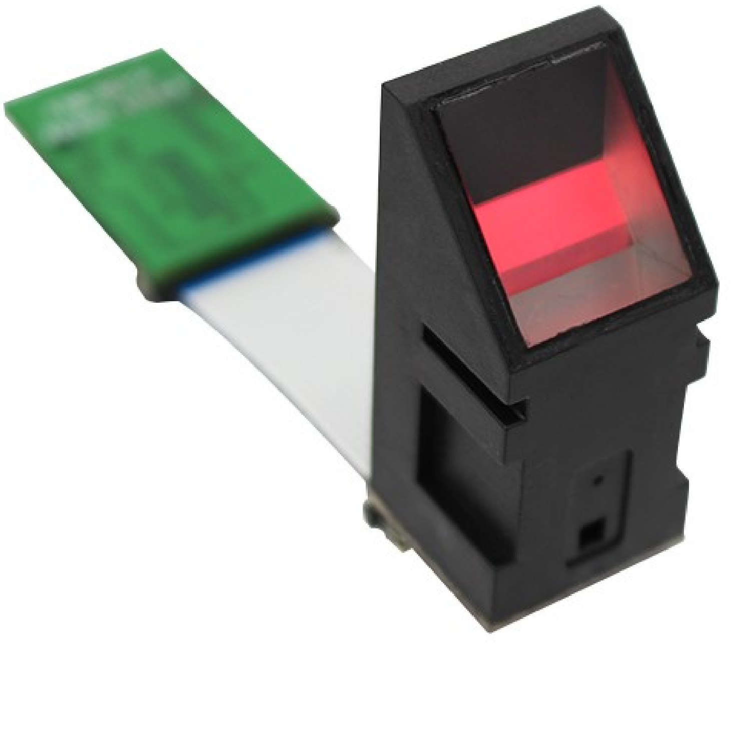
USB Fingerprint Sensor (Model: FS-01)



User Manual

Content

No.	Editor	Modification time	Version	Description
1	Mr. Coyote	20170307	Version 1.0.1.1	Definition
2	Mr. Coyote	20170501	Version 1.0.0.2	Upgraded

1 Summary

FS-01 is the fingerprint module for secondary development which has integrated fingeprint optical sensor, CMOS image sensor, CPU, and Flash memory together. It has many features like small size, low power consumption, simple ports, high reliability, small fingerprint template, large fingerprint capacity, etc. It is convenient to be embedded to fingerprint verification products.

Main Functions:

- Communication interface: USB
- Optic sensor is reliable and Low-cost, High ESD Protection
- 1: N Identification (One-to-Many)
- 1:1 Verification (One-to-One)
- High speed fingerprint identification algorithm engine
- Self study function
- Fingerprint template data read from /write to FLASH memory function
- Get Feature Data of Captured fingerprint and Verify/Identify Downloaded Feature with Captured fingerprint (Specially designed for fingerprint stored in IC card)
- Identify Downloaded Feature with Captured fingerprint
- Security Level setting
- Able to set Baud Rate/ Device ID/Device Password

Applications:

- Access control systems
- Time & Attendance
- Locks, safes
- POS, handheld terminals

1.1 Specifications

Electric Parameters	
Operating Voltage	5V (Typical Value) Range: 4.5V-6.0V
Operating Current	60mA (Typical Value), Peak Current: <100mA 30uA (Sleep Mode)
Fingerprint Image Input Time	<0.2S
Operating Temperature	-10℃-+60℃
Store Temperature	-20℃ - +80℃
Operating/Store Humidity	20%—90% /16%—95%
Performance Parameters	
Fingerprint Sensor	Optical Sensor
Sensor Dimensions:	21*17.4mm
Effective Image Size	256*288 pixels
Image Size	256*288 pixel
Image Pixel	500DPI
Matched Mode	Compared Mode (1:1) Search Mode (1：N)(900 PCS Full registration) < 1 second
Fingerprint Features	496 bytes (Extracting Time < 0.4 second)
Collected Mode	Plane Press
Storage Capacity	900 PCS
Security Level	Five level (from low to high：1、2、3、4、5)
False Accept Rate (FAR)	<0.001% (Security Level：3)
False Reject Rate (FRR)	<0.1 % (Security Level：3)
Search Time	<1.0 second (When 1:1000, average)
Communication Interface	USB1.1/2.0 compatibility

1.2 Factory default Settings

Project	Initial Value
Security Level (1~5)	3
Allow/prohibit to repeatedly detect by fingerprint (Duplication Check (ON/OFF))	ON
Timeout waiting (Time Over)	5s

2 Hardware Description

2.1 Power Delay Time

After the module is powered on, the initialization is needed around 100mS. During this period, the module cannot respond the PC command.

