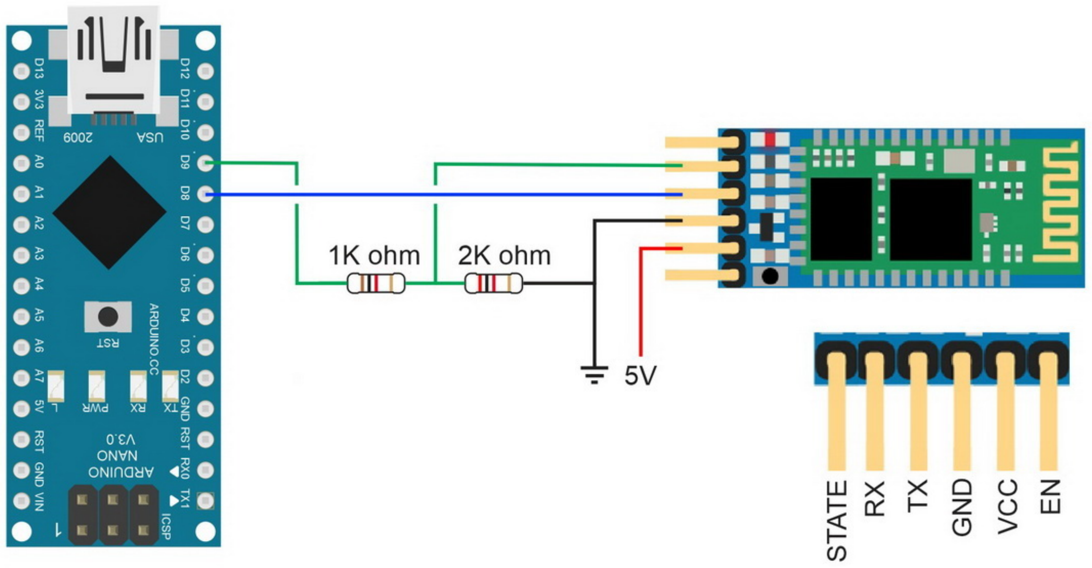
Hardware Synthesis Lab I

Semester 1 Academic Year 2017

Keyless entry through Bluetooth protocol with STM32F4 and ESP8266



Team Members	Responsibilities
5730290621 Narit Kemtongcharoen	Front-end development
5831037521 Parin Meparinya	System Architecture
5831024321 Tanapol Oonmongkol	User Interface
5831071821 Supanat Limjitti	Embedded System Development

Table of Contents

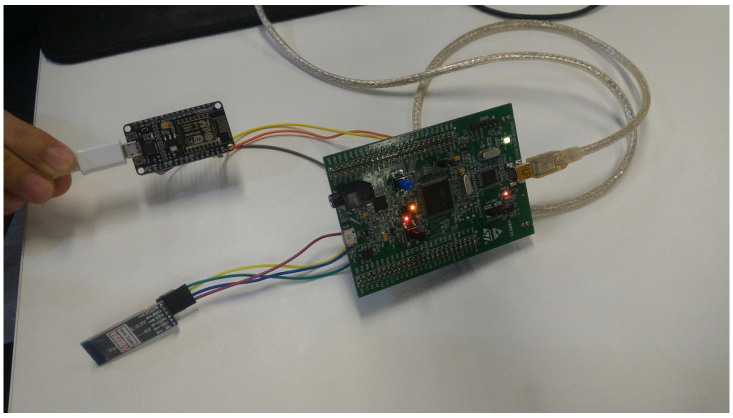
Preface

For this project we decided to utilize the Bluetooth technology to solve an everyday problem, i.e. having to carry physical keys in the digital age.

Bluetooth technology can be found in a wide range of devices from smartwatches to smartphone and from fitness tracker to digital cameras which is why we chose this technology, as it is common enough that users do not have to buy a new device but also powerful enough to be utilized in this application.

