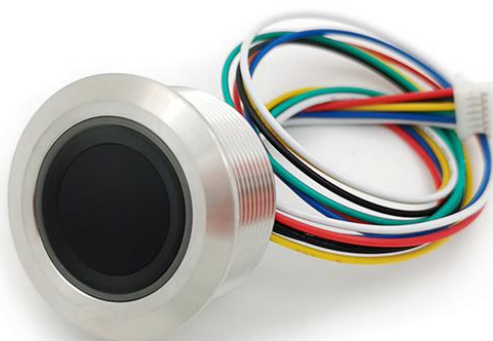




# R558 Fingerprint Module User Manual



**Hangzhou Grow Technology Co., Ltd.**

2024.04 Ver: 1.1

## Preface & Declaration

Thank you for your selection of R558 Fingerprint Identification Module of GROW.

The Manual is targeted for hardware & software development engineer, covering module function, hardware and software interface etc. To ensure the developing process goes smoothly, it is highly recommended the Manual is read through carefully.

Because of the products constantly upgraded and improved, module and the manual content may be changed without prior notice. If you want to get the latest information, please visit our company website ([www.hzgrow.com](http://www.hzgrow.com)).

We have been trying our best to ensure you the correctness of the Manual. However, if you have any question or find error, feel free to contact us or the authorized agent. We would be very grateful.

The Manual contains proprietary information of Hangzhou Grow Technology Co., Ltd., which shall not be used by or disclosed to third parties without the permission of GROW, nor for any reproduction and alteration of information without any associated warranties, conditions, limitations, or notices.

No responsibility or liability is assumed by GROW for the application or use, nor for any infringements of patents or other intellectual property rights of third parties that may result from its use.

[www.hzgrow.com](http://www.hzgrow.com)

## Revised Version

Version Number	Date	Revise Content	Modifier
V1.1	2024.04	Set up	Grow Tech

# Catalog

Preface & Declaration .....	I
Revised Version .....	II
Catalog .....	III
I Introduction .....	- 1 -
Operation Principle .....	- 1 -
II Hardware Interface .....	- 2 -
Exterior Interface .....	- 2 -
Serial Communication .....	- 2 -
Hardware Connection .....	- 3 -
Serial communication protocol .....	- 3 -
Power-on delay time .....	- 3 -
Power Supply Requirements .....	- 4 -
Ripple noise .....	- 4 -
III System Resources .....	- 5 -
Notepad .....	- 5 -
Buffer .....	- 5 -
Fingerprint Library .....	- 5 -
System Configuration Parameters .....	- 5 -
Baud rate control (Parameter Number: 4) .....	- 6 -
Security Level (Parameter Number: 5) .....	- 6 -
Data package length (Parameter Number: 6) .....	- 6 -
System status register .....	- 6 -
Module password .....	- 6 -
Module address .....	- 6 -
Random number generator .....	- 7 -
Features and templates .....	- 7 -
IV Communication Protocol .....	- 8 -
Data package format .....	- 8 -
Instruction Table .....	- 9 -
Check and acknowledgement of data package .....	- 10 -
V Module Instruction System .....	- 12 -
To collect finger image      GetImg .....	- 12 -
To collect finger image in register      GetEnrollImage .....	- 12 -
To generate character file from image      GenChar .....	- 13 -
To carry out precise matching of two finger templates      Match .....	- 13 -
To search finger library      Search .....	- 14 -
To generate template      RegModel .....	- 15 -
To store template      Store .....	- 15 -
To read template from Flash library      LoadChar .....	- 16 -
To upload image      UpImage .....	- 16 -
To download the image      DownImage .....	- 17 -
To delete template      DeletChar .....	- 18 -
To empty finger library      Empty .....	- 18 -

Write system registers	WriteReg .....	- 19 -
Read system Parameter	ReadSysPara .....	- 20 -
To read information page	ReadInfPage .....	- 21 -
Note 2: Package identifier=02: Data packet, and have subsequent packets;; .....		- 22 -
Package identifier=08: the last packet, that is end packet. ....		- 22 -
When the UART reads a flash information page packet, it is sent in packets of a preset length. ....		- 22 -
To write note pad	WriteNotepad .....	- 22 -
To read note pad	ReadNotepad .....	- 22 -
Read fingerprint template index table	ReadIndexTable .....	- 23 -
Get the unique serial number of the chip	GetChipSN .....	- 23 -
Hand shake	HandShake .....	- 24 -
Check sensor	CheckSensor .....	- 24 -
Sleep	Sleep .....	- 25 -
Read valid template number	TemplateNum .....	- 25 -
Set password	SetPwd .....	- 26 -
Verify password	VfyPwd .....	- 26 -
LED Control	ControlBLN .....	- 27 -
Restore factory setting	RestSetting .....	- 29 -
To generate a random code	GetRandomCode .....	- 29 -
Set chip address	SetChipAdder .....	- 30 -
Automatic registration template	AutoEnroll .....	- 30 -
Automatic fingerprint verification	AutoIdentify .....	- 34 -
Cancel instruction	Cancel .....	- 35 -
VI Operation Process .....		- 37 -
6.1 Process of the UART command package .....		- 37 -
6.2 UART Packet Sending Process .....		- 38 -
6.3 UART packet receiving process .....		- 39 -
6.4 Register fingerprint process .....		- 40 -
6.5 Fingerprint search process .....		- 41 -
6.6 Master loads a fingerprint feature or template for accurate matching .....		- 42 -
6.7 Sleep Process .....		- 43 -

# I Introduction

Supply Voltage	DC 3.3±0.3V	Interface	UART(3.3V TTL logical level)
Maximum Current	<40(mA)/VDD	Matching Mode	1:1 and 1:N
Standby Current	<12(uA)/VADD	Baud Rates(UART)	57600
Presence of Self-learning	Yes	LED	RGB
Image acquiring time	<180ms	Image resolution	508dpi
Sensing Array	112*88 pixel	Starting time	<50ms
Storage capacity	100 Pieces	IP Grade	Front side waterproof IP54
FAR	<0.001%	FRR	<1%
Algorithm Search Time	< 100ms(50 Pieces)	Overall Identification Time	<0.6s(50 Pieces)
Working Environment	Temp: -30℃- +70℃	Storage Environment	Temp: -40℃- +85℃
	RH: 20%-85%		
Antistatic Capacity(ESD)	Contact discharge±8KV, non-contact discharge±15KV.		
Operating Vibration	10-150HZ,0.5G,3 axis,1min/oct,1 time/axis.		

\*Note: When the module is not powered on at 3.3V, if the UART signal of the module is connected to an external device, you need to set the UART interface signal of the external device connected to the module to a low level.

## Operation Principle

Fingerprint processing includes two parts: fingerprint enrollment and fingerprint matching (the matching can be 1:1 or 1:N).

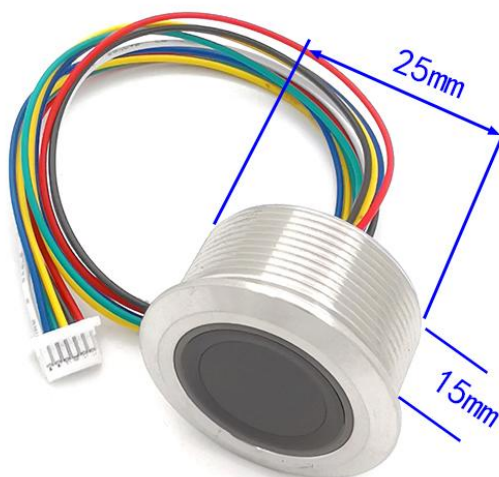
When enrolling, user needs to enter the finger two times. The system will process the two time finger images, generate a template of the finger based on processing results and store the template. When matching, user enters the finger through optical sensor and system will generate a template of the finger and compare it with templates of the finger library. For 1:1 matching, system will compare the live finger with specific template designated in the Module; for 1:N matching, or searching, system will search the whole finger library for the matching finger. In both circumstances, system will return the matching result, success or failure.

## II Hardware Interface

### Exterior Interface

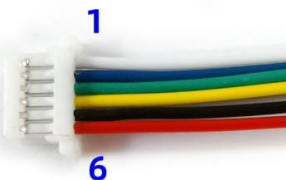
#### R558 Info

Connector: SH1.0--6P Thread:M25



### Serial Communication

Connector: SH1.0--6P

Pin	Name	Description	Pic
1	VCC_3.3V	Fingerprint sensor power supply 3.3V.	
2	TOUCH_OUT	Fingerprint sensor interrupt signal.	
3	VCC_MCU	MCU power supply 3.3V. (Application sleep command: 33H)	
4	UART_TX	Date Output. TTL Logical Level	
5	UART_RX	Date Input. TTL Logical Level	
6	GND	Signal Ground	

Note:  
Please ignore the color of the cable.  
The pin sequence prevails.

## Hardware Connection

The RX of the module is connected with the TX of the upper computer, and the TX of the module is connected with the RX of the upper computer. The IRQ signal can be connected with the middle fracture or IO port of the upper computer.

To reduce the system standby power consumption, when the upper computer needs to use the fingerprint module, then power on the main power supply of the fingerprint module. At this time, the fingerprint module is powered on, and complete the corresponding instructions sent by the upper computer. When the upper computer does not need to use the fingerprint module, disconnect the fingerprint module from the main power supply.

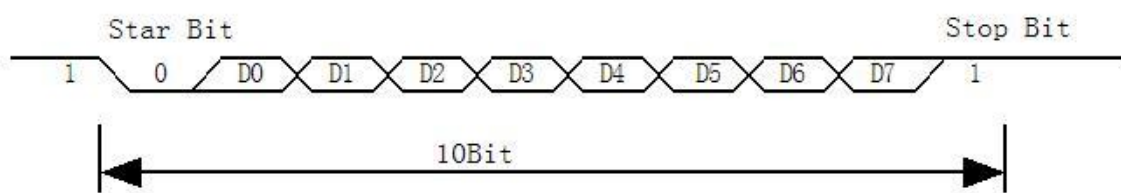
When the upper computer is in standby mode, in order to keep the finger touch detection, the touch power supply needs to be powered all the time. The working voltage of the touch power supply is 3V~5V, and the average current of the touch power supply is about 2uA. When there is no finger touch, the default touch sensing signal outputs high level; When a finger touches, the default touch sensing signal outputs low level. After detecting the touch sensing signal, the upper computer supplies power to the fingerprint module and the fingerprint module starts to work.

