ФЕДЕРАЛЬНОЕ АГЕНТСТВО ЖЕЛЕЗНОДОРОЖНОГО ТРАНСПОРТА
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Иркутский государственный университет путей сообщения»
Сибирский колледж транспорта и строительства

Методические указания по выполнению лабораторных работ по дисциплине

МДК.02.01. Микропроцессорные системы

профессионального модуля
ПМ.02 Применение микропроцессорных систем, установка и настройка периферийного оборудования

для специальности
09.02.01 Компьютерные системы и комплексы

Иркутск 2022

РАССМОТРЕНО:
Цикловой методической
комиссией специальности 09.02.01
Компьютерные системы и комплексы
«08» июня 2022 г.
Председатель:_____/Арефьева Н.В.

СОГЛАСОВАНО:
Заместитель директора по УВР
_____/А.П.Ресельс
«09» июня 2022 г.

Разработчик : Подгорнов С.В., преподаватель СКТиС

Оглавление

Предисловие

Сборник содержит задания для лабораторных работ, имеющих целью более глубокое изучение дисциплины, систематизацию и закрепление полученных знаний и практических умений. Он предназначен для углубления и расширения теоретических и практических знаний; формирования умений использовать специальную, справочную литературу. В нем содержатся методические указания по выполнению предложенных заданий и список литературы, необходимой для изучения дисциплины.

Методические указания к выполнению лабораторных работ

- К выполнению работы необходимо подготовиться до начала занятия в лаборатории.
- Помимо данного методического пособия рекомендуется использовать дополнительную литературу и конспект лекций.
- При подготовке необходимо продумывать ответы на контрольные вопросы.
- К выполнению работы допускаются только подготовленные студенты.

Правила оформления отчета о лабораторной работе

Лабораторная работа представляет собой небольшое, но вполне законченное учебное исследование. Отчет о лабораторной работе является документом, отражающим результаты выполненного исследования с максимальной полнотой и объективностью.

К оформлению технической документации предъявляются единые требования. В определенной мере этим требованиям должен удовлетворять и отчет о лабораторной работе.

Требования к оформлению отчета

Отчет должен быть выполнен на бумаге стандартного размера (формат А4) с полями по обеим сторонам текста. Материал отчета должен иметь четкую рубрикацию, каждый раздел необходимо снабдить заголовком.

Примерный состав отчета по лабораторной работе:

цель работы;

порядок выполнения лабораторной работы;

принципиальные электрические схемы и (или) схемы соединений;

перечень приборов и оборудования с указанием;

таблицы экспериментальных исследований и выполненных вычислений;

диаграммы и графики характеристик функциональных зависимостей;

выводы или оформленный результат проделанной работе.

ПРАВИЛА ТЕХНИКИ БЕЗОПАСНОСТИ ПРИ ВЫПОЛНЕНИИ ЛАБОРАТОРНЫХ РАБОТ

В начале семестра каждый студент ДОЛЖЕН внимательно знакомиться с настоящими правилами и расписаться в журнале учета инструктажа по технике безопасности.

Студент ОБЯЗАН выполнять следующие правила:

1. При сборке цепи использовать провода с исправной изоляцией.

2. Включать источники питания (подавать напряжение на схему) только после проверки цепи преподавателем.
3. При проведении любых изменений в схеме отключайть источник питания.
4. Отключить питание по завершению измерений.
5. Курить в лаборатории, находиться в верхней одежде или головных уборах.

По всем возникающим вопросам студентам следует обращаться к преподавателю. За порчу оборудования студенты несут материальную ответственность.

Указания к оцениванию практических работ

Оценивание индивидуальных образовательных достижений по результатам выполнения практических работ производится в соответствии с универсальной пятибалльной шкалой.

Процент результативности (правильных ответов)	Качественная оценка индивидуальных образовательных достижений	
	балл (отметка)	вербальный аналог
90 – 100	5	отлично
80 – 89	4	хорошо
70 – 79	3	удовлетворительно
менее 70	2	неудовлетворительно

Количество часов на освоение лабораторных занятий – 6 часов

Лабораторная работа №1

«Отладка прикладного программного обеспечения микроконтроллеров»

ЦЕЛЬ РАБОТЫ: формировать навыки практической работы по отладке ПО, анализу и нахождению ошибок в тексте программы а также их корректному устранению.

