2c 8d 65 85 db 4e 0d d8 7b fc 7d 59 67 cd d7 43 82 6a 0a 10 0e 1b 1f 8a ca 89 bf 1f b2 e2 27 13 2e 92 ac 0d 7f 55 b8 58 18 96 0d fc 2d 31 43 f6 d7 c3 df cc 0b 53 4c 3b 0e 52 c7 53 c6 c0 97 45 81 b0 4e ea ac 8d e2 80 a4 0e e4 c7 bd f2 84 8d fc 9c 5e 64 c0 dd c3 7c d4 b4 77 03 7f 9a b2 a2 67 ea 25 11 ca 30 19 c2 9f b3 28 2a 6d 37 29 d0 31 5e f4 3b df 7a 9c d1 da 74 3e b0 d8 91 e3 48 c3 e4 bb 76 0a 63 58 12 54 17 17 e8 90 d6 77 f1 74 04 73 f4 6e cc da 27 48 c0 e9 46 25 a4 f6 a6 f5 16 01 94 be d0 92 5b 49 77 7a 83

Recv:ef aa 00 00 06 f7 00 **00 08** 00 00 f9

Send:ef aa f7 00 f8 **00 09** 7e e6 67 48 27 03 9d d1 ec 51 0d 86 2c c6 8d 0f d8 de b2 44 7d 55 48 8e 59 bd d5 f3 94 08 d7 0c 09 07 44 2c 16 7a db e4 9a 31 60 69 2d c3 e5 dc d4 91 15 18 e5 59 8d c8 d2 0a b7 f5 7b 64 e9 f7 41 6b 0f cf 18 16 e8 3b 70 9f 1a d7 4a d0 ba 06 92 7d 1e c5 b8 0e 57 fc df f4 0d 2d 90 d5 c2 0e 92 44 c0 f3 7c 86 14 9f 83 87 b8 ff ee 40 84 50 1f 1e 8e ec 5f 04 67 00 ab b7 07 59 b9 ac fc 43 28 9e 74 13 06 85 55 a3 a2 08 a0 0c df cb ad e2 2b e5 8a 18 31 c9 8b 59 44 59 7d ec 60 d6 23 5e 62 60 94 8d 27 87 1c db 43 4e 74 15 f9 5f 29 1e a4 4d 30 a5 d8 8b 8f b5 98 d8 3e 3c 37 d3 1f 9e 00 8f 5e b3 94 d6 67 eb 72 51 ad 91 49 78 ac 7f 69 4e 6d 26 16 5e 22 06 09 3a 14 fe 44 d9 73 4d fd 5c 22 fd 85 21 ee 6b bf c0 84 81 6c 26 b3 fc b4 82 af 58 d9 14 88

Recv:ef aa 00 00 06 f7 00 **00 09** 00 00 f8

Send:ef aa f7 00 f8 **00 0a** 17 bc b5 db ac 60 be 47 17 21 09 6a 18 be 22 f0 59 6a d5 41 d3 53 66 25 23 72 c3 84 8e ac 4d 5d e8 2a 0f 1f 03 09 29 e8 46 d5 9e 4c d3 a0 62 30 37 62 6d 6f 91 e4 cc fd bd 08 87 72 78 ff 40 09 e0 c4 7a cd 68 e3 2d 6d 2b 60 fe eb b7 6b 27 0d 62 a2 dc 9f 36 20 4b 4d da 56 a1 a9 55 c5 b2 a3 16 5d 5c fa b1 53 22 9a ee 5e 15 1f ea cb 23 64 36 55 5a 8c 66 63 74 8b aa 69 f2 08 e7 72 cc d7 77 aa 88 d1 29 f8 8e a5 2e cd 5b 6d 39 fb b5 cc 24 7c 2d 6a b3 0d 5f 60 49 8e 58 37 41 b2 b9 96 78 58 0d b1 c1 40 23 8c 65 84 b9 c4 fa df 40 70 7c d6 58 07 c8 69 a8 01 11 6d 7b a1 08 d7 d4 dd 68 4a 2e bb 20 f3 fd d8 3e 3e f7 04 00 f1 46 b5 72 f4 5c 08 c1 6f eb a6 20 bb be a1

