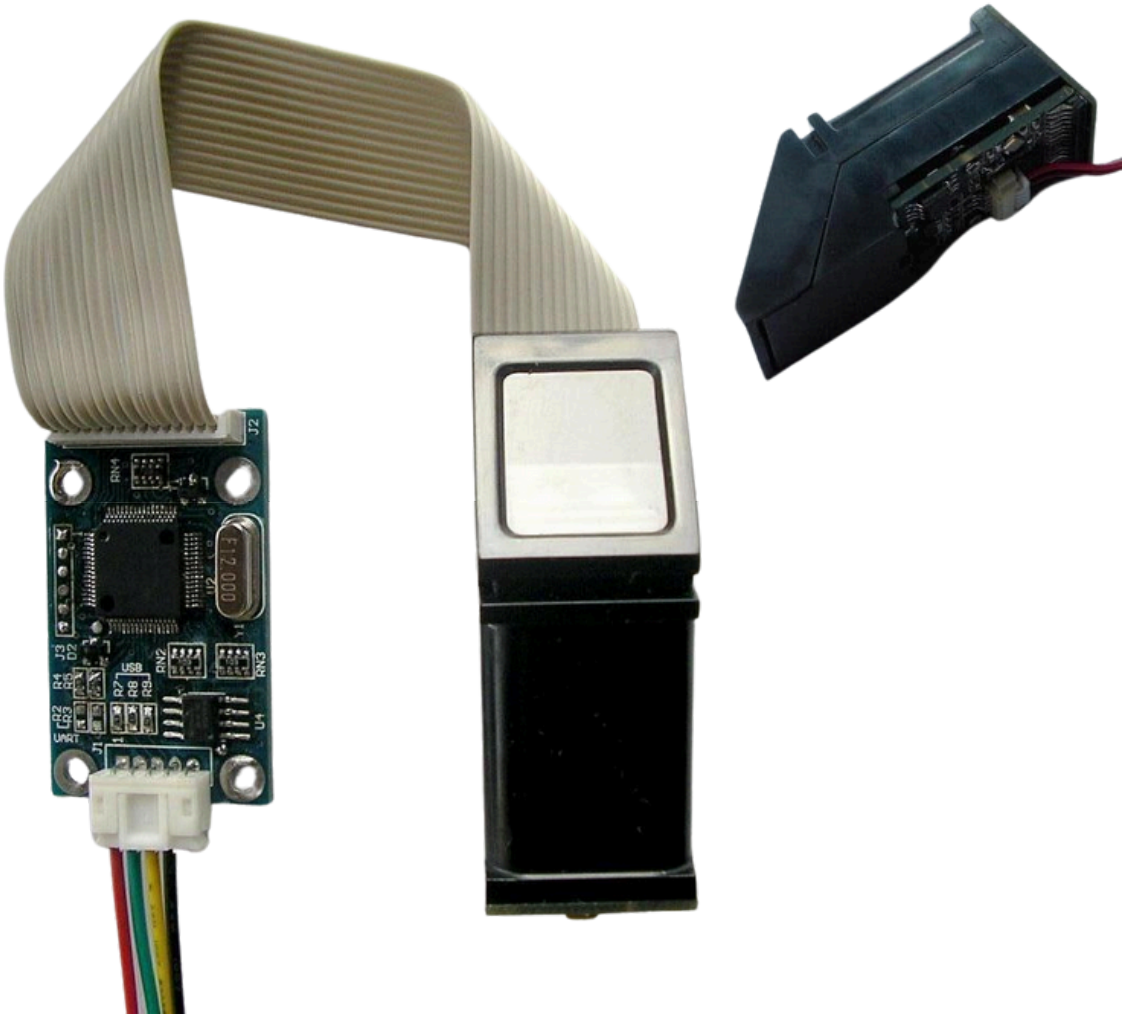
# ZFM-20 Series
# Fingerprint Identification Module

# User Manual



Hangzhou Zhian Technologies Co., Ltd

Sep 2008  Ver: 1.4

# Preface & Declaration

Thank you for you selection of ZFM-20 series Fingerprint Identification Module (Module) of Zhiantec Technologies Co., Ltd. (Zhiantec).

The Manual is targeted for hardware & software developing engineer, covering hardware interface, system resource, instruction system, installment information, etc. To ensure the developing process goes smoothly, it is highly recommended the Manual is read through carefully.

We will try our best to assure you the correctness of the Manual. However, should you find any problem or error with it, feel free to contact us or the sales representative of us. We would be very grateful.

Holding the principle of constantly improving and perfecting products, so both the module and contents of the Manual might subject to changes. Sorry for separate notice. You may visit our website or call us for the latest information.

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

No responsibility or liability is assumed by Zhiantec 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.

