南京林業大學
NANJING FORESTRY UNIVERSITY

Course design for the first semester of the 2022-2023 academic year

Course Name: Embedded Development and Applications

Class Number	20108011, 20108012			
Pro	Computer Science and Technology			
Student number	200860421	200860420	200860426	200860422
Name	Wang Bin	Tao Junfei	Xu Zhen	Wang Xinqiang
title	Smart Dorm based on Blinker - smart door locks and smart toilets			

Contents

Project name: Smart dormitory based on Blinker -- smart door lock and smart toilet

1. Prior Art (background technology)

Figure 1.1 A dormitory door lock is generally composed of a lock core, a pin, a spring, a lock tongue, a key, etc. The lock core is divided into an inner lock core and an outer lock core. The inner lock core is where you insert the key. Copper pins are divided into inner pins and outer pins, cylindrical in shape and of different lengths. The spring is installed in the round hole of the outer lock core to support the marble. The lock tongue is the telescopic part of the lock; The key has "teeth" of different heights for marbles of different lengths.

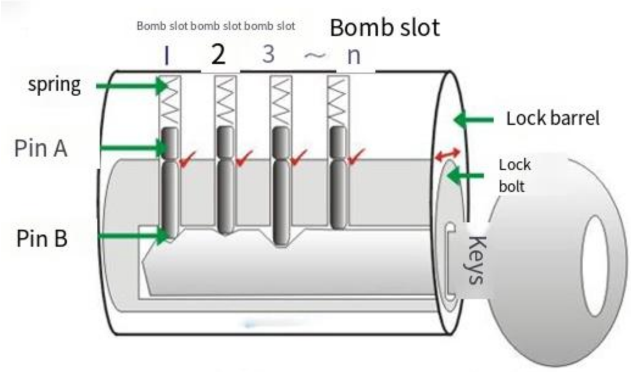


Figure 1.1 Mechanical door lock

Traditional mechanical lock has many disadvantages: 1. Single opening mode, poor safety performance, 2. Mechanical lock core is easy to rust and wear, 3. The key is easy to lose, 4 mechanical lock there is a mutual opening rate, there is no way to avoid, but the level is different. 5 Mechanical lock can not identify, data authentication, networking, damage can not be warned, users can not be the first time to know, nor can effectively monitor the user. It has been unable to meet modern demands.

In our daily dormitory life, we often encounter situations such as forgetting the key and missing the key. This project aims to solve such problems. On the premise of not destroying the lock structure of the dormitory itself, the steering engine and AS608 are used to complete the purpose of fingerprint unlocking, and the Blinker Internet of Things solution is used to solve the problems of remote unlocking and electronic key allocation.

At the same time, as shown in Figure 1.2, due to multiple people sharing a toilet, narrow toilet space and power failure at night, there will be problems such as pit grabbing, cumbersome lighting, poor air flow, etc., this project intends to use the Blinker Internet of Things solution to realize non-inductive lighting, non-inductive ventilation, remote lights on and off, pit viewing and other functions.

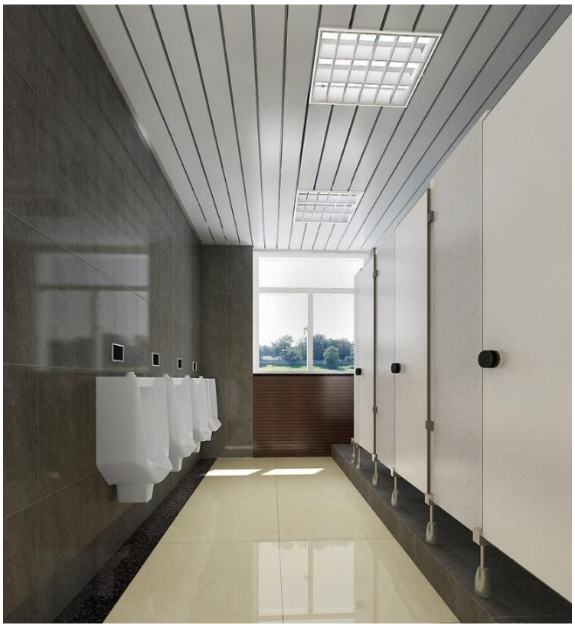


Figure 1.2 Schematic diagram of the toilet

2. Project purpose

As the current dormitory door lock and toilet use simple physical structure, not intelligent, in order to improve the convenience of use, some functions of the door lock and toilet intelligent.

Therefore, the purpose of this project is to enable users to enter the dormitory using fingerprint recognition by inputting their fingerprints in advance, open the dormitory door remotely through mobile phones, check whether the toilet is occupied on mobile phones, and remotely turn on and off the lights.

Through the increase of the above intelligent functions, the dormitory becomes more intelligent and convenient, improve the quality of life.

3. Technical solutions

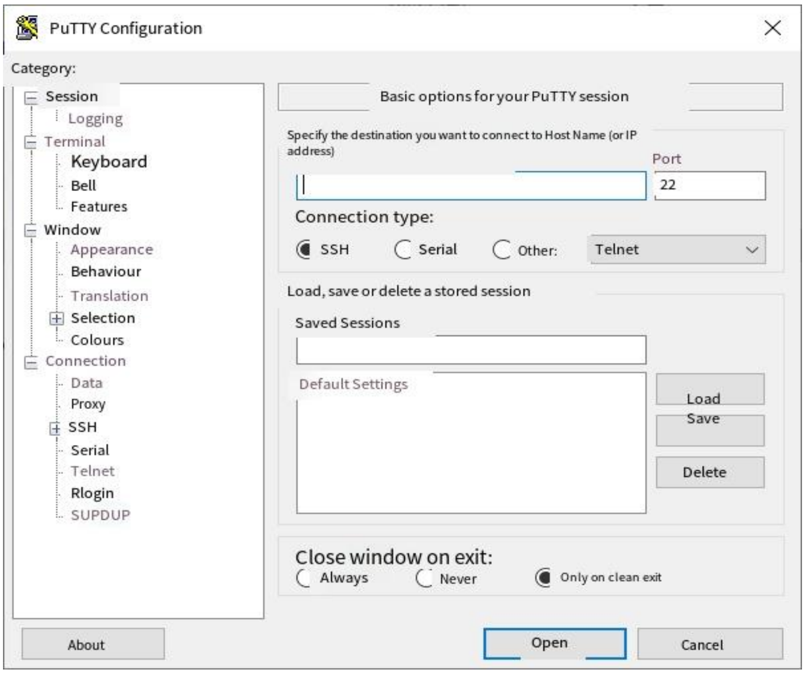
This project plans to utilize Raspberry PI 3B+ and Arduino as processors, and complete fingerprint unlocking with AS608 and steering gear processing. Human body infrared sensor, collision sensor, fan and esp8266 realize the network connection of arduino. In combination with Python's wiringPi, GPIO and other libraries and Blinker Internet of Things scheme, the front-end interface provides users with an interactive interface, realizing fingerprint unlock, remote door opening, non-inductive light opening, non-inductive ventilation, checking pit site occupancy and other functions.

3.1 SSH and VNC solve remote login

3.1.1. The puTTY

PuTTY is a Telnet, SSH, rlogin, pure TCP, and serial interface connection software. PuTTY is an open source software, primarily maintained by Simon Tatham and licensed under the MIT licence. With the popularity of Linux on the server side, Linux system management is increasingly dependent on remote. Among the various remote login tools, Putty is one of the outstanding tools.

After connecting the Raspberry PI to the computer through a network cable and setting the shared Ethernet, run the arp -a command on the cmd interface to query the IP address of the Raspberry PI. Enter the IP address in the following figure to remotely connect to the Raspberry PI over SSH. In the Raspberry PI terminal, you can use the command vncserver to start the service of vnc remote connection and assign the port of vnc remote, as shown in Figure 3.1.1-1.



Basic options for your PuTTY session

Specify the destination you want to connect to Host Name (or IP address)

Port: 22

Connection type:

● SSH ○ Serial ○ Other: Telnet

Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit:

○ Always ○ Never ● Only on clean exit

Figure 3.1.1-1 The PuTTY configuration screen

3.1.2 VNC Viwer

VNC (Virtual Network Console) is short for Virtual Network Console. VNC is a free open source software based on UNIX and Linux operating systems. The remote control capability is powerful, efficient and practical, and its performance can be comparable to any remote control software in Windows and MAC. VNC is basically composed of two parts: one is the client application (vncviewer); The other is the server-side application (vncserver). VNC basically works much like some remote control software for Windows. VNC's server-side applications are adaptable in UNIX and Linux operating systems and have a user-friendly graphical user interface that looks similar to Windows software interfaces. Any Linux machine with the vncviewer installed can easily interconnect with the vncserver installed on the machine.

Figure 3.1.2-1 below shows the start page of VNC Viewer software. Enter the IP address assigned by vncserver in the address bar to connect to PI and operate in the visual interface.

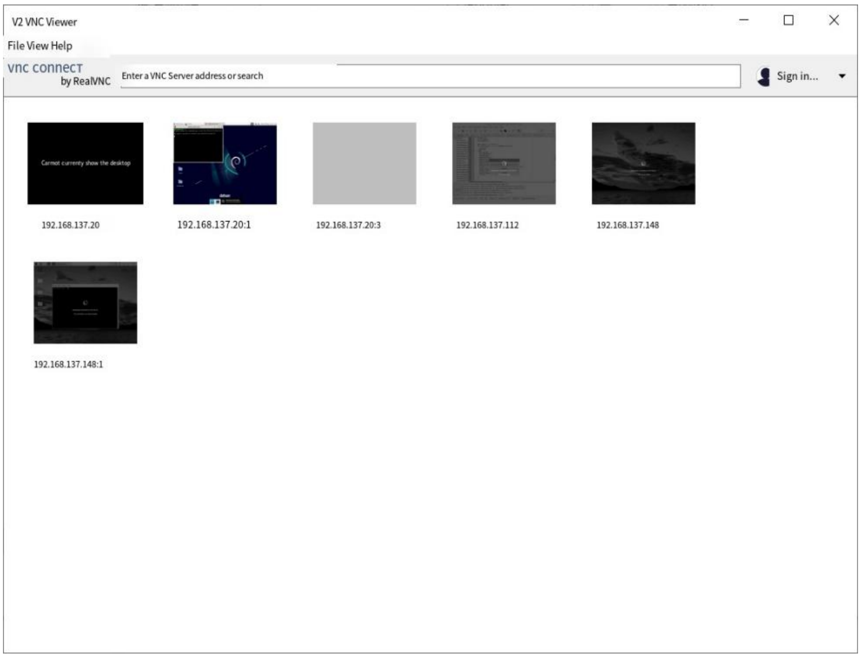
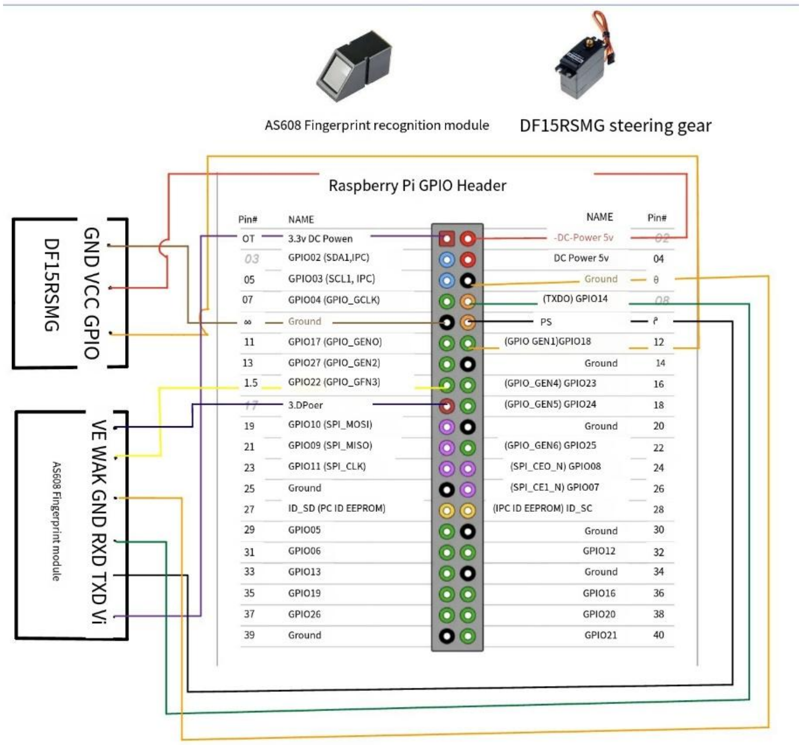