The maximum response time of the touch function is about 120mS @ $V_t=3.3V$ . When the module is not touched, the recalibration period is about 4.0sec; the touch signal output is CMOS output, and the output voltage is roughly the same as the input voltage.

## Serial communication protocol

The mode is semiduplex asynchronism serial communication. And the default baud rate is 57600bps. User may set the baud rate in 9600~115200bps.

Transferring frame format is 10 bit: the low-level starting bit, 8-bit data with the LSB first, and 2 ending bit. There is no check bit.



## Power-on delay time

At power on, it takes about 50ms for initialization. During this period, the module can't accept commands for upper computer. After completing the initialization, the module will immediately send a byte (0x55) to the upper computer, indicating that the module can work normally and receive instructions from the upper computer.



## Power Supply Requirements

The power supply is DC +3.3V. The power input is allowed only after the R558 is properly connected.

Electrical components of the R558 may be damaged if you insert or remove the cable (with the electric plug) when the cable is live. Ensure that the power supply is switched off when you insert or remove the cable.

The R558 may not work properly due to poor power connections, short power off/on intervals, or excessive voltage drop pulses. So pls keep the power is stable. After the power is turned off, the power must be turned on at least two seconds later.

## Ripple noise

Since the power input of R558 is directly supplied to the image sensor and decoding chip.

To ensure stable operation, pls use low ripple noise power input.

It is recommended that the ripple noise not exceed 50mV (peak-to-peak).

## III System Resources

To address demands of different customer, Module system provides abundant resources at user's use.

### Notepad

The system sets aside a 512-bytes memory (16 pages\* 32 bytes) for user's notepad, where data requiring power-off protection can be stored. The host can access the page by instructions of PS\_WriteNotepad and PS\_Read Notepad.

Note: when write on one page of the pad, the entire 32 bytes will be written in wholly covering the original contents.

The user can run the module address or random number command to configure the unique matching between the module and the system. That is, the system identifies only the unique module. If a module of the same type is replaced, the system cannot access the system.

### Buffer

The module RAM resources are as follows:

An ImageBuffer: ImageBuffer

6 feature buffers: CharBuffer[1:6]

All buffer contents are not saved without power.

The user can read and write any buffer by instruction. CharBuffer can be used to store normal feature files or store template feature files.

When uploading or downloading images through the UART port, only the high four bits of pixel bytes are used to speed up the transmission, that is, use gray level 16, two pixels are combined into one byte. (The high four bits are a pixel, the low four bits are a pixel in the next adjacent column of the same row, that is, two pixels are combined into one byte and transmitted)

Since the image has 16 gray levels, when it is uploaded to PC for display (corresponding to BMP format), the gray level should be extended (256 gray levels, that is, 8bit bitmap format).

### Fingerprint Library

System sets aside a certain space within Flash for fingerprint template storage, that's fingerprint library. The contents of the fingerprint database are protected by power-off, and the serial number of the fingerprint database starts from 0.

Capacity of the library changes with the capacity of Flash, system will recognize the latter automatically. Fingerprint template's storage in Flash is in sequential order. Assume the fingerprint capacity N, then the serial number of template in library is 0, 1, 2, 3 ... N. User can only access library by template number.

### System Configuration Parameters

The system allows the user to individually modify a specified parameter value (by parameter serial number) by command. Refer to *SetSysPara*. After the upper computer sets the system parameter instructions, the system must be powered on again so that the module can work according to the new

## Baud rate control (Parameter Number: 4)

The Parameter controls the UART communication speed of the Module. Its value is an integer N,  $N = [1/2/4/6/12]$ . Corresponding baud rate is  $9600 * N$  bps.

## Security Level (Parameter Number: 5)

The Parameter controls the matching threshold value of fingerprint searching and matching. Security level is divided into 5 grades, and corresponding value is 1, 2, 3, 4, 5. At level 1, FAR is the highest and FRR is the lowest; however at level 5, FAR is the lowest and FRR is the highest.

## Data package length (Parameter Number: 6)

The parameter decides the max length of the transferring data package when communicating with upper computer. Its value is 0, 1, 2, 3, corresponding to 32 bytes, 64 bytes, **128 bytes**, 256 bytes respectively.

## System status register

System status register indicates the current operation status of the Module. Its length is 1 word, and can be read via instruction *ReadSysPara*. Definition of the register is as follows:

Bit Num	15	4	3	2	1	0
Description	Reserved		ImgBufStat	PWD	Pass	Busy

Note:

Busy: 1 bit. 1: system is executing commands; 0: system is free;

Pass: 1 bit. 1: find the matching finger; 0: wrong finger;

PWD: 1 bit. 1: Verified device's handshaking password.

ImgBufStat: 1 bit. 1: image buffer contains valid image.

## Module password

The default password of the module is 0x00000000. If the default password is modified, after the module is powered on, the first instruction of the upper computer to communicate with the module must be verify password. Only after the password verification is passed, the module will enter the normal working state and receive other instructions.

The new modified password is stored in Flash and remains at power off. (the modified password cannot be obtained through the communication instruction. If forgotten by mistake, the module cannot communicate, please use with caution)

Refer to instruction *SetPwd* and *VfyPwd*.

## Module address

Each module has an identifying address. When communicating with upper computer, each instruction/data is transferred in data package form, which contains the address item. Module system only responds to data package whose address item value is the same with its identifying



address.

The address length is 4 bytes, and its default factory value is 0xFFFFFFFF. User may modify the address via instruction *SetAddr*. The new modified address remains at power off.

## Random number generator

Module integrates a hardware 32-bit random number generator (RNG) (without seed). Via instruction *GetRandomCode*, system will generate a random number and upload it.

## Features and templates

The chip has one image buffer and six feature file buffers, all buffer contents are not saved after power failure.

A template can be composed of 2-6 feature files. The more feature files in the synthesis template, the better the quality of the fingerprint template.

It is recommended to take at least four templates to synthesize features.

## IV Communication Protocol

The protocol defines the data exchanging format when R558 series communicates with upper computer. The protocol and instruction sets applies for both UART communication mode.

Baud rate 57600, data bit 8, stop bit 2, parity bit none.

### Data package format

When communicating, the transferring and receiving of command/data/result are all wrapped in data package format. For multi-bytes, the high byte precedes the low byte (for example, a 2 bytes 00 06 indicates 0006, not 0600).

#### Data package format

Header	Adder	Package identifier	Package length	Package content (instruction/data/Parameter)	Checksum
--------	-------	--------------------	----------------	---	----------

#### Definition of Data package

Name	Symbol	Length	Description	
Header	Start	2 bytes	Fixed value of 0xEF01; High byte transferred first.	
Adder	ADDER	4 bytes	Default value is 0xFFFFFFFF, which can be modified by command. High byte transferred first and at wrong adder value, module will reject to transfer.	
Package identifier	PID	1 byte	01H	Command packet;
			02H	Data packet; Data packet shall not appear alone in executing process, must follow command packet or acknowledge packet.
			07H	Acknowledge packet;
			08H	End of Data packet.
Package length	LENGTH	2 bytes	Refers to the length of package content (command packets and data packets) plus the length of Checksum( 2 bytes). Unit is byte. Max length is 256 bytes. And high byte is transferred first.	
Package contents	DATA	—	It can be commands, data, command's parameters, acknowledge result, etc. (fingerprint character value, template are all deemed as data);	
Checksum	SUM	2 bytes	The arithmetic sum of package identifier, package length and all package contents. Overflowing bits are omitted. high byte is transferred first.	

## Instruction Table

Code	Identifier	Description	Code	Identifier	Description
01H	GenImg	To collect finger image	19H	ReadNotepad	To read note pad
29H	GetEnrollImage	To collect finger image in register	1FH	ReadIndexTable	Read fingerprint template index table
02H	Genchar	To generate character file from image	34H	GetChipSN	Get the unique serial number of the chip
03H	Match	To carry out precise matching of two finger templates	35H	HandShake	Hand shake
04H	Search	To search finger library	36H	CheckSensor	Check sensor
05H	RegModel	To generate template	33H	Sleep	Sleep command
06H	Store	To store template	1DH	TemplateNum	Read valid template number
07H	LoadChar	To read template from Flash library	12H	SetPwd	Set password
0AH	UpImage	To upload image	13H	VfyPwd	Verify password
0BH	DownImage	To download the image	3CH	ControlBLN	LED control
0CH	DeletChar	To delete template	3BH	RestSetting	Restore factory setting
0DH	Empty	To empty the library	14H	GetRandomCode	To generate a random code
0EH	WriteReg	Write system registers	15H	SetChipAdder	Set chip address
0FH	ReadSysPara	Read system Parameter	31H	AutoEnroll	Automatic registration template
16H	ReadInfPage	To read information page	32H	AutoIdentify	Automatic fingerprint verification
18H	WriteNotepad	To write note pad	30H	Cancel	Cancel instruction

## Check and acknowledgement of data package

**Note: Commands shall only be sent from upper computer to the Module, and the Module acknowledges the commands.**

Upon receipt of commands, Module will report the commands execution status and results to upper computer through acknowledge packet. Acknowledge packet has parameters and may also have following data packet. Upper computer can't ascertain Module's package receiving status or command execution results unless through acknowledge packet sent from Module. Acknowledge packet includes 1 byte confirmation code and maybe also the returned parameter.

*Confirmation code's definition is :*

- 00H: command execution complete or OK;
- 01H: error when receiving data package;
- 02H: no finger on the sensor;
- 03H: failed to record the fingerprint image;
- 04H: fingerprint images are too dry and faint to produce features;
- 05H: fingerprint images are too wet and mushy to produce features;
- 06H: failed to generate a character file due to the over-disorderly fingerprint image;
- 07H: fingerprint image is normal, but there are too few feature points to generate a feature;
- 08H: fingerprint doesn't match;
- 09H: failed to find the fingerprint;
- 0AH: failed to combine the character files;
- 0BH: addressing PageID is beyond the finger library;
- 0CH: error when reading template from library or the template is invalid;
- 0DH: error when uploading feature;
- 0EH: module can't receive the following data packages;
- 0FH: error when uploading image;
- 10H: failed to delete the template;
- 11H: failed to clear fingerprint library;
- 12H: failed to enter low-power state;
- 13H: wrong password;
- 14H: failed to make the system reset;
- 15H: failed to generate the image due to the lack of valid primary image;
- 16H: failed to upgrade online;
- 17H: residual fingerprints or fingers not moving between two acquisitions;
- 18H: error reading and writing flash;
- 19H: failed to generation random number;
- 1AH: invalid register number;
- 1BH: incorrect configuration of register;
- 1CH: wrong notepad page number;
- 1DH: failed to operate the communication port;
- 1EH: failed to enroll automatically;
- 1FH: fingerprint library is full;
- 20H: the address code is incorrect;
- 21H: error password;
- 22H: fingerprint template is not empty;
- 23H: fingerprint template is empty;

24H: fingerprint library is empty;  
25H: incorrect setting of input count;  
26H: timeout;  
27H: fingerprints already exist;  
28H: fingerprint features are correlated;  
29H: fail to back to factory setting;  
2AH: module information is not empty;  
2BH: module information is empty;  
2CH: OTP operation failed;  
2DH: Key generation failed;  
2EH: Key does not exist;  
2FH: Security algorithm execution failed;  
30H: Security algorithm encryption and decryption results are incorrect;  
31H: Function does not match encryption level;  
32H: Key locked;  
33H: Small image area;  
34H: Static foreign object in the image (Orange);  
35H: Illegal data;  
36H: Reverse mould(Orange);  
37H: Static foreign object in the feature (Orange);  
38H: Dynamic foreign object in the feature (Orange);  
39H-3aH: Reserved;  
3bH: Poor template quality;  
3cH-efH: Reserved.