All products are sold subject to Zhiantec¡s terms and conditions of sale supplied at the time of order acknowledgment. Testing, tool and other quality control techniques are used to the extent Zhiantec considers necessary to support the warranty of relevant performance of its products to the specifications, except as expressly agreed to in writing by government requirements, testing of all parameters of each product is not necessarily performed.

Copyright ? 2007¨2008 by Hangzhou Zhian Technologies Co., Ltd. All rights reserved.

# Contents


I Introduction---------- 1 -
1.1 Operation Principle....- 1 -
1.2 Order Information....- 1 -
II Main Parameters---------- 2 -
III Hardware Interface---------- 3 -
3.1 Connecting with upper computer (J1 on board)....- 3 -
3.1.1 Serial Communication....- 3 -
3.1.2 USB communication....- 4 -
3.2 Connecting with sensor (J2 on board)....- 4 -
IV System Resources---------- 5 -
4.1 Notepad....- 5 -
4.2 Buffer....- 5 -
4.2.1 Image buffer....- 5 -
4.2.2 Character file buffer....- 5 -
4.3 Fingerprint Library....- 5 -
4.4 System Confiuration Parameter....- 5 -
4.4.1 Baud rate control (Parameter Number: 4)....- 6 -
4.4.2 Security Level (Parameter Number: 5)....- 6 -
4.4.3 Data package length (Parameter Number: 6)....- 6 -
4.5 System status register....- 6 -
4.6 Module address....- 6 -
4.7 Random number generator....- 6 -
V Communication Protocol---------- 7 -
5.1 Data package format....- 7 -
5.2 Check and acknowledgement of data package....- 7 -
VI Module Instruction System---------- 9 -
6.1 System-related instructions....- 9 -
6.1.1 Communicate link: handshake....- 9 -
Confirm that communicate is connect between module and upper monitor....- 9 -
6.1.2 Set Module address: SetAdder....- 9 -
6.1.3 Set module system¡s basic parameter: SetSysPara....- 10 -
6.1.4 Read system Parameter: ReadSysPara....- 10 -
6.1.5 Read valid template number: TempleteNum....- 11 -
6.2 Fingerprint-processing instructions....- 11 -
6.2.1 To collect finger image: GenImg....- 11 -
6.2.2 Upload image: UpImage....- 12 -
6.2.3 Download the image: DownImage....- 12 -
6.2.4 To generate character file from image: Img2Tz....- 13 -
6.2.5 To generate template: RegModel....- 13 -
6.2.6 To upload character or template: UpChar....- 14 -
6.2.7 To download character file or template: DownChar....- 14 -
6.2.8 To store template: Store....- 15 -
6.2.9 To read template from Flash library: LoadChar....- 15 -
6.2.10 To delete template: DeletChar....- 16 -

6.2.1 To empty finger library: Empty.....- 16 -
6.2.11 To carry out precise matching of two finger templates: Match.....- 17 -
6.2.12 To search finger library: Search.....- 17 -
6.3 Other instructions.....- 18 -
6.3.1 To generate a random code : GetRandomCode.....- 18 -
6.3.2 To write note pad: WriteNotepad.....- 18 -
6.3.3 To read note pad:ReadNotepad.....- 19 -
6.4 Instruction Table.....- 20 -
6.4.1 Classified by functions.....- 20 -
6.4.2 Classified by instruction code.....- 20 -
Dimentions.....- 21 -
F1.1 Dimentions of spilt type Module.....- 21 -
F1.2 Dimentions of integral type Module.....- 21 -

# I Introduction

ZFM-20 series are separate fingerprint identification modules proposed by Hangzhou Zhian Technologies Co., Ltd., which takes Synochip DSP as the main processor and optical sensor with Zhian¡s own intellectual property rights. The module performs series of functions like fingerprint enrollment, image processing, fingerprint matching, serching and template storage.

## 1.1 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 specifc 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.

## 1.2 Order Information

Naming of our fingerprint modules follows the following rule. When placing order with us, please fill the correct the type name, so that we can provide better service.

ZFM －20 x x x －x x x －Vxx

Software edition (option)

Wire Length (unit: mm; option)

finger library capacity A：120

B：375

C：880

Interface U： USB1.1

S：Series：UART

Structure mode 0： split

1： integral

optical 20 series

Zhiantec fingerprint module

Note: 1） Wire length means the length of parallel wire which connects optical sensor and main board. It is on mode.

2) Software edition can be omitted in first order or neglected at all. By default, we take it as the latest edition.