Figure 3.1.2-1VNC operation page

3.2 AS608, steering gear



Raspberry Pi GPIO Header

Pin#	NAME	NAME	Pin#
OT	3.3v DC Powen	-DC-Power 5v	02
03	GPIO02 (SDA1,IPC)	DC Power 5v	04
05	GPIO03 (SCL1, IPC)	Ground	θ
07	GPIO04 (GPIO_GCLK)	(TXDO) GPIO14	08
∞	Ground	PS	ʔ
11	GPIO17 (GPIO_GENO)	(GPIO GEN1)GPIO18	12
13	GPIO27 (GPIO_GEN2)	Ground	14
1.5	GPIO22 (GPIO_GFN3)	(GPIO_GEN4) GPIO23	16
17	3.DPoer	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Ground	20
21	GPIO09 (SPI_MISO)	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	(SPI_CEO_N) GPIO08	24
25	Ground	(SPI_CE1_N) GPIO07	26
27	ID_SD (PC ID EEPROM)	(IPC ID EEPROM) ID_SC	28
29	GPIO05	Ground	30
31	GPIO06	GPIO12	32
33	GPIO13	Ground	34
35	GPIO19	GPIO16	36
37	GPIO26	GPIO20	38
39	Ground	GPIO21	40

FIG3.2-1 AS608 Circuit connection diagram of steering gear and Raspberry PI

ATK-AS608 Fingerprint recognition module is a high-performance optical fingerprint recognition module introduced by ALIENTEK. The ATK-AS608 module uses the AS608 fingerprint recognition chip of Hangzhou Shengyuan Chip Technology Co., LTD. (Synochip), a famous domestic fingerprint recognition chip company. The chip has a built-in DSP arithmetic unit and integrates the fingerprint recognition algorithm, which can efficiently and quickly collect images and identify fingerprint features. Module is equipped with serial port, USB communication interface, users do not need to study complex image processing and fingerprint recognition algorithm, just through a simple serial port, USB in accordance with the communication protocol can control the module. This module can be applied to all kinds of attendance machine, safe cabinet, fingerprint access control system, fingerprint lock and other occasions. The technical indicators are as follows:

Table 3-1 Technical specifications

project	Instructions
Operating voltage (V)	3.0 to 3.6V, typical value: 3.3V
Operating current (mA)	30 to 60mA, typical value: 40mA
USART Communications	Baud rate (9600×N), N=1 to 12. Default N=6,bps= 57600 (data bit :8 Stop bit :1 Check bit :none TTL level)
USB communication	2.0FS (2.0 Full speed)
Sensor image Size (pixel)	256*288pixel
Image processing time (S)	< 0.4 (S)
Power-on delay (S)	<0.1(S), it takes about 0.1S to initialize the module after it is powered on
Search time (S)	< 0.3 (S)
Rejection rate (FRR)	< 1%
Fingerprint storage capacity	300 prints (ID:0~299)

Control AS608 by transmitting packets

To realize the control of the AS608 fingerprint module, it is necessary to send and process serial port data according to the communication protocol format stipulated by the AS608 chip, and the realization of the functions of the whole module is also carried out around the protocol format.

Table 3-2 Format of the command package

Number of bytes	2bytes	4bytes	1 bytes	2 bytes	1 bytes				2 bytes
name	Baotou	Chip address	Package identification	Package length	Instructions	Parameter 1	...	Parameter n	checksum
Content	0xEF01	XXXX	01	N=					

Table 3-3 Format of data packets

Byte number	2bytes	4bytes	1 bytes	2 bytes	N bytes	2 bytes
Name	Baotou	Chip address	Package	Package	Data	Checksu

			identification	length		m
Content	0xEF01	XXXX	02			

Table 3-4 End Pack formats

Number of bytes	2bytes	4bytes	1 bytes	2 bytes	N bytes	2 bytes
Name	Baotou	Chip address	Package identification	Package length	Data	checksum
Content	0xEF01	XXXX	08			

We can use the corresponding instruction in AS608 by sending the corresponding packet to AS608, and receive and process the reply. According to the above principle, we can add user function (add), delete user function (delete), compare function (search) and show the current user function (list).

3.2.2 Use the steering gear to open the door and twist the handle

Steering gear model -- DF15RSMG

As shown in Figure 3.2.2-1, the DF15RSMG is different from the market. The steering gear for model aircraft vehicle is specially designed to solve some problems in the application of robot steering gear. It is very suitable for players and professionals to build multi-degree of freedom mechanical arms and multi-legged robots, which require high accuracy of steering gear. DF15RSMG steering gear can greatly improve the bearing capacity of the robot arm and walking platform, while ensuring the accuracy and stability of the operation.

The steering gear has two sets of standard enclosures, both of which adopt thickened design and strong structure. One has four standard M4 mounting holes, while the other housing has a dual bearing system specifically designed for the arm for increased wear and load bearing. The gear is a durable copper-steel mixed gear structure, rather than the plastic gears available on the market. The PWM drive frequency can be up to 4KMHz, and the jitter after load is small, without any noise. With pulse locking function, an action only needs to send one PWM pulse to lock. Reduce processor loss. The following is the description of some specific parameters of the steering gear.

● Mechanical characteristics:
Size: 40mmx40mmx20mm
Quality: 60g
360° continuous rotation, direction controlled through PWM signal
No-load speed: 60%/0.16s (7.2v)
Lock-in torque: 15kg.cm (7.2v)

Gear: steel, copper mixed gear
Protection: 2 special copper sleeves
● Electrical characteristics:
Power supply voltage: 5v-7.2v
Stoppage current: 2.5A (7.2v)
PWM signal voltage: 3v-5v
Positive pulse width range of PWM signal: 500us-2500us
Middle position of PWM signal :1500us
● Wiring instructions:
Brown wire -GND
Red -VCC
Orange -PWM signal input diagram

3.3.2-1 Physical picture of steering gear

Analog control of steering gear

DF15RSMG is a 360-degree analog steering gear. The frequency of the steering gear is generally 50HZ, which is a time base pulse of about 20ms, and the high level part of the pulse is generally in the range of 0.5ms-2.5ms. To control the different speed or Angle of the steering gear.

Use the hardware PWM interface

The default PWM mode of the Raspberry PI is Balanced mode. The PWM frequency in this mode is much higher than the 50Hz of the steering engine, so the frequency of the PWM needs to be changed. At this time, we need to switch the PWM mode to Mark:Space mode (duty cycle mode). According to the data, the basic frequency of PWM clock of the Raspberry PI is 19.2MHz. The frequency division is set by pwmSetClock() function (the default is 32 frequency division). The calculation formula of PWM frequency is as follows:

pwmFrequency(Hz) = 19.2*10^6 Hz/pwmClock/pwmRange

In order to get better control accuracy, range is set to 2000 here, that is, the cycle is divided into 2000 steps and the control accuracy is higher. To obtain the PWM frequency of 50Hz, then divisor is:

divsior = 19200*1000 / (50 * 2000) = 192

360 degrees steering gear can rotate 360 degrees, so there is quite a big difference with 180 degrees steering gear, first 360 degrees steering gear can not control the rotation Angle, the general steering gear is given a specific PWM, steering gear will turn to the corresponding

Angle, and 360 degrees steering gear is only able to control the direction and rotation speed, so 360 degrees steering gear given a PWM signal, It will rotate at a specific speed and direction.

So there are:

0.5ms---------------- forward maximum RPM;
1.5ms---------------- speed is 0;
2.5ms---------------- reverse maximum speed;

3.3 Sensor, esp8266

3.3.1 Human infrared sensor

As shown in Figure 3.3.1-1, the pyroelectric infrared motion sensor can detect the infrared emitted by moving human beings or animals and output switch signals. It can be applied to various occasions where it is necessary to detect moving human beings. The traditional pyroelectric infrared sensor needs human pyroelectric infrared probe, special chip and complex peripheral circuit to realize, the volume is slightly larger, the circuit is complex, and the reliability is slightly lower. Our newly launched infrared pyroelectric motion sensor, specially designed for Arduino, uses digital integration to integrate human pyroelectric infrared probe, featuring small volume, high reliability, low power consumption, simple peripheral circuit and so on.

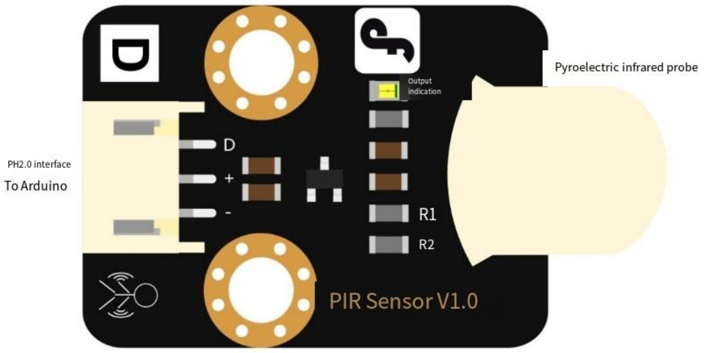


Figure 3.3.1-1 Human body infrared sensor

This pyroelectric infrared motion sensor adopts the digital integrated human pyroelectric infrared probe AM412. AM412 is a pyroelectric infrared probe that integrates digital intelligent control circuit and human body detection sensor in the electromagnetic shield. The body detection sensor couples the sensed human movement signal to a digital intelligent integrated circuit chip through a very high impedance differential input circuit, which converts the signal into a 15-bit ADC digital signal. When the PIR signal exceeds the selected digital threshold, there is a delayed high level output. All signal processing is done on a single chip.

As shown in Figure 3.3.1-2, when the pyroelectric infrared signal received by the probe exceeds the trigger threshold inside the probe, a counting pulse will be generated inside the probe. When the probe receives such signal again, it will think that it has received a second pulse. Once two pulses are received within 4 seconds, the probe will generate an alarm signal, and the output pin will output a high level. In addition, as long as the received signal amplitude is more than 5 times the trigger threshold, then only one pulse is needed to trigger the high level output at the output. An example of the trigger logic diagram is shown below. For multiple triggers, the holding time of the output high level is timed from the last active pulse.