# V Module Instruction System

## To collect finger image      GetImg

Description: detecting finger and store the detected finger image in ImageBuffer while returning successfully confirmation code; If there is no finger, returned confirmation code would be “can’t detect finger”.

Input Parameter: none

Return Parameter: Confirmation code (1 byte)

Instruction code: 01H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	0003H	01H	0005H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	Sum

Note: Confirmation code=00H: finger collection success;

Confirmation code=01H: error when receiving package;

Confirmation code=02H: no finger on the sensor.

## To collect finger image in register      GetEnrollImage

Description: enroll finger and store the detected finger image in ImageBuffer while returning successfully confirmation code; If there is no finger, returned confirmation code would be “can’t detect finger”.

Input Parameter: none

Return Parameter: Confirmation code (1 byte)

Instruction code: 29H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	0003H	29H	002DH

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	Sum

Note: Confirmation code=00H: finger collection success;  
 Confirmation code=01H: error when receiving package;  
 Confirmation code=02H: no finger on the sensor

## To generate character file from image      GenChar

Description: to generate character file from the original finger image in ImageBuffer

Input Parameter: BufferID (character file buffer number)

Return Parameter: Confirmation code (1 byte)

Instruction code: 02H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Buffer number	Checksum
0xEF01	xxxx	01H	0004H	02H	BufferID	sum

**Note: In the registration process, BufferID indicates that how many times press your finger; in other cases, BufferID has a corresponding default value.**

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: generate feature success;  
 Confirmation code=01H: error when receiving package;  
 Confirmation code=07H: fingerprint image is normal, but there are too few feature points to generate a feature;  
 Confirmation code=08H: require similar finger areas to be entered each time during the enrollment process, where the current feature is not similar to the previous feature; (this feature is turned off by default)  
 Confirmation code=0aH: feature merge failure;  
 Confirmation code=28H: require different finger areas to be entered each time during the enrollment process, when the current feature overlaps too much with the previous one. (this feature is turned off by default)

## To carry out precise matching of two finger templates

### Match

Description: Precisely compare the feature files in the template buffer.

Input Parameter: none

Return Parameter: Confirmation code (1 byte), matching score.

Instruction code: 03H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	0003H	03H	0007H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Score	Checksum
0xEF01	xxxx	07H	0005H	xxH	xxxxH	sum

Note 1: Confirmation code=00H: templates of the two buffers are matching;

Confirmation code=01H: error when receiving package;

Confirmation code=08H: fingerprint doesn't match;

Confirmation code=31H: function does not match encryption level.

Accurate comparison e.g.:

[18:47:50.049] 发→	EF 01 FF FF FF FF 01 00 06 07 02 00 00 00 10	Read template commands
[18:47:50.057] 收←	EF 01 FF FF FF FF 07 00 03 00 00 0A	Get image command
[18:47:54.857] 发→	EF 01 FF FF FF FF 01 00 03 01 00 05	Generate feature command
[18:47:55.003] 收←	EF 01 FF FF FF FF 07 00 03 00 00 0A	Accurate comparison of two feature commands
[18:47:59.817] 发→	EF 01 FF FF FF FF 01 00 04 02 01 00 08	
[18:48:00.106] 收←	EF 01 FF FF FF FF 07 00 03 00 00 0A	
[18:48:06.944] 发→	EF 01 FF FF FF FF 01 00 03 03 00 07	
[18:48:06.968] 收←	EF 01 FF FF FF FF 07 00 05 00 00 BC 00 C8	

## To search finger library

## Search

Description: to search the whole or part of finger library with the feature files in the template buffer. When found, PageID will be returned.

Input Parameter: CharBufferID + StartID + Num

Return Parameter: Confirmation code+ModelID(template number)+ MatchScore

Instruction code: 04H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Buffer number	Parameter	Parameter	Checksum
0xEF01	xxxx	01H	0008H	04H	BufferID	StartPage	PageNum	sum

**Note: BufferID defaults to 1 and searches the whole or part of the fingerprint library with the fingerprint templates in the template buffer.**

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Page	Score	Checksum

0xEF01	xxxx	07H	0007H	xxH	PageID	MatchScore	sum
--------	------	-----	-------	-----	--------	------------	-----

Note 1: Confirmation code=00H: found the matching finer;

Confirmation code=01H: error when receiving package;

Confirmation code=09H:No matching in the library (both the PageID and matching score are 0);

Confirmation code=31H:function does not match encryption level.

## To generate template RegModel

Description: To combine information of character files from CharBuffer1 and CharBuffer2 and generate a template which is stored back in both CharBuffer1 and CharBuffer2.

Input Parameter: none

Return Parameter: Confirmation code (1 byte)

Instruction code: 05H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	0003H	05H	0009H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: operation success;

Confirmation code=01H: error when receiving package;

Confirmation code=3bH: poor template quality and merge failure.

## To store template Store

Description: to store the template of specified buffer (Buffer1/Buffer2) at the designated location of Flash library.

Input Parameter: BufferID, Page ID

Return Parameter: Confirmation code (1 byte)

Instruction code: 06H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Buffer number	Location number	Checksum
0xEF01	xxxx	01H	0006H	06H	BufferID	PageID	sum

**Note:** BufferID defaults to 1.

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: storage success;

Confirmation code=01H: error when receiving package;

Confirmation code=18H: error when writing Flash;

Confirmation code=31H: function does not match encryption level;

Confirmation code=35H: illegal data.

## To read template from Flash library      LoadChar

Description: to load template at the specified location (PageID) of Flash library to template buffer CharBuffer1/CharBuffer2

Input Parameter: BufferID, PageID

Return Parameter: Confirmation code (1 byte)

Instruction code: 07H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Buffer number	Page number	Checksum
0xEF01	xxxx	01H	0006H	07H	Buffer ID	Page ID	sum

**Note: BufferID defaults to 2.**

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: load success;

Confirmation code=01H: error when receiving package.

## To upload image      UpImage

Description: to upload the image in Img\_Buffer to upper computer.

Input Parameter: none

Return Parameter: Confirmation code (1 byte)

Instruction code: 0AH

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	0003H	0AH	000EH

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
---------	--------	--------	---------	--------	---------

Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note 1: Confirmation code=00H: ready to transfer the following data packet;  
Confirmation code=01H: error when receiving package.

Data package format:

2 bytes	4bytes	1 byte	2 bytes	N bytes	2 bytes
Header	Module address	Package identifier	Package length	Package data	Checksum
0xEF01	xxxx	xxH	xxxxH	xx	sum

Note 2: Package identifier=02:packets with subsequent packets;

Package identifier=08:the last packet, the end packet.

When the UART uploads image data packets, they are sent in packets of a preset length.

A byte contains two pixels, each pixel occupies 4bits.

## To download the image      DownImage

Description: to download image from upper computer to Img\_Buffer.

Input Parameter: none

Return Parameter: Confirmation code (1 byte)

Instruction code: 0BH

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	0003H	0BH	000FH

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note: 1: Confirmation code=00H: subsequent packets can be received;

Confirmation code=01H: error when receiving package;

Confirmation code=31H: function does not match encryption level;

Data package format:

2 bytes	4bytes	1 byte	2 bytes	N bytes	2 bytes
Header	Module address	Package identifier	Package length	Package data	Checksum
0xEF01	xxxx	xxH	xxxxH	xx	sum

Note 2: Package identifier=02:packets with subsequent packets;

Package identifier=08:the last packet, the end packet.

When the UART downloads image data packets, they are received in packets of a preset length.

A byte contains two pixels, each pixel occupies 4bits.

## To delete template

## DeletChar

Description: to delete a segment (N) of templates of Flash library started from the specified location (or PageID);

Input Parameter: PageID + Num

Return Parameter: Confirmation code (1 byte)

Instruction code: 0CH

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes	2bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Page number	Number of templates to be deleted	Checksum
0xEF01	xxxx	01H	0007H	0CH	PageID	N	sum

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: delete templates success;

Confirmation code=01H: error when receiving package;

Confirmation code=10H: failed to delete the templates.

## To empty finger library

## Empty

Description: to delete all the templates in the Flash library

Input Parameter: none

Return Parameter: Confirmation code (1 byte)

Instruction code: 0DH

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	0003H	0DH	0011H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: empty success;

Confirmation code=01H: error when receiving package;

Confirmation code=11H: failed to clear fingerprint library.

## Write system registers

## WriteReg

Description: Write module registers.

Input Parameter: Register number+Contents

Return Parameter: Confirmation code (1 byte)

Instruction code: 0EH

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1byte	1byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Register number	Contents	Checksum
0xEF01	xxxx	01H	0005H	0EH	0~13	xx	sum

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note 1: Confirmation code=00H: OK;

Confirmation code=01H: error when receiving package;

Confirmation code=18H: error when write FLASH;

Confirmation code=1aH: wrong register number;

Confirmation code=1bH: register setting content error;

Note 2: When the write system register (PS\_WriteReg) instruction is executed, it first answers according to the original configuration, and after answering, it modifies the system settings and records the configuration in the FLASH, and works according to the new configuration at the same time.



