4357. Embedded Firmware Essentials

**Final Project Report**

**6/1/2015**

**Jae Yang Park (**[**jaeyangp@gmail.com**](mailto:jaeyangp@gmail.com)**)**

**Objective**

This project is to implement the Fingerprint recognition system using a fingerprint sensor module, and mbed platform and mbed application board hardware.

What’s fingerprint recognition?

“It refers to automated method of verifying a match between two human fingerprints. Finger prints are one of many forms of biometrics used to identify individuals and verify their identity.”

**The fundamental of fingerprint**

**Patterns**

The three basic patterns of fingerprint ridges are the arch, loop, and whorl.

* Arch: The ridges enter from one side of the finger, rise in the center forming an arc, and then exit the other side of the finger.
* Loop: The ridges enter from one side of a finger, form a curve, and then exit on that same side.
* Whorl: Ridges form circularly around a central point on the finger.

|  |  |  |
| --- | --- | --- |
| Fingerprint_Arch.jpg | Fingerprint_Loop.jpg | Fingerprint_Whorl.jpg |
| The arch pattern | The loop pattern | The whorl pattern |

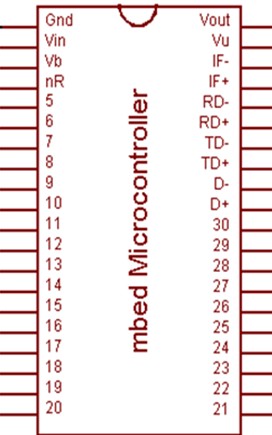
**Minutia**

The major minutia features of fingerprint ridges are ridge ending, bifurcation, and short ridge (or dot).

Minutiae and patterns are very important in the analysis of fingerprints since no two fingers have been shown to be identical.

|  |  |  |
| --- | --- | --- |
| 585px-Ridge_ending.svg.png | 585px-Bifurcation.svg.png | 585px-Short_ridge.svg.png |
| Ridge ending | Bifurcation | Short ridge (dot) |

**Hardware Diagram**

****

Blue

Green

Red

Left

Right

Up

Center

Down

A0

SCK

MOSI

nRESET

128x32 LCD module

5 Way Joystick

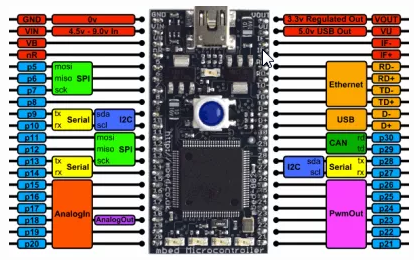
RGB LED

Speaker

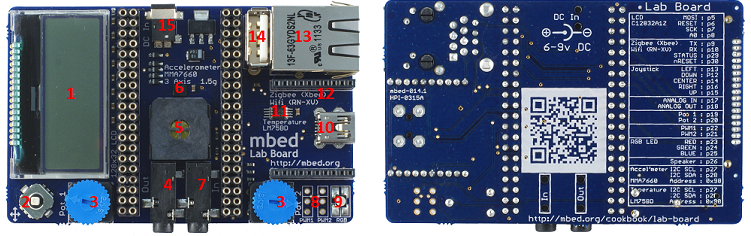
Fingerprint sensor module

GT-511C1R

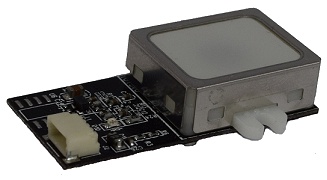
**mbed LPC1768**



**mbed application board**



**Fingerprint reader module GT-511C1R**



**Specification of Fingerprint reader module GT-511C1R**

* High-accuracy and high-speed fingerprint identification technology
* Ultra-thin optical sensor
* 1:1 verification, 1:N identification