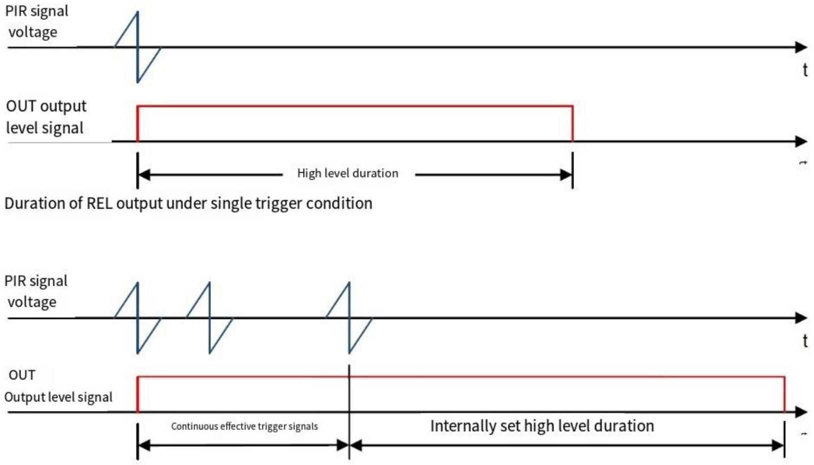
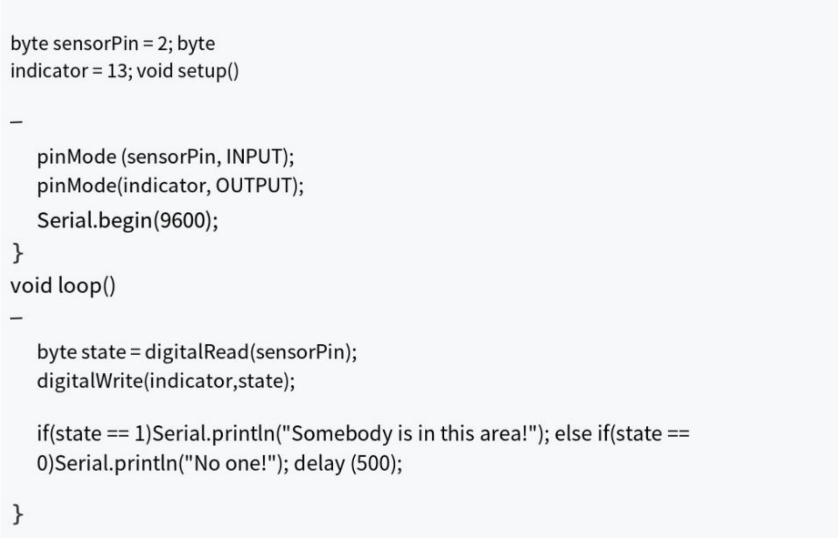

PIR signal voltage
t
OUT output level signal
High level duration
Duration of REL output under single trigger condition
PIR signal voltage
t
OUT
Output level signal
Continuous effective trigger signals
Internally set high level duration

Figure 3.3.1-2 Trigger mode of infrared sensor

In Figure 3.2.1-3, the wiring diagram of human infrared sensor and arduino uno and the code sample are shown



```
byte sensorPin = 2; byte
indicator = 13; void setup()

–
    pinMode (sensorPin, INPUT);
    pinMode(indicator, OUTPUT);
    Serial.begin(9600);
}
void loop()
–
    byte state = digitalRead(sensorPin);
    digitalWrite(indicator,state);

    if(state == 1)Serial.println("Somebody is in this area!"); else if(state ==
    0)Serial.println("No one!"); delay (500);

}
```

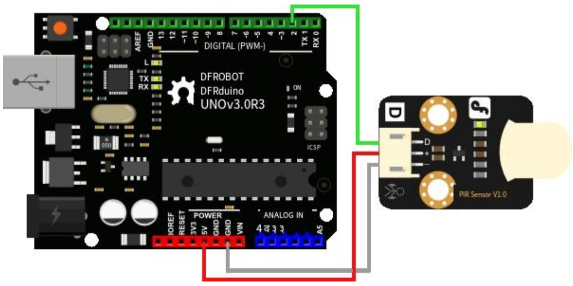


Figure 3.2.1-3 shows the wiring of the infrared sensor

3.3.2 Collision Sensor

A miniature quick-acting switch, also known as a switch and often referred to as a microswitch, is an electric switch driven by a very small physical force. Microswitches are used in a wide range of applications; Their applications include appliances, machinery, industrial controls, vehicles, and many other places where they are used to control circuits. They are usually rated to carry current only in the control circuit, although certain switches can be used directly to control small motors, solenoids, lights, or other devices.

Figure 3.3.2-1 This is a small microswitch sensor designed for use with Arduino microcontrollers. It can be connected directly to the IO expansion board. It integrates a pull-up resistor and an onboard status indicator LED.

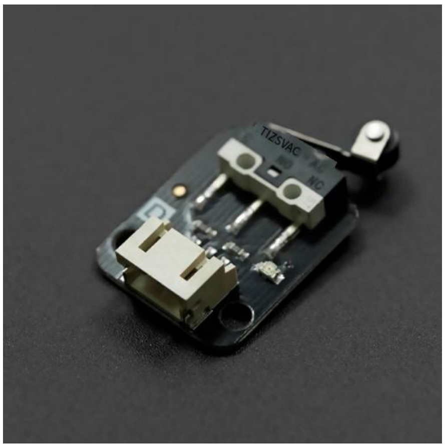


Figure 3.3.2-1 Physical image of the collision sensor

In the project, we used collision sensor to simulate the action of door being opened and closed. When there is no collision, it is low. When there is a collision (being pressed down), it is high. Its wiring diagram to the arduino is shown in Figure 3.3.2-2.

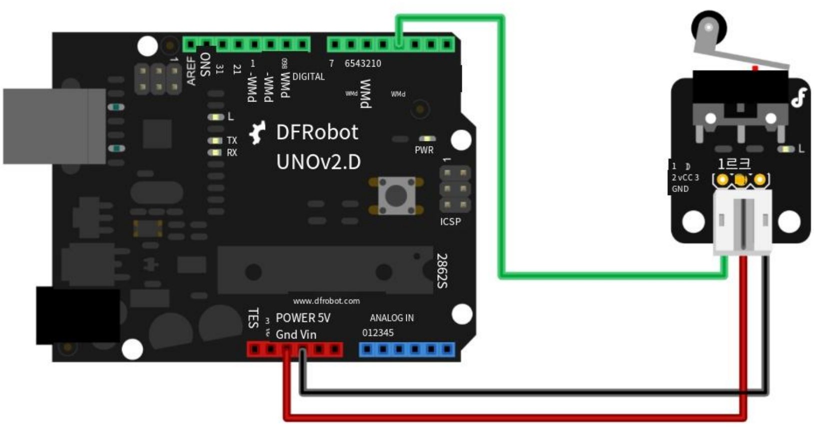
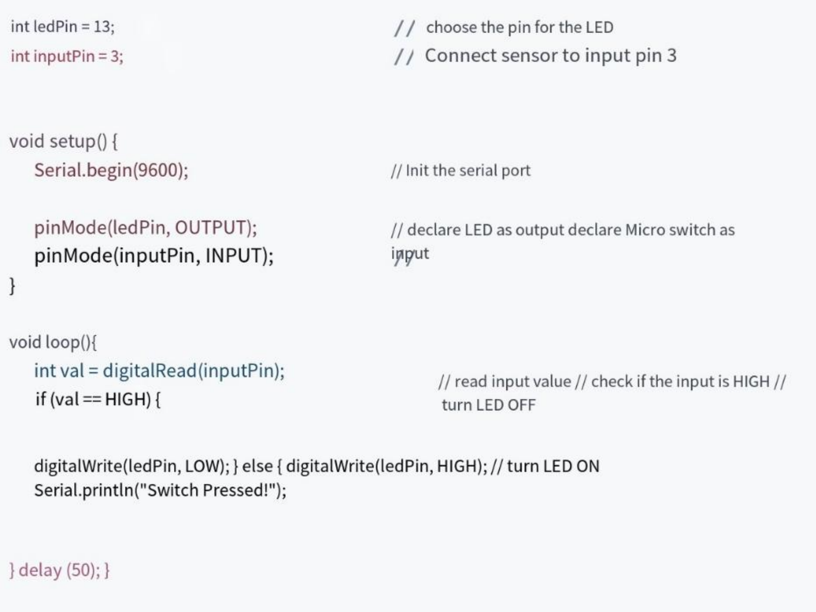


Figure 3.3.2-2 Connection diagram of the collision sensor and Arduino

Sample code:



```
int ledPin = 13;                          //  choose the pin for the LED
int inputPin = 3;                         //  Connect sensor to input pin 3


void setup() {
    Serial.begin(9600);                   // Init the serial port

    pinMode(ledPin, OUTPUT);              // declare LED as output declare Micro switch as
    pinMode(inputPin, INPUT);             input
}

void loop(){
    int val = digitalRead(inputPin);
    if (val == HIGH) {                        // read input value // check if the input is HIGH //
                                              turn LED OFF

    digitalWrite(ledPin, LOW); } else { digitalWrite(ledPin, HIGH); // turn LED ON
    Serial.println("Switch Pressed!");


} delay (50); }
```

3.3.3 esp8266WIFI module

ESP8266 is a complete and self-contained WiFi network solution that can run independently or as a slave on other hosts. The ESP8266 module is shown in Figure 3.3.3-1.

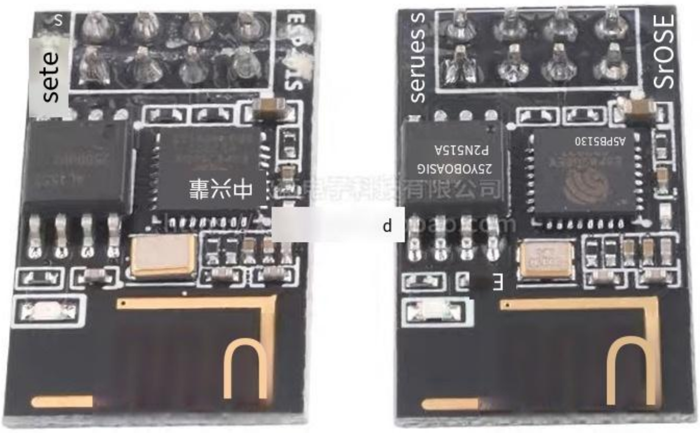


Figure 3.3.3-1 ESP8266WIFI module

The ESP8266 can boot directly from the external flash memory when it is loaded with applications and is the only application processor in the device. The built-in cache helps improve system performance and reduces memory requirements.
Alternatively, when wireless access takes on the task of a WiFi adapter, it can be added to any microcontroller-based design with a simple connection via the SPI/SDIO interface or the CPU

AHB bridge interface.

The ESP8266's powerful on-chip processing and storage capabilities enable it to integrate sensors and other application-specific devices via the GPIO port, achieving minimal up-front development and minimal system resource footprint during operation.

The ESP8266 supports softAP mode, station mode, and softAP + station coexistence mode. ESP8266 can realize very flexible networking mode and network topology. SoftAP: A wireless access point is the central node of a wireless network. A commonly used wireless router is a wireless access point. Station: A wireless terminal is the terminal of a wireless network. In this project, esp8266 is used as a wireless terminal.

As a station, ESP8266 is connected to the internet through a router (AP) and can upload and download data to the cloud server. Users can use mobile terminals (mobile phones, laptops, etc.) at any time to monitor the status of ESP8266 module through the cloud and send control commands to ESP8266 module, as shown in Figure 3.3.3-2.