Выполнение данной работы способствует формированию следующих компетенций: ОК-6, ОК-7, ОК-8, ОК-4, ПК 2.2, ПК 2.4

Бюджет времени выполнения работы - 6 часов.

Ход работы

1*) Повторение теоретических основ (в парах, взаимопроверка).*

- Что такое внешняя библиотека и способы ее подключения (пример).
- Как оформить и применить макрокоманду в тексте программы (пример).
- Указать способы оформления комментариев.

2) Практическая часть

- запустить отладчик аппаратной платформы Arduino (язык C/C++) и загрузить скетч программы паяльной станции solderingstation.ino, в который намеренно внесены ошибочные строки.
- Прокомментировать работу программы, используя описание функций работы жидкокристаллического экрана LiquidCrystal.h и стандартные ф-и.
- Найти и исправить обнаруженные ошибки (*Verify/Compile*).

***3)* Оформить результаты и защитить работу.**

Перечень учебных изданий, Интернет-ресурсов, дополнительной литературы

Основные источники:

1. Микропроцессорная техника А.В. Кузин, М.А. Жаворонков. М: Академия 2013г 7изд стер

Дополнительные источники:

1. **Цифровые устройства и микропроцессорные системы Б. А. Калабеков М.: Просвещение, 2008.**

Программное обеспечение и Интернет-ресурсы:

1. Интернет-университет информационных технологий (ИНТУИТ.ру): www.intuit.ru
2. Алгоритмизация и программирование : Учебное пособие / С.А. Канцедал. - М.: ИД ФОРУМ: НИЦ ИНФРА-М, 2013. - 352 с.: ил.; 60x90 1/16. - (Профессиональное образование). (переплет) ISBN 978-5-8199-0355-1(электронная библиотечная сеть ZNANIUM.COM)
3. http://www.knigafund.ru/books/106073

4. http://www.ozon.ru/context/detail/id/17981415/
5. http://madelectronics.ru/book/shemotehnika/index-2.htm
6. http://madelectronics.ru/book/shemotehnika/index-2.htm
7. Микропроцессорные системы (Электронный ресурс) : электрон. учеб. пособие / О. В. Непомнящий, Е. А. Вейсов, Г. А. Скотников, М. В. Савицкая.– Красноярск : ИПК СФУ, 2009.

Приложение 1

Необходимые сведения об Ардуино

Язык программирования для аппаратной платформы Arduino является стандартным C++ (используется компилятор AVR-GCC) с некоторыми особенностями, облегчающими новичкам написание первой работающей программы:

− программы, написанные программистом Arduino называются скетчи (от англ. Sketch- набросок) и сохраняются в файлах с расширением ino. Эти файлы перед компиляцией обрабатываются препроцессором Ардуино. Также существует возможность создавать и подключать к проекту стандартные файлы C++;

− программист должен написать две обязательные для Arduino функции setup() и loop(). Первая вызывается однократно при старте, вторая выполняется в бесконечном цикле;

− в текст своей программы (скетча) программист не обязан вставлять заголовочные файлы стандартных библиотек, их добавит препроцессор Arduino в соответствии с конфигурацией проекта. Но пользовательские библиотеки нужно указывать;

Приложение 1

Пользовательская библиотека LiquidCrystal для Arduino (работа с жидкокристаллическим дисплеем, извлечения)

Библиотека LiquidCrystalсодержит необходимые функции управления жидкокристаллическим дисплеем, созданным на базе контроллеров, совместимых с Hitachi 44780. В число этих дисплеев входит большое количество популярных алфавитно-цифровых устройств, для которых в библиотеке содержатся высокоуровневые функции инициализации, управления и вывода информации на экран.

LiquidCrystal поддерживает работу как в 8-ми, так и в 4-х проводном режиме обмена данными.

Подключение библиотеки и дисплея

Использование данной библиотеки стандартно для платформы Arduino. Перед первым вызовом любых функций необходимо прописать режим, в котором работает дисплей и номера используемых выводов Arduino. Могут быть использованы любые цифровые вывода, в произвольном порядке. Этот факт значительно повышает универсальность библиотеки и позволяет использовать устройства индикации, совместно с другой периферией в уже готовых проектах.