|  |  |  |
| --- | --- | --- |
| **Item** | | **Value** |
| CPU | | ARM Cortex M3 Core |
| Sensor | | Optical sensor |
| Effective area of the sensor | | 12 x 12.5 (mm) |
| Image size | | 240 x 216 pixels |
| Resolution | | 450 dpi |
| The maximum number of fingerprints | | 20 fingerprints |
| Matching mode | | 1:1, 1:N |
| The size of template | | 504 bytes (template) + 2 bytes (checksum) |
| Communication interface | | UART, default baud rate = 9600 bps after power on  USB ver 1.1, Full speed |
| False Acceptance Rate (FAR) | | < 0.001% |
| False Rejection Rate (FRR) | | < 0.1% |
| Enrollment time | | < 3 sec (3 fingerprints) |
| Identification time | | < 1.5 sec (20 fingerprints) |
| Operating voltage | | DC 3.3 ~ 6V |
| Operating current | | < 130 mA |
| Operating environment | Temperature | -20˚C ~ +60˚C |
| Humidity | 20% ~ 80% |
| Storage environment | Temperature | -20˚C ~ +60˚C |
| Humidity | 10% ~ 80% |

LCD Screen design

0 deleted

[5] Delete ID

<< MENU >> \* RUN

Press finger to Identify

Press finger to Identify

ID = 0

<< MENU >> \* CHANGE

Press finger to Enroll (1ST)

Fingerprint System v2.0

[5] Delete ID

<< MENU >> \* RUN

Fingerprint System v2.0

[4] Identify ID

<< MENU >> \* RUN

Fingerprint System v2.0

[3] Verify ID

<< MENU >> \* RUN

Fingerprint System v2.0

[2] Enroll ID

<< MENU >> \* RUN

Fingerprint System v2.0

[1] Select ID

<< MENU >> \* RUN

Fingerprint System v2.0

[0] Menu

<< MENU >> \* RUN

finger.h

//

// UCSC ext.

// 2015 Embedded Firmware Essential

// Project

// Fingerprint Security Lock

// Jae-Yang Park

// jaeyangp@gmail.com

//

#ifndef FINGER\_H

#define FINGER\_H

Serial debug(USBTX,USBRX);

DigitalOut myled(LED1);

GT511C3 finger(p9,p10);

C12832 LCD(p5, p7, p6, p8, p11);

InterruptIn JOY\_CENTER(p14);

InterruptIn JOY\_UP(p15);

InterruptIn JOY\_DOWN(p12);

InterruptIn JOY\_LEFT(p13);

InterruptIn JOY\_RIGHT(p16);

PwmOut SPEAKER(p26);

PwmOut LEDR(p23);

PwmOut LEDG(p24);

PwmOut LEDB(p25);

Timer debounce;

#endif

main.cpp

//

// UCSC ext.

// 2015 Embedded Firmware Essential

// Project

// Fingerprint Security Lock

// Jae-Yang Park

// jaeyangp@gmail.com

//

#include "mbed.h"

#include "GT511C3.hpp"

#include "C12832.h"

#include "finger.h"

int progress(int, char \*);

void err\_beep(void);

void good\_beep(void);

void red\_on(void);

void green\_on(void);

void blue\_on(void);

void rgb\_off(void);

void init(void);

void disp\_GT\_info(void);

void disp\_menu(void);

void chk\_Enrolled(int);

void enroll\_id(void);

void verify\_id(void);

void identify\_id(void);

void delete\_id(void);

void select\_id(void);

void joy\_center\_ISR(void);

void joy\_right\_ISR(void);

void joy\_left\_ISR(void);

void joy\_up\_ISR(void);

void joy\_down\_ISR(void);

int EnrollID = 0;

int current = 0;

#define MENU\_SZ 6

#define MAX\_ID 20

char \*menu[] = {"[0] Menu >>",

"[1] Select ID << >>",

"[2] Enroll ID << >>",

"[3] Verify ID << >>",

"[4] Identify ID << >>",

"[5] Delete ID << "};

void (\*ptr\_func[MENU\_SZ])(void) = {disp\_menu, select\_id, enroll\_id, identify\_id, identify\_id, delete\_id};

/////////////////////

void err\_beep(void)

{

SPEAKER.period(1.0 / 2000.0);

SPEAKER = 0.5;

wait(0.1);

SPEAKER = 0.0;

}

void good\_beep(void)

{

SPEAKER.period(1.0 / 2000.0);

SPEAKER = 0.8;

wait(0.1);

SPEAKER = 0.0;

}

void red\_on(void)

{

LEDR.period(0.001);