# II Main Parameters

| | | | |
|---|---|---|---|
| Power | DC 3.6V-6.0V | Interface | UART(TTL logical level)/ USB 1.1 |
| Working current | Typical: 100mA<br>Peak: 150mA | Matching Mode | 1:1 and 1:N |
| Baud rate | (9600*N)bps,<br>N=1～12 (default N=6) | Character file size | 256 bytes |
| Image acquiring time | <1s | Template size | 512 bytes |
| Storage capacity | 120/ 375/ 880 | Security level | 5 (1, 2, 3, 4, 5(highest)) |
| FAR | <0.001% | FRR | <0.1% |
| Average searching time | < 1s (1:880) | Window dimension | 14mm*18mm |
| Working environment | Temp: -10℃- +40℃<br>RH: 40%-85% | Storage environment | Temp: -40℃- +85℃<br>RH: <85% |
| Outline Dimention | Split type | Module: 42*25*8.5mm<br>(install dimension: 31.5*19mm)<br>Sensor:56*20*21.5mm | |
| | Integral type | 56*20*21.5mm | |

# III Hardware Interface

## 3.1 Connecting with upper computer(J1 on board)

Whether the interface is UART or USB (hardware setting is different when out of factory, please don¡t misuse), on PCB board the connector is uniform. For split type, 5-pin connector (J1) with 2.0mm space between; for integral type, 4-pin connector (J1) with 1.27mm space between.

Unless required specially by user, the connecting cable with upper computer is 150mm.

### 3.1.1 Serial Communication

When the FP module communicates with user device, definition of J1 is as follows:

| Pin Nmuber | Name | Type | Function Description |
|---|---|---|---|
| 1 | Vin | in | Power input（cable color: red） |
| 2 | TD | out | Data output. TTL logical level (cable color: green） |
| 3 | RD | in | Data input. TTL logical level (cable color: whrite) |
| 4 | GND | — | Signal ground. Connected to power ground (cable color: black) |
| 5 | NC | — | Not connect. (doesn¡t exist with integral type) |

#### 3.1.1.1 Hardware connection

Via serial interface, the Module may communicate with MCU of 3.3V or 5V power: TD (pin 2 of J1) connects with RXD (receiving pin of MCU), RD (pin 3 of J1) connects with TXD (transferring pin of MCU). Should the upper computer (PC) be in RS-232 mode, please add level converting circuit, like MAX232, between the Module and PC.

#### 3.1.1.2 Serial communication protocol

The mode is semiduplex asychronism 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 an ending bit. There is no check bit.

| 起始位 | | | | | | | | | 停止位 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | D0 | D1 | D2 | D3 | D4 | D5 | D6 | D7 | 1 |

1 0 位帧格式

#### 3.1.1.3 Reset time

At power on, it takes about 500ms for initialization. During this period, the Module can¡t accept commands for upper computer.

3.1.1.4 Electrical paramenter (All electrical level takes GND as reference)

1. Power supply

| Item | Parameter | | | Unit | Note |
|---|---|---|---|---|---|
| | Min | Typ | Max | | |
| Power Voltage (Vin) | 3.6 | | 6.0 | V | Normal working value. |
| Maximum Voltage (Vinmax) | －0.3 | | 7.0 | V | Exceeding the Maximum rating may cause permant harm to the Module. |
| Operation Current (Icc) | 90 | 100 | 110 | mA | |
| Peak Current (Ipeak) | | | 150 | mA | |

2. TD（output, TTL logic level）

| Item | Condition | Parameter | | | Unit | Note |
|---|---|---|---|---|---|---|
| | | Min | Typ | Max | | |
| $V_{OL}$ | $I_{OL}$=－4mA | | | 0.4 | V | Logic 0 |
| $V_{OH}$ | $I_{OH}$= 4mA | 2.4 | | 3.3 | V | Logic 1 |

3. RD（input, TTL logic level）

| Item | Condition | Parameter | | | Unit | Note |
|---|---|---|---|---|---|---|
| | | Min | Typ | Max | | |
| VIL | | | | 0.6 | V | Loigc 0 |
| VIH | | 2.4 | | | V | Logic 1 |
| IIH | | | | | mA | |
| VImax | VIH =5V | | 1 | | uA | |
| | VIH =3.3V | | 30 | | V | |
| | | －0.3 | | 5.5 | | Maximum input voltage |

## 3.1.2 USB communication

When it¡s USB communication, definition of J1 is:

| Pin Num | Name | Type | Function Description |
|---|---|---|---|
| 1 | Vin | in | Power supply input（refer to 3.1.1.4 for electrical parameter） |
| 2 | DP+ | In/Out | USB data。 USB data。 Signal ground. Connected to power ground. Earth Ground. Float or connect to the shield layer of cable. (doesn¡t exist with Integral type) |
| 3 | DP- | In/Out | |
| 4 | GND | — | |
| 5 | END | — | |

# 3.2 Connecting with sensor (J2 on board)

15-pin single-row connector (J2) (connecting a 15-wire flat cable) with 1.25mm space between, serves as the connector between the main board and the optical sensor. Unless as specially required by usder, the default cable length is 150mm。

For integral type, user needn¡t to worry, it¡s connected internally.