Для 4-х проводного режима обмена данными объявление библиотеки управления жидкокристаллическим дисплеем выглядит следующим образом:

LiquidCrystal lcd(RS, E, D4,D5,D6,D7);

Здесь lcd – имя, по которому производится вызов функций управления дисплеем. Параметрами являются номера выводов Arduino, к которым подключены соответствующие линии ЖК-индикатора:

- RS– выбор регистра адрес/данные
- E– разрешение чтения с шины
- D4-D7 – 4 линии шины данных индикатора

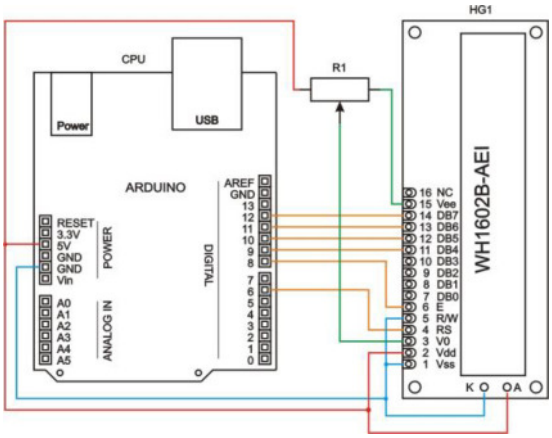


Схема подключения ЖК-дисплея к Arduino

После объявления имени используемого дисплея проводится инициализация дисплея.

lcd.begin(Ch,S, ChSz);

Данная функция выполняет инициализацию дисплея, а также задает количество строк S и символов в строке Ch. Также можно задать размер шрифта ChSz. По умолчанию последний равен 5x8.

Функции библиотеки LiquidCrystal

- **Вывод на дисплей**

lcd.print();

Функция отображает на дисплее произвольную информацию, начиная с текущего положения курсора. В качестве аргумента можно использовать текстовую строку или переменную.

- **Установка положения курсора**

lcd.setCursor(Pos,Str);

Устанавливает курсор ЖК-дисплея в позицию Pos, строки Strдисплея.

- **Установка нулевого положения курсора**

lcd.home();

Устанавливает курсор дисплея в нулевую позицию, строки 0 дисплея.

- **Очистка дисплея**

lcd.clear();

Стирает всю информацию с экрана

- **Включение/выключение дисплея**

lcd. Display();

lcd. noDisplay();

- **Включение/выключение курсора**

lcd. Cursor();

lcd. noCursor();

- **Включение/выключение мигания курсора**

lcd. blink();

lcd. noblink();

- **Сдвиг информации на дисплее**

Влево **lcd. scrollDisplayLeft();**

Вправо **lcd. scrollDisplayRight();**

- **Установка направления вывода текстовой строки**

Слева направо **lcd.leftToRight();**

Справа налево **lcd. rightToLeft();**

- **Создание произвольного символа**

Lcd. createChar(Adr, CH[]);

Команда записывает в память дисплея произвольный символ. Запись выполняется по одному из первых 8 адресов (от 0 до 7) в памяти знакогенератора. Символ CH[] должен представлять собой массив из 8 байт. Для задания пикселей, в каждом байте используется только 8 младших бит.

Пример использования библиотеки

```
// Подключение библиотеки
#include <LiquidCrystal.h>

LiquidCrystal lcd( 6, 8, 12, 11, 10, 9); // Объявление дисплея

void setup() {
lcd.begin(16, 2); // Инициализация ЖК-дисплея
lcd.print("hello, world!"); // Вывод на дисплей
lcd.rightToLeft(); // Установка обратного порядка вывода
lcd.setCursor(12, 1); // Установка начала вывода (строка 2 поз.12)
```

```
lcd.print("hello, world!"); // Вывод на дисплей
}

// Бесконечный цикл

void loop() {
}
```

