Home Automation System Implementation and Simulation
CSCE493002 - Selected Topics in CSCE (2021 Spring)
Instructor: Tamer ElBatt

Ismael Elsharkawi 900173030
Mohamed Basuony 900182339

Link to YouTube Demo: https://youtu.be/4LBKP6YRZG4

The code:

First, we installed the CayenneMQTTESP32 because we were planning on using the Cayenne IoT cloud.

Cayenne cloud was chosen because it has an organized UI as IoT systems can be built using drag and drop. It also has a variety of libraries that can be installed to integrate many chips such as ESP32, Raspberry Pi, Arduino, LoRa and many others.

How to upload the code to ESP32?

Click upload then click the boot button on the ESP32 then click reset on the ESP32

Library Prerequisites:

```
#include <CayenneMQTTESP32.h>
#include <Time.h>
#include <TimeLib.h>
```

Defining Cayanne's attributes in our code:

In order to be able to connect to Cayanne's cloud. Defining the Debug and print keywords is essential. The username, password and ClientID needs to be defined in order to be able to transfer the data to our cloud. The ssid and WIFI password are also needed to connect to the local network.

```
#define CAYANNE_DEBUG
#define CAYANNE_PRINT Serial
#include <CayenneMQTTESP32.h>

// Cayenne authentication info. This should be obtained from the Cayenne Dashboard.
char username[] = "3e3aa4b0-a47d-11eb-8779-7d56e82df461";
char password[] = "b6e05921421ce3269dc0624b00fd2b036a3f996c";
char clientID[] = "7fe15340-a47e-11eb-b767-3f1a8f1211ba";

char ssid[]= "WE7EA1CC";
char wifiPassword[]="kb450229";
```

Defining the LED's, their LDR values, and their position on the circuit:

The LDR values are defined as digital values in the D0 pin of the LDR sensor. Note that they are defined 3 times, so that if there are 3 separate LDR sensors, only the values of the LDR_D0_green, LDR_D0_blue and LDR_D0_yellow would be changed to the appropriate pins. The green, red and yellow led values are also defined at the top of the code. Room channels also are given values. Those are the Cayenne channels that we post the readings of the sensors onto.

```
#define LDR_D0_green 18
#define LDR_D0_blue 18
#define LDR_D0_yellow 18
#define green_LED 15
#define blue_LED 22
#define yellow_LED 23

#define ROOM_1_CHANNEL 1
#define ROOM_2_CHANNEL 2
#define ROOM_3_CHANNEL 3
```

Setup Function:

The pinMode function is used to configure the LDR D0 values as input and the LEDs as output. The serial begin function initializes the baud rate is set to 9600. Also, to initialize uploading data to the cloud, Cayenne begin function gets the username, password, client ID, SSID, and WIFI Password.

```
// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  pinMode(LDR_D0_green, INPUT);
  pinMode(LDR_D0_blue, INPUT);
  pinMode(LDR_D0_yellow, INPUT);
  pinMode(green_LED, OUTPUT);
  pinMode(blue_LED, OUTPUT);
  pinMode(yellow_LED, OUTPUT);
  Serial.begin(9600);
  Cayenne.begin(username, password, clientID,ssid,wifiPassword);
}
```

turn_on_off Function:

The function reads the values coming from the LDR sensor and controls the circuit accordingly. If the sensor is sensing light, it returns the value of 1 and therefore, it signals for the LEDs to turn on. If the sensor reads the value of 0, it signals for the LEDs to turn off. All that data is sent to Cayenne by the function virtualWrite().

```
void turn_on_off(int pin_LDR_no, int pin_out_no, int room_channel){
  int sensorValue = digitalRead(pin_LDR_no);
  if(sensorValue==1) digitalWrite(pin_out_no, HIGH);
  if(sensorValue==0) digitalWrite(pin_out_no, LOW);
  Cayenne.virtualWrite(room_channel, sensorValue,"digital_sensor", "d");
}
```