# IV System Resources

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

## 4.1 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_ReadNotepad.

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

## 4.2 Buffer

There are an image buffer and two 512-byte-character-file buffer within the RAM space of the module. Users can read & write any of the buffers by instructions.

Note: Contents of the above buffers will be lost at power-off.

### 4.2.1 Image buffer

ImageBuffer serves for image storage and the image format is 256*288 pixels.
When transferring through UART, to quicken speed, only the upper 4 bits of the pixel is transferred (that is 16 grey degrees). And two adjacent pixels of the same row will form a byte before the transferring. When uploaded to PC, the 16-grey-degree image will be extended to 256-grey-degree format. That¡s 8-bit BMP format.

When transferring through USB, the image is 8-bit pixel, that¡s 256 grey degrees.

### 4.2.2 Character file buffer

Character file buffer, CharBuffer1, CharBuffer2, can be used to store both character file and template file.

## 4.3 Fingerprint Library

Synstem sets aside a certain space within Flash for fingerprint template storage, that¡s fingerprint library. Contents of the library remain at power off.
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.

## 4.4 System Confiuration Parameter

To facilitate user¡s developing, Module opens part system parameters for use. And the basic instructions are SetSysPara & ReadSysPara. Both instructions take Parameter Number as parameter.

When upper computer sends command to modify parameter, Module first responses with original configurations, then performs the parameter modification and writes configuration record into Flash. At the next startup, system will run with the new configurations.

### 4.4.1 Baud rate control (Parameter Number: 4)

The Parameter controls the UART communication speed of the Modul. Its value is an integer N, N= [1, 12]. Cooresponding baud rate is 9600*N bps。

### 4.4.2 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 cooresponding 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.

### 4.4.3 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.

## 4.5 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:

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

Note: ① Busy：1 bit. 1: system is executing commands; 0: system is free; ② Pass：1 bit. 1: find the matching finger; 0: wrong finger; ③ PWD： 1 bit. 1: Verified device¡s handshaking password. ④ ImgBufStat：1 bit. 1: image buffer contains valid image.

## 4.6 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 *SetAdder.* The new modified address remains at power off.

## 4.7 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.

# V Communication Protocol

The protocol defines the data exchanging format when ZFM-20 series communicates with upper computer. The protocol and instruction sets apples for both UART and USB communication mode. For PC, USB interface is strongly recommended to improve the exchanging speed, especially in fingerprint scanning device.

## 5.1 Data package format

When communicating, the transferring and receiving of command/data/result are all wrapped in data package format.

Data package format

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

Definition of Data package

| Name | Symbol | Length | Description |
|---|---|---|---|
| Header | Start | 2 bytes | Fixed value of EF01H; High byte transferred first. Default value is 0xFFFFFFFF, which can be modified by command. High byte transferred first and at wrong adder value, module will reject to transfer. |
| Adder | ADDER | 4 bytes | |
| Package identifier | PID | 1 byte | 01H Command packet;<br>02H Data packet; Data packet shall not appear alone in executing processs, must follow command packet or acknowledge packet.<br>07H Acknowledge packet;<br>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 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 contens. Overflowing bits are omitted. high byte is transferred first. |

## 5.2 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 :*

1. 00h: commad execution complete;
2. 01h: error when receiving data package;
3. 02h: no finger on the sensor;
4. 03h: fail to enroll the finger;
5. 06h: fail to generate character file due to the over-disorderly fingerprint image;
6. 07h: fail to generate character file due to lackness of character point or over-smallness of fingerprint image
7. 08h: finger doesn¡t match;
8. 09h: fail to find the matching finger;
9. 0Ah: fail to combine the character files;
10. 0Bh: addressing PageID is beyond the finger library; 11. 0Ch: error when reading template from library or the template is invalid; 12. 0Dh: error when uploading template; 13. 0Eh: Module can¡t receive the following data packages. 14. 0Fh: error when uploading image; 15. 10h: fail to delete the template; 16. 11h: fail to clear finger library; 17. 15h: fail to generate the image for the lackness of valid primary image; 18. 18h: error when writing flash; 19. 19h: No definition error; 20. 1Ah: invalid register number; 21. 1Bh: incorrect configuration of register; 22. 1Ch: wrong notepad page number; 23. 1Dh: fail to operate the communication port; 24. others: system reserved;

# VI Module Instruction System

ZFM-20 series provide 23 instructions. Through combination of different instructions, application program may realize muti finger authentication functions. All commands/data are transferred in package
format. *Refer to 5.1 for the detailed information of package.*

## 6.1 System-related instructions

### 6.1.1 Communicate link: handshake

Description:

Confirm that communicate is connect between module and upper monitor

Input Parameter: control code 0

Return Parameter: confirmation code;