8f ba 4a bd 58 bb ba 53 ab c1 cb 45 79 d4 a7 7f fa cb f9 0a a6 37 e0 c0 83 db be

Recv:ef aa 00 00 06 f7 00 **00 0a** 00 00 fb

Send:ef aa f7 00 f8 **00 0b** fe 81 4e 39 23 55 a4 7c 77 3e 92 05 19 08 fd 8b a6 b4 5f 28 cf a6 61 ba 31 36 12 b8 c0 0c c7 fd d0 2d 57 90 2c ae 9f cd 47 88 c5 57 5e 92 fe 20 d1 f8 95 c0 25 85 d4 71 0c 08 ef eb 25 4b a8 27 b3 c9 e7 11 ab 55 e1 74 d4 d0 d6 ce 70 9f a3 3f 30 d3 e5 f7 42 e7 b7 30 f6 fe f6 c8 6c 54 d5 62 a0 a3 47 0e fd 51 a3 f0 c7 13 df 5d 08 b8 2a 98 63 0a 18 bf ae 24 f8 9d 6f cf 67 43 07 fb e8 44 cd b4 ae a2 a4 c1 43 9b 54 b8 7f 51 28 5e bb c0 b2 17 47 c2 3c 28 45 ec a2 51 f3 53 d6 20 d5 91 61 b0 14 7d 1c 37 c3 fd c9 89 ee 7c 38 0b b2 a8 4a ca e3 9a 13 3b da f5 8d 5b 28 8d 8e 81 9e 66 23 60 d3 c0 21 4e 9f da 44 03 28 8f 85 ef 81 4e f1 73 88 04 c0 db 4d 95 fc ed 0c d3 69 87 b6 77 f1 73 fe 7a d2 14 06 03 97 0f 56 9d ec c3 a6 71 a1 f8 9b a9 01 f9 e5

Recv:ef aa 00 00 06 f7 00 **00 0b** 00 00 fa

Send:ef aa f7 00 3b **00 0c** 77 02 96 95 8f fb 54 aa e2 db 08 57 9e 55 73 79 51 98 e7 0a 81 05 15 ff 23 e6 05 8d d9 b6 e6 14 76 80 72 22 ae 91 d5 e1 6c aa 45 0a 6d 9a dd 43 64 6c 91 34 ee a5 3b 92 27 24

Recv:ef aa 00 00 06 f7 00**00 0c00 01** fcSend complete, UserID=1

(iii) Supplementary explanation

1. MID_REPLY

Each command sent by the master to the module will eventually receive a MID_REPLY reply from the module, which contains the command mid sent by the master,the execution result of the command, and the data that may be returned.

```
typedef struct {
uint8_t mid; // the command(message) id to reply (the request message ID)
uint8_t result; // command handling result: success or failed -> s_msg_result
uint8_t data[0];
} s_msg_reply_data;
```

The result of the command execution may be the following:

```
/* message result code */
typedef uint8_t s_msg_result;
const uint8_t MR_SUCCESS = 0; // success
const uint8_t MR_REJECTED = 1; // module rejected this command
const uint8_t MR_ABORTED = 2; // algo aborted
const uint8_t MR_FAILED4_CAMERA =4; // camera open failed
const uint8_t MR_FAILED4_UNKNOWNREASON = 5; // UNKNOWN_ERROR
```

```
const uint8_t MR_FAILED4_INVALIDPARAM = 6; // invalid param
const uint8_t MR_FAILED4_NOMEMORY = 7; // no enough memory
const uint8_t MR_FAILED4_UNKNOWNUSER = 8; // exceed limitation
const uint8_t MR_FAILED4_MAXUSER = 9; // exceed maximum user number
const uint8_t MR_FAILED4_FACEENROLLED = 10; // this face has been enrolled
const uint8_t MR_FAILED4_LIVENESSCHECK = 12;// liveness check failed
const uint8_t MR_FAILED4_TIMEOUT = 13; // exceed the time limit
const uint8_t MR_FAILED4_AUTHORIZATION = 14;// authorization failed
const uint8_t MR_FAILED4_READ_FILE = 19; // read file failed
const uint8_t MR_FAILED4_WRITE_FILE = 20; // write file failed
const uint8_t MR_FAILED4_NO_ENCRYPT = 21; // encrypt must be set
const uint8_t MR_FAILED4_NO_RGBIMAGE= 23; // rgb image is not ready
/* message result code end */
```