Parameter number	Name	Content
0	Delay Time	Delay Time
1	Enroll Times	Enroll Times
2	Image Format Register	0:format 0
3	Enroll Logic Register	0:mode 0 1:mode 1 2:mode 2
4	Baud Rate Control Register	9600 * N bps
5	Comparison Threshold Register	1:level 1 2:level 2 3:level 3 4:level 4 5:level 5
6	Packet Size Register	0:32bytes 1:64bytes 2:128bytes 3:256bytes
7	Encryption Level Register	0~255: Reserved
10	Product Model Register	Reserved
11	Automatic lighting Register	Led Ctr

## Read system Parameter      ReadSysPara

Description: Reads the basic parameters of the module (baud rate, packet size, etc.). The first 16 bytes of the parameter table store the basic communication and configuration information of the module, called the basic parameters of the module.

Input Parameter: none

Return Parameter: Confirmation code (1 byte) + basic parameter (16bytes)

Instruction code: 0FH

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	0003H	0FH	0013H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	16 bytes	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Basic parameter list	Checksum

0xEF01	xxxx	07H	0013H	xxH	See following table	sum
--------	------	-----	-------	-----	---------------------	-----

Note: Confirmation code=00H: read complete;

Confirmation code=01H: error when receiving package;

Name	Description	Offset (word)	Size (word)
Enroll Times	Maximum number of entries at registration	0	2
Fingerprint Template Size	Fingerprint template size	2	2
Fingerprint Database Size	Fingerprint database capacity	4	2
Score level	Score level code (1/2/3/4/5)	6	2
Device Address	32-bit device address	8	4
Data Packet Size	Data packet size code: 0:32bytes 1:64bytes 2:128bytes 3:256bytes	12	2
Baud Settings	N (baud = 9600*N bps)	14	2

## To read information page      ReadInfPage

Description: read information page(512bytes)

Input Parameter: none

Return Parameter: Confirmation code (1 byte)

Instruction code: 16H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	0003H	16H	001AH

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note 1: Confirmation code=00H: ready to transfer the following data packet;

Confirmation code=01H: error when receiving package;

Data package format:

2 bytes	4bytes	1 byte	2 bytes	N bytes	2 bytes
---------	--------	--------	---------	---------	---------

Header	Chip address	Package identifier	Package length	Data	Checksum
0xEF01	xxxx	xxH	xxxxH	xx	sum

Note 2: Package identifier=02: Data packet, and have subsequent packets;;

Package identifier=08: the last packet, that is end packet.

When the UART reads a flash information page packet, it is sent in packets of a preset length.

## To write note pad      WriteNotepad

Description: Inside the module, a 256bytes FLASH space is opened for the user to store the user's data, which is called the user's notepad, and the notepad is logically divided into 16 pages, and the Write Notepad command is used to write the user's 32bytes of data to the specified notepad page.

Input Parameter: NotePageNum, user content (or data content)

Return Parameter: Confirmation code (1 byte)

Instruction code: 18H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1byte	32 bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Page number	User information	Checksum
0xEF01	xxxx	01H	0024H	18H	0~15	User content	sum

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: write success;

Confirmation code=01H: error when receiving package;

Confirmation code=1cH: wrong notepad page number;

Confirmation code=18H: error reading and writing flash.

## To read note pad      ReadNotepad

Description: Reads 128bytes of data from the FLASH user area.

Input Parameter: NotePageNum

Return Parameter: Confirmation code (1 byte) + User content

Instruction code: 19H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1byte	2 bytes
Header	Module	Package	Package	Instruction	Page	Checksum

	address	identifier	length	code	number	
0xEF01	xxxx	01H	0004H	19H	0~15	Sum

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	32bytes	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	User information	Checksum
0xEF01	xxxx	07H	0023H	xxH	User content	sum

Note: Confirmation code=00H: read success;

Confirmation code=01H: error when receiving package;

Confirmation code=1cH: wrong notepad page number.

## Read fingerprint template index table

## ReadIndexTable

Description: Read the index table of the entry template

Input Parameter: Index table page number, page number 0, 1 corresponds to the index of the template from 0-256, 256-512 respectively, each 1 bit represents a template, 1 means that the template in the corresponding storage area has been recorded, 0 means not recorded.

Return Parameter: Confirmation code+Fingerprint template index table

Instruction code: 1FH

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Index page	Checksum
0xEF01	xxxx	01H	0004H	1FH	0~1	Sum

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	32 bytes	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Index page	Check-sum
0xEF01	xxxx	07H	0023H	xxH	Index	sum

Note: Confirmation code=00H: read success;

Confirmation code=01H: error when receiving package;

Confirmation code=0bH: address serial number out of range of fingerprint database when accessing fingerprint database.

## Get the unique serial number of the chip

## GetChipSN

Description: Get the unique serial number of the chip.

Input Parameter: Reserve

Return Parameter: Confirmation code (1 byte) +unique serial number(SN)

Instruction code: 34H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Parameter	Checksum
0xEF01	xxxx	01H	0004H	34H	0	0039H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	32 bytes	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Unique serial number	Checksum
0xEF01	xxxx	07H	0023H	xxH	SN	sum

Note : Confirmation code=00H: read success;

Confirmation code=01H: error when receiving package.

## Hand shake

## HandShake

Description: Check whether the module works properly.

Input Parameter: none

Return Parameter: Confirmation code

Instruction code: 35H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	0003H	35H	0039H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note 1: Confirmation code=00H: the device is normal and can receive instructions;

Confirmation code=01H: error when receiving package.

## Check sensor

## CheckSensor

Description: Check whether the sensor is work normal

Input Parameter: none

Return Parameter: Confirmation code

Instruction code: 36H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	0003H	36H	003AH

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note 1: Confirmation code=00H: the sensor is normal;  
Confirmation code=01H:error when receiving package;  
Confirmation code=29H: the sensor is abnormal.

## Sleep

## Sleep

Description: Setting the sensor to sleep mode.

Input Parameter: none

Return Parameter: Confirmation code (1 byte)

Instruction code: 33H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	0003H	33H	0037H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package Length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note: Confirmation code = 00H: success;  
Confirmation code = 01H: error when receiving package;  
Confirmation code = 29H: Sensor operation fail;

## Read valid template number      TemplateNum

Description: read the current valid template number of the Module

Input Parameter: none

Return Parameter: Confirmation code (1 byte), template number:N

Instruction code: 1DH

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	0003H	1DH	0021H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes	2 bytes
---------	--------	--------	---------	--------	---------	---------

Header	Module address	Package identifier	Package length	Confirmation code	Valid template number	Checksum
0xEF01	xxxx	07H	0005H	xxH	ValidN	sum

Note: Confirmation code=00H: read success;

Confirmation code=01H: error when receiving package.

## Set password

## SetPwd

Description: Set Module's handshaking password. The default password of the fingerprint module system is 0, if the default password is not modified, the system does not require the verification password during communication, and the main controller can communicate with the module directly; If the password is modified, the first command of the master controller and the device module must be the verification password, and only after the password is verified mthe module can receive other commands.

Input Parameter: PassWord (4 bytes)

Return Parameter: Confirmation code (1 byte)

Instruction code: 12H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	4 bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Password	Checksum
0xEF01	xxxx	01H	0007H	12H	PassWord	sum

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package Length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: password setting complete;

Confirmation code=01H: error when receiving package.

## Verify password

## VfyPwd

Description: Verify Module's handshaking password.

Input Parameter: PassWord (4 bytes)

Return Parameter: Confirmation code (1 byte)

Instruction code: 13H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	4 bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Password	Checksum
0xEF01	xxxx	01H	0007H	13H	PassWord	sum

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package Length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

Note: Confirmation code = 00H: Correct password;  
 Confirmation code = 01H: error when receiving package;  
 Confirmation code = 13H: Wrong password.

## LED Control

## ControlBLN

Description: Control light instructions are mainly divided into two categories: general indicator light and colorful programmed breathing light.

Input Parameter: Function code, starting color, ending color, cycle times

Return Parameter: Confirmation code (1 byte)

Instruction code: 3CH

Command (or instruction) package format:

general indicator light command package format

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	1 byte	1 byte	1byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Function number	Starting color	Ending color	Cycle times	Checksum
0xEF01	xxxx	01H	0007H	3CH	xxH	xxH	xxH	xxH	sum

### Auxiliary Description:

**Function code:** LED light mode control bits, 1-normal breathing light, 2-flashing light, 3-normally on light, 4-normally off light, 5-Gradually on, 6- Gradually off, other function codes are not applicable to this command packet format.

**Starting color:** When set to normal breathing light, the color from off to on limited to normal breathing light (function code 01) function, for other functions, the color should be consistent with the ending color.

bit0 is the blue light control bit, bit1 is the green light control bit, bit2 is the red light control bit, set 1: light on, Set 0: light off.

For example, 0x01 blue light on, 0x02 green light on, 0x03 cyan light on, 0x04 red light on, 0x05 purple light on, 0x06 yellow light on, 0x07 white light on, 0x00 all off.

**Ending color:** When set to normal breathing light, the color from on to off is limited to normal breathing light (function code 0x01), for other functions, it is consistent with the starting color, and the setting method is the same as the starting color.

**Cycle times:** Indicates the number of breathing or flashing, when set to 0, it means infinite cycle, when set to other values, it means a limited number of times to breathe, the number of cycle times applies to the breathing, flashing function, it is invalid in other functions, for example, it is invalid in the normally open, normally closed, gradually open and gradually closed.

### Colorful programmed breathing light command package format



2 bytes	4 bytes	1 byte	2 bytes	1 byte	1 byte	1 byte	1 byte	1 byte	1 byte	1 byte	1 byte	1 byte	2 bytes
Header	Chip address	Package identifier	Package length	Instruction code	Function code	Time bit	Color 1	Color 2	Color 3	Color 4	Color 5	Cycle times	Check sum
0xEF01	xxxx	01H	000BH	3CH	xxH	xxH	xxH	xxH	xxH	xxH	xxH	xxH	sum

#### Auxiliary Description:

**Function code:** 7-colorful programmed breathing light, other function codes are not applicable to this command packet format.

**Time bit:** It is used to control the time for the light to breathe once, i.e. the time from on to off and back to on again. The time range of a single breath is about 0.1s~10.0s, which is expressed by the number between 1-100, and the number beyond this range is invalid. If the time bit is set to 1, it corresponds to 0.1s, and if the time bit is set to 100, it corresponds to 10.0s. We recommend setting the time bit to 36, and the breathing time is the same as the normal breathing light (function code 0x01), which is about 3.6s.