Instruction code: 17H

Command (or instruction) package format:

| 2 bytes Header | 4bytes Chip address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 1byte Control code | 2 bytes Checksum |
|---|---|---|---|---|---|---|
| EF01H | xxxx | 01H | 0004H | 17H | 0 | 001CH |

Acknowledge package format:

| 2 bytes Heade | 4bytes Chip address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | xxxx | 07H | 0003H | xxH | sum |

Note: Confirmation code=00H: Port operation complete;

Confirmation code=01H: error when receiving package;

Confirmation code=1dH: fail to operate the communication port;

### 6.1.2 Set Module address: SetAdder

Description: Set Module address. (Refer to 4.7 for adderss information)

Input Parameter: None;

Return Parameter: Confirmation code (1 byte)

Instruction code: 15H

Command (or instruction) package format:

| 2 bytes Heade | 4bytes Original Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 4 bytes New Module address | 2 bytes Checksum |
|---|---|---|---|---|---|---|
| EF01H | xxxx | 01H | 0007H | 15H | xxxx | sum |

Acknowledge package format:

| 2 bytes Heade | 4bytes New Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | xxxx | 0007H | 07H | xxH | Sum |

Note: Confirmation code=00H: address setting complete;
Confirmation code=01H: error when receiving package;

### 6.1.3 Set module system¡s basic parameter:SetSysPara

Description: Operation parameter settings. (Refer to 4.4 for more information)
Input Parameter: Parameter number;
Return Parameter: Confirmation code (1 byte)
Instruction code: 0eH
Command (or instruction) package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 1byte Parameter number | 1byte Contents | 2 bytes Checksum |
|---|---|---|---|---|---|---|---|
| EF01H | xxxx | 01H | 0005H | 0eH | 4/5/6 | xx | sum |

Acknowledge package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | xxxx | 0007H | 03H | xxH | Sum |

Note: Confirmation code=00H: parameter setting complete;
Confirmation code=01H: error when receiving package;
Confirmation code=1aH: wrong register number;

### 6.1.4 Read system Parameter:ReadSysPara

Description: Read Module¡s status register and system basic configuration parameters;Refer to 4.4 for system configuration parameter and 4.5 for system status register)
Input Parameter：none
Return Parameter：Confirmation code (1 byte) + basic parameter (16 bytes)
Instuction code: 0fH
Command (or instruction) package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | 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 |
| EF01H | xxxx | 07H | 3+16 (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) |
|---|---|---|---|
| Status register | Contents of system status register | 0 | 1 |
| System identifier code | Fixed value: 0x0009 | 1 | 1 |

| | | | |
|---|---|---|---|
| Finger library size | Finger library size | 2 | 1 |
| Security level | Security level (1, 2, 3, 4, 5) | 3 | 1 |
| Device address | 32-bit device address | 4 | 2 |
| Data packet size | Size code (0, 1, 2, 3) | 6 | 1 |
| Baud settings | N (baud = 9600*N bps) | 7 | 1 |

### 6.1.5 Read valid template number:TempleteNum

Description: read the current valid template number of the Module
Input Parameter: none
Return Parameter: Confirmation code (1 byte)，template number:N
Instuction code: 1dH
Command (or instruction) package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | xxxx | 01H | 0003H | 1DH | 0021H |

Acknowledge package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Template number | 2 bytes Checksum |
|---|---|---|---|---|---|---|
| EF01H | xxxx | 07H | 5 | xxH | N | sum |

Note: Confirmation code=00H: read complete;
Confirmation code=01H: error when receiving package;

## 6.2 Fingerprint-processing instructions

### 6.2.1 To collect finger image: GenImg

Description: detecting finger and store the detected finger image in ImageBuffer while returning successfull 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)
Instuction code: 01H
Command (or instruction) package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | Xxxx | 01H | 0003H | 01H | 0005H |

Acknowledge package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | Xxxx | 07H | 0003H | xxH | Sum |

Note: Confirmation code=00H: finger collection successs;
Confirmation code=01H: error when receiving package;
Confirmation code=02H: can¡t detect finger;
Confirmation code=03H: fail to collect finger;

### 6.2.2 Upload image: UpImage

Description: to upload the image in Img_Buffer to upper computer. Refer to 1.1.1 for more about image buffer.
Input Parameter: none
Return Parameter: Confirmation code (1 byte)
Instuction code: 0aH
Command (or instruction) package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | Xxxx | 01H | 0003H | 0AH | 000EH |

Acknowledge package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | Xxxx | 07H | 0003H | xxH | sum |

Note 1： Confirmation code=00H: ready to transfer the following data packet;
Confirmation code=01H: error when receiving package;
Confirmation code=0fH: fail to transfer the following data packet;
2：Module shall transfer the following data packet after responding to the upper computer.