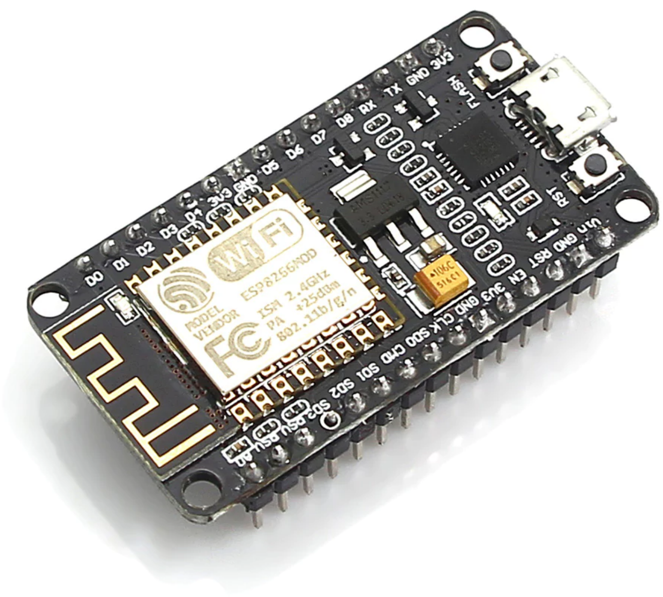
We came up with this application as we were thinking how we could improve our everyday lives with ordinary technology found lying around. We found that even though we carry electronic keys with us everywhere but we do not utilize it, instead we still rely heavily on cumbersome physical keys which is both difficult to carry and highly insecure.

The solution we came up with is to use existing devices on our body whether it is a smartphone, smartwatch or even a digital camera to pair with a door lock to act as a key. With the security measures provide in the Bluetooth protocol[1], you can be sure that no unauthorized person can access the device and unlock the door.



[1] http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-121r1.pdf

System architecture/NodeMCU back-end

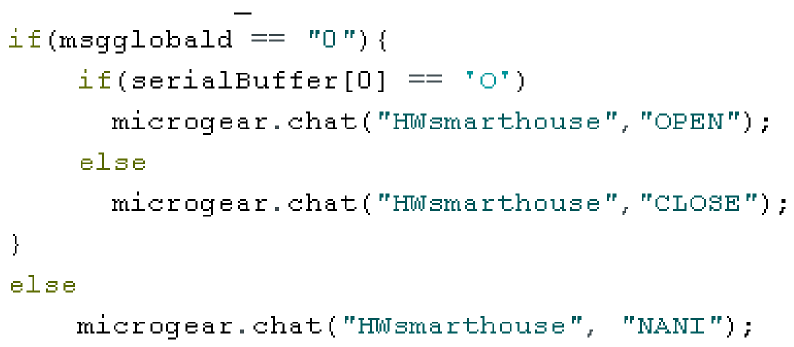


Since board STM32F4 can't connect to WIFI,so we use NodeMCU to help about send sensor data from STM32F4 to netpie via WIFI .

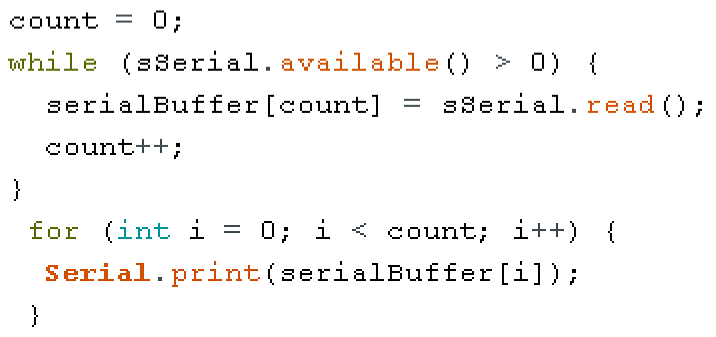
We also use microgear to help NodeMCU connect to NETPIE.

Microgear-esp8266-arduino is a client library that help NodeMCU connect with netpie platform for develop IOT application.

We can use microgear.chat() in microgear library to communicate from one to another device in the send APPID. as you can see in the rightside, we use microgear.chat() to send string "CLOSE" and "OPEN" when bluetooth sensor provide matched string .



```
if(msgglobald == "0"){
    if(serialBuffer[0] == 'O')
      microgear.chat("HWsmarthouse","OPEN");
    else
      microgear.chat("HWsmarthouse","CLOSE");
}
else
    microgear.chat("HWsmarthouse", "NANI");
```



