/\*

IoT SMART FARM\_V2

\*/

#include <DHT\_U.h> //교재 참고

#include <VitconBrokerComm.h>// 이미 구현된 네트워크 (데이터 전송 수신역할 )

#include "OLED\_logo.h" // oled에 띄위기 위한 헤더

#include "SoftPWM.h" // PWM의 위치와 사용을 위한 헤더

using namespace vitcon; // <VitconBrokerComm.h>안의 네임스페이스 vitcon : BrokerComm -> 넷웤

#define DHTPIN A1 // Digital pin connected to the DHT sensor = A1센서에(온습도) 연결위한 재정의

#define DHTTYPE DHT22 // 이 타입은 DHT22임요 (11도 있는게 이건 기계타입임)

#define LAMP 17 // D17 핀을 DC LAMP(LED)와 연결하기 위한 재정의

#define PUMP 16 // D16 핀을 DC PUMP(PUMP)와 연결하기 위한 재정의

#define SOILHUMI A6 // 토양핀은 A6= A6에 연결

DHT\_Unified dht(DHTPIN, DHTTYPE); //dht 클래스 생성 DHT = 온습도 센서 생성

SOFTPWM\_DEFINE\_CHANNEL(A3); //A3핀연결 -> 이건 FAN에 연결된 위치

/\* 잠깐! PWM이란 0과 1 즉 0V와 5V 출력만 있는 디지털 출력으로 아날로그 출력과 같은 출력을 낼 수 있는 방법인데 이 0과 5의 주기를 조절하는데 아두이노 자세한건 공부 3번 참고! \*/

uint32\_t DataCaptureDelay = 4000; // ms단위 = 1/1000초 4000=>4초

uint32\_t StartTime1; // 시작시간 저장을 위한 변수

uint32\_t StartTime2; // 시작시간 저장을 위한 변수

uint32\_t StartTime3; // 시작시간 저장을 위한 변수

int Soilhumi = 0; // 토양 습도 저장을 위한 변수

float Temp = 0; // 온도 저장을 위한 변수

float Humi = 0; // 습도 저장을 위한 변수

int fanVal = 0; // 팬의 속도 저장을 위한 변수 (duty rate)

bool timeset = false; // 시간타이머 설정을 위한 변수

bool autoMode = false; // 오토모드를 위한 변수

// 이건 필요가 없는건가 확인해보자

bool soilstatus = true; // 토양 습도 저장을 위한 변수

bool tempstatus = false; // 토양 습도 저장을 위한 변수

bool lampflag = true; // 토양 습도 저장을 위한 변수

bool auto\_1\_execute = false; // 오토모드 실행하는지 변수

bool manu\_1\_execute = false; // 수동모드 실행하는지 변수

bool oledcnt = false; // 아두이노 oled 출력 on/off를 위한 변수

bool fan\_out\_status; // fan이 켜졌냐 꺼졌냐 저장을 위한 변수

bool pump\_out\_status; // pump가 켜졌냐 꺼졌냐 저장을 위한 변수

bool lamp\_out\_status; // lamp가 켜졌냐 꺼졋냐 저장을 위한 변수

bool Interval\_Minute\_Up\_status; // 토양 습도 저장을 위한 변수

bool Interval\_Hour\_Up\_status; // 토양 습도 저장을 위한 변수

uint32\_t Hour = 0; // 시간단위 저장 변수

uint32\_t Minute = 1; // 분 저장 변수

uint32\_t TimeSum = 0; // 시간의 총합

uint32\_t TimeStatus; //시간의 상태

/\* A set of definition for IOT items \*/

#define ITEM\_COUNT 18 // 우리가 만든 iot의 아이템 인덱스 (이름) 총 18개

//모드 변경을 위한 함수

void mode\_out(bool val) {

autoMode = val;

}

//Interval 설정 모드로 들어가기 위한 함수

void timeset\_out(bool val) {

timeset = val;

}

//Interval 시간 단위를 설정하는 함수

void Interval\_Hup(bool val) {

Interval\_Hour\_Up\_status = val;

}

//Interval 분 단위를 설정하는 함수

void Interval\_Mup(bool val) {

Interval\_Minute\_Up\_status = val;

}

//manual mode일 때 FAN을 제어하는 함수

void fan\_out(bool val) {

fan\_out\_status = val;

}

//manual mode일 때 PUMP를 제어하는 함수

void pump\_out(bool val) {

pump\_out\_status = val;

}

//manual mode일 때 LAMP를 제어하는 함수

void lamp\_out(bool val) {

lamp\_out\_status = val;