### 6.2.3 Download the image: DownImage

Description: to download image from upper computer to Img_Buffer. Refer to 1.1.1 for more about the image buffer.
Input Parameter: none
Return Parameter: Confirmation code (1 byte)
Instuction code: 0bH
Command (or instruction) package format:

| 2 bytes Header | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | Xxxx | 01H | 0003H | 0bH | 000FH |

Acknowledge package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | Xxxx | 07H | 0003H | xxH | sum |

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

Confirmation code=01H: error when receiving package;
Confirmation code=0eH: fail to transfer the following data packet;

2：Module shall transfer the following data packet after responding to the upper computer. Data package length must be 64, 128, or 256。

## 6.2.4 To generate character file from image:Img2Tz

Description: to generate character file from the original finger image in ImageBuffer and store the file in CharBuffer1 or CharBuffer2.

Input Parameter: BufferID (character file buffer number)

Return Parameter: Confirmation code (1 byte)

Instuction code: 02H

Command (or instruction) package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 1 byte Buffer number | 2 bytes Checksum |
|---|---|---|---|---|---|---|
| EF01H | xxxx | 01H | 0004H | 02H | BufferID | sum |

Note: BufferID of CharBuffer1 and CharBuffer2 are 1h and 2h respectively. Other values (except 1h, 2h) would be processed as CharBuffer2.

Acknowledge package format:

| 2 bytes | 4bytes | 1 byte | 2 bytes | 1 byte | 2 bytes |
|---|---|---|---|---|---|
| Header | Module address | Package identifier | Package length | Confirmation code | Checksum |
| EF01H | xxxx | 07H | 0003H | XxH | sum |

Note: Confirmation code=00H: generate character file complete;
Confirmation code=01H: error when receiving package;
Confirmation code=06H: fail to generate character file due to the over-disorderly fingerprint image;
Confirmation code=07H: fail to generate character file due to lackness of character point or over-smallness of fingerprint image;
Confirmation code=15H: fail to generate the image for the lackness of valid primary image;

## 6.2.5 To generate template: RegModel

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

Input Parameter：none

Return Parameter：Confirmation code (1 byte)

Instuction code: 05H

Command (or instruction) package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | xxxx | 01H | 0003H | 05H | 09H |

 Acknowledge package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | xxxx | 07H | 0003H | xxH | sum |

Note: Confirmation code=00H: operation success;
Confirmation code=01H: error when receiving package;
Confirmation code=0aH: fail to combine the character files. That¡s, the character files don¡t belong to one finger.

## 6.2.6 To upload character or template:UpChar

Description: to upload the character file or template of CharBuffer1/CharBuffer2 to upper computer;
Input Parameter: BufferID (Buffer number)
Return Parameter: Confirmation code (1 byte)
Instuction code: 08H
Command (or instruction) package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 1 byte Buffer number | 2 bytes Checksum |
|---|---|---|---|---|---|---|
| EF01H | xxxx | 01H | 0004H | 08H | BufferID | sum |

Note: BufferID of CharBuffer1 and CharBuffer2 are 1h and 2h respectively. Other values (except 1h, 2h) would be processed as CharBuffer2.
Acknowledge package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | xxxx | 07H | 0003H | xxH | sum |

Note 1: Confirmation code=00H: ready to transfer the following data packet;
Confirmation code=01H: error when receiving package;
Confirmation code=0dH: error when uploading template;
2: Module shall transfer following data packet after responding to the upper computer.;
3: The instruction doesn¡t affect buffer contents.

## 6.2.7 To download character file or template:DownChar

Description: to download character file or template from upper computer to the specified buffer of Module;
Input Parameter: BufferID (buffer number)
Return Parameter: Confirmation code (1 byte)
Instuction code: 09H
Command (or instruction) package format:

| 2 bytes Header | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 1 byte buffer number | 2 bytes Checksum |
|---|---|---|---|---|---|---|

| | | | | | | |
|---|---|---|---|---|---|---|
| EF01H | xxxx | 01H | 0004H | 09H | BufferID | sum |

Note: BufferID of CharBuffer1 and CharBuffer2 are 1h and 2h respectively. Other values (except 1h, 2h) would be processed as CharBuffer2.

Acknowledge package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | xxxx | 07H | 0003H | xxH | sum |

Note 1: Confirmation code=00H: ready to transfer the following data packet;
Confirmation code=01H: error when receiving package;
Confirmation code=0eH: fail to receive the following data packages.
2: Module shall transfer the following data packet after responding to the upper computer.

## 6.2.8 To store template: Store

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

Input Parameter: BufferID(buffer number), PageID（Flash location of the template, two bytes with high byte front and low byte behind）

Return Parameter: Confirmation code (1 byte)

Instuction code: 06H

Command (or instruction) package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 1 byte buffer number | 2 bytes Location number | 2 bytes Checksum |
|---|---|---|---|---|---|---|---|
| EF01H | xxxx | 01H | 06H | 0006H | BufferID | PageID | sum |

Note: BufferID of CharBuffer1 and CharBuffer2 are 1h and 2h respectively. Other values (except 1h, 2h) would be processed as CharBuffer2.