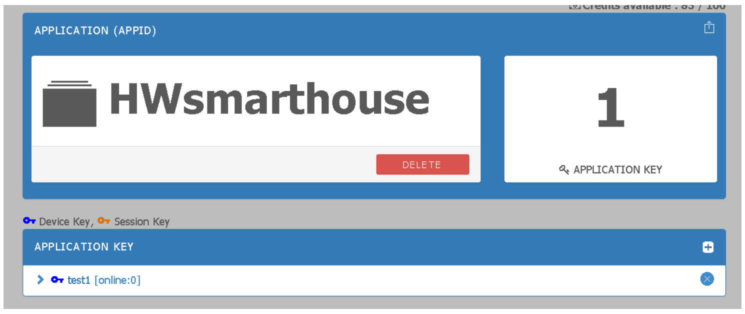
```
count = 0;
while (sSerial.available() > 0) {
  serialBuffer[count] = sSerial.read();
  count++;
}
 for (int i = 0; i < count; i++) {
  Serial.print(serialBuffer[i]);
 }
```

sSerial is a softwareserial that contain rxPin and txPin. so we can use sSerial.read() to recieve data and corrected them in char array name serialbuffer.

NETPIE.io

NETPIE is stand for Network Platform for Internet of Everything. that mean it will help one device connected to another device easier.

In this project, we only use Application part of netpie to help our website can use sensor data that read from board STM32F4.


APPLICATION (APPID)

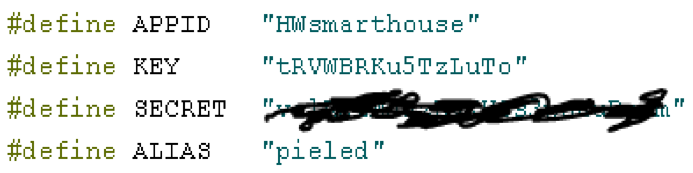
HWsmarthouse

1

APPLICATION KEY

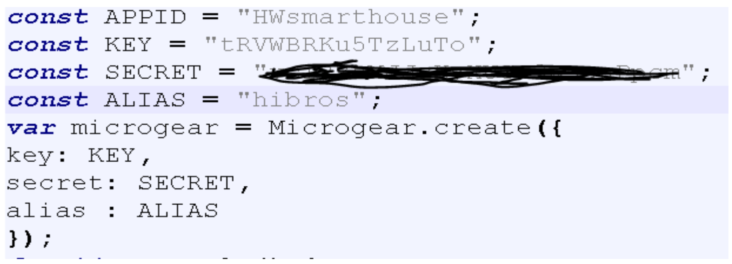
test1 [online:0]

to connect two parts(NodeMCU and Website) together, we must have our own appid and application key to access data.



```
#define APPID    "HWsmarthouse"
#define KEY      "tRVWBRKu5TzLuTo"
#define SECRET   "[redacted]"
#define ALIAS    "pieled"
```

in left side,it's a code in arduino that make our NodeMCU can send and receive data .

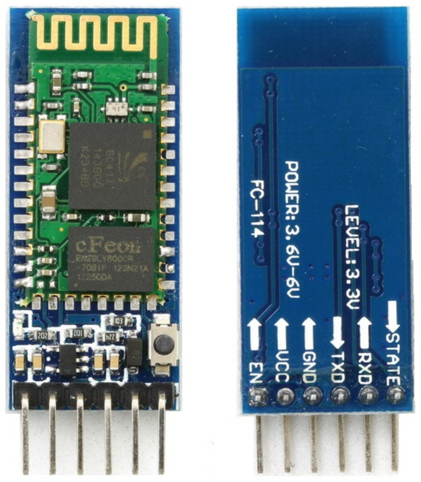


```
const APPID = "HWsmarthouse";
const KEY = "tRVWBRKu5TzLuTo";
const SECRET = "[redacted]";
const ALIAS = "hibros";
var microgear = Microgear.create({
key: KEY,
secret: SECRET,
alias : ALIAS
});
```

and this is code from html part, you can see that it has the same APPID,key and secret .

Sensor/Bluetooth Module

As this is a project in which we utilize an STM32F4-DISCOVERY board, we chose to use the HC-05 Bluetooth module which provides Serial connection capabilities through the UART protocol.[2] The HC-05 Bluetooth module is connected to the STM32F4 board as shown in the figure on the cover of this report comprising of RX, TX, VCC and GND wires.