**Color code:** A total of 5 bytes, as shown in the table below, each byte of the color code is divided into 2 units, each unit has 4 bits, starting from the high bit is divided into one valid bit, and 3 color control bits, each unit controls a certain color of the light from off to on, and then off process. In addition, the programmed sequence of the breathing light cycle starts from unit 1 of color 1, then unit 2 of color 1, then unit 1 of color 2, and so on.

**Color control bit diagram table**

Color(1 byte)							
Unit 1				Unit 2			
Valid bit	Red light bit	Green light bit	Blue light bit	Valid bit	Red light bit	Green light bit	Blue light bit
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

**Valid bits:** 0 - This unit and all units after this unit are invalid, 1- This unit is valid;

**Red light bit:** 0-red light off, 1-red light on;

**Green light bit:** 0-green light off, 1-green light on;

**Blue light bit:** 0-blue light off, 1-blue light on;

**Cycle times:** indicates the number of breathing light, when set to 0, it means infinite cycle, when set to other values, it means a limited number of breathing.

#### Acknowledge package format:

2 bytes	4 bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	Sum

Note: Confirmation code=00H: generate address success;

Confirmation code=01H: error when receiving package;

Confirmation code=35H: Illegal data.

### Auxiliary settings (light shortcut commands):

To enable the following function: Send the command corresponding to the function

Function	Directive
Turn on light auto	EF 01 FF FF FF FF 01 00 05 0E 11 01 00 26
Turn off light auto	EF 01 FF FF FF FF 01 00 05 0E 11 00 00 25
Blue light always on	EF 01 FF FF FF FF 01 00 07 3C 03 01 01 00 00 49
Green light always on	EF 01 FF FF FF FF 01 00 07 3C 03 02 02 00 00 4B
Red light always on	EF 01 FF FF FF FF 01 00 07 3C 03 04 04 00 00 4F
Lights off	EF 01 FF FF FF FF 01 00 07 3C 04 00 00 00 00 48

## Restore factory setting

## RestSetting

Description: After the module receives the command and clears the internal data and deletes the internal key pairs.

Input Parameter: None

Return Parameter: Confirmation code (1 byte)

Instruction code: 3BH

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	0003H	3BH	sum

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	05H	0003H	xxH	Sum

Note: Confirmation code=00H: address setting complete;

Confirmation code=01H: error when receiving package;

Confirmation code=18H: error reading and writing flash;

Confirmation code=31H: function does not match encryption level.

## To generate a random code

## GetRandomCode

Description: to command the Module to generate a random number and return it to upper computer.

Input Parameter: none

Return Parameter: Confirmation code (1 byte)+RandomCode

Instruction code: 14H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum

0xEF01	xxxx	01H	0003H	14H	0018H
--------	------	-----	-------	-----	-------

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	4 bytes	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Random number	Checksum
0xEF01	xxxx	07H	0007H	xxH	xxxx	sum

Note: Confirmation code=00H: generation success;

Confirmation code=01H: error when receiving package.

## Set chip address

## SetChipAdder

Description: Set chip address.

Input Parameter: None

Return Parameter: Confirmation code (1 byte)

Instruction code: 15H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	4 bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Chip address	Checksum
0xEF01	xxxx	01H	0007H	15H	xxxx	sum

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0007H	xxH	Sum

Note: Confirmation code=00H: address setting success;

Confirmation code=01H: error when receiving package;

▪When the master transmits the command packet, the device address adopts the default address: 0xffffffff, and the address field of the respond packet adopts the newly generated address.

▪After this instruction is executed, the device address is fixed and remains unchanged.

Only clearing the FLASH can change the chip address.

▪After this instruction is executed, all packets have to use this generated address.

## Automatic registration template

## AutoEnroll

Description: Including the functions of capturing fingerprints, generating features, combining templates, and storing templates.

Input Parameter: ID number, input times, parameters

Return Parameter: Confirmation code Parameters

Instruction code: 31H

Command (or instruction) package format:

2 bytes	4 bytes	1 byte	2 bytes	1 byte	2 bytes	1 byte	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	ID Number	Number of entries	Parameters	Check sum
0xEF01	xxxx	01H	0008H	31H	xxxxH	xxH	xxH	sum

● Auxiliary Description

ID number: high byte first, low byte last. For example, if fingerprint #1 is entered, it would be 0001H.

Number of entries: 1byte, enter twice, it is 02H, enter four times, it is 04H.

Parameter: bit0 for the lowest bit.

- 1) bit0: Reserve;
- 2) bit1: Reserve;
- 3) bit2: During the registration process, whether the module is required to return the current status in key steps, 0- required to return, 1- not required to return;
- 4) bit3: Whether the ID number is allowed to be overwritten. 0-Not allowed, 1-allowed;
- 5) bit4: Allow the fingerprint to repeat registration control bits, 0-allowed, 1-Not allowed;
- 6) bit5: When registering, in the process of multiple fingerprint collection, whether the finger is required to leave before entering the next fingerprint image collection, 0- Leave is required; 1- No request to leave;
- 7) Bit6~bit15: Reserve.

Acknowledge package format:

**Auto Enrollment Template Instruction Normal Flow Response Packet Format**

Header	Chip address	Package identifier	Package length	Confirmation code	Parameter 2 bytes		Checksum	Note
					Parameter 1	Parameter 2		
2bytes	4bytes	1byte	2bytes	1byte	1byte	1byte	2bytes	
0xEF01	xxxx	07H	0005H	xxH	00H	00H	sum	Instruction validity test: Legal
0xEF01	xxxx	07H	0005H	xxH	01H	1	sum	Collect image result: Success/Timeout
0xEF01	xxxx	07H	0005H	xxH	02H	1	sum	Generate feature results: success/failure
0xEF01	xxxx	07H	0005H	xxH	03H	1	sum	Fingers leave, First entry success: Success/Time

								out
						...		
0xEF01	xxxx	07H	0005H	xxH	01H	n	sum	Collect image result:Success/Timeout
0xEF01	xxxx	07H	0005H	xxH	02H	n	sum	Generate feature results: success/failure
0xEF01	xxxx	07H	0005H	xxH	04H	F0H	sum	Merge templates
0xEF01	xxxx	07H	0005H	xxH	05H	F1H	sum	Registered Testing
0xEF01	xxxx	07H	0005H	xxH	06H	F2H	sum	Template storage results

- Confirmation code, return value of parameter 1 and parameter 2:

**Auto-registration Template Reply Packet Description Quick reference table**

Confirmation code	Paraphrase	Parameter 1	Paraphrase	Parameter 2	Paraphrase
00H	Success	00H	Instruction validity test:	00H	Instruction validity test:
01H	Fail	01H	Get Image	F0H	Merge templates
07H	Failed to generate features	02H	Generate feature	F1H	Check if the finger is registered
08H	During the registration process, similar finger areas are required to be entered each time, when the second feature is not similar to the previous feature (this function is disabled by default).	03H	Judgement of Finger Leaving	F2H	Storage Templates
0aH	Failed merger	04H	Merge templates	n	Current enter times
0bH	ID number out of range	05H	Register and check		
18H	Error reading or writing FLASH	06H	Storage Templates		
1fH	Fingerprint database is full.				

22H	Fingerprint template is not empty				
25H	Enter times setting wrong				
26H	Timeout				
27H	Fingerprint enrolled				
28H	During the registration process, similar finger areas are required to be entered each time, when the second feature is not similar to the previous feature (this function is disabled by default).				
31H	The function does not match the encryption level level				
35H	Illegal data				
3bH	Poor quality of templates				

● **Instruction Description:**

- 1) If the specified ID number is invalid, the confirmation code, parameter 1, and parameter 2 return (hereinafter directly described as return): 0b 00 00H. Validity test:
  - If the specified ID number is invalid, return: 0b 00 00H.
  - If the number of entries is incorrectly configured, it returns 25 00 00H. When the fingerprint is not overwritten, 1f 00 00H is returned if the fingerprint database is full.
  - Returns 22 00 00H if the template already exists for the specified ID number.
  - If the validity of the command is successfully checked, 00 00 00H is returned and start register the first fingerprint.
- 2) Waiting for collect image successfully. (return 00 01 0nH).
- 3) Waiting for generate feature successfully (00 02 0nH), if generate feature fails (07 02 0nH), if merge features fails (0a 02 0nH), wait again until the collection image successfully.
- 4) ① If the registration logic mode is configured to 1, which requires a different finger area to be entered each time, if the current and previous features overlap too much area when generating features (returning 28 02 0nH), wait again until the collection image successfully.  
 ② If the registration logic mode is configured to 2, i.e., it is required to enter a similar finger area each time, and if this is not similar to the previous feature when generating the feature (returning 08 02 0nH), wait again until the collection image successfully.
- 5) Wait for the finger to leave, the first entry is successful (00 03 0nH), after the finger leaves, jump to step 2 and enter the next cycle until n is the number of times set for entry. Note: If you set the finger not to leave during the recording process, then it will return to the first successful recording and jump to step 2; the last fingerprint collection, don't have the response for finger left, enter successful.  
 Merge the template, get the template quality of the template, return 00 04 F0H if successfully, return 3B 04 F0H if the template quality is too poor.
- 6) Fingerprint Repeat Check means to match the newly recorded finger with the stored finger (enable or disable this function by setting the parameter bit4), if there is the same fingerprint, it will return to 27 05 F1H and end the process; if there is no same fingerprint, it will return to 00 05 F1H.
- 7) Registering this template data, storage failure returns 01 06 F2H, ending the process; success returns 00 06 F2H.

8) If the PS\_Cancel command is received, it is terminated and an answer is returned.

## Automatic fingerprint verification

## AutoIdentify

Description: Automatically collect a fingerprint, searches for the target template or the entire fingerprint template in the fingerprint database, and returns the search result. If the score of comparison between the target template and the currently collected fingerprint is greater than the maximum threshold, and the target template is incomplete, the blank area of the target template is updated with the collected features. One-stop search includes image acquisition, feature generation, fingerprint search and other functions.

Input Parameter: Security level, ID number, and Parameters

Return Parameter: Confirmation code Page number

Instruction code: 32H

Command (or instruction) package format:

2 bytes	4 bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Security Level	ID Number	Parameters	Checksum
0xEF01	xxxx	01H	0008H	32H	xxH	xxxxH	xxxxH	sum

### ● Auxiliary Description:

ID Number: 2 bytes. For example, if fingerprint #1 is entered, it is 0001H. ID number is 0xFFFF, then a 1:N search is performed; if not, a 1:1 match is performed.

Parameters: The lowest bit is bit0.