Acknowledge package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | Xxxx | 07H | 0003H | xxH | sum |

Note: Confirmation code=00H: storage success;
Confirmation code=01H: error when receiving package;
Confirmation code=0bH: addressing PageID is beyond the finger library;
Confirmation code=18H: error when writing Flash.

## 6.2.9 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(buffer number), PageID (Flash location of the template, two bytes with high byte front and low byte behind)。

Return Parameter: Confirmation code (1 byte)

Instuction 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 |
| EF01H | xxxx | 01H | 0006H | 07H | BufferID | PageID | sum |

Note: BufferID of CharBuffer1 and CharBuffer2 are 1h and 2h respectively. Other values (except 1h, 2h) would be processed as CharBuffer2.

Acknowledge package format:

| 2 bytes Header | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | xxxx | 07H | 0003H | XxH | sum |

Note: Confirmation code=00H: load success;

Confirmation code=01H: error when receiving package;

Confirmation code=0cH: error when reading template from library or the readout template is invalid;

Confirmation code=0BH: addressing PageID is beyond the finger library;

## 6.2.10 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 (template number in Flash), N (number of templates to be deleted)

Return Parameter: Confirmation code (1 byte)

Instuction code: 0cH

Command (or instruction) package format:

| 2 bytes Header | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 2 bytes Page number | 2bytes number of templates to be deleted | 2 bytes Checksum |
|---|---|---|---|---|---|---|---|
| EF01H | Xxxx | 01H | 0007H | 0cH | PageID | N | sum |

Acknowledge package format:

| 2 bytes Header | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | Xxxx | 07H | 0003H | xxH | sum |

Note: Confirmation code=00H: delete success;

Confirmation code=01H: error when receiving package;

Confirmation code=10H: faile to delete templates;

## 6.2.11 To empty finger library: Empty

Description: to delete all the templates in the Flash library

Input Parameter: none

Return Parameter: Confirmation code (1 byte)

Instuction code: 0dH

Command (or instruction) package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | Xxxx | 01H | 0003H | 0dH | 0011H |

Acknowledge package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | Xxxx | 07H | 0003H | xxH | sum |

Note: Confirmation code=00H: empty success;
Confirmation code=01H: error when receiving package;
Confirmation code=11H: fail to clear finger library;

## 6.2.12 To carry out precise matching of two finger templates: Match

Description: to carry out precise matching of templates from CharBuffer1 and CharBuffer2, providing matching results.

Input Parameter: none

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

Instuction code: 03H

Command (or instruction) package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | Xxxx | 01H | 0003H | 03H | 0007H |

Acknowledge package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Matching score | 2 bytes Checksum |
|---|---|---|---|---|---|---|
| EF01H | Xxxx | 07H | 0005H | XxH | XxH | sum |

Note 1: Confirmation code=00H: templates of the two buffers are matching!
Confirmation code=01H: error when receiving package;
Confirmation code=08H: templates of the two buffers aren¡t matching;
2: The instruction doesn¡t affect the contents of the buffers.

## 6.2.13 To search finger library: Search

Description: to search the whole finger library for the template that matches the one in CharBuffer1 or CharBuffer2. When found, PageID will be returned.

Input Parameter: BufferID, StartPage (searching start address), PageNum（searching numbers）

Return Parameter: Confirmation code (1 byte), PageID (matching templates location)

Instuction 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 |
|---|---|---|---|---|---|---|---|---|
| EF01H | xxxx | 01H | 0008H | 04H | BufferID | StartPage | PageNum | sum |

Note: BufferID of CharBuffer1 and CharBuffer2 are 1h and 2h respectively. Other values (except 1h, 2h) would be processed as CharBuffer2.

Acknowledge package format:

| 2 bytes Header | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes 页码 | 2 bytes 得分 | 2 bytes Checksum |
|---|---|---|---|---|---|---|---|
| EF01H | 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);
2: The instruction doesn¡t affect the contents of the buffers.