HC-05 Bluetooth module shown above

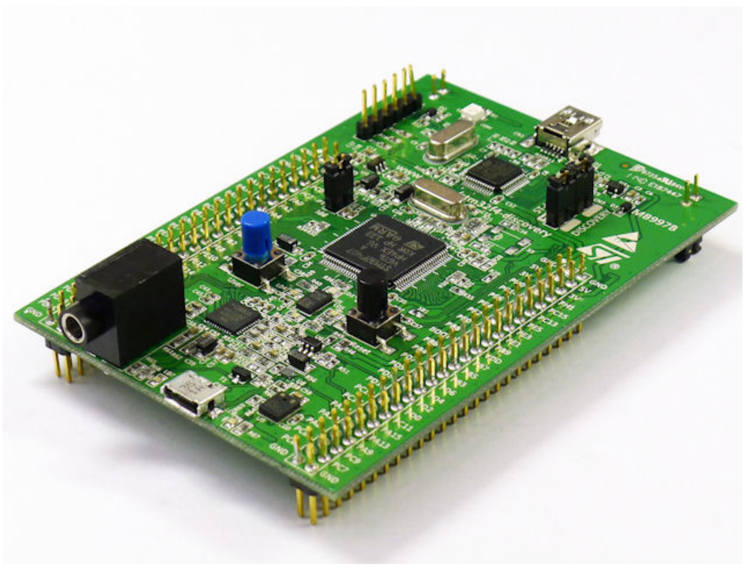
After you have connected your Bluetooth device to the module, you can send commands to operate the door, e.g. OPEN and CLOSE. The command is then sent from the provided mobile application through serial communication to the Bluetooth module which the accepts the command then pass along the information also through serial communications to the ESP8266 NodeMCU to be displayed on the website as will be explained further in the front-end and web development section.

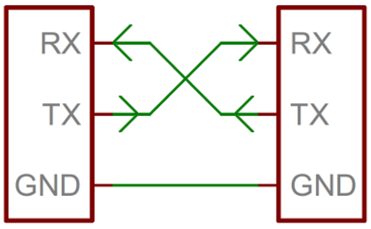
[2] https://www.itead.cc/wiki/Serial_Port_Bluetooth_Module_(Master/Slave)_:_HC-05

STM32F4 board development

As we all know, for this project, the STM32F4 is one of the crucial elements that combines to make this project a possibility. With the capabilities of processing two serial communication channels simultaneously we can seamlessly devise a method to receive serial communication from the aforementioned Bluetooth module then save the data received in a character array `char[] buff` with the included function, `HAL_UART_Receive(&huart2, &buff, 2, 100)`. As a result of the implementation of strings by the C language, the string received is stored in the form of `char[]`.

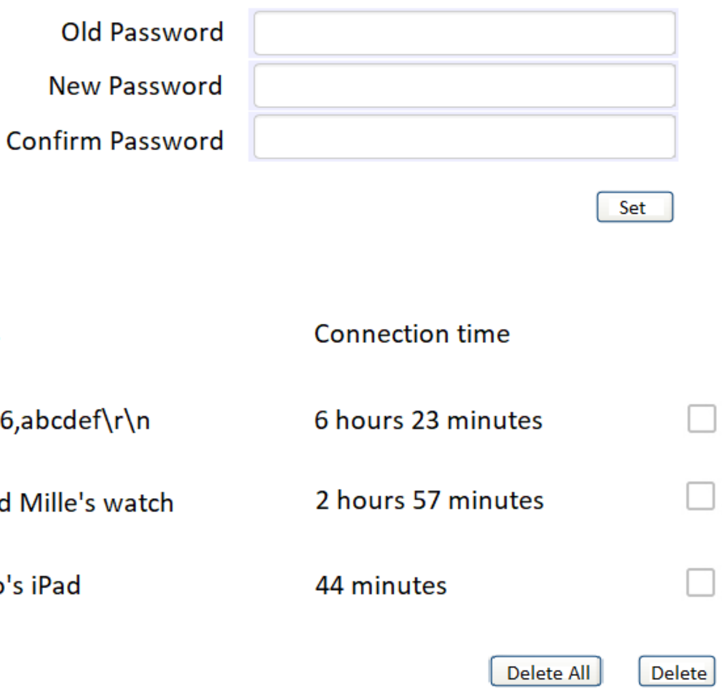
After that the received string or command will then be transmitted to the ESP8266 module, in this case, the NodeMCU, also through a serial communication channel designated by `huart3` as to not be conflicting to `huart2` which is connected to the Bluetooth module. The function used to transmit the data serially is as follows; `HAL_UART_Transmit(&huart3, buff2, sizeof(buff2), 100)`. After the transmission, the program aboard the STM32F4 then returns to a state in which it is ready and is constantly listening for data from the connected module.