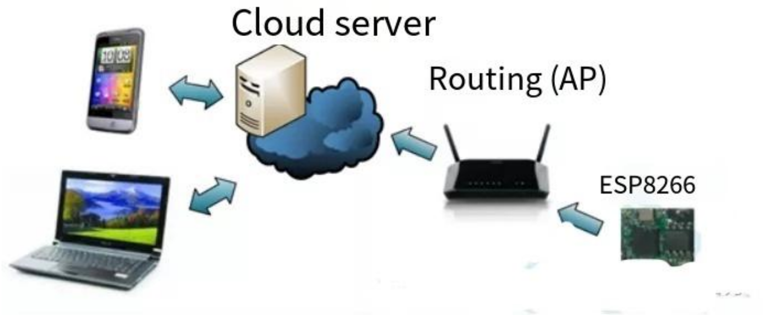
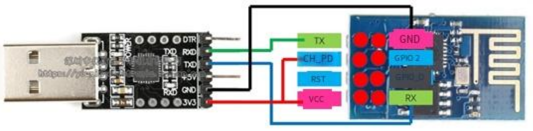


Figure 3.3.3-2 ESP8266 as a wireless terminal

ESP8266 has two burning modes. Users need to connect different circuits according to their needs:

（I）Burn through AT instruction. In this mode, ESP8266 needs to be connected with USB to TTL and burned through the computer's burning software. The specific connection diagram is shown in Figure 3.3.3-3:



https://blog.csdn.net/sunshineQY

Figure 3.3.3-3 Burning wiring diagram of esp8266

（II）Code burning through Arduino. In this mode, it is necessary to connect ESP8266 to Arduino, and burn Arduino board by writing code through Arduino IDE. The specific connection is shown in Figure 3.3.3-4:

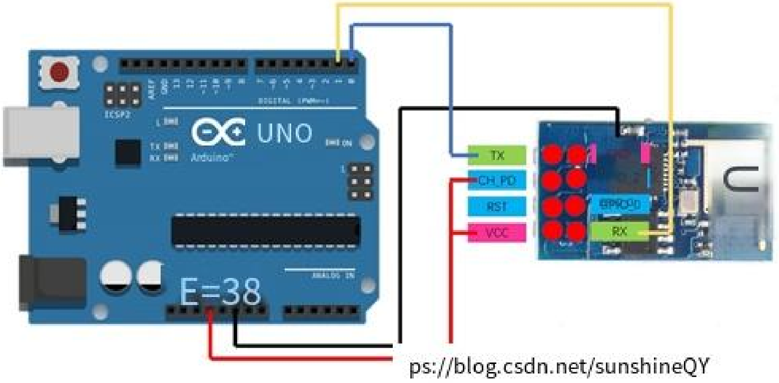
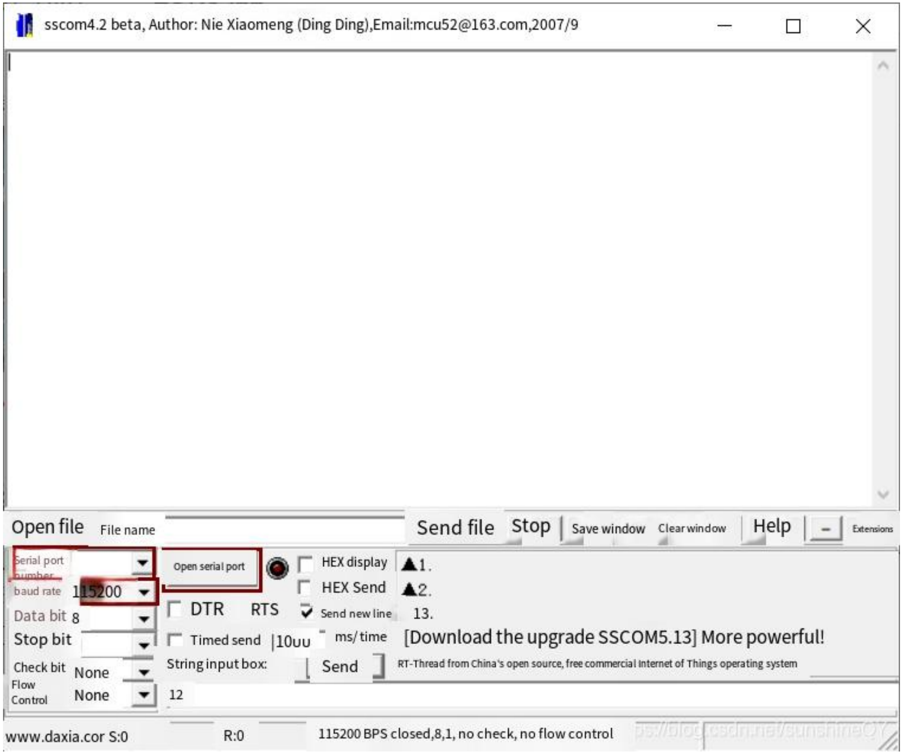


Figure 3.3.3-4 Connection diagram of ESP8266 and Arduino

2. Burn ESP8266 directly

Connect the circuit according to Figure 1, insert the USB port into the computer terminal, check the port number in my computer management, the first use may not show the problem can be installed CH340 driver, after identifying the port number to remember, it will be used when burning.

Open SSCOM serial port debugging tool or other serial port debugging tools, according to the serial port number you see in the management to select your own serial port, open the serial port as shown in Figure 3.3.3-5.



sscom4.2 beta, Author: Nie Xiaomeng (Ding Ding),Email:mcu52@163.com,2007/9

Serial port number

baud rate 115200

Data bit 8

Stop bit

Check bit None

Flow Control None

□ HEX display

□ HEX Send

□ DTR RTS

☑ Send new line

□ Timed send |10uu ms/ time

String input box:

12

[Download the upgrade SSCOM5.13] More powerful!

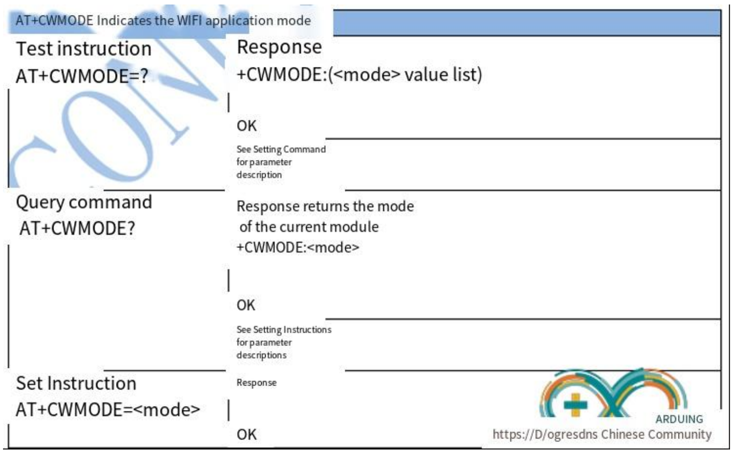
RT-Thread from China's open source, free commercial Internet of Things operating system

www.daxia.cor S:0 R:0 115200 BPS closed,8,1, no check, no flow control

Figure 3.3.3-5 SSCOM debugging software

Send AT instruction. AT instruction set is shown in Figure 3.3.3-6 (only WIFI application mode selection is displayed) :

1. AT+CWMODE Select the wifi application mode



AT+CWMODE Indicates the WIFI application mode		
Test instruction AT+CWMODE=?	Response +CWMODE:(<mode> value list) OK	See Setting Command for parameter description
Query command AT+CWMODE?	Response returns the mode of the current module +CWMODE:<mode> OK	See Setting Instructions for parameter descriptions
Set Instruction AT+CWMODE=<mode>	Response OK	

Figure 3.3.3-6 Command operation

STA mode (1) : Station, similar to wireless network terminal, receives wireless access.

AP mode (2) : ACCESS POINT provides wireless network services and allows other wireless devices to access.

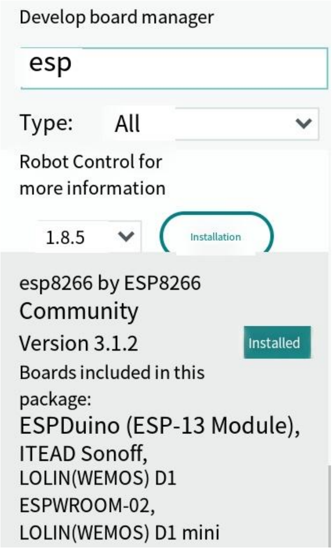
AP+STA mode (3) : It can not only provide wireless network terminals to receive wireless access, but also provide wireless network services to allow other wireless devices to access.

3.4 Configuring the Blinker Environment

First Raspblian needs to download the blinker library using git and configure the environment as follows

```
git clone https://github.com/blinker-iot/blinker-py

cd blinker-py

sudo python3 setup.py install

sudo pip3 install -r requirements.txt
```

Arduino needs to download the ESP8266WIFI module library in the IDE as shown in Figure 3.4.1. In addition, you need to burn the code connecting to wifi and communicating with Arduino through the serial port to esp8266 through CH340.



Develop board manager

esp

Type: All

Robot Control for more information

1.8.5

esp8266 by ESP8266 Community

Version 3.1.2

Boards included in this package:

ESPDuino (ESP-13 Module),
ITEAD Sonoff,
LOLIN(WEMOS) D1
ESPWROOM-02,
LOLIN(WEMOS) D1 mini

Figure 3.4.1

4 Specific implementation

The overall flow chart of this project is shown in Figure 4.1-1

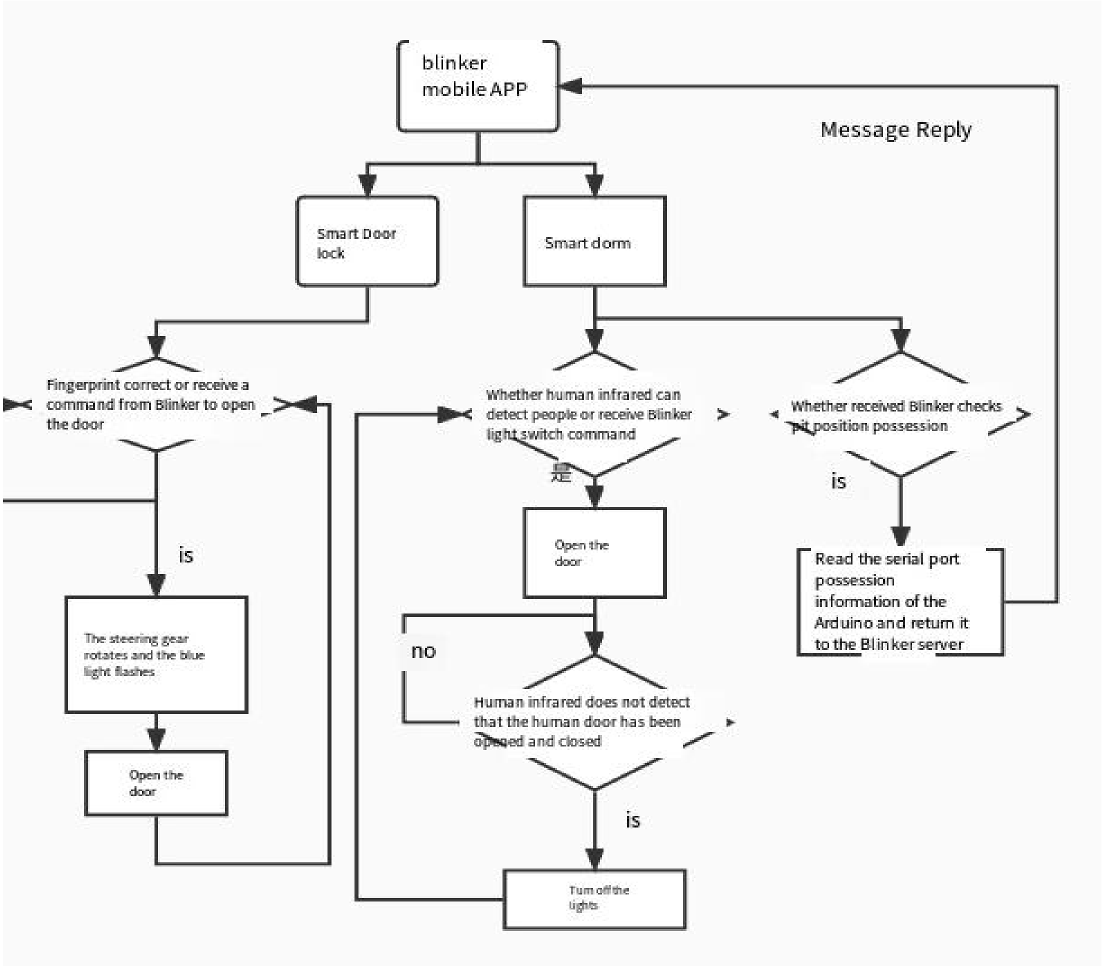

blinker mobile APP
Message Reply
Smart Door lock
Smart dorm
Fingerprint correct or receive a command from Blinker to open the door
Whether human infrared can detect people or receive Blinker light switch command
Whether received Blinker checks pit position possession
是
is
Open the door
Read the serial port possession information of the Arduino and return it to the Blinker server
is
The steering gear rotates and the blue light flashes
no
Human infrared does not detect that the human door has been opened and closed
Open the door
is
Turn off the lights