Результат работы программы

Приложение 2

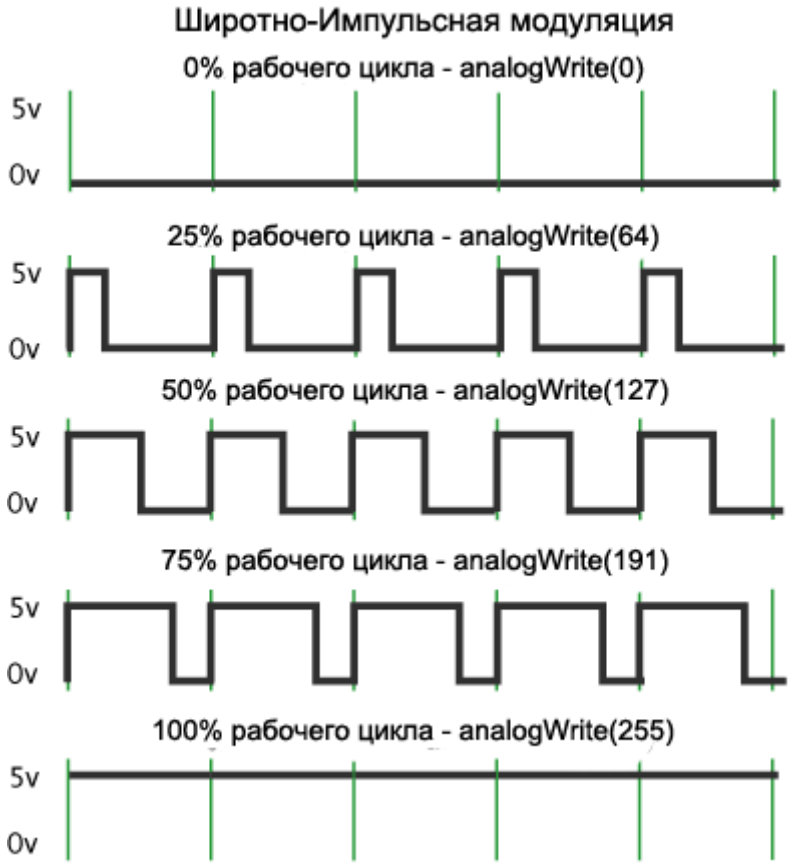
Функции стандартной библиотеки

Широтно-импульсная модуляция (ШИМ)

ШИМ- это операция получения изменяющегося аналогового значения посредством цифровых устройств. Устройства используются для получения прямоугольных импульсов - сигнала, который постоянно переключается между максимальным и минимальным значениями. Данный сигнал моделирует напряжение между максимальным значением (5 В) и минимальным (0 В), изменяя при этом длительность времени включения 5 В

относительно включения 0 В. Длительность включения максимального значения называется шириной импульса. Для получения различных аналоговых величин изменяется ширина импульса. При достаточно быстрой смене периодов включения-выключения можно подавать постоянный сигнал между 0 и 5 В на светодиод, тем самым управляя яркостью его свечения.

На графике зеленые линии отмечают постоянные временные периоды. Длительность периода обратно пропорциональна частоте ШИМ. Т.е. если частота ШИМ составляет 500 Гц, то зеленые линии будут отмечать интервалы длительностью в 2 миллисекунды каждый. Вызов функции analogWrite() с масштабом 0 – 255 означает, что значение analogWrite(255) будет соответствовать 100% рабочему циклу (постоянное включение 5 В), а значение analogWrite(127) – 50% рабочему циклу.



Для примера можно взять платформу и начать трясти ее взад и вперед. Для наших глаз данное движение превращает в светящиеся линии мигание светодиода. Наращивание или уменьшение ширины импульса на светодиоде будет увеличивать или уменьшать светящиеся линии светодиода. Теперь вы знаете что такое ширина импульса.

Пример

Управление яркостью светодиода

В секции *setup()* кода устанавливаем режим выхода для вход/выхода 9 (pin 9).

Для управление яркостью светодиода мы будем изменять значение ширины импульса, передаваемое в функцию *analogWrite()*. 0- светодиод выключен, при 255 светодиод светит на полную яркость. Функция *analogWrite()*, которая циклически вызывается в теле скетча, принимает два аргумента: номер выхода и значение ширины импульса ШИМ в диапазоне

от 0 до 255 (переменная *brightness)*. Шаг изменения этого значения задан переменной *fadeAmount.*Для плавного изменения яркости мы вводим задержку в конце главного цикла (тела) скетча — *delay(30).*

```
/* Пример управления яркостью светодиода
 на выходе 9 контроллера Arduino
 функцией analogWrite().
 */
int brightness = 0;    // уставливаем начально значение яркости
int fadeAmount = 5;    // шаг приращения/убывания яркости

void setup()  {
  // устанваливаем пин 9 в режим выхода
  pinMode(9, OUTPUT);
}

void loop()  {
  // устанавливаем значение широты импульса на выходе 9
  // задавая яркость светодиода
  analogWrite(9, brightness);

  // измением значение в переменной для яркости
  brightness = brightness + fadeAmount;

  // при достижение крайних значений для яркости
  // меняем знак переменной шага приращения/убывания яркости
  if (brightness == 0 || brightness == 255) {
    fadeAmount = -fadeAmount ;
  }
  // делаем паузу для достижения плавного наращивания/убывания яркости
  delay(30);
}
```

map(value, fromLow, fromHigh, toLow, toHigh)