}

//Interval을 0시 0분으로 리셋하는 함수

void IntervalReset(bool val) {

if (!timeset && val) {

Hour = 0;

Minute = 0;

}

}

/\*widget toggle switch 온오프 그 스위치

Bin = bool 타입

Int = int 타입

Flo = float 타입

\*/

IOTItemBin ModeStatus; // ModeStatus가 true인가 false인가

IOTItemBin Mode(mode\_out); //Mode(mode\_out)의 값이 true인가 false인가

IOTItemBin StopStatus;

IOTItemBin Stop(timeset\_out);

IOTItemBin FanStatus;

IOTItemBin Fan(fan\_out);

IOTItemBin PumpStatus;

IOTItemBin Pump(pump\_out);

IOTItemBin LampStatus;

IOTItemBin Lamp(lamp\_out);

/\*widget push button 시간 설정 그 버튼\*/

IOTItemBin IntervalHUP(Interval\_Hup);

IOTItemBin IntervalMUP(Interval\_Mup);

IOTItemBin IntervalRST(IntervalReset);

/\*widget label 정보 출력 라벨 \*/

IOTItemInt label\_Hinterval;

IOTItemInt label\_Minterval;

IOTItemFlo dht22\_temp;

IOTItemFlo dht22\_humi;

IOTItemInt soilhumi;

IOTItem\* items[ITEM\_COUNT] = { &ModeStatus, &Mode,

&StopStatus, &Stop,

&FanStatus, &Fan,

&PumpStatus, &Pump,

&LampStatus, &Lamp,

//index num : 0 ~ 9

&IntervalHUP, &IntervalMUP, &IntervalRST,

&label\_Hinterval, &label\_Minterval,

&dht22\_temp, &dht22\_humi, &soilhumi,

//index num : 10 ~ 17

};

/\* IOT server communication manager \*/

const char device\_id[] = ""; // Change device\_id to yours

BrokerComm comm(&Serial, device\_id, items, ITEM\_COUNT); //이게 네트워크 설정 함수

void setup() {

Serial.begin(250000); //시리얼 통신 속도 설정

comm.SetInterval(200); //네트워크 속도 설정

pinMode(LAMP, OUTPUT); // 램프는 출력모드

pinMode(PUMP, OUTPUT); // 펌프도 출력모드

pinMode(SOILHUMI, INPUT); // 토양은 저항값을 받아야 하므로 입력모드

// pinMOde (a,b) a에는 19개의 디지털, 6개의 아날로그 핀중 어떤걸 선택할지 b가 output이면 출력, input이면 입력

//초기설정

// digitalWrite , digitalRead, analogWrite, analogRead

// digital - 디지털 핀, analog - 아날로그 핀

// read - 신호값 읽어오기, write - (HIGH) 5볼트를 줌 (LOW) 0볼트를 줌

digitalWrite(LAMP, LOW); // 램프 처음에는 꺼

digitalWrite(PUMP, LOW); // 펌프도 처음에는꺼

u8g2.begin(); // 글씨쓰는거 시작해

dht.begin(); // 온습도 체크도 시작해

StartTime1 = millis(); // 이제 처음 시작 시간

millis(); // 모드링크 보드가 현재 프로그램을 돌리기 시작한 후 지난 밀리 초 숫자 반환

// begin with 60hz pwm frequency

SoftPWM.begin(490); //490주기로 시작해

}

void loop() {

/\* 온습도 정보 가져오는 구성

\*/

if ((millis() - StartTime1) > DataCaptureDelay) {

// (시작후 지난시간 – 시작시간) > 4초 4초 간격으로 실행

//sensors\_event\_t 반환할 수 있는 다양한 유형의 센서 데이터들의 조합

sensors\_event\_t event1;

sensors\_event\_t event2;

dht.temperature().getEvent(&event1); //DHT22\_Temperature

Temp = event1.temperature;

dht.humidity().getEvent(&event2); //DHT22\_Humidity

Humi = event2.relative\_humidity;

Soilhumi = map(analogRead(SOILHUMI), 0, 1023, 100, 0); //soil humiditiy

// map에관한건 교재32p 참고

drawLogo();

StartTime1 = millis();

}

/\*Mode change\*/

if (!autoMode) { //수동모드일 때

auto\_1\_execute = false; //네트워크의 auto\_1\_execute는 false

ManualMode(); // 수동모드를 실행해라

}

if (autoMode) { //자동모드일 때

manu\_1\_execute = false; //네트워크의 auto\_1\_execute는 false

AutoMode();// 자동모드를 실행해라

}

/\* Interval time set \*/

if (timeset && autoMode) //Auto모드에서 시간설정 스위치가 ON일 때

{

TimeStatus = (millis() - StartTime2) / TimeSum; // 지난시간 – 자동화시점 시간/ 지금설정해놓은시간 = 켜야하는 시간