The returneddatavariesaccording tomid.Inthe header filemessage.h,the structure starting with "s_msg_reply_"is the data carried when each instruction sends a reply message. Take verify as an example:

```
/* message reply data definitions */
typedef struct {
uint8_t user_id_heb;
uint8_t user_id_leb;
uint8_t user_name[USER_NAME_SIZE];
uint8_t admin;
uint8_t unlockStatus;
} s_msg_reply_verify_data;
```

2. MID_NOTE

MSG::NOTE is the information that the module actively reports to the master control. For example, when the module is powered on, it will actively send a NOTE to the master control, indicatingthat it has READY and can receive the command of the master control. After receiving the NOTE message, the master control can start sending the entry or verification command.

The data contained inthe

NOTE message is: typedef

```
struct {
uint8_t nid; // note id
```

uint8_t data[0];

} s_msg_note_data;

/* module -> host note end */

3. MID_RESET

After the master sends this command, the module will cancel the previously executed command (such as entry, unlock, etc.) and return to STANDBY state.

4. MID_GETSTATUS

After the master sends the command, the module returns to its current state, mainly including: MS_STANDBY(0): module is in idle state, waitingfor master command MS_BUSY(1): module is in working state MS_ERROR(2): module error, unable to work normally MS_INVALID(3):module is not initialized

5. MID_VERIFY

MSG::VERIFY is the most commonly used function, which carries thefollowingdata

// verify

typedef struct {

uint8_t pd_rightaway; // power down right away after verifying

uint8_t timeout; // timeout, unit second, default 10s

} s_msg_verify_data;

[pd_rightaway] indicates whether to shut down immediately after unlocking;[timeout] is the unlocking timeout time (unit s), which is implicitlyrecognizedas10s andcan be set when sending the unlocking command. The timeout time can be set arbitrarily by the user, and the maximum is255s.

During the unlock process, the module returns NOTEandREPLYmessages.

NOTE mainly returns the state of the face in the current frame picture, as well as the position and posture of the face.

REPLYmainly returns the final result after the unlocking process is completed, and the possible returned messages include MR_SUCCESS, MR_FAILED4_NOSUCHFACE, MR_ABORTED, etc. When the returned result is MR_SUCCESS, it indicates that the unlocking is successful. REPALY will carry is what kind of way to unlock. The data structure is as follows:

typedef struct

{ uint8_t user_id_heb;

```
uint8_t user_id_leb;
uint8_t user_name[USER_NAME_SIZE];
uint8_t admin;
```

uint8_t unlockStatus;

} s_msg_reply_verify_data;

6. MID_ENROLL/MID_ENROLL_SINGLE

MSG::ENROLL is also a common function, and its accompanyingdata is as follows:

// enroll user

typedef struct {

uint8_t admin; // the user will be set to admin

uint8_t user_name[32];

s_face_dir face_direction;

uint8_t timeout;

} s_msg_enroll_data;

[admin] indicates that the person who entered the entry is the administrator;[timeout] is the timeout time (ins)duringthe entry process,which can beset by default when sending the entry command. The timeout time can be set freely by the user, and the maximum is255s[face_direction] is onlyvalidin MID_ENROLL

/* msg face direction */

typedef uint8_t s_face_dir;

const uint8_t FACE_DIRECTION_UP = 0x10; // face up

const uint8_t FACE_DIRECTION_DOWN = 0x08; // face down

const uint8_t FACE_DIRECTION_LEFT = 0x04; // face left

const uint8_t FACE_DIRECTION_RIGHT = 0x02; // face right

const uint8_t FACE_DIRECTION_MIDDLE = 0x01; // face middle

const uint8_t FACE_DIRECTION_UNDEFINE= 0x00;// face undefine

Two messages, NOTEand

NOTE mainly returns the state of the face in the current frame picture, as well as the position and posture of the face.

