# Cs122a\_p1\_final\_project.ino

//stream link: http://192.168.245.163/mjpeg/1

// Blynk device template information

#define BLYNK\_TEMPLATE\_ID "TMPL2FEVyWppL"

#define BLYNK\_TEMPLATE\_NAME "Quickstart Device"

#define BLYNK\_AUTH\_TOKEN "3e8SJ427t5bXDpBCkXc1ni6jjobGn-lp"

// Blynk prints, can be commented out.

#define BLYNK\_PRINT Serial

//Pins for the DC motor.

#define ENABLE 5

#define UP 9

#define

10

//Baud rate for ESP8266 ((data rate in bits per second))

#define ESP8266\_BAUD 115200

//DHT11 Pin

#define DHTPIN 7

//DHT11 type definition

#define DHTTYPE DHT11

#include <ESP8266\_Lib.h>

#include <BlynkSimpleShieldEsp8266.h>

#include <DHT.h>

#include <SoftwareSerial.h>

//WIFI Credentials

char ssid[] = "Verizon-SM-G955U-1B9A";

char pass[] = "llhw778\*";

//Current on/off state of the DC motor

int fanState = 0;

//Software Serial for Uno R3

SoftwareSerial EspSerial(2, 3);

//Define baud rate

ESP8266 wifi(&EspSerial);

DHT dht(DHTPIN, DHTTYPE);

BlynkTimer timer;

typedef struct task {

int state;

unsigned long period;

unsigned long elapsedTime;

int (\*TickFct)(int);

} task;

int delay\_gcd;

const unsigned short tasksNum = 2;

task tasks[tasksNum];

enum SM1\_States { SM1\_INIT, SM1\_S0, SM1\_S1};

int SM1\_Tick(int state){

switch (state) { // State transitions

case SM1\_INIT:

state = SM1\_S0;

break;

case SM1\_S0:

state = SM1\_S0;

break;

case SM1\_S1:

break;

}

switch (state) { // State Action

case SM1\_INIT:

break;

case SM1\_S0:

Blynk.run();

break;

case SM1\_S1:

break;

}

return state;

}

enum SM2\_States { SM2\_INIT, SM2\_S0, SM2\_S1};

int SM2\_Tick(int state){

switch (state) {

case SM2\_INIT:

state = SM2\_S0;

break;

case SM1\_S0:

state = SM1\_S0;

break;

case SM2\_S1:

break;

}

switch (state) {

case SM2\_INIT:

break;

case SM2\_S0:

timer.run();

break;

case SM2\_S1:

break;

}

return state;

}

BLYNK\_WRITE(V1) {

fanState = param.asInt();

fanControl();

Serial.print("fanState after digitalWrite: ");

Serial.println(fanState);

}

void fanControl(){

if (fanState == 0) {

digitalWrite(ENABLE,LOW);

} else {

digitalWrite(ENABLE,HIGH);

}

}

//Sends DHT readings to Blynk app

void sendSensor()

{

float h = dht.readHumidity();

float t = dht.readTemperature(true);

if (isnan(h) || isnan(t)) {

Serial.println("Failed to read from DHT sensor!");

return;

}

Blynk.virtualWrite(V5, h);

Blynk.virtualWrite(V6, t);

Serial.print(F("Humidity: "));

Serial.print(h);

Serial.print(F("% Temperature: "));

Serial.print(t);

Serial.println(F("°F"));

if (t > 79.00){

digitalWrite(ENABLE,HIGH);

}

else if (t < 79.00 && fanState != 1){

digitalWrite(ENABLE,LOW);

}

}

void setup()

{

pinMode(ENABLE,OUTPUT);

pinMode(UP,OUTPUT);

pinMode(DOWN,OUTPUT);

digitalWrite(UP,LOW);

digitalWrite(DOWN,HIGH);

digitalWrite(ENABLE,LOW);

Serial.begin(115200);

EspSerial.begin(ESP8266\_BAUD);

delay(10);

Blynk.begin(BLYNK\_AUTH\_TOKEN, wifi, ssid, pass);

dht.begin();

timer.setInterval(1000L, sendSensor);

unsigned char i = 0;

tasks[i].state = SM1\_INIT;

tasks[i].period = 500;

tasks[i].elapsedTime = tasks[i].period;

tasks[i].TickFct = &SM1\_Tick;

i++;

tasks[i].state = SM2\_INIT;

tasks[i].period = 500;

tasks[i].elapsedTime = tasks[i].period;

tasks[i].TickFct = &SM2\_Tick;

delay\_gcd = 500;

}

void loop()

{

unsigned char i;

for (i = 0; i < tasksNum; ++i) {

if ( tasks[i].elapsedTime >= tasks[i].period) {

tasks[i].state = tasks[i].TickFct(tasks[i].state);

tasks[i].elapsedTime = 0;

}

tasks[i].elapsedTime += delay\_gcd;

}

delay(delay\_gcd);

}

# Design Schematic