LEDR = 0.8;

LEDG = 1.0;

LEDB = 1.0;

}

void green\_on(void)

{

LEDG.period(0.001);

LEDR = 1.0;

LEDG = 0.8;

LEDB = 1.0;

}

void blue\_on(void)

{

LEDB.period(0.001);

LEDR = 1.0;

LEDG = 1.0;

LEDB = 0.8;

}

void rgb\_off(void)

{

LEDR = 1.0;

LEDG = 1.0;

LEDB = 1.0;

}

///////////////////

int progress(int status, char \*msg)

{

debug.printf("%s", msg);

LCD.cls();

LCD.locate(0, 1);

LCD.printf("%s", msg);

good\_beep();

if (status >= 1 && status <= 100) {

blue\_on();

}

return 0;

}

void disp\_menu(void)

{

LCD.cls();

LCD.locate(0, 1);

// 1234567890123456789012345

LCD.printf("Fingerprint System v2.0 \n");

LCD.locate(0, 10);

LCD.printf("%s\n", menu[current]);

LCD.printf(" << MENU >> \* RUN \n");

finger.CmosLed(0);

rgb\_off();

}

void init(void)

{

debug.format(8, Serial::None, 1);

debug.baud(115200);

debug.printf("Fingerprint reader module \"GT-511C3 / GT-511C31\" test program.\n");

good\_beep();

rgb\_off();

debug.printf("Build: %s %s\n", \_\_DATE\_\_, \_\_TIME\_\_);

debug.printf("Open\n");

}

void disp\_GT\_info(void)

{

int i;

debug.printf("FirmwareVersion = %lx\n", finger.FirmwareVersion);

debug.printf("IsoAreaMaxSize = %ld\n", finger.IsoAreaMaxSize);

debug.printf("DeviceSerialNumber = ");

for (i = 0; i < sizeof(finger.DeviceSerialNumber); i++){

debug.printf("%02X", finger.DeviceSerialNumber[i]);

}

debug.printf("\n");

}

void chk\_Enrolled(int EnrollID)

{

if (finger.CheckEnrolled(EnrollID) == 0) {

debug.printf("EnrollID(%d) is already enrolled.\nDelete!\n", EnrollID);

LCD.cls();

LCD.printf("EnrollID(%d) is already enrolled.\nDelete!\n", EnrollID);

if (finger.DeleteID(EnrollID) == 0){

debug.printf("Delete OK!\n");

LCD.printf("Delete OK!\n");

}

}

}

void select\_id(void)

{

LCD.locate(0, 10);

LCD.printf("ID = %d \n", EnrollID);

LCD.printf(" << MENU >> \* CHANGE\n");

}

void enroll\_id(void)

{

finger.Open();

chk\_Enrolled(EnrollID);

finger.Enroll(EnrollID, progress);

finger.CmosLed(1);

LCD.locate(0, 10);

LCD.printf(" ");

LCD.locate(0, 10);

LCD.printf("%s\n", menu[current]);

LCD.printf(" << MENU >> \* RUN \n");

}

void verify\_id(void)

{

LCD.locate(0, 10);

LCD.printf("%s\n", menu[current]);

// 1234567890123456789012345

LCD.printf(" << MENU >> \* RUN \n");

}

void identify\_id(void)

{

int id;

finger.CmosLed(1);

debug.printf("Press finger for Identify\n");

LCD.locate(0, 20);

LCD.printf("Press finger for Identify\n");

finger.WaitPress(1);

if (finger.Capture(1) != 0) {

debug.printf("Press finger for Identify\n");

LCD.cls();

LCD.locate(0, 20);

// 1234567890123456789012345

LCD.printf("Press finger for Identify\n");

}

id = finger.Identify();

debug.printf("ID = %d\n", id);

debug.printf("Remove finger\n");

if (id == -1) {

err\_beep();

red\_on();

LCD.cls();

LCD.locate(0, 1);

LCD.printf("ID = %d, Not matched!\n", id);

}

else {

good\_beep();

green\_on();

LCD.cls();

LCD.locate(0, 1);

LCD.printf("ID = %d, Matched!\n", id);

}

LCD.locate(0, 10);

LCD.printf("%s\n", menu[current]);