Figure 4.1-1 Project flow chart

4.1 Physical operation of steering gear and fingerprint module

As shown in Figure 4.1-1, the flow chart of the intelligent door lock is as follows:

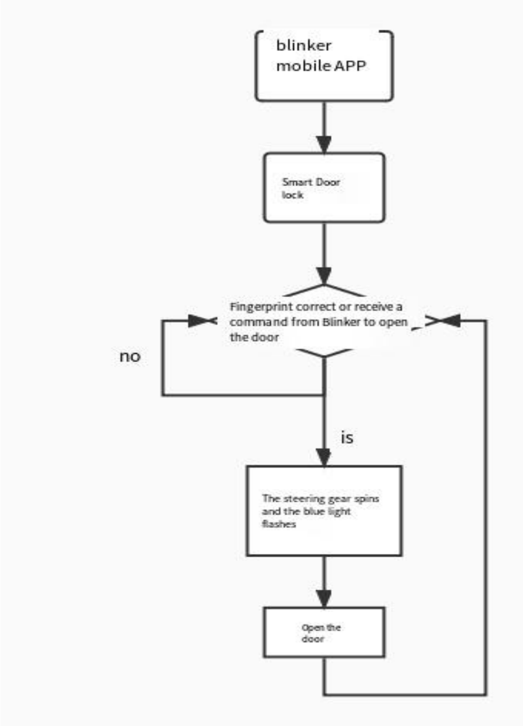

blinker mobile APP
Smart Door lock
Fingerprint correct or receive a command from Blinker to open the door
no
is
The steering gear spins and the blue light flashes
Open the door

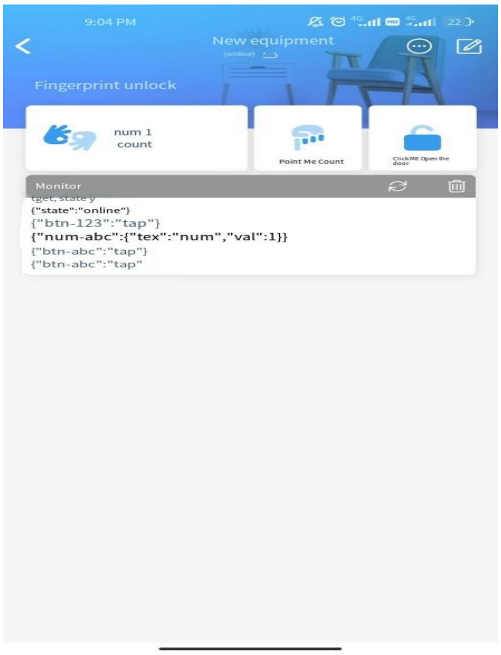
Figure 4.1-2 Intelligent door lock flow chart

There are two kinds of door opening:

1、Fingerprint module comparison is correct

2、Raspberry PI received an open door request from the Blinker server

At the same time, a remote door opening button is added to the phone operator. When a data request from the phone is sent to PI via the Blinker server, PI calls the main program.



```
tget, state y
{"state":"online"}
{"btn-123":"tap"}
{"num-abc":{"tex":"num","val":1}}
{"btn-abc":"tap"}
{"btn-abc":"tap"
```

Figure 4.1-3 Door opening request sent to Raspberry PI

4.1.1 Implementation of the AS608 Identification fingerprint

By inquiring the user manual, we found out several major instructions and recorded their instruction package format and reply package format, such as: input image (PS_GetImage) instruction package format and reply package format as shown in Table 4-1, Table 4-2:

Table 4-1 PS_GetImage instruction pack format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
baotou	Chip address	Package identification	Package length	Instruction code	Checksum
0xEF01	xxxx	01H	03H	01H	05H

Table 4-2 Format of PS_GetImage Reply packet:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Baotou	Chip address	Package identification	Package length	Confirmation code	Checksum
0xEF01	xxxx	07H	03H	xxH	sum

Then when we call the corresponding function in the terminal, Raspberry PI sends the corresponding format data packet to the fingerprint module of AS608 to call the instruction in AS608, receives the reply packet, and determines the instruction result according to the reply packet.

4.1.2 Realization of steering gear control

How to control the rotation of 360° steering gear:

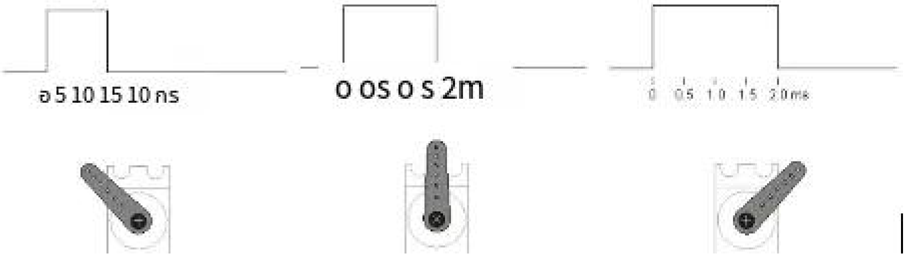


Figure 4.1-4 Rotation angles of steering gear with different PWM waves

1. The control signal is a pulse width modulation (PWM) signal, which can be easily generated by the microcontroller.

2. The pulse stays at a high level for 1 to 2 milliseconds (ms), or 1000 to 2000 microseconds (μs). For a 180 degree steering gear, at 1000μs, the steering gear is left full rudder. At 2000μs, full right rudder. However, a larger or smaller range of motion can be achieved by adjusting the pulse width. The control mode of 360 degree steering gear is different from that of 180

degree steering gear in that 360 steering gear can only control direction and speed. As shown in the figure above, Angle can be interpreted as direction. For example, in 20msPWM square wave, the maximum speed at 0.5ms is left turn (reverse), the maximum speed at 1.5ms is stop rotation, and the maximum speed at 2.5ms is right turn (positive turn).

3. The low level of the control pulse lasts for 20ms. Every 20 milliseconds (50 times per second), it is necessary to jump to the high level again, otherwise the steering gear may strike and it will be difficult to maintain stability.

4. The default PWM mode of Raspberry PI is Balanced mode, whose PWM frequency is much higher than 50Hz of the steering engine, so the PWM frequency needs to be modified. At this time, we need to switch the PWM mode to Mark:Space mode (duty cycle mode). pwmSetClock() function is used to set the frequency division. The specific setting code is as follows:

```
pinMode(PWM_PIN, PWM_OUTPUT); // Set PWM output
pwmSetMode(PWM_MODE_MS); // Set traditional mode
pwmSetClock(192); // Set the frequency division
pwmSetRange(2000); // The setup cycle is divided into 2000 steps
```

The code below controls the speed of the steering gear

C language:

```
pwmWrite(PWM_PIN, 200);
delay(900);
pwmWrite(PWM_PIN, 150);
delay(3000);
```

Python:

```
wiringpi.pwmWrite(PWM_PIN, 200)
wiringpi.delay(800)
wiringpi.pwmWrite(PWM_PIN, 150)
wiringpi.delay(3000)
```

4.2 Construction of smart toilet model

As shown in Figure 4.2-1, the flow chart of the smart toilet is as follows:

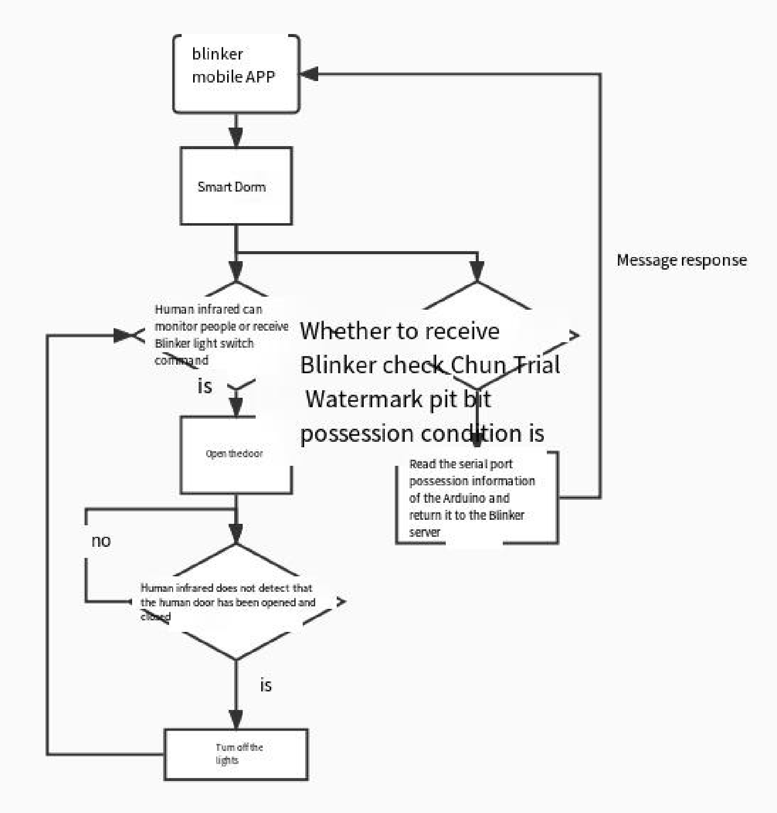

blinker mobile APP
Smart Dorm
Message response
Human infrared can monitor people or receive Blinker light switch command
Whether to receive Blinker check Chun Trial Watermark pit bit possession condition is
is
Open the door
Read the serial port possession information of the Arduino and return it to the Blinker server
no
Human infrared does not detect that the human door has been opened and closed
is
Turn off the lights

Figure 4.2-1 Smart toilet flow chart

There are two types of lights on:

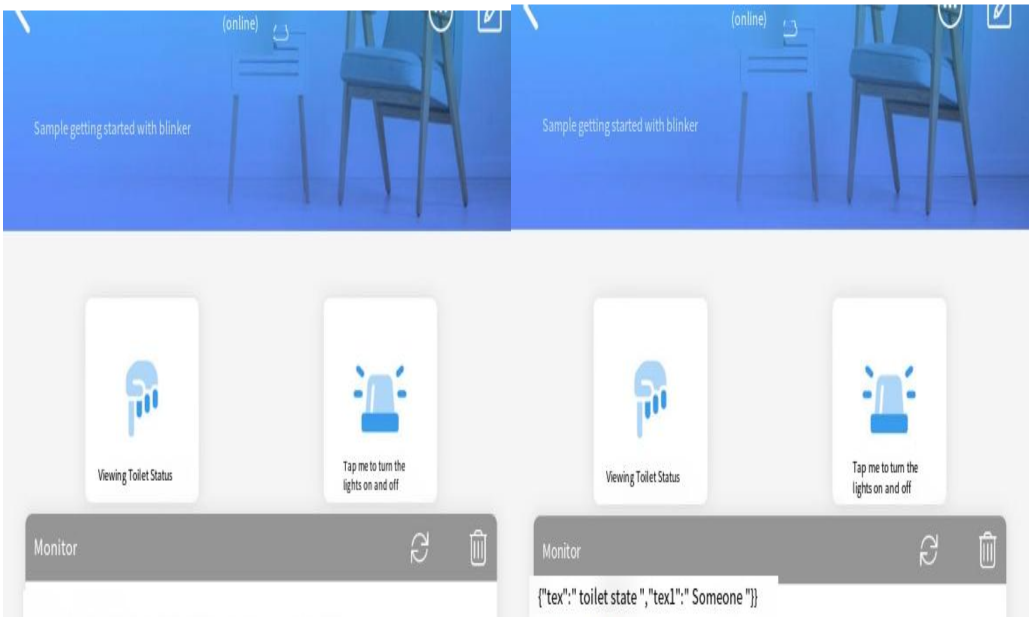
1 Infrared sensors detect people

2 Esp8266 received a light switch request from the Blinker server

There is only one way to turn off the lights:

1 The toilet detects that no one is present and the door has been opened and closed -- the collision sensor has passed the collision.