UI design

At first, our project tried to provide a function for administrator such as deleting authentic devices and set new password for bluetooth access. So the interface should be like this



Old Password	
New Password	
Confirm Password	

Device	Connection time	
1234,56,abcdef\r\n	6 hours 23 minutes	☐
Richard Mille's watch	2 hours 57 minutes	☐
Naruto's iPad	44 minutes	☐

Unfortunately, There was a problem with embeded system. So our projected got shrunk and the interface haven’t been used.

The goal of our project is to authentify via bluetooth device to lock/unlock the door. But the sensor we only got is bluetooth sensor so we can’t build a model for the house. My plan was to create second interface to mock-up a lock for the house

Welcome : Master Bruce

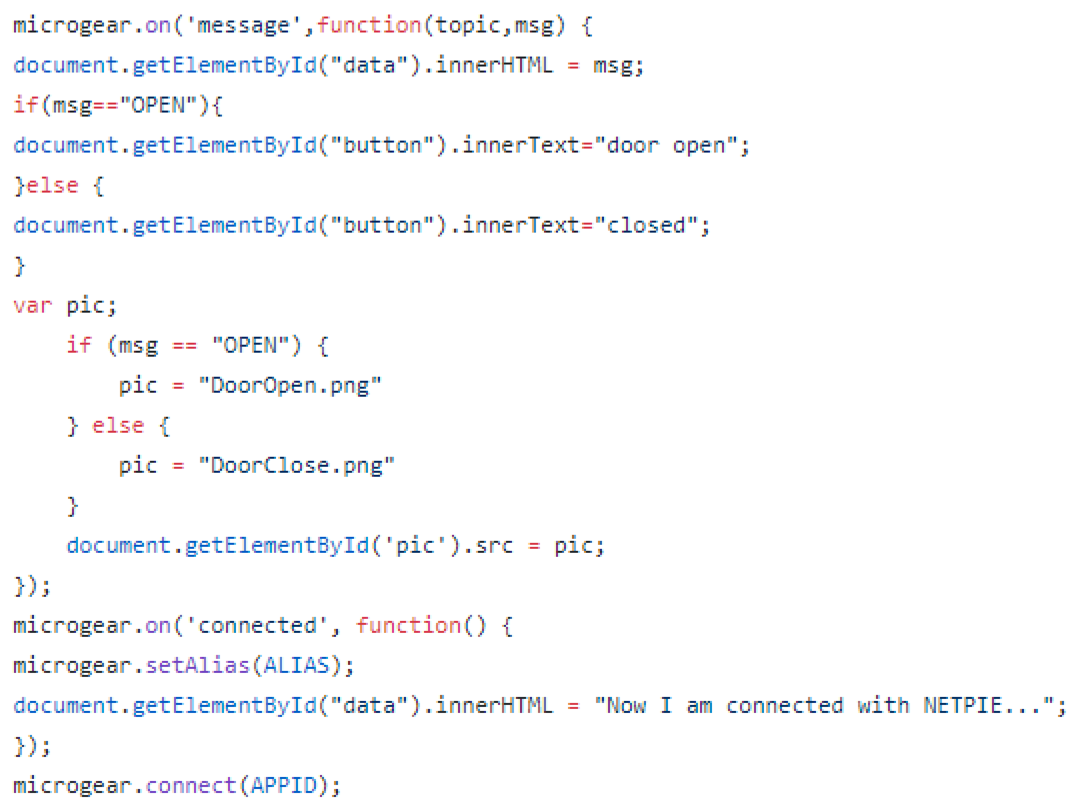
LOCKED

The second interface shows whether the door is locked or not which stands for whether the house is locked or not.

Front-end development

After the design part, we use the mock-up interface to make it become code. The first interface was canceled by some embedded system problem and new interface to let us know the status of the door at which we control. Front-end website get the necessary value from back-end system that send from the board and sensors.

The script function for HTML website is so simple



```
microgear.on('message',function(topic,msg) {
document.getElementById("data").innerHTML = msg;
if(msg=="OPEN"){
document.getElementById("button").innerText="door open";
}else {
document.getElementById("button").innerText="closed";
}
var pic;
    if (msg == "OPEN") {
        pic = "DoorOpen.png"
    } else {
        pic = "DoorClose.png"
    }
    document.getElementById('pic').src = pic;
});
microgear.on('connected', function() {
microgear.setAlias(ALIAS);
document.getElementById("data").innerHTML = "Now I am connected with NETPIE...";
});
microgear.connect(APPID);
```

so the signal parameter is signal that we receive from the board.