LCD.printf(" << MENU >> \* RUN \n");

finger.WaitPress(0);

finger.CmosLed(0);

}

void delete\_id(void)

{

if (finger.DeleteID(EnrollID) == 0) {

LCD.locate(0, 1);

// 1234567890123456789012345

LCD.printf("%d deleted \n", EnrollID);

}

else {

LCD.locate(0, 1);

LCD.printf("%d is not found \n", EnrollID);

}

wait(0.5);

LCD.locate(0, 10);

LCD.printf("%s\n", menu[current]);

LCD.printf(" << MENU >> \* RUN \n");

}

void joy\_center\_ISR1(void)

{

ptr\_func[current]();

}

void joy\_right\_ISR(void)

{

current++;

if (current > MENU\_SZ - 1)

current = 0;

LCD.locate(0, 10);

LCD.printf(" ");

LCD.locate(0, 10);

LCD.printf("%s\n", menu[current]);

LCD.printf(" << MENU >> \* RUN \n");

debounce.reset();

}

void joy\_left\_ISR(void)

{

current--;

if (current < 0)

current = MENU\_SZ - 1;

LCD.locate(0, 10);

LCD.printf(" ");

LCD.locate(0, 10);

LCD.printf("%s\n", menu[current]);

LCD.printf(" << MENU >> \* RUN \n");

debounce.reset();

}

void joy\_up\_ISR(void)

{

EnrollID++;

if (EnrollID > (MAX\_ID - 1))

EnrollID = 0;

debounce.reset();

}

void joy\_down\_ISR(void)

{

EnrollID--;

if (EnrollID < 0)

EnrollID = MAX\_ID - 1;

debounce.reset();

}

void set\_ISR(void)

{

debounce.start();

JOY\_CENTER.rise(&joy\_center\_ISR1);

JOY\_RIGHT.rise(&joy\_right\_ISR);

JOY\_LEFT.rise(&joy\_left\_ISR);

JOY\_UP.rise(&joy\_up\_ISR);

JOY\_DOWN.rise(&joy\_down\_ISR);

}

int main()

{

int sts = 0;

init();

disp\_menu();

set\_ISR();

sts = finger.Open();

debug.printf("sts = %d\n", sts);

if (sts == 0) disp\_GT\_info();

}

**Menu selection**

Due to limitation of switch, menu selection is implemented with switch interrupt. When joystick is pressed with direction or center, each interrupt service routine is executed. The main action of ISR (up, down, left, and right) is changing value for menu index, and the center button ISR launch the functions for menu respectively.

**Further work**

In this project, few functions were implemented, and some improvement is needed for already implemented functions, such as, Verify and Identify function.

At this time, these functions, “Get Image”, “Get Raw Image”, “Get / Set Template”, “Get / Set Database”, are not implemented. With these function, handling image file function (copy, transfer) from fingerprint module to mbed or external SD memory, or PC file system is one of want-to-be functions.

**Conclusion**

Through this project and course I can learn the trend of small embedded hardware system and its potential capabilities of IoT. COMSIS-DAP, pyOCD, mbed platform, these are things I learned from the class newly.

**Project Pictures**

|  |  |  |  |
| --- | --- | --- | --- |
| 20150528_143656.jpg | 20150528_143716.jpg | 20150528_143731.jpg | 20150528_143752.jpg |
|  |  |  |  |
| 20150528_143844.jpg | 20150528_143913.jpg | 20150528_143944.jpg | 20150528_144017.jpg |
|  |  |  |  |

**References**

1. Fingerprint recognition, <http://en.wikipedia.org/wiki/Fingerprint_recognition>
2. GT-511C1R datasheet, [http://www.adh-tech.com.tw/files/GT-511C1R\_datasheet\_V1%205\_20140312[1].pdf](http://www.adh-tech.com.tw/files/GT-511C1R_datasheet_V1%205_20140312%5b1%5d.pdf)
3. Toshihisa T, GT511C3 library, <https://developer.mbed.org/users/tosihisa/code/GT511C3/>
4. Toshihisa T, GT511C3test wiki, <https://developer.mbed.org/users/tosihisa/code/GT511C3test/wiki/Homepage>