At the same time, a button to check toilet occupancy was added to the mobile phone operator. Through TX and RX serial ports, the Arduino transmits the infrared detection results of human body to esp8266 in real time. When the data request of the mobile phone is transmitted to esp8266 through the Blinker server, esp8266 replies to the server in json format, so that the data is transmitted back to the mobile end. See Figure 4.2-2.


(online)
Sample getting started with blinker
Viewing Toilet Status
Tap me to turn the lights on and off
Monitor
(online)
Sample getting started with blinker
Viewing Toilet Status
Tap me to turn the lights on and off
Monitor

```
{"tex":" toilet state ","tex1":" Someone "}
```

Figure 4.2-2 Json format data sent back to the mobile end

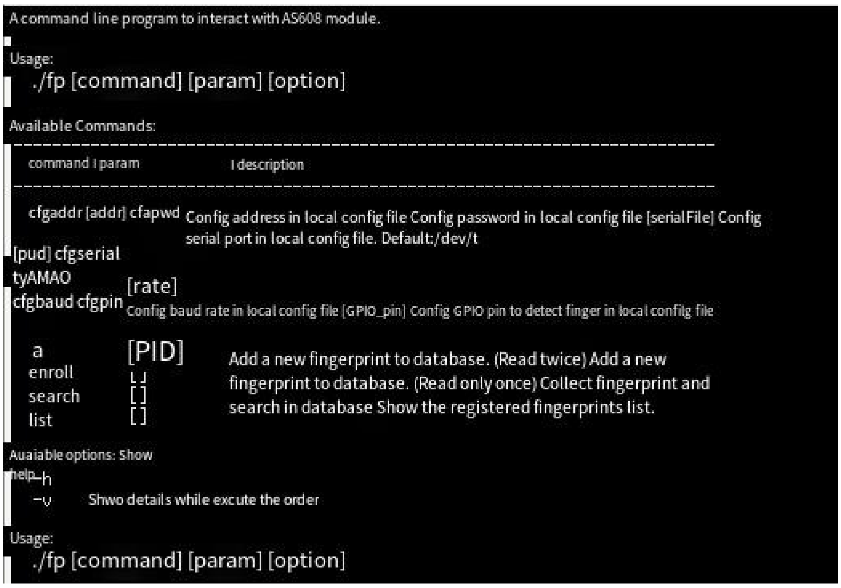
5. Test cases and running results

5.1 Intelligent door lock test

5.1.1 Fingerprint opens the door lock

(1) Main interface of fingerprint module

As shown in Figure 5.1.1-1, after running the main program of the intelligent door lock, the user interface will be displayed. Different operations can be realized by calling commands, such as adding fingerprint (add), deleting fingerprint (delete), checking the number of fingerprint in the fingerprint database (list), comparing fingerprint (search) and judging whether the fingerprint is correct to decide whether to rotate the steering engine and light the blue light.

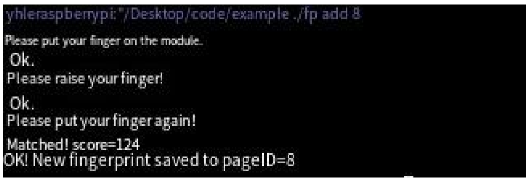


```
A command line program to interact with AS608 module.

Usage:
    ./fp [command] [param] [option]

Available Commands:
 ------------------------------------------------------------------------
   command I param              I description
 ------------------------------------------------------------------------
   cfgaddr [addr] cfapwd  Config address in local config file Config password in local config file [serialFile] Config
                          serial port in local config file. Default:/dev/t
[pud] cfgserial
tyAMAO               [rate]
cfgbaud cfgpin       Config baud rate in local config file [GPIO_pin] Config GPIO pin to detect finger in local confilg file

   a          [PID]     Add a new fingerprint to database. (Read twice) Add a new
   enroll     []        fingerprint to database. (Read only once) Collect fingerprint and
   search     []        search in database Show the registered fingerprints list.
   list       []
Auaiable options: Show
help -h
   -v      Shwo details while excute the order

Usage:
    ./fp [command] [param] [option]
```

Figure 5.1.1-1 Main interface of fingerprint module

(2) Add a fingerprint

Call in./fp add [fingerNum] format, as shown in Figure 5.1.1-2, collect fingerprint information for two consecutive times and input the fingerprint.

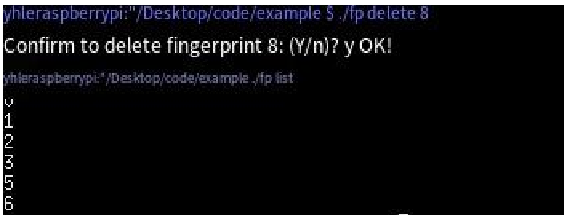


```
yhleraspberrypi:"/Desktop/code/example ./fp add 8
Please put your finger on the module.
 Ok.
Please raise your finger!
 Ok.
Please put your finger again!
Matched! score=124
OK! New fingerprint saved to pageID=8
```

Figure 5.1.1-2 Add fingerprint

(3) Remove the fingerprint

Call in./fp delete [fingerNum] format, as shown in Figure 5.1.1-3, to delete the fingerprint labeled fingerNum.

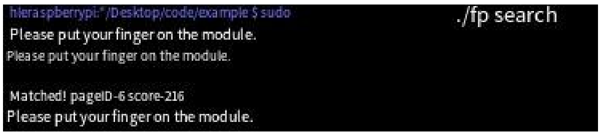


```
yhleraspberrypi:"/Desktop/code/example $ ./fp delete 8
Confirm to delete fingerprint 8: (Y/n)? y OK!
yhleraspberrypi:"/Desktop/code/example ./fp list
0
1
2
3
5
6
```

Figure 5.1.1-3 Deleting a fingerprint

(4) Verify the fingerprint

Call in./fp search format, as shown in Figure 5.1.1-4, and loop all the way through. Wait for two seconds, if the fingerprint is detected, compare it with the fingerprint in the fingerprint database, if the fingerprint feature conforms to higher than the standard, it is judged that the fingerprint is correct, the blue light indicates that the identification is successful, and rotate the steering engine to pull the door handle to open the door. If no fingerprint is detected, the cycle will continue.



```
hleraspberrypi:"/Desktop/code/example $ sudo                    ./fp search
 Please put your finger on the module.
Please put your finger on the module.

 Matched! pageID-6 score-216
Please put your finger on the module.
```

Figure 5.1.1-4 Verifying a fingerprint

Figure 5.1.1-5 below shows the initial state of the door lock:

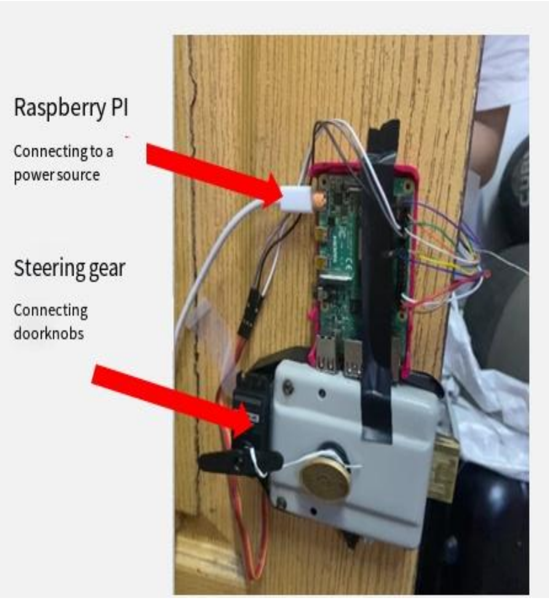


Figure 5.1.1-5 Fingerprint unlock

After the correct fingerprint is pressed, the blue light blinks to indicate successful unlocking. Then the steering engine pulls the door lock, as shown in Figure 5.1.1-6 below

:

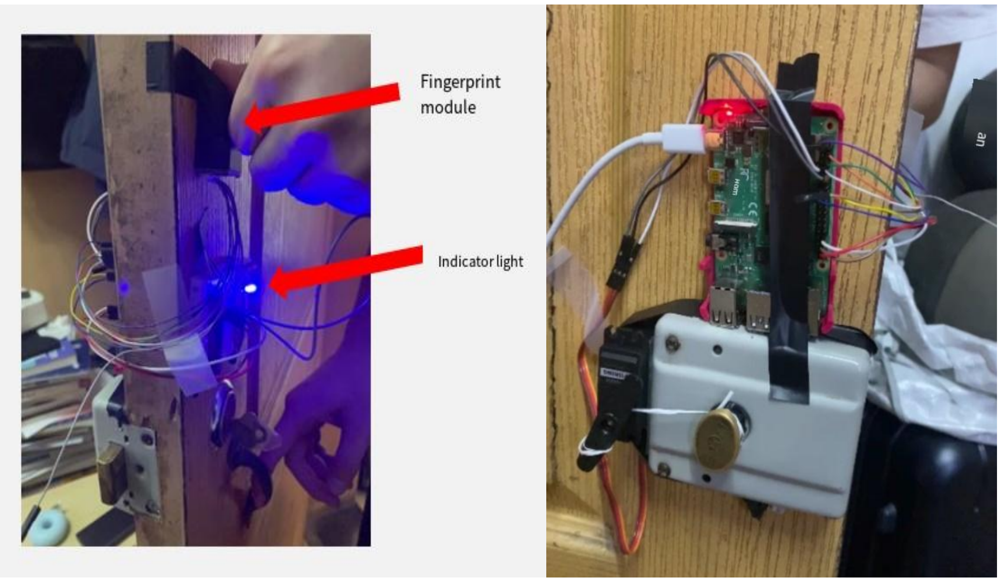


Figure 5.1.1-6 Fingerprint correct unlocking diagram

5.1.2 Remotely unlocking the lock on your phone

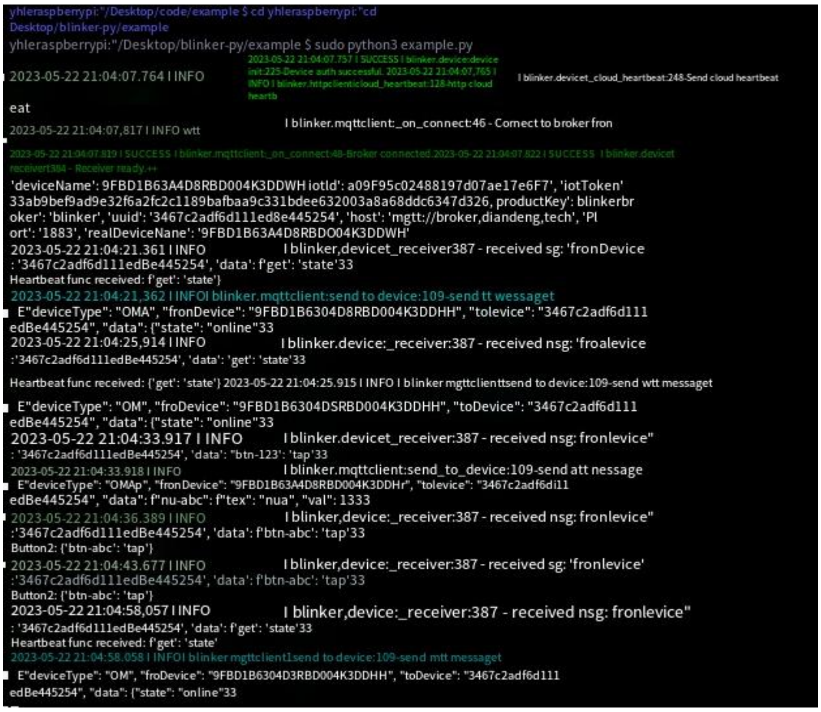
Blinker is a professional and easy-to-use Internet of Things solution that provides server, application, and device-side sdk support. Developed based on a high-performance asynchronous framework, the server side can host a large number of device connections, allowing device owners to easily manage devices. Simple and convenient application with multi-device support sdk, can allow developers to achieve device access in a short time. The device can be connected to the program by setting the key of the Device, for example, device = Device("3f07e35c3649"). You can also edit different buttons for different interactive operations. For example

```
button1 = device.addWidget(ButtonWidget('btn–123'))

button2 = device.addWidget(ButtonWidget('btn–abc'))

number1 = device.addWidget(NumberWidget('num–abc'))
```