- 1) bit0: Reserve;
- 2) bit1: Reserve;
- 3) bit2: During the registration process, whether the module is required to return the current status in key steps, 0 - required to return, 1 - not required to return;
- 4) bit3~bit15: Reserve

Acknowledge package format:

2 bytes	4 bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes	2 bytes	2 bytes	
Header	Chip address	Package identifier	Package length	Confirmation code	Parameters	ID Number	Score	Checksum	Note
0xEF01	xxxx	0x07	0x0008	xxH	00H	xxxxH	xxxxH	Sum	Instruction validity test: Legal
0xEF01	xxxx	0x07	0x0008	xxH	01H	xxxxH	xxxxH	Sum	Collect image result: Success/Timeout
0xEF01		0x07	0x0008	xxH	05H	xxxxH	xxxxH	Sum	Search Results: Success/Failure

- Confirmation code, return value of parameter 1 and parameter 2

**Auto Verify Fingerprint Reply Packet Description Quick reference table**

Confirmation code	Paraphrase	Parameters	Paraphrase
00H	Success	00H	Instruction validity test: Legal
01H	Fail	01H	Get image
07H	Failed to generate features	05H	Enrolled fingerprint matching
08H	Fingerprint mismatch		
09H	No fingerprints.		
0bH	ID number out of range		
17H	Residual fingerprint		
23H	Fingerprint template is empty		
24H	Fingerprint database is empty		
26H	Timeout		
27H	Fingerprint already exist		
31H	Mismatch between function and encryption level		

● **Instruction Description:**

- 1) If the fingerprint database is empty, the confirmation code and parameter return (hereinafter directly described as return): 24 00H. If the specified ID number is invalid, 0b 00H is returned. 23 00H is returned if the registered template does not exist.
- 2) The command validity check succeeded, return 00 00H, and enter register fingerprint.
- 3) Within the set timeout period, if a complete fingerprint entry is not completed, return 26 00H to end the process.
- 4) Check the correctness of the input fingerprint image. If it is not correct, wait for the next image capture.
- 5) If the input fingerprint is correct, return 00 01H, i.e. the fingerprint entry to acquire image is successful.
- 6) If the feature generation fails, return 09 05H to end the process.
- 7) After the feature is successfully generated,, compare between the current captured fingerprint template and the registered fingerprint template and return its result. If the comparison fails, return 09 05H and end the process; if the comparison succeeds, return 00 05H with the correct ID number and score.
- 8) If PS\_Cancel command is received, terminate the command and return the answer.

## Cancel instruction

## Cancel

Description:Cancel AutoEnroll and AutoIdentify.

Input Parameter: none





Return Parameter: Confirmation code

Instruction code: 30H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	0003H	30H	xxxxH

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	0003H	xxH	sum

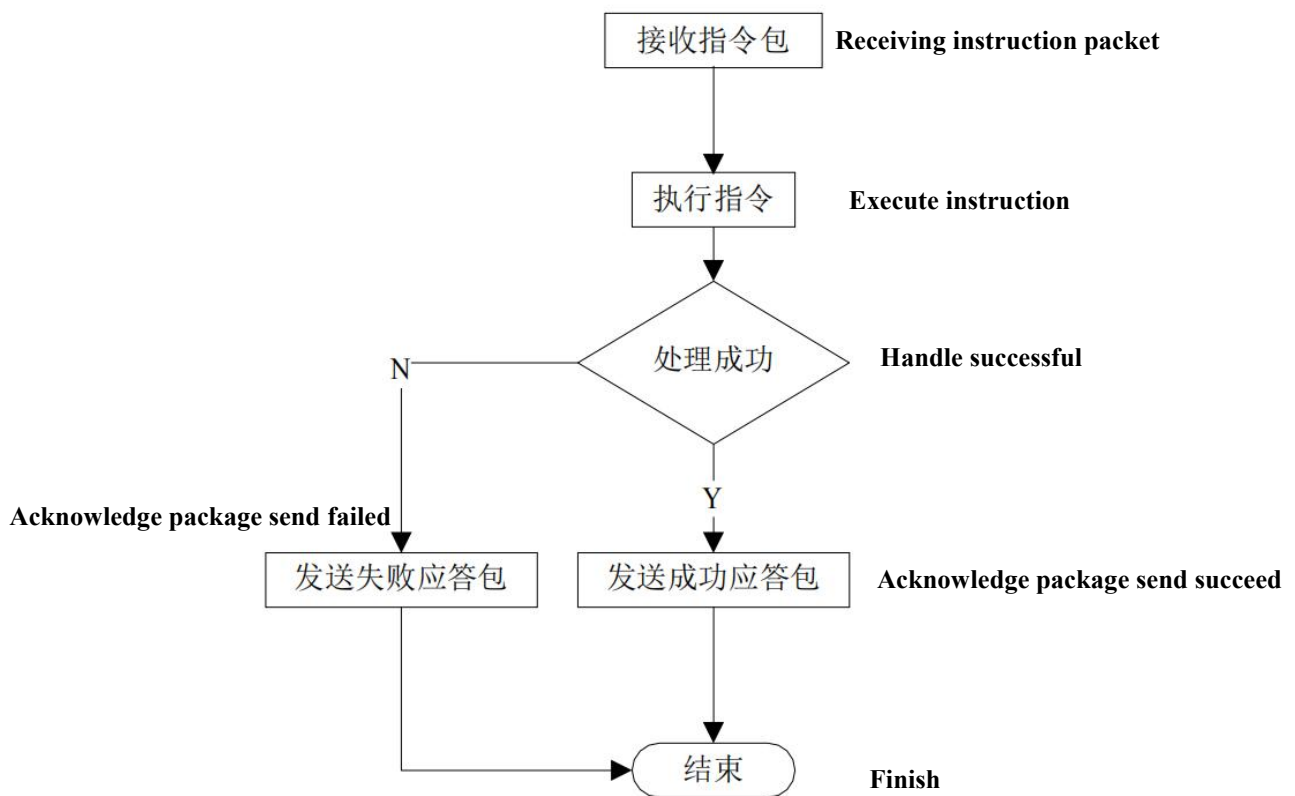
Note 1: Confirmation code=00H: cancel setting successful;

Confirmation code=01H: cancel setting failed;

Confirmation code=31H: function does not match encryption level.

## VI Operation Process

### 6.1 Process of the UART command package



功能实现示例 1：UART命令包的处理过程

Function implementation e.g. 1: Process of UART command packets

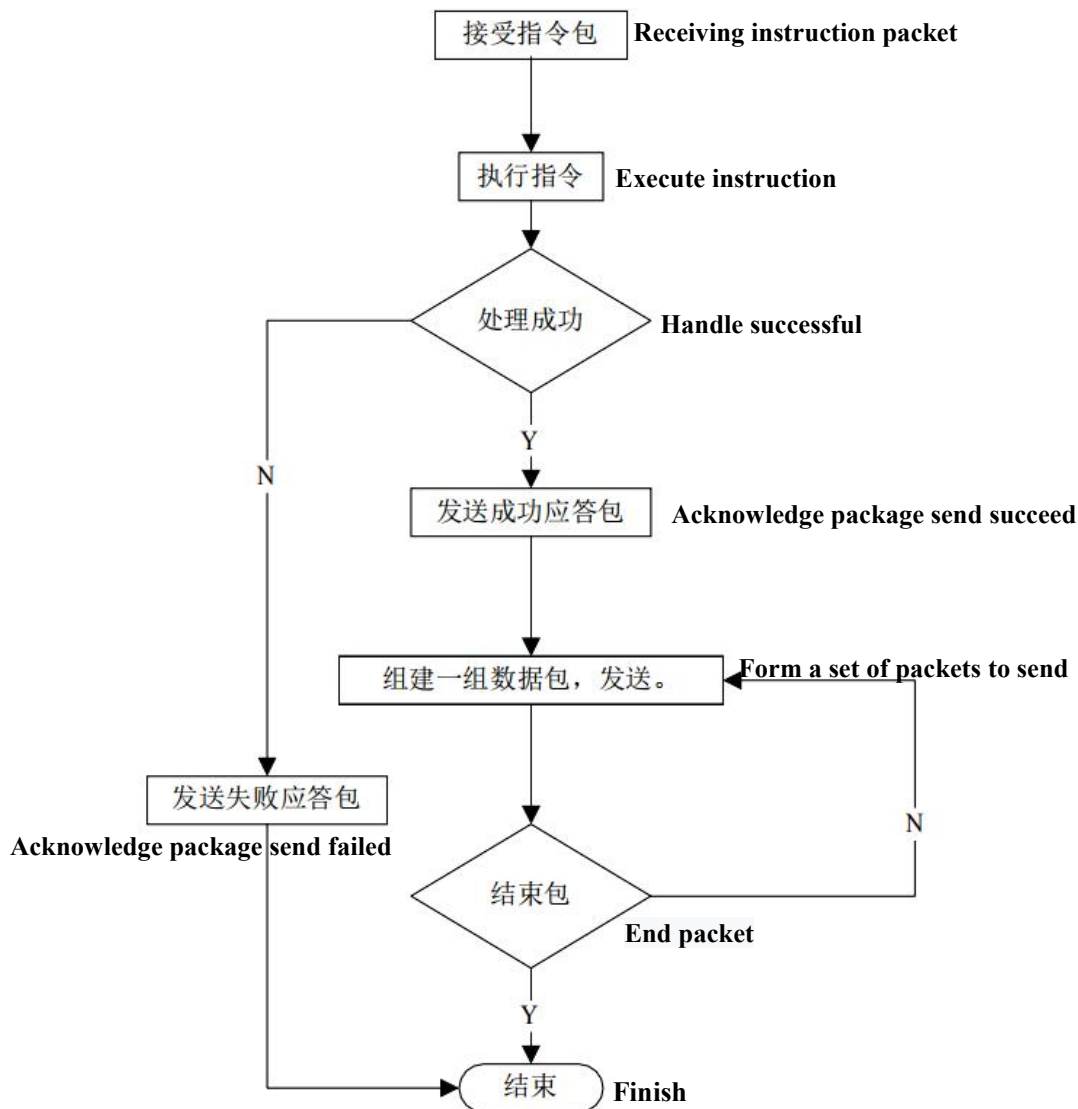
## 6.2 UART Packet Sending Process

Before transmitting data packets, the UART should be received the instruction packet for transmitting data packets first, makes preparations for transmission, then sends a successful response packet, and finally starts transmitting the data packets. Packet mainly includes: packet header, chip address, packet identity, packet length, data and checksum.