# 6.3 Other instructions

## 6.3.1 To generate a random code :

GetRandomCode

Description: to command the Module to generate a random number and return it to upper computer;Refer to 4.8 for more about Random Number Generator;

Input Parameter: none

Return Parameter: Confirmation code (1 byte)

Instuction code: 14H

Command (or instruction) package format:

| 2 bytes Header | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | xxxx | 01H | 0003H | 14H | 0018H |

Acknowledge package format:

| 2 bytes Header | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 4 bytes Random number | 2 bytes Checksum |
|---|---|---|---|---|---|---|
| EF01H | xxxx | 07H | 0007H | xxH | xxxx | sum |

Note: Confirmation code=00H: generation success;
Confirmation code=01H: error when receiving package;

## 6.3.2 To write note pad: WriteNotepad

Description: for upper computer to write data to the specified Flash page (refer to 4.1 for more about Note pad). Also see ReadNotepad;

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

Return Parameter: Confirmation code (1 byte)

Instuction code: 18H

Command (or instruction) package format:

| 2 | 4bytes | 1 byte | 2 bytes | 1 byte | 1byte | 32 bytes | 2 bytes |
|---|---|---|---|---|---|---|---|
| bytes Heade | Module address | Package identifier | Package length | Instruction code | Page number | Data content | Checksum |
| EF01H | xxxx | 01H | 4+32(0024H) | 18H | 0~15 | content | sum |

Acknowledge package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 2 bytes Checksum |
|---|---|---|---|---|---|
| EF01H | xxxx | 07H | 0003H | xxH | sum |

Note: Confirmation code=00H: write success;

Confirmation code=01H: error when receiving package;

## 6.3.3 To read note pad: ReadNotepad

Description: to read the specified page¡s data content; Refer to 4.1 for more about user note pad. Also see WriteNotepad.

Input Parameter: none

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

Instuction code: 19H

Command (or instruction) package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Instruction code | 1byte Page number | 2 bytes Checksum |
|---|---|---|---|---|---|---|
| EF01H | xxxx | 01H | 0004H | 19H | 0~15 | xxH |

Acknowledge package format:

| 2 bytes Heade | 4bytes Module address | 1 byte Package identifier | 2 bytes Package length | 1 byte Confirmation code | 32bytes User content | 2 bytes Checksum |
|---|---|---|---|---|---|---|
| EF01H | xxxx | 07H | 3+32(0023H) | xxH | User content | sum |

Note: Confirmation code=00H: read success;

Confirmation code=01H: error when receiving package;

## 6.4 Instruction Table

### 6.4.1 Classified by functions

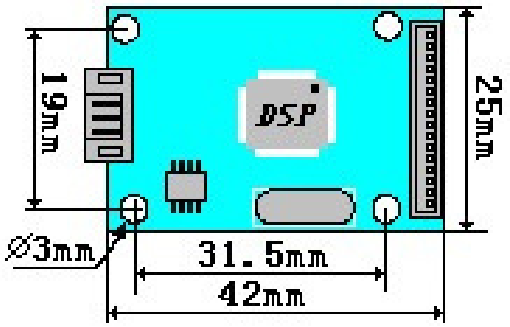
| type | num | code | description | Type | num | Code | description |
|---|---|---|---|---|---|---|---|
| *System-related* | 1 | 17H | Communicate link | *Finerprint processing* | 12 | 09H | To download template |
| | 2 | 15H | To set device address | | 13 | 06H | To store template; |
| | 3 | 0EH | To set system Parameter | | 14 | 07H | to read/load template |
| | 4 | 0FH | To read system Parameter | | 15 | 0CH | to delete tempates |
| | 5 | 1DH | To read finger template numbers | | 16 | 0DH | to empty the library |
| | 6 | 01H | Collect finger image | | 17 | 03H | Carry out precise matching of two templates; |
| *Finerprint processing* | 7 | 0AH | To upload image | | 18 | 04H | Search library finger to get random |
| | 8 | 0BH | To download image | | 19 | 14H | code |
| | 9 | 02H | To generate character file from image | | 20 | 18H | to write note pad |
| | 10 | 05H | To combine character files and generate template | *others* | 21 | 19H | To read note pad |
| | 11 | 08H | to upload template | | | | |

### 6.4.2 Classified by instruction code

| code | identifier | Description | Code | Identifier | Description |
|---|---|---|---|---|---|
| 01H | GenImg | Collect finger image | 0CH | DeletChar | to delete tempates |
| 02H | Img2Tz | To generate character file from image | 0DH | Empty | to empty the library |
| 03H | Match | Carry out precise matching of two templates; | 0EH | SetSysPara | To set system Parameter |
| 04H | Serach | Search the finger library | 0FH | ReadSysPara | To read system Parameter |
| 05H | RegModel | To combine character files and generate template | 14H | GetRandomCode | to get random code |
| 06H | Store | To store template; | 15H | SetAdder | To set device address |
| 07H | LoadChar | to read/load template | 17H | handshake | Communicate link |
| 08H | UpChar | to upload template | 18H | WriteNotepad | to write note pad To read |
| 09H | DownChr | to download template | 19H | ReadNotepad | note pad To read finger |
| 0AH | UpImage | To upload image | 1DH | TempleteNum | template numbers |
| 0BH | DownImage | To download image | | | |

# Dimentions

## F1.1 Dimentions of spilt type Module




## F1.2 Dimentions of integral type Module