After running the program, the Raspberry PI terminal can receive the returned json format information to realize two-way data communication. Press the button to open the door in the Blinker mobile phone client, and the Raspberry PI terminal will receive the information

```
Received msg:{‘fromDevice’:’3467c2adf6d111ed8e445254’,’data’:{‘btn-abc’:’tap’}}
```

This means the open door button is pressed and the raspberry pie receiving the message is programmed to rotate the steering gear. Figure 5.2.1–1.



```
yhleraspberrypi:"/Desktop/code/example $ cd yhleraspberrypi:"cd
Desktop/blinker-py/example
yhleraspberrypi:"/Desktop/blinker-py/example $ sudo python3 example.py
                                      2023-05-22 21:04:07.757 I SUCCESS I blinker.device:device
2023-05-22 21:04:07.764 I INFO        init:225-Device auth successful. 2023-05-22 21:04:07,765 I      I blinker.devicet_cloud_heartbeat:248-Send cloud heartbeat
                                      INFO I blinker.httpclienticloud_heartbeat:128-http cloud
                                      heartb
eat
2023-05-22 21:04:07,817 I INFO wtt                  I blinker.mqttclient:_on_connect:46 - Comect to broker fron
2023-05-22 21:04:07.819 I SUCCESS I blinker.mqttclient:_on_connect:48-Broker connected 2023-05-22 21:04:07.822 I SUCCESS  I blinker.devicet
receivert384 - Receiver ready.++
'deviceName': 9FBD1B63A4D8RBD004K3DDWH iotld': a09F95c02488197d07ae17e6F7', 'iotToken'
33ab9bef9ad9e32f6a2fc2c1189bafbaa9c331bdee632003a8a68ddc6347d326, productKey': blinkerbr
oker': 'blinker', 'uuid': '3467c2adf6d111ed8e445254', 'host': 'mgtt://broker,diandeng,tech', 'Pl
ort': '1883', 'realDeviceNane': '9FBD1B63A4D8RBDO04K3DDWH'
2023-05-22 21:04:21.361 I INFO              I blinker,devicet_receiver387 - received sg: 'fronDevice
: '3467c2adf6d111edBe445254', 'data': f'get': 'state'33
Heartbeat func received: f'get': 'state'}
2023-05-22 21:04:21,362 I INFOI blinker.mqttclient:send to device:109-send tt wessaget
  E"deviceType": "OMA", "fronDevice": "9FBD1B6304D8RBD004K3DDHH", "tolevice": "3467c2adf6d111
edBe445254", "data": {"state": "online"33
2023-05-22 21:04:25,914 I INFO              I blinker.device:_receiver:387 - received nsg: 'froalevice
:'3467c2adf6d111edBe445254', 'data': 'get': 'state'33
Heartbeat func received: {'get': 'state'} 2023-05-22 21:04:25.915 I INFO I blinker mgttclienttsend to device:109-send wtt messaget
  E"deviceType": "OM", "froDevice": "9FBD1B6304DSRBD004K3DDHH", "toDevice": "3467c2adf6d111
edBe445254", "data": {"state": "online"33
2023-05-22 21:04:33.917 I INFO          I blinker.devicet_receiver:387 - received nsg: fronlevice"
: '3467c2adf6d111edBe445254', 'data': "btn-123': 'tap'33
2023-05-22 21:04:33.918 I INFO              I blinker.mqttclient:send_to_device:109-send att nessage
  E"deviceType": "OMAp", "fronDevice": "9FBD1B63A4D8RBD004K3DDHr", "tolevice": "3467c2adf6di11
edBe445254", "data": f"nu-abc": f"tex": "nua", "val": 1333
2023-05-22 21:04:36.389 I INFO              I blinker,device:_receiver:387 - received nsg: fronlevice"
:'3467c2adf6d111edBe445254', 'data': f'btn-abc': 'tap'33
Button2: {'btn-abc': 'tap'}
2023-05-22 21:04:43.677 I INFO              I blinker,device:_receiver:387 - received sg: 'fronlevice'
:'3467c2adf6d111edBe445254', 'data': f'btn-abc': 'tap'33
Button2: {'btn-abc': 'tap'}
2023-05-22 21:04:58,057 I INFO              I blinker,device:_receiver:387 - received nsg: fronlevice"
: '3467c2adf6d111edBe445254', 'data': f'get': 'state'33
Heartbeat func received: f'get': 'state'
2023-05-22 21:04:58.058 I INFOI blinker mgttclient1send to device:109-send mtt messaget
  E"deviceType": "OM", "froDevice": "9FBD1B6304D3RBD004K3DDHH", "toDevice": "3467c2adf6d111
edBe445254", "data": {"state": "online"33
```

Figure 5.2.1-1 Raspberry PI receiving Blinker server information

After the Raspberry PI is connected to the LAN through esp8266WIFI module, we can remotely control the PI through the connection with the Blinker software. The Blinker user interface is shown in Figure 5.2.1-2 and Figure 5.2.1-3 below. The door can be opened by pressing the open button.

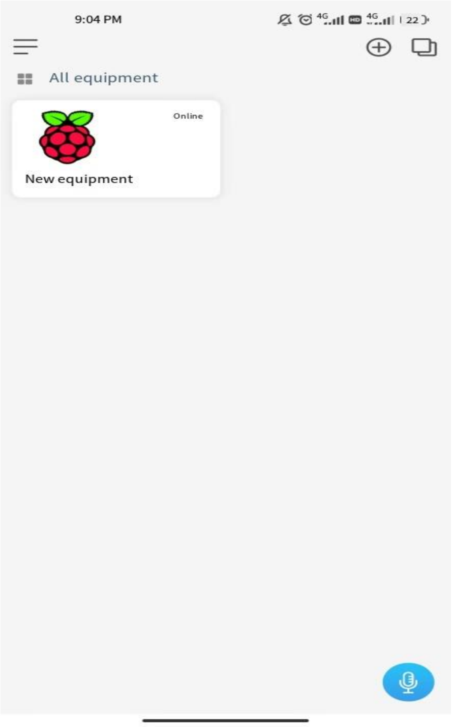
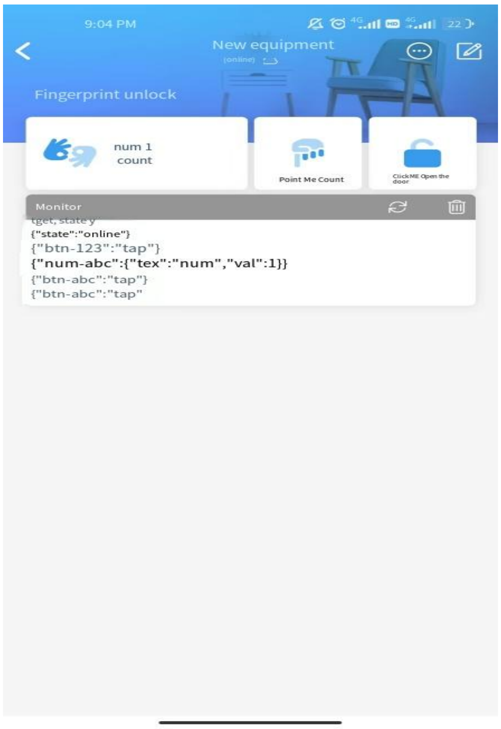


Figure 5.2.1-2 Software interface



```
{get, state y
{"state":"online"}
{"btn-123":"tap"}
{"num-abc":{"tex":"num","val":1}}
{"btn-abc":"tap"}
{"btn-abc":"tap"
```

Figure 5.2.1-3 Remote door opening operation interface

The following figure 5.2.1-4 shows the test state. After pressing the button, the steering engine will rotate counterclockwise to drive the door handle so as to realize the operation of opening the door. After waiting for three seconds, the steering engine will rotate back to the original position for the convenience of closing the door:

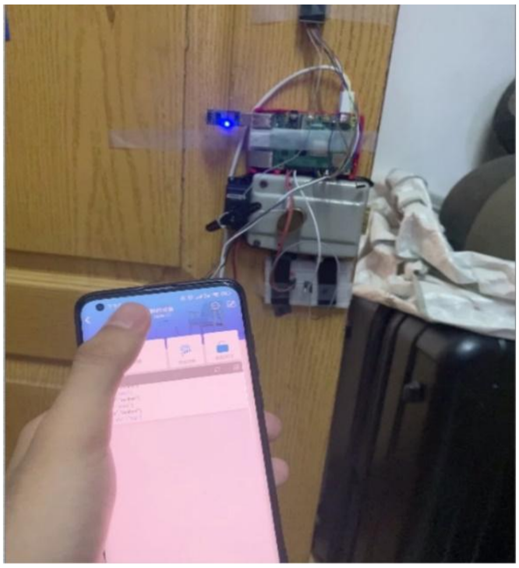


Figure 5.2.1-4 Running state diagram of steering gear

5.2 Smart toilet test

5.2.1 Human body infrared detection Turn on lighting

As shown in Figure 5.2.1-1, considering the large space of the toilet, the Dupont line is difficult to meet in the test directly installed. We used cardboard boxes to simulate the toilet situation.