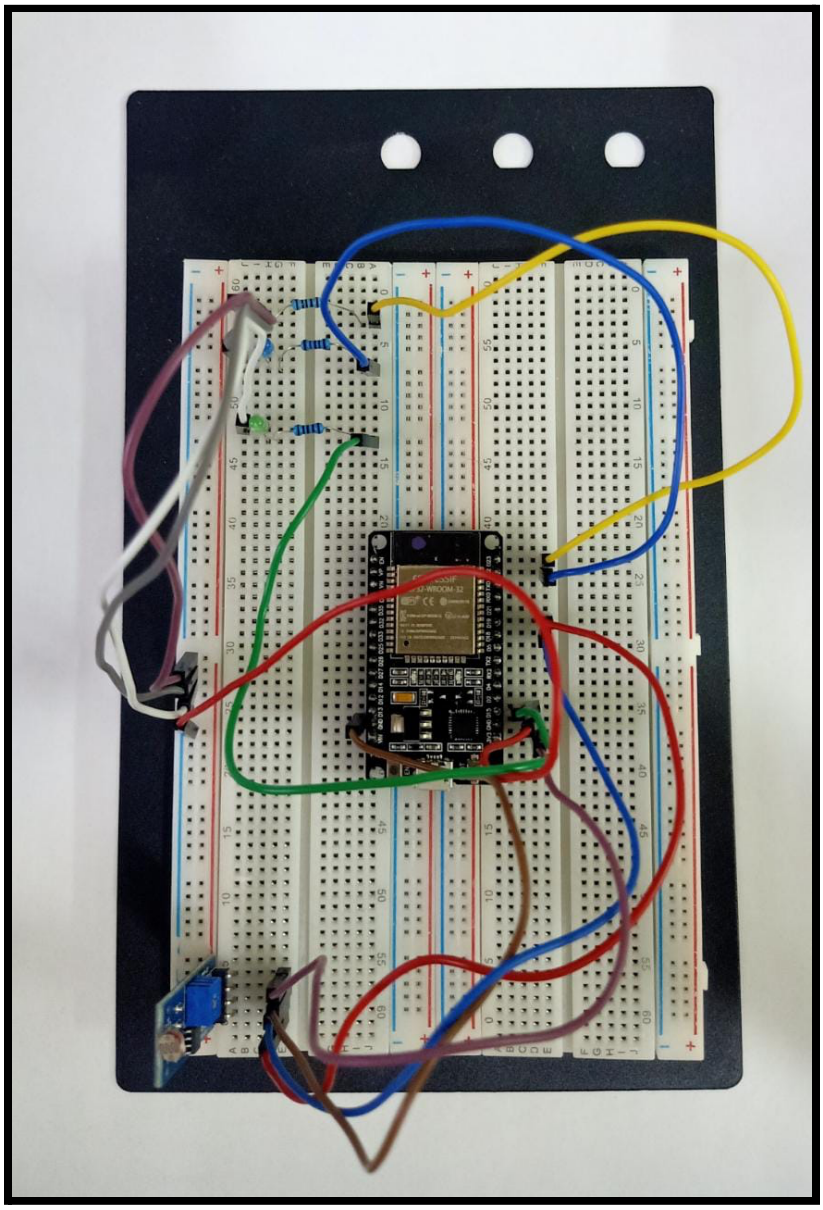
loop function:

The function loops over the functionality of the previous code to keep the sensor reading values and controlling the LEDs. Furthermore, loop() function is used in Cayenne to keep the

transferring of data to the cloud.

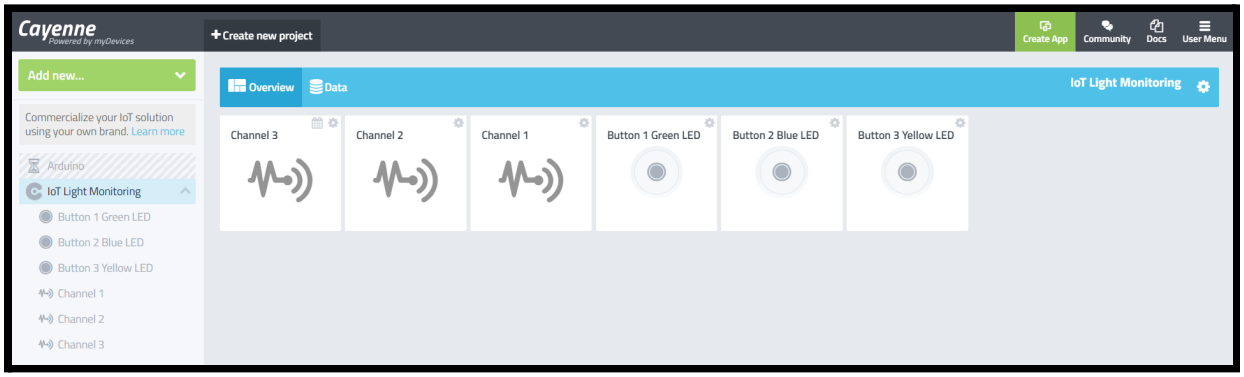
```
void loop() {
  // read the input on analog pin 0:
  Cayenne.loop();
  turn_on_off(LDR_D0_green, green_LED, ROOM_1_CHANNEL);
  turn_on_off(LDR_D0_blue, blue_LED, ROOM_2_CHANNEL);
  turn_on_off(LDR_D0_yellow, yellow_LED, ROOM_3_CHANNEL);
}
```

The circuit:



The connections in the following circuit are as follows. The 3v3, GND and D18 are connected to the LDR sensor with their respective pins. D15, D22, D23 are connected to the LEDs. The LEDs are connected to the ground and 3 resistors with the value of 100 ohms are used in series to the LEDs.