There are two types of packet identifiers: 02H and 08H. 02H: indicates the data packet and subsequent packets. 08H: indicates the last packet, that is, the end packet. Data length is preset, mainly divided into: 32, 64, 128, and 256 four types.

For example, if the length of the data to be transmitted is 1K bytes and the preset length of the data packet is 128 bytes, the 1K bytes of data must be divided into eight data packets. Each packet includes: 2 bytes header, 4 bytes chip address, 1 bytes packet identifier, 2 bytes packet length, 128 bytes data and 2 bytes check sum, each packet length is 139 bytes.

In addition, of the eight packets, the packet ID of the first seven packets is 02H and the packet ID of the last end data packet is 08H. Finally, note that if the end packet does not reach 139 bytes in length, it is transmitted at the actual length and is not otherwise expanded to 139 bytes.



功能实现示例 2: UART 数据包的发送过程

Function implementation e.g.2: UART Packet Sending Process

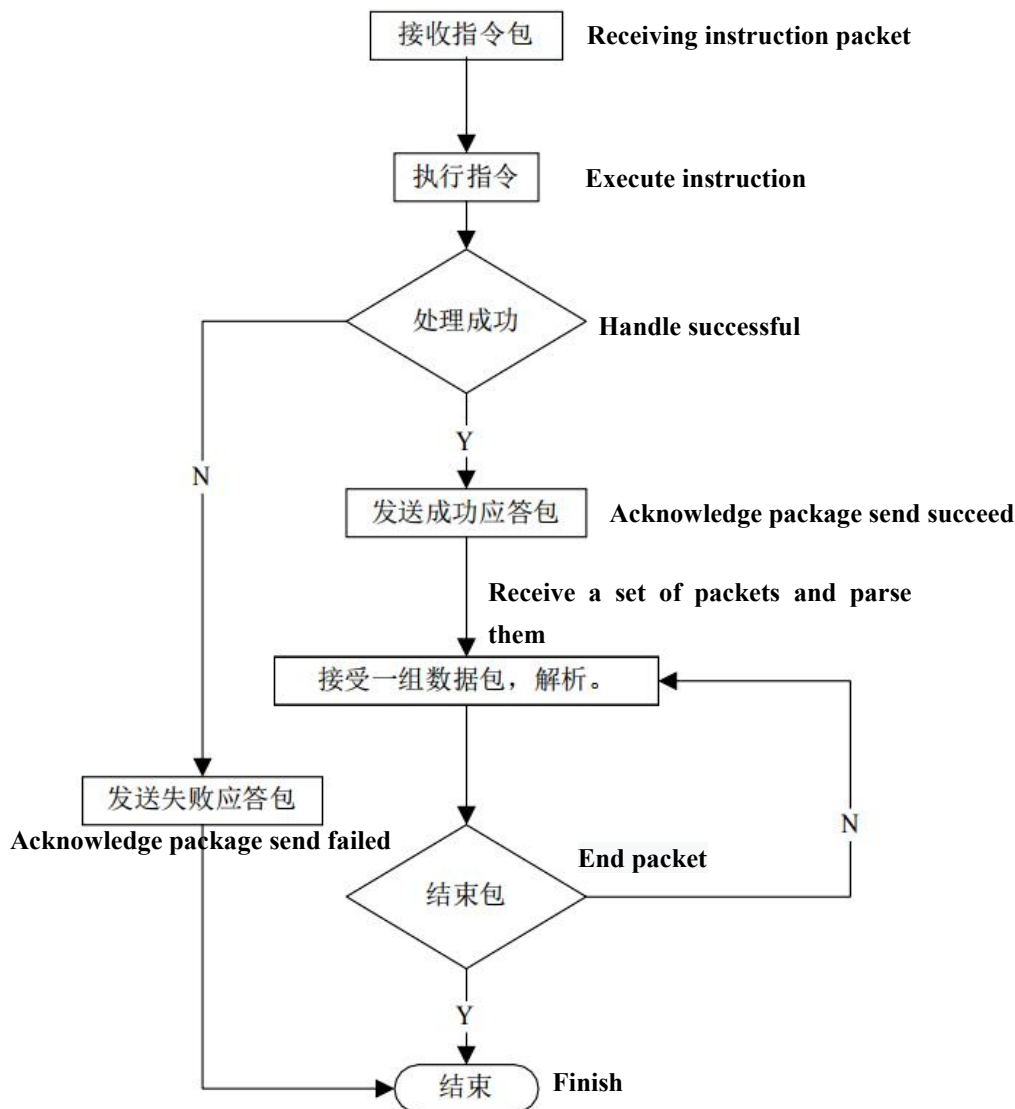
## 6.3 UART packet receiving process

Before transmitting data packets, the UART should be received the instruction packet for transmitting data packets first, makes preparations for transmission, then sends a successful response packet, and finally starts transmitting the data packets. Packet mainly includes: packet header, chip address, packet identity, packet length, data and checksum.

There are two types of packet identifiers: 02H and 08H. 02H: indicates the data packet and subsequent packets. 08H: indicates the last packet, that is, the end packet. Data length is preset, mainly divided into: 32, 64, 128, and 256 four types.

For example, if the length of the data to be transmitted is 1K bytes and the preset length of the data packet is 128 bytes, the 1K bytes of data must be divided into eight data packets. Each packet includes: 2 bytes header, 4 bytes chip address, 1 bytes packet identifier, 2 bytes packet length, 128 bytes data and 2 bytes check sum, each packet length is 139 bytes.

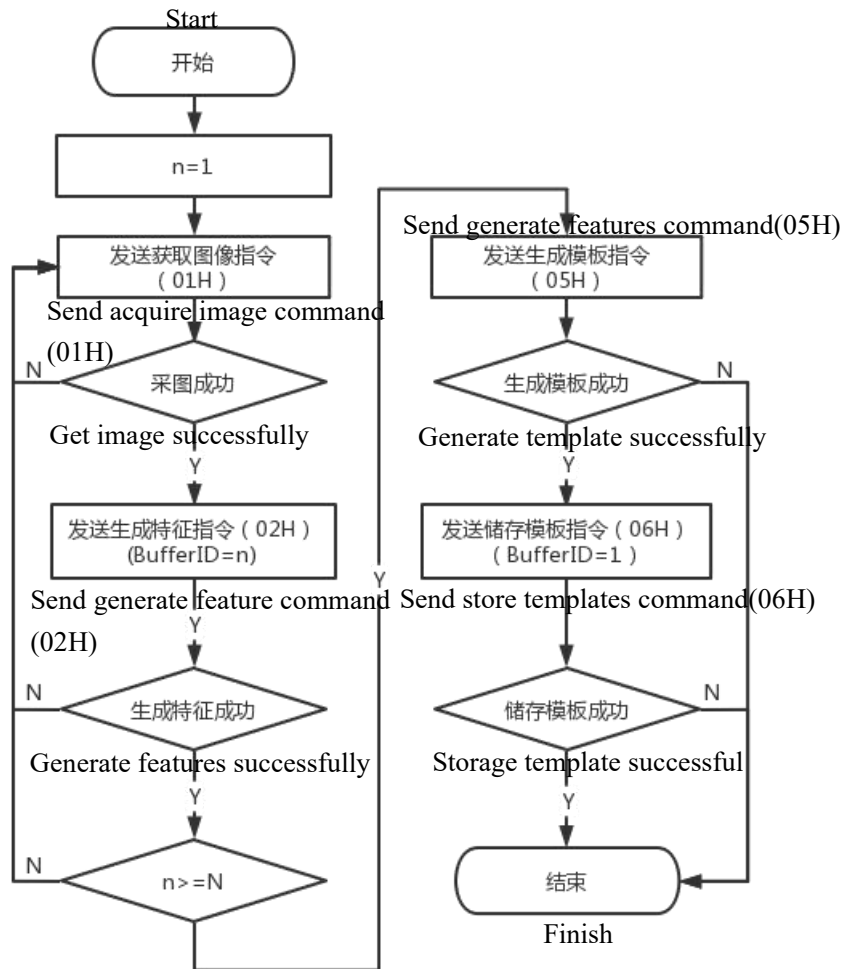
In addition, of the eight packets, the packet ID of the first seven packets is 02H and the packet ID of the last end data packet is 08H. Finally, note that if the end packet does not reach 139 bytes in length, it is transmitted at the actual length and is not otherwise expanded to 139 bytes.



功能实现示例 3: UART 数据包的接收过程

Function implementation e.g.3: UART packet receiving process

## 6.4 Register fingerprint process



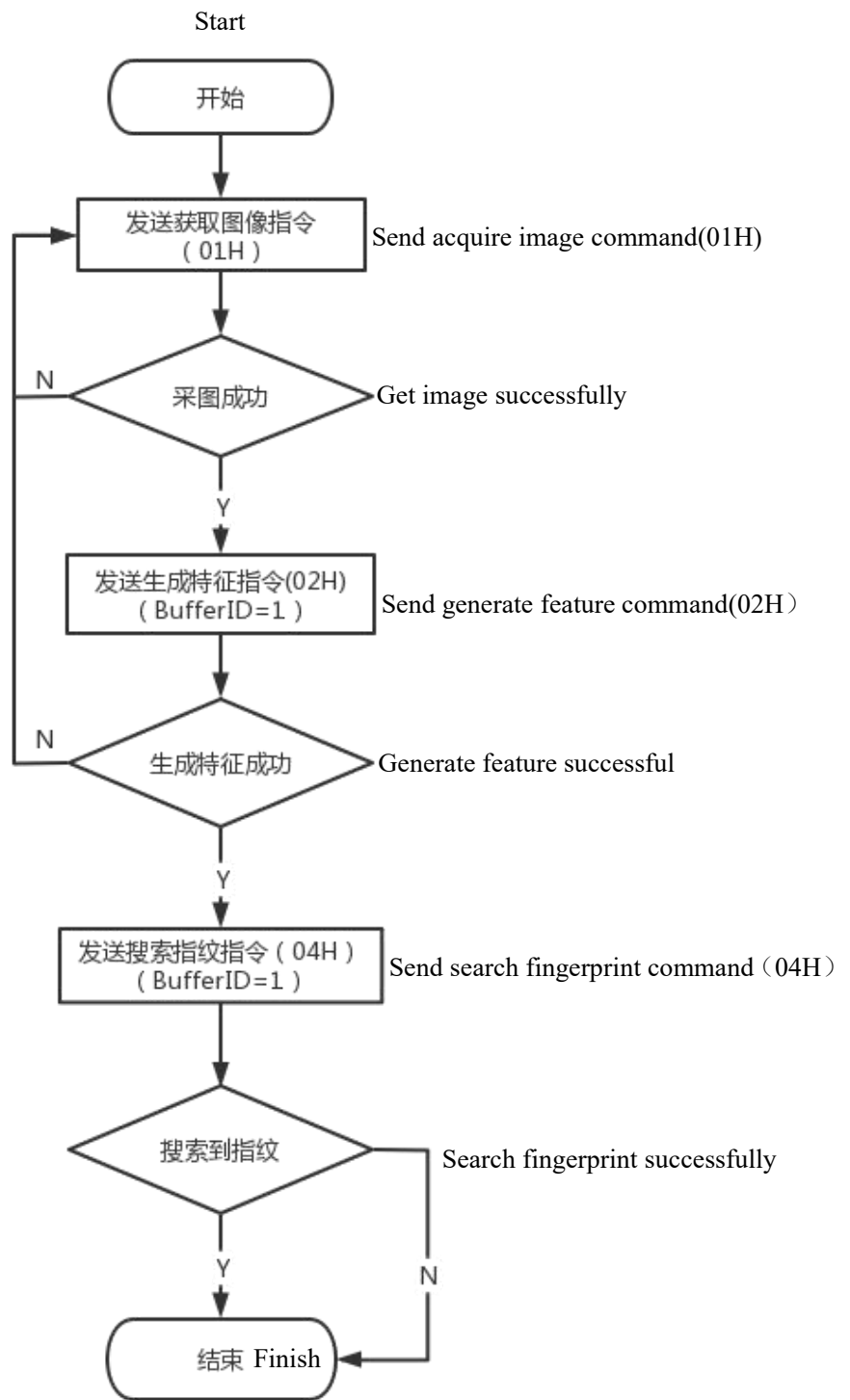
Function implementation e.g.4: Register fingerprint process