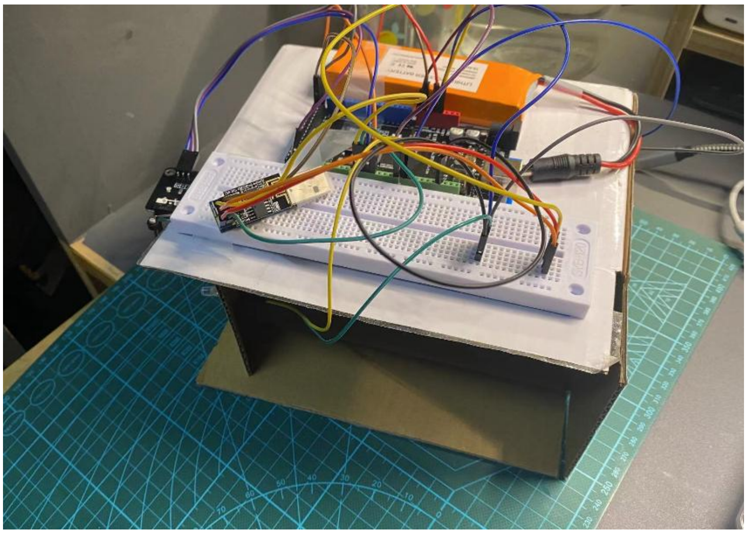


Figure 5.2.1-1 Smart toilet model

When the infrared sensor of human body detects someone (the module lights up blue), it will turn on the headlight, fan and the warning light outside the door. See Figure 5.2.1-2

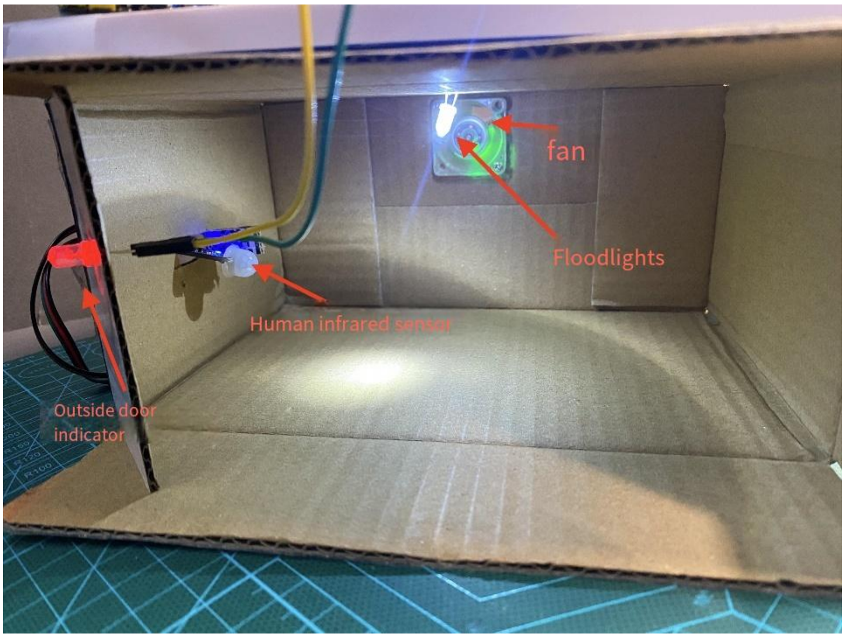


Figure 5.2.1-2 Model Startup lighting diagram

Wait for some time, when the person leaves, and press the collision sensor (green light) to indicate that the door has been opened and closed, turn off the lighting, fans, and outside warning lights.

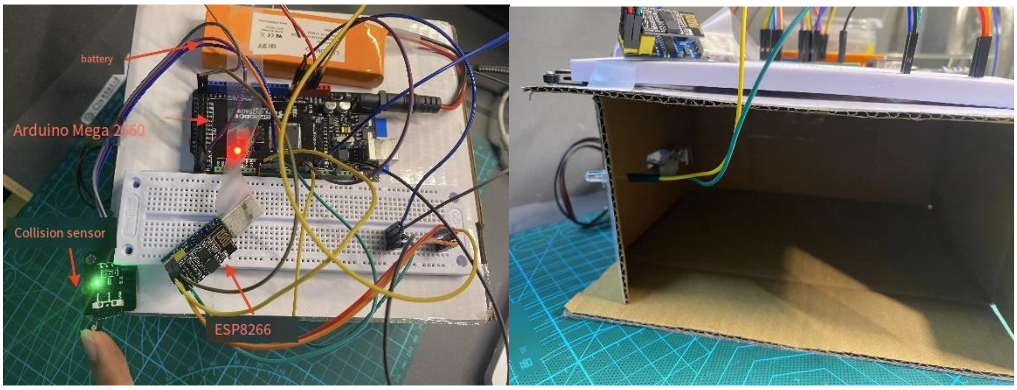


Figure 5.2.1-3 Device not operating status

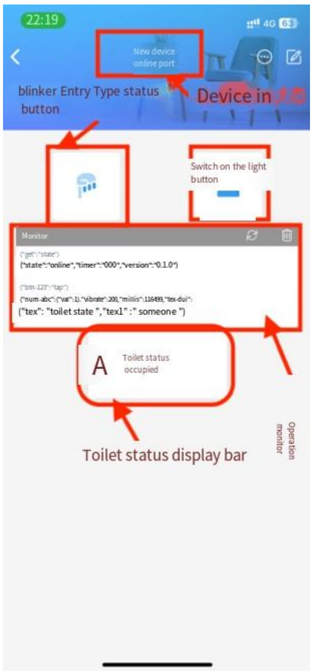
5.2.2 Blinker Remote Operation

In Figure 5.22-1 and Figure 5.22-2, the software operation is shown below.

1. When an iot device is connected to the Blinker server, the software interface will appear online right above it.
2. The Monitor will display device status and button clicks. Blue is the phone action reaction, black is the message returned by the server.

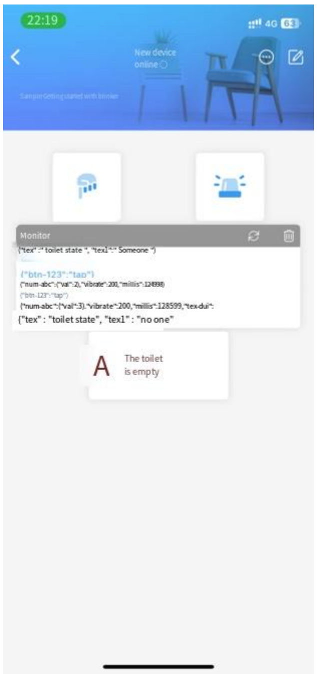
For example: get state indicates that the phone sent a request to the Blinker server. If an iot device is connected to the Blinker server, the Blinker server will send a message to the phone in Json format: state online.

When the "Click me to turn lights on and off" button is clicked, the monitor will have the information of the btn-abc "tap". btn-123 is to view the status of the toilet, and when it is clicked, the server will return the status of the toilet in Json format. Tex-dui is the name of the text message, divided into tex and tex1: "Toilet status" and "Someone".



```
{"get":"state"}
{"state":"online","timer":"000","version":"0.1.0"}

{"btn-123":"tap"}
{"num-abc":{"val":1},"vibrate":200,"millis":116499,"tex-dui":
("tex": "toilet state ","tex1" :" someone ")
```

Figure 5.2.2-1



```
{"tex" :" toilet state ", "tex1":" Someone "}

{"btn-123":"tap"}
{"num-abc":{"val":2},"vibrate":200,"millis":124998}
{"btn-123":"tap"}
{"num-abc":{"val":3},"vibrate":200,"millis":128599,"tex-dui":
{"tex" : "toilet state", "tex1" : "no one"
```

Figure 5.2.2-2

6. References

Human body infrared Sensors:
https://wiki.dfrobot.com.cn/_SKU_SEN0171_PIR_motion_sensor_%E7%BA%A2%E5%A4%96%E7%83%AD%E9%87%8A%E7%94%B5%E8%BF%90%E5%8A%A8%E4%BC%A0%E6%84%9F%E5%99%A8

Collision sensor:
https://wiki.dfrobot.com/Crash_Sensor__SKU__SEN0138_

Modify the communication serial port:
https://blog.csdn.net/guet_gjl/article/details/85164072?ydreferer=aHR0cHM6Ly9naXRodWIuY29tL0xlb3BhcmQtQy9BUzYwOA%3D%3D

ESP8266 serial port communication, burning:
https://blog.csdn.net/sunshineQY/article/details/80084882

DF15RSMG steering gear:
https://www.dfrobot.com/product-959.html

7. conclusion

This project basically realizes the development of fingerprint module, steering gear and sensor, which can handle door opening and induction in reality, and uses Blinker to form intelligent dormitory Internet of things for remote control. A complete embedded development process has been built.

In the process of the project, we cooperate with each other to select the information conducive to the progress of the project from the complicated information of the business, and apply it in the process of development. From the individual inquiry, connection and testing of each module to the assembly of the whole project, the blinker environment construction of the Internet of Things system is often stuck because of the absence of a step. However, with the help of each other, the Blinker is always enlightened. In the process, I am more familiar with apt-get, pip and other downloading tools, and have a more in-depth understanding of computer network.

However, there are still many deficiencies in this project:

1. Sustainability: The mechanical mechanism used by the steering engine to turn the door handle is just a simple pull and pull. Due to the time relationship, it is impossible to think of a solution without damaging the door lock. In reality, the pull-and-pull door of this project is easy to slip and malfunction after many experiments.

2. Security issues: Although fingerprint unlocking is convenient, fingerprint information may be imitated or embezzled, resulting in security risks. Remote unlocking of mobile phones may be subject to hacker attacks, and security measures need to be strengthened to protect user privacy and prevent unauthorized access.

3 Power dependence: A stable power supply is required and may not work properly if the battery runs out or external power is disconnected, resulting in failure to unlock.

4. Compatibility problems: Different brands and models may have compatibility problems. Different mobile operating system versions may support different remote unlock functions.

5. Technical failure: Smart dormitory involves a variety of technical components, such as fingerprint sensors, control modules, etc., and there is a risk of technical failure.

The process from conception to completion of the Raspberry PI development project needs to be realized step by step. The selection of topic at the beginning requires an overall sense of direction for the project, and the project needs should be within the scope of the team's ability to achieve, rather than blindly innovate, and should be combined with the reality. In the concrete implementation of the project, I learned to use Raspberry PI to control various electronic devices and obtain the values of various sensors, which was partly due to what I learned in the course, which made me easy to learn.

8. Project division

Person in charge: 200860421 Wang Bin (37h)

Tasks	Time
Product design and project planning writing	3h
Sensor selection of Arduino module and project material procurement	1h
Data retrieval and testing of Arduino infrared module	1h
ArduinoIDE and esp8266 libraries install Blinker environment configuration	3h
Blinker environment configuration	1h
Construction of smart toilet model, module assembly	3h
Debugging and commands for the Esp8266 serial port	3h
Smart toilet main program writing	3h
Raspberry PI system reburn (memory card burns out)	1h
The Esp8266 communicates with the serial port of the arduino	3h
Door lock drag mount	1h
Report PPT content structure design	1h
Demo video shooting	1h
Preparation of various documents	10h
English report	2h
Total	37h

Team Member: 200860420 Tao Junfei (32h)

Tasks	Time
Find and learn the steering gear related information (understand 180 degree and 360 degree steering gear, etc.)	3h
Pick and purchase steering gear	2h
Find ways to learn to control the steering gear	3h
Write the code of steering gear rotation in C language	2h

Embed the written steering gear rotation code into the main program and debug	3h
Learn to use Blinker software and connect Raspberry PI with Python debug	3h
Write the code to control the steering gear in Python and embed it in the linker	2h
Debug run two programs Fingerprint recognition unlock and Blinker Remote unlock	2h
Raspberry PI SD card is broken, reintegrate the saved code for debugging	3h
PPT making	1h
Preparation of various documents	6h
Report in English	2h
Total	32h

Team Member: 200860426 Xu Zhen (30h)

Task	Time
AS608 module selection and procurement	1h
AS608 module data reading	5h
AS608 module data test	1h
Raspberry PI Environment Configuration	2h
AS608 main program writing	6h
Door lock drag mount	1h
Intelligent door lock project integration	3h
Report the design of PPT content structure	1h
Demo video shooting	1h
Preparation of various documents	7h

English report	2h
Total	30h

Group Member: 200860422 Xinqiang Wang (28h)

Task	Time
Smart toilet model construction, module assembly	2h
Smart toilet main program writing	3h
Esp8266 Modules Study and Test	2h
Esp8266 Serial debugging and commands	3h
The Esp8266 communicates with the serial port of the arduino	3h
AS608 module data reading	5h
AS608 Main program writing	3h
Intelligent door lock indicator light programming	1h
PPT Making	1h
Preparation of various documents	5h
Total	28h