}

Else // 수동 모드에서는

{

TimeSum = (Hour \* 60 + Minute) \* 60 \* 1000; //ms단위로 변환

StartTime2 = millis(); //지속적으로 타임 초기화

if (millis() > StartTime3 + 500) //위젯 버튼 누르는 시간 딜레이주기 0.5초

{

Hour += Interval\_Hour\_Up\_status;

if (Hour >= 24) Hour = 0;

Minute += Interval\_Minute\_Up\_status;

if (Minute >= 60) Minute = 0;

StartTime3 = millis();

}

}

SoftPWM.set(fanVal); //fan값을 설정해 네트워크에 넘겨준다

PumpStatus.Set(digitalRead(PUMP));// 펌프 값을 네트워크에 넘겨준다

LampStatus.Set(digitalRead(LAMP)); ));// led 값을 네트워크에 넘겨준다

if (fanVal > 60) FanStatus.Set(true); ));// fan이 60이상의 비율로 돌아가면 돌아가라고 전해

else FanStatus.Set(false); ));// 아님 안돌아간다고 전해

ModeStatus.Set(autoMode); // 모드상태(자동/수동)를 계속 넣어주자

StopStatus.Set(timeset); //시간 상태를 계속 넣어주자

label\_Hinterval.Set(Hour); ); //시간을 계속 넣어주자

label\_Minterval.Set(Minute); ); //분을 계속 넣어주자

dht22\_temp.Set(Temp);// 온도를 전해주자

dht22\_humi.Set(Humi);// 습도를 전해주자

soilhumi.Set(Soilhumi);// 토양 습도를 전해주자

comm.Run(); // 네트워크는 계속 달려!

}

void ManualMode() { // 오토 모드일 때 함수

if (fan\_out\_status == true) // 팬이 돌아가라는 상태라면 팬의 속도는 65

{

fanVal = 65;

}

Else // 아니면 0

{

fanVal = 0;

}

digitalWrite(PUMP, pump\_out\_status); // 이신호를 펌프에게 전달! On or off

digitalWrite(LAMP, lamp\_out\_status); // 이신호를 LED에게 전달! On or off

}

void AutoMode() { //

/\* a LAMP auto control 대략 1분 간격으로 LED 조명이 ON/OFF된다\*/

if (timeset) // {

if (TimeStatus % 2 == 1) { //TimeSatus값이 2배수가 아니면 꺼

digitalWrite(LAMP, LOW);

}

else {//그외에는 켜!

digitalWrite(LAMP, HIGH);

}

}

else if (!timeset) {

digitalWrite(LAMP, LOW);

}

/\* a pump auto control \*/

if (Soilhumi <= 30) { //토양 습도값이 30%이하일 때

digitalWrite(PUMP, HIGH);

}

else if (Soilhumi >= 60) { //토양 습도값이 60%이상일 때

digitalWrite(PUMP, LOW);

}

/\* a fan auto control \*/

if (Temp >= 25) { // 화분 주변 온도가 30\*C이상일 때

fanVal = 65;

}

else if (Temp <= 20) { // 화분 주변 온도가 28\*C이하일 때

fanVal = 0;

}

}

void drawLogo()

{

oledcnt = !oledcnt; //화면 전환

if (!oledcnt) {

u8g2.clearBuffer();

u8g2.setFontMode(1); // Transparent

u8g2.drawXBM(0, 0, LOGO\_WIDTH, LOGO\_HEIGHT, LOGO);

u8g2.sendBuffer();

}

else if (oledcnt) {

u8g2.clearBuffer();

u8g2.setFont(u8g2\_font\_ncenB08\_te);

u8g2.drawStr(1, 15, " SMART FARM");

u8g2.drawStr(15, 36, "Temp.");

u8g2.setCursor(85, 36); u8g2.print(Temp);

u8g2.drawStr(114, 36, "\xb0");

u8g2.drawStr(119, 36, "C");

u8g2.drawStr(15, 47, "Humidity");

u8g2.setCursor(85, 47); u8g2.print(Humi);

u8g2.drawStr(116, 47, "%");

u8g2.drawStr(15, 58, "Soil Humi.");

u8g2.setCursor(85, 58); u8g2.print(Soilhumi);

u8g2.drawStr(116, 58, "%");

u8g2.sendBuffer();

}

}