During the registration process, if you want to control the finger area for each entry, you can configure the registration logic mode by writing a system register instruction (the registration logic mode defaults to 0, and there is no control over the finger area for each entry).

1.If it is required to record different finger areas each time, then configure the registration logic mode as 1, send the instruction EF 01 FF FF FF FF 01 00 05 0E 03 01 00 18, If there is too much overlap area between the second and the previous feature when the feature is generated, the confirmation code in the module response packet is 0x28;

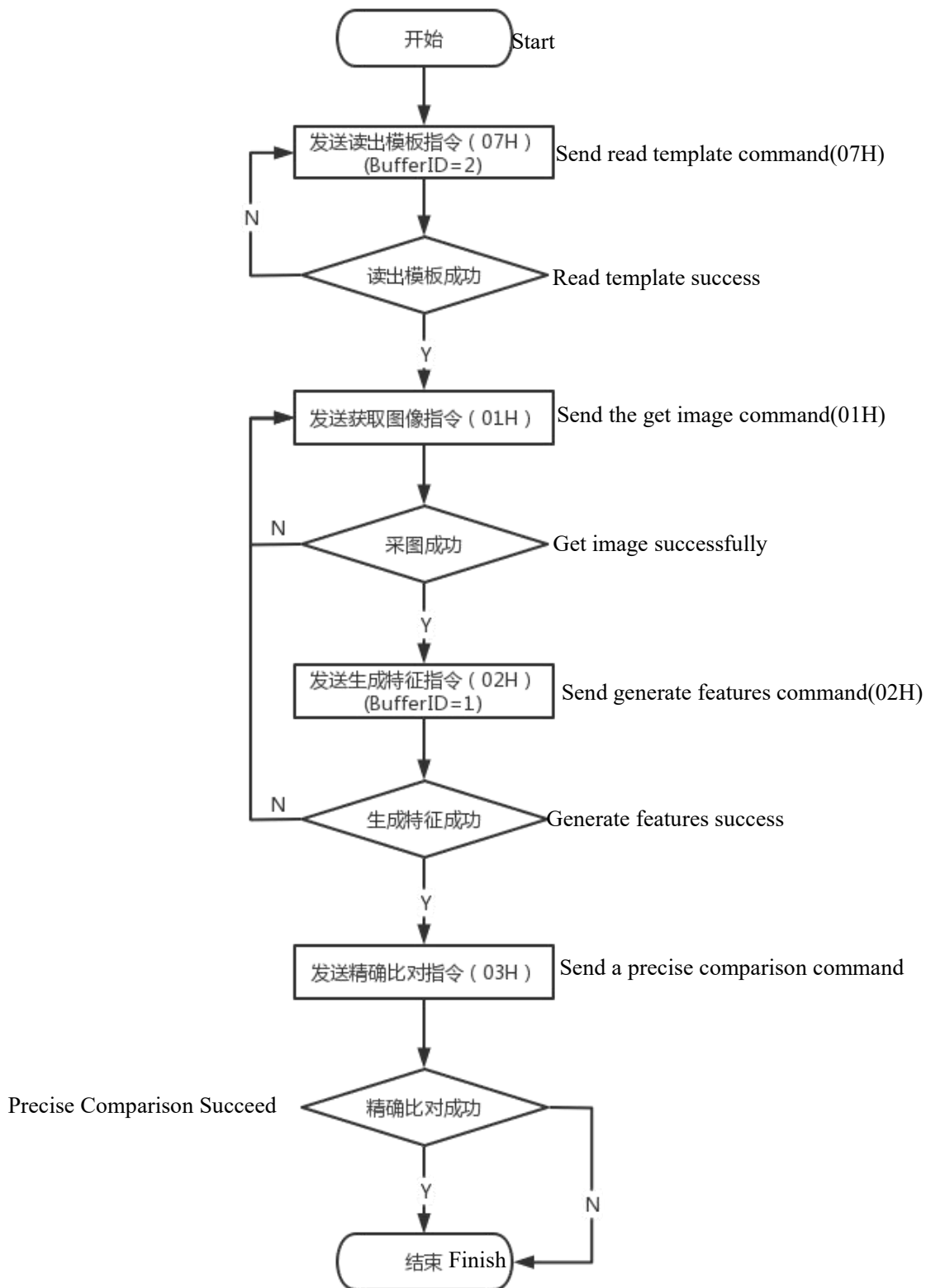
2.If it is required to record a similar finger area each time, then configure the registration logic mode to 2, send the instruction EF 01 FF FF FF FF 01 00 05 0E 03 02 00 19, when the feature is generated, if it is not similar to the previous feature, the confirmation code in the module response packet is 0x08.

## 6.5 Fingerprint search process



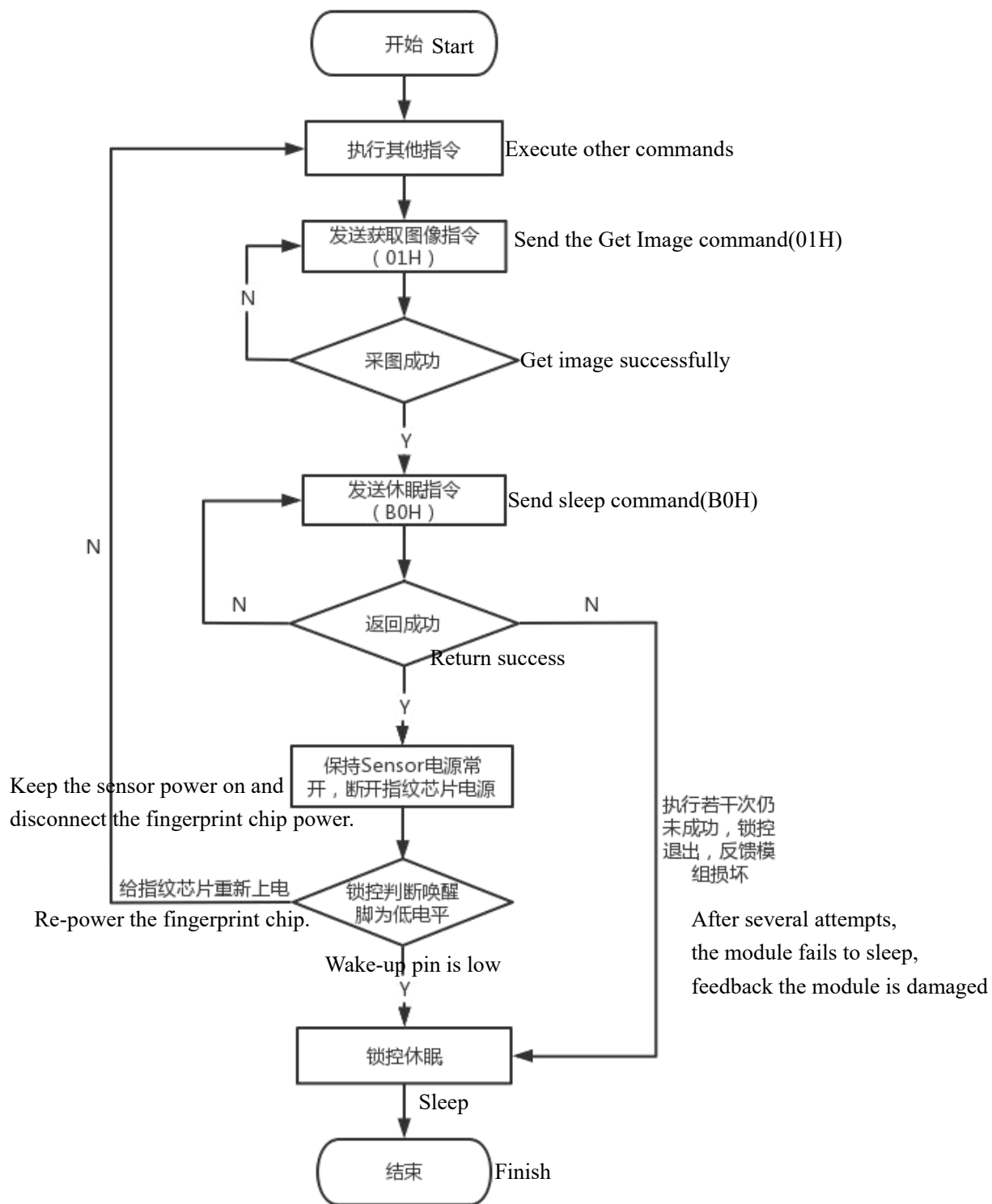
Function implementation e.g.5:Fingerprint search process

## 6.6 Master loads a fingerprint feature or template for accurate matching



Function implementation e.g.6: Master loads a fingerprint feature or template for accurate matching

## 6.7 Sleep Process



Function implementation e.g.7: Sleep process