2.2 The Interface Definition (Communication)

Definition of USB Interface pin is as follows:

Pins Numbe	Names	Types	Functional Description
1	Vcc	in	Positive input end of power supply. (5v)
2	D-	In/Out	USB cable (D-)
3	D+	In/Out	USB cable (D+)
4	GND	-	Signal ground. Interior and power ground are onnected

Definition of Usart Interface pin is as follows:

Pins Number	Names	Types	Functional Description
1	VCC	in	positive input end of power supply. (4.5-6v)
2	TXD	In/Out	Equipment serial port of sending end
3	RXD	In/Out	Equipment serial port of receiving end
4	GND	—	Signal ground. Interior and power ground are
5	TOUT	OUT	Induction level output

Notes: In the type column, “in” indicates input to the module, “out” indicates output from the module.

2.3 Fingerprint Resources

In order to meet the requirements from all kinds of customers, a large number of resources are supplied to the user system by the module system.

2.3.1 Buffer

Module RAM resources are as follows:

1) An Image Buffer[256*288]
It is used for depositing the current collected image data and the internal image processing of module.

2) CharBuffer1 [496] and CharBuffer2 [496]
They are used for depositing the current generated characteristic documents and the characteristic documents of module.
The users can read and write any buffer by the command. When the power supply is cut off, the content of image buffer and the characteristic documents buffer is not saved.

2.3.2 Flash Area

Flash resources of module are as follows:

1) Fingerprint database

The module opens a template which is use for depositing the users' fingerprint, it is also called the fingerprint database. When the power supply is cut off, the fingerprint database is saved.

In the fingerprint database, the position of template is deposited orderly by the serial numbers, If the size of fingerprint database is N, the definition of serial numbers in the fingerprint database is as follows:

1、2、3......N-2、N-1、N。They users can only according to the serial numbers visit the content of fingerprint database.

2) Parameters area

In order to convenient the customers to use, the system parameters are opened in the template part for the customers to set according to their developed requirements.

2.3.3 Structure of Fingerprint Template Data

Serial Numbers	(Template)	Checksum
1	496byte	2byte
2	Characteristic Data	Carry out the arithmetic and operation for the characteristic data by the bytes, and take the lower 2 bytes.
Each fingerprint template data is：Template + Checksum		

2.3.4 Fingerprint Characteristic & Definition Of Fingerprint Template

Fingerprint Characteristic:

Each fingerprint image will generate a unique fingerprint characteristic data through the algorithm, the data size is 496 byte.

Fingerprint Template:

It is a combination of two two fingerprint characteristics, that is to say two fingerprint characteristics will generate a unique fingerprint template through the algorithm, the template size is 496 byte.

2.3.5 Structure of Fingerprint Resources

<table>
<tr><td rowspan="8">MCU</td><td rowspan="4">RAM</td><td>Serial No.</td><td>Use</td><td>Size</td></tr>
<tr><td>1</td><td>CharBuf1</td><td>496byte</td></tr>
<tr><td>2</td><td>CharBuf2</td><td>496byte</td></tr>
<tr><td>3</td><td>ImageBuf</td><td>256*288byte</td></tr>
<tr><td rowspan="2">ROM</td><td>1</td><td>parameter list</td><td>128byte</td></tr>
<tr><td>2</td><td>DB</td><td>900*512byte</td></tr>
<tr><td>TTL</td><td>1</td><td colspan="2">TTL communication cache</td></tr>
<tr><td>USB</td><td>1</td><td colspan="2">USB communication cache</td></tr>
</table>

3 API Call Interface

API Call Interface defines the interface function of USB communication that let users easily develop on Windows / Linux / Android / Wince etc.