REPLY mainly returns the final result after the entry process is completed. When MR_SUCCESSS is returned, it indicates that the entry is successful. TheREPLY also returns some information, as follows:

typedef struct

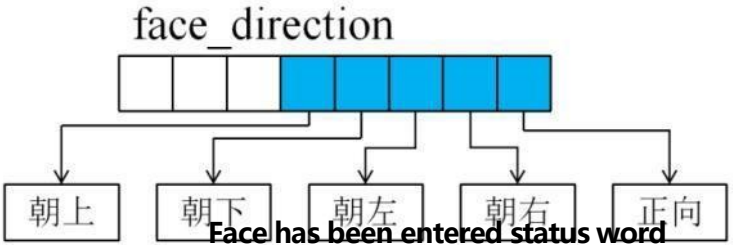
{ uint8_t user_id_heb;

uint8_t user_id_leb;

```
    uint8_t face_direction;
```

} s_msg_reply_enroll_data;

[face_direction], the lower5bits of the data from high to low respectively represent the upward, downward, left, right and positive directions of the face direction,as shown in the following figure, 1 indicates that the direction has been entered, and 0 indicates that the direction has not been entered.



Face has been entered status word

7. MID_ENROLL_ITG

MSG::ENROLL_ITisa supplement and extensionto the 1.15.6 ENROLL
, and allows users to choose

whether to repeat the

entry;the accompanying data

is as follows:

```
// enroll user intergrated
typedef struct {
uint8_t admin; // the user will be set to admin, 1:admin user, 0:normal user
uint8_t user_name[USER_NAME_SIZE];
uint8_t face_direction; // reference FACE_DIRECTION_*
uint8_t enroll_type; // reference FACE_ENROLL_TYPE_*
uint8_t enable_duplicate; // enable user enroll duplicatly, 1:enable, 0:disable
  // when enroll_type is equal to FACE_ENROLL_TYPE_RGB，  the enable_duplicate can be set to 2, it means that cant duplicate enroll with username
uint8_t timeout; // timeout unit second default 10s
```

```
uint8_t reserved[3]; // reserved feild

} s_msg_enroll_itg;
```

[enroll_type] FACE_ENROLL_TYPE_INTERACTIVE, interactive entry;
FACE_ENROLL_TYPE_SINGLE, single frame entry

[enable_duplicate] functions as follows:

0: The same person cannot be entered, but the user name can be repeated.

1: The same person can be entered repeatedly, and the user name can also be repeated.

2: The same person can be entered, but the username cannot be repeated (only valid when RGB is registered).

8. MID_DELUSER

MSG:DELUSER is to delete a registered user, specified by user_id.

9. MID_DELALL

MSG::DELALL is to delete all registered user information.

10. MID_GETUSERINFO

The master specifies the registered user to be returnedthrough the userid.After receiving the command, the module will return the information of the specified user, mainly including the followinginformation:

```
typedef struct
{  uint8_t user_id_heb;
uint8_t user_id_leb;
uint8_t user_name[32];
uint8_t admin;
} s_msg_reply_getuserinfo_data;
```

11. MID_ ENROLL_WITH_PHOTO

Master initiates photo registration, Seqis set to 0, Photo datais 4 bytes (big end mode) Photo size; Module returns Result and Seq,

WhenResultisMR_SUCCESS, the master control accumulatesSeqby1 (Seqstarts to send photo datafrom1)Photodatais photodata, and the packet size of photo data is 246 bytes. When the last packet is insufficient, it is sent for its actual byte number.

12. MID_DEMOMODE

After the control sends the command, the module enters the demonstration mode. In this mode, the module authentication process will not do feature comparison, that is, everyone can unlock, but the living body detection will still be performed.

13. MID_GET_ALL_USERID

Gets the number of all registered users and the IDs of all registered users.