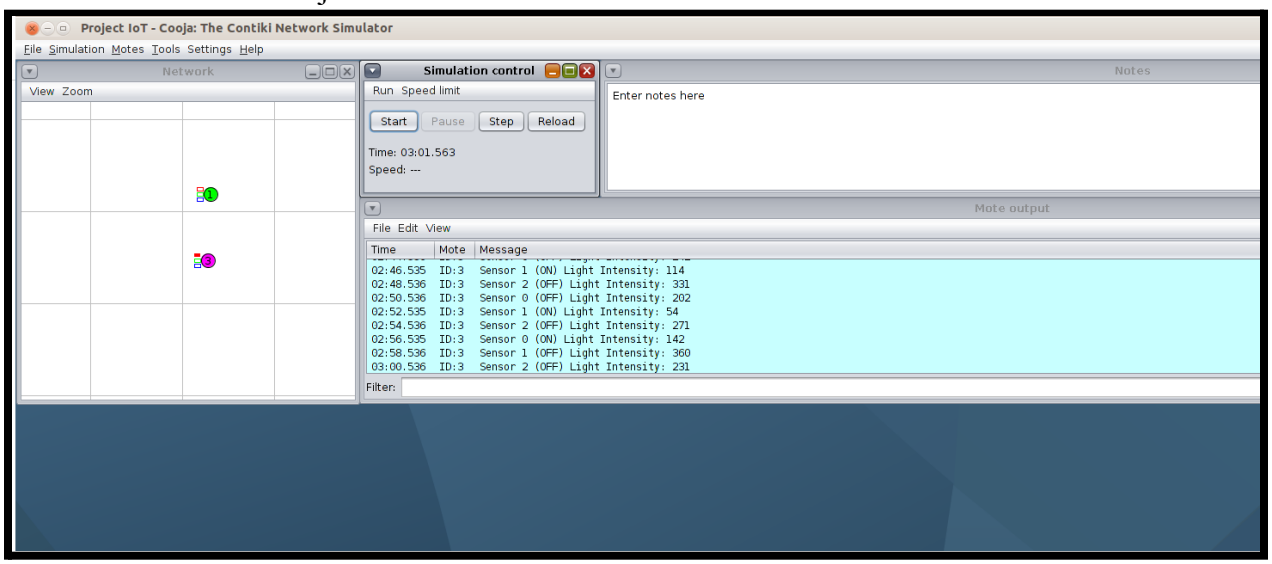
Cayenne:



The simulation:

We opened skywebsense file and we choose the light sensors as our motes. Since the skywebsense have temperature and light sensors, we are not concerned with the temperature readings we are only concerned with the light. We then set up a threshold value to compare it with our simulation readings. If our readings are lower than the threshold value, then the red/blue/green LEDs will be on. If the readings are higher than the threshold, then the LEDs will turn off. The readings are displayed below at different settings. In the simulation below we have 1 skywebsense motes and one RPL border router to display the data on our browser. The skymote simulates 3 light intensity sensors, that is why the value of History is set to 3 to have only 3 sensors. In order to be able to see the results from firefox, the following command should be entered on the terminal:

make connect-router-cooja.



The Bonus:

1. We made the preset timers, where the light is turned on at a specific preset time and is turned off at a specific preset time. This is made for each led of the 3 LEDs.

```
void turn_on_off_green_schedule(){
  if(month()==5 && day()==3 && hour()==1 && minute()==5 && second()>=30){
      digitalWrite(green_LED, HIGH);
      flag_set_from_button[0] = 1;
  }
  if(month()==5 && day()==3 && hour()==1 && minute()==5 && second()>=30){
      digitalWrite(green_LED, LOW);
      flag_set_from_button[0] = 0;
  }
}

void turn_on_off_yellow_schedule(){
  if(month()==5 && day()==3 && hour()==1 && minute()==5 && second()>=30){
      digitalWrite(yellow_LED, HIGH);
      flag_set_from_button[1] = 1;
  }
  if(month()==5 && day()==3 && hour()==1 && minute()==5 && second()>=30){
      digitalWrite(yellow_LED, LOW);
      flag_set_from_button[1] = 0;
  }
}

void turn_on_off_blue_schedule(){
  if(month()==5 && day()==3 && hour()==1 && minute()==5 && second()>=30){
      digitalWrite(blue_LED, HIGH);
      flag_set_from_button[2] = 1;
  }
  if(month()==5 && day()==3 && hour()==1 && minute()==5 && second()>=30){
      digitalWrite(blue_LED, LOW);
      flag_set_from_button[2] = 0;
  }
}
```

2. Allow the user to control the room light independent of the LDR sensor value using the cloud system. This part was done using the drag and drop feature on the cloud system, Cayenne. We added 3 widgets (buttons) on Cayenne. These widgets posted their data on separate channels. The Arduino code read from these channels what data values are available. These 3 widgets represented the 3 LEDs in our circuit. With every widget (button) push, the LEDs change their values accordingly.

```
CAYENNE_IN(4){
  digitalWrite(green_LED, getValue.asInt() );
  flag_set_from_button[0] = getValue.asInt();
}

CAYENNE_IN(5){
  digitalWrite(blue_LED, getValue.asInt() );
  flag_set_from_button[1] = getValue.asInt();
}

CAYENNE_IN(6){
  digitalWrite(yellow_LED, getValue.asInt() );
  flag_set_from_button[2] = getValue.asInt();
}
```