3.1 Function Declaration List:

3.1.1 Get Template Version: int WINAPI FPVersion(char *p_vers);
3.1.2 Reset Device: int WINAPI InitDevice();
3.1.3 Release Device: int WINAPI CloseDevice();
3.1.4 Communication Test: int WINAPI ConnectTest();
3.1.5 Detect FP Image: int WINAPI IsHaveFinger();
3.1.6 Get FP Character: int WINAPI GetFpchar(int buf);
3.1.7 Get FP Tamplate: int WINAPI GetFpMoudle();
3.1.8 FP Comparison: int WINAPI FpMatch(int *iScore);
3.1.9 Search FP: int WINAPI FpSearch(int *iMbAddress,int *iScore);
3.1.10 Save FP: int WINAPI FpStore(int FPID);
3.1.11 Delete FP: int WINAPI FpDelete(int FPID);
3.1.12 Clear FP: int WINAPI FpEmpty();
3.1.13 Load FP Character: int WINAPI Loadchar(int iPageID);
3.1.14 Upload FP Character: int WINAPI Upchar(byte *pTemplet);
3.1.15 Download FP Character: int WINAPI Downchar(byte *pTemplet);
3.1.16 Upload FP Image: int WINAPI UpImage(byte *pImageData,int *iImageLength);
3.1.17 Download FP Image: int WINAPI DownImage(byte *pImageData,int iLength);
3.1.18 Display FP Image: int WINAPI ShowImage(HWND hWnd,byte *pFingerData);
3.1.19 Baud Rate Level: int WINAPI Set_BaudLevel(int nBaud,int nLevel);
3.1.20 Read BMP Image: int WINAPI ReadBmpFile(const char* pFileName,
byte * pImageData,int im_x,int im_y);
3.1.21 Write BMP Image: int WINAPI WriteBmpFile(const char* pFileName,
byte * pImageData,int im_x,int im_y);

3.2 Detailed Description of Function

3.2.1 Get Fingerprint Version:int WINAPI FPVersion(char *p_vers)

Function: This Function is to get the current version of the software (reference the version table), if any changes, please upgrade API Interface Library files accordingly.

Parameter: char *p_vers Means the current version No.

Returned value: 0 (Success)
Other values please reference the ERROR Returned Value

3.2.2 Reset the Device: int WINAPI InitDevice()

Function: This Function is to reset the fingerprint device to make it enter into working status. Please note that besides get the version, the other Functions must call this Function first before using.

Parameter: None

Returned value: 0 (Success)
Other values please reference the ERROR Returned Value

3.2.3 Release the Device: int WINAPI CloseDevice()
Function: This Function is to release the fingerprint device

Parameter: None
Returned value: 0 (Success)
Other values please reference the ERROR Returned Value

3.2.4 Device Communication Test: int WINAPI ConnectTest()
Function: This Function is to test the communication, it's usually called after device resetting

Parameter: None

Returned value: 0 (Success)
Other values please reference the ERROR Returned Value

3.2.5 Detect Fingerprint Image: int WINAPI IsHaveFinger()
Function: After calling this Function, the device will enroll the image immediately, and check if the fingerprint enrolled or not.

Parameter: None

Returned value: 0 (Success) 2 (No fingerprint)
Other values please reference the ERROR Returned Value

3.2.6 Get Fingerprint Characteristic: int WINAPI GetFpchar(int buf)
Function: After calling this Function, the device will generate a fingerprint characteristic basing on the fingerprint image enrolled.
(Note: Each time when call this Function, please call "Detect Fingerprint Image" first)

Parameter: None

Returned value: 0 (Success)
Other values please reference the ERROR Returned Value

3.2.7 Get Fingerprint Template: int WINAPI GetFpMoudle()
Function: After calling this Function, the device will combine CharBuf 1 and CharBuf2 to generate a fingerprint template, which will be saved in CharBuf 1 and CharBuf 2, and those two buffers are equal.

Parameter: None

Returned value: 0 (Success)

Other values please reference the ERROR Returned Value

(Note: Before asking this Function, please call "Get Fingerprint Characteristic" to get CharBuf 1 and CharBuf2 first)

3.2.8 Fingerprint Comparison: int WINAPI FpMatch(int *iScore)

Function: After calling this Function, the device will compare CharBuf 1 and CharBuf2 through the algorithm, and then return value and output the comparison score.

Parameter: int *iScore Output comparison score

Returned value: 0 (Success)

Other values please reference the ERROR Returned Value

3.2.9 Search Fingerprint: int WINAPI FpSearch(int *iMbAddress,int *iScore)

Function: After calling this Function, the device will search all the fingerprints to find out the ID of the fingerprint which match CharBuf 1 and CharBuf2, and will search the output score at the same time.

