УДК 636.4.087.61

Студент **Д. А. Тимофеев**

(ФГБОУ ВО СПбГАУ)
Канд. техн. наук **Ю.В. Иванов**

(ФГБОУ ВО СПбГАУ)

**Устройство измерения температуры при создании микроклимата в теплице на базе микроконтроллерной платы «Ардуино».**

Теплица - это сооружение, имеющее светопропускающие стены и кровлю и предназначенное для выращивания различных растений в весенне-осенний период, когда погодные условия не позволяют снимать несколько урожаев за один год.

Для каждого растения существует оптимальная температура выращивания. На температуру в теплице большое влияние имеет солнце. В солнечный день теплица нагревается из-за большого тепла от солнца, что плохо влияет на растения находящиеся в данной теплице. Для того чтобы вернуть температуру в норму можно использовать устройство автоматического проветривание и нагрева, к сожалению в наше время установка проветривание основана на определенных температурных критериях. Например, «Дуся сан», белорусского производства, открывает проветривание в приделах 15-25 градусов, что может плохо сказаться на определенных растениях в связи с тем, что нет точного контроля температуры. Для этого более выгодно использовать систему микроклимата, позволяющую уравновесить температурный режим в пределах необходимого значения.

Для примера данной системы может служить система микроклимата на базе микроконтроллерной платы «Ардуино».



 Основным элементом которой является цифровой датчик температуры DS18B20. Данный датчик измеряет температуру в теплице и передает ее на ардуино После чего по средствам программы производится перевод данных в градусы Цельсия или Фаренгейта, а затем производится анализ полученных значений температуры с необходимым параметром данной теплицы. Если температура находится выше необходимой, то программа передает команду на открытие вентиляции. Если же температура ниже необходимой, то программа отправляет команду на обогрев.

Программа :

#include <Wire.h> // Подключаем библиотеку Wire

#include <LiquidCrystal\_I2C.h> // Подключаем библиотеку LiquidCrystal\_I2C

#include <OneWire.h> // Подключаем библиотеку

OneWire ds(2); // Указываем, к какому выводу подключена DQ

byte simvol[8] = {B11100,B10100,B11100,B00000,B00000,B00000,B00000,B00000,}; // Символ градуса

 LiquidCrystal\_I2C lcd(0x27,16,2); // Задаем адрес и размер дисплея

 byte type\_s;

 byte i;

 byte present = 0;

 byte data[12];

 byte addr[8];

 int red = 8;

 int blu = 9;

 void setup()

{ pinMode(red, OUTPUT);

 pinMode(blu, OUTPUT);

 lcd.begin(); // Инициализация lcd

 lcd.setCursor(0,0); // Устанавливаем курсор на 1 строку, ячейка 2

 lcd.print("TEMP ="); // Выводим текст

}

void loop(void) {

 byte i;

 byte present = 0;

 byte type\_s;

 byte data[12];

 byte addr[8];

 float celsius, fahrenheit;

 if ( !ds.search(addr)) {

 ds.reset\_search();

 delay(250);

 return;

 } // the first ROM byte indicates which chip

 ds.reset();

 ds.select(addr);

 ds.write(0x44, 1); // start conversion, with parasite power on at the end

 delay(1000); // maybe 750ms is enough, maybe not

 // we might do a ds.depower() here, but the reset will take care of it.

 present = ds.reset();

 ds.select(addr);

 ds.write(0xBE); // Read Scratchpad

 for ( i = 0; i < 9; i++) { // we need 9 bytes

 data[i] = ds.read();

 }

 // Convert the data to actual temperature

 // because the result is a 16 bit signed integer, it should

 // be stored to an "int16\_t" type, which is always 16 bits

 // even when compiled on a 32 bit processor.

 int16\_t raw = (data[1] << 8) | data[0];

 if (type\_s) {

 raw = raw << 3; // 9 bit resolution default

 if (data[7] == 0x10) {

 // "count remain" gives full 12 bit resolution

 raw = (raw & 0xFFF0) + 12 - data[6];

 }

 } else {

 byte cfg = (data[4] & 0x60);

 // at lower res, the low bits are undefined, so let's zero them

 if (cfg == 0x00) raw = raw & ~7; // 9 bit resolution, 93.75 ms

 else if (cfg == 0x20) raw = raw & ~3; // 10 bit res, 187.5 ms

 else if (cfg == 0x40) raw = raw & ~1; // 11 bit res, 375 ms

 //// default is 12 bit resolution, 750 ms conversion time

 }

 celsius = (float)raw / 16.0;

 fahrenheit = celsius \* 1.8 + 32.0;

 lcd.createChar(1, simvol); // Создаем символ под номером 1

 lcd.setCursor(7,0); // Устанавливаем курсор на 1 строку, ячейка 7

 lcd.print(celsius); // Выводим показания температуры

 lcd.setCursor(12,0); // Устанавливаем курсор на 1 строку, ячейка 12

 lcd.print("\1"); // Выводим символ градуса

 lcd.setCursor(13,0); // Устанавливаем курсор на 1 строку, ячейка 13

 lcd.print("C");

 lcd.setCursor(7,1);// Выводим текст lcd.setCursor(7,0); // Устанавливаем курсор на 1 строку, ячейка 7

 lcd.print(fahrenheit); // Выводим показания температуры

 lcd.setCursor(12,1); // Устанавливаем курсор на 1 строку, ячейка 12

 lcd.print("\1"); // Выводим символ градуса

 lcd.setCursor(13,1); // Устанавливаем курсор на 1 строку, ячейка 13

 lcd.print("F");

 if (celsius<25){

 digitalWrite(blu, HIGH);

 }

 else{digitalWrite(blu, LOW);

 if (celsius>27){

 digitalWrite(red, HIGH);

 }

 else{

 digitalWrite(red, LOW);

 digitalWrite(blu, LOW);

 }

}}

**Литература:**

1. Соммер У. Программирование микроконтроллерных плат Arduino/Freeduino // БХВ-Петербург- СПБ, 2012 — 256 с. ил - (Электроника)
2. Петли В. А. Проекты е использованием контроллера Arduino. **/**/ БХВ-Петербург- СПБ, 2015 — 464 с. ил — (Электроника)
3. С. Монк Программируем Arduino. Профессиональная работа со скетчами // ООО Издательство «Питер» СПб.:, 2017 — 252 с.
4. Блум Джереми Изучаем Arduino: инструменты и методы технического волшебства// Перевод с английского — БВХ-Петербург — Спб, 2015-336 с. ил - (Электроника)