Parameter: int *iMbAddress Search the ID of the matched fingerprint characteristic

int *iScore Search the output score

Returned value: 0 (Success)

Other values please reference the ERROR Returned Value

3.2.10 Save Fingerprint: int WINAPI FpStore(int FPID)

Function: This Function is to save the data from CharBuf 1 to a specified database.

Parameter: int FPID Fingerprint location ID

Returned value: 0 (Success)

Other values please reference the ERROR Returned Value

3.2.11 Delete Fingerprint: int WINAPI FpDelete(int FPID)

Function: This Function is to delete the specified fingerprint from the database.

Parameter: int FPID Fingerprint location ID

Returned value: 0 (Success)

Other values please reference the ERROR Returned Value

3.2.12 Clear Fingerprint: int WINAPI FpEmpty()

Function: This Function is to clear all fingerprint characteristics from the database.

Parameter: None

Returned value: 0 (Success)
Other values please reference the ERROR Returned Value

3.2.13 Load Fingerprint Characteristic: int WINAPI Loadchar(int FPID)

Function: This Function is to load the fingerprint characteristic with specified ID to CharBuf 1.

Parameter: int FPID   Fingerprint location ID

Returned value: 0 (Success)
Other values please reference the ERROR Returned Value

3.2.14 Upload Fingerprint Characteristic: int WINAPI Upchar(byte *pTemplet)

Function: This Function is to upload the data from CharBuf 1 to pTemplet.

Parameter: byte *pTemplet   Fingerprint Data (498 byte)
Returned value: 0 (Success)
Other values please reference the ERROR Returned Value

3.2.15 Download Fingerprint Characteristic: int WINAPI Downchar(byte *pTemplet)

Function: This Function is to download the date from pTemplet to CharBuf 1.

Parameter: byte *pTemplet   Fingerprint Data (498 byte)
Returned value: 0 (Success)
Other values please reference the ERROR Returned Value

3.2.16 Upload Fingerprint Image: int WINAPI UpImage(byte *pImageData,int *iImageLength)

Function: This Function is to upload the ImageBuf data from RAM to pImageData and output the iImageLength at the same time.

Parameter: byte *pImageData   Fingerprint Image Date (256*288byte)
Int *iImageLength   Fingerprint Image Size

Returned value: 0 (Success)
Other values please reference the ERROR Returned Value

3.2.17 Download Fingerprint Image: int WINAPI DownImage(byte *pImageData,int iLength)

Function: This Function is to download the date from pImageData to ImageBuf in RAM.

Parameter: byte *pImageData   Fingerprint Image Date (256*288byte)
Int *iImageLength   Fingerprint Image Size

Returned value: 0 (Success)
Other values please reference the ERROR Returned Value

3.2.18 Display Fingerprint Image: int WINAPI ShowImage(HWND hWnd,byte *pFingerData)

Function: This Function is to display the data from pImageData to hwd pc handle.

Parameter: byte *pImageData Fingerprint Image Date (256*288byte)
Int *iImageLength Fingerprint Image Size

Returned value: 0 (Success)
Other values please reference the ERROR Returned Value

3.2.19 Baud Rate Level: int WINAPI Set_BaudLevel(int nBaud,int nLevel)

Function: This Function is to set the Baud Rate and Security Level.

Parameter: int nBaud Baud Rate(9600,38400,57600,115200....)
Int nLevel Security Level

Returned value: 0 (Success)
Other values please reference the ERROR Returned Value

3.2.20 Read BMP Image: int WINAPI ReadBmpFile(const char* pFileName, byte * pImageData,int im_x,int im_y)

Function: This Function is to generate a fingerprint image (Width im_x, height im_y) basing on the data from pImageData.

Parameter: const char* pFileName Fingerprint Image (for example:\\fp.bmp)
Int im_x Width of Image
Int im_y Height of Image

Returned value: 0 (Success)

Other values please reference the ERROR Returned Value

3.2.21 Write BMP Image: int WINAPI WriteBmpFile(const char* pFileName, byte * pImageData,int im_x,int im_y)

Function: This Function is to Read the fingerprint image data from pFileName to pImageData, and read the Width im_x and height im_y of the fingerprint image to pFileName.

Parameter: const char* pFileName Fingerprint Image (for example:\\fp.bmp)
Int im_x Width of Image
Int im_y Height of Image

Returned value: 0 (Success)

Other values please reference the ERROR Returned Value