Функция пропорционально переносит значение (**value**) из текущего диапазона значений (**fromLow** .. **fromHigh**) в новый диапазон (**toLow** .. **toHigh**), заданный параметрами. Функция возвращает значение в новом диапазоне

Пример

```
/* Переносим значение с аналогового входа (возможные значения от 0 до 1023) в 8 бит (0..255) */
void setup() {}

void loop()
{
 int val = analogRead(0);
 val = map(val, 0, 1023, 0, 255);
 analogWrite(9, val);
}
```

Функция analogRead(*pin*)

Функция считывает значение с одно из 6 каналов аналогового входа.
Напряжение поданное на аналоговый вход (обычно от 0 до 5 вольт) будет преобразовано в значение от 0 до 1023, это 1024 шага с разрешением 0.0049 Вольт.
pin: номер порта аналогового входа

Возвращаемое значение

int (0 to 1023)

Пример

```
int analogPin = 3;     // номер порта к которому подключен потенциометр
int val = 0;           // переменная для хранения считываемого значения

void setup()
{
  Serial.begin(9600);              //  установка связи по serial
}

void loop()
{
  val = analogRead(analogPin);     // считываем значение
  Serial.println(val);             // выводим полученное значение
}
```

Функция digitalRead(*pin*)

Функция считывает значение с заданного входа - HIGH или LOW.

Возвращаемое значение

HIGH или LOW

Пример

```
int ledPin = 13;                // Светодиод подключенный к вход/выходу 13
int inPin = 7;                  // кнопка на входе 7
int val = 0;                    // переменная для хранения значения

void setup()
{
  pinMode(ledPin, OUTPUT);       // устанавливает режим работы - выход для 13го
вход/выхода (pin)
  pinMode(inPin, INPUT);         //  устанавливает режим работы - вход для 7го вход/выхода
(pin)
}
void loop()
{
  val = digitalRead(inPin);      // считываем значение с входа
  digitalWrite(ledPin, val);     // устанавливаем значение на светодиоде равным значению
входа кнопки
}
```

Функция pinMode(*pin, mode)*

Устанавливает режим работы заданного вход/выхода(pin) как входа или как выхода.

Параметры

- pin: номер вход/выхода(pin), который Вы хотите установить
- mode: режим одно из двух значение - INPUT или OUTPUT, устанавливает на вход или выход соответственно.

Пример

```
int ledPin = 13;              // Светодиод, подключенный к вход/выходу 13
void setup()
{
  pinMode(ledPin, OUTPUT);      // устанавливает режим работы - выход
}

void loop()
{
  digitalWrite(ledPin, HIGH);   // включает светодиод
  delay(1000);                  // ждет секунду
  digitalWrite(ledPin, LOW);    // выключает светодиод
  delay(1000);                  // ждет секунду
}
```

Приложение 3

solderingstation.ino

```
/*

 ПАЯЛЬНАЯ СТАНЦИЯ ver. 0.5


 Дата создания 2015

 Автор Олег Андреев

 d-serviss@inbox.lv

 http://www.d-serviss.lv


 */


#include <LiquidCrystal.h>

LiquidCrystal lcd(13, 12, 11, 10, 9, 8);  // Выходы для дисплея 1602


int pinSolderOut = 5;  // Выход для паяльника
```

```
int pinSolderIn = A4;  // Потенциометр паяльника
int pinSolderTCouple = A3;  // Термопара паяльника
int pinSolderButton = 2;  // Кнопка вкл. и выкл. паяльника

int pinHotAirOut = 6;  // Выход для фена
int pinHotAirIn = A2;  // Потенциометр фена
int pinHotAirTCouple = A1;  // Термопара фена
int pinHotAirCoolerOut = 3;  // Выход для вентилятора фена ( PWM )
int pinHotAirCoolerIn = A0;  // Потенциометр вентилятора фена
int pinHotAirButton = 4;  // Кнопка вкл.и выкл. фена

uint8_t char_cel[8] = {
  B00111, B00101, B00111, B00000, B00000, B00000, B00000
};

void setup()
{
  TCCR2B = TCCR2B & 0b11111000 | 0x01; // Частота ШИМ 11 и 3
  pinMode(pinSolderOut, OUTPUT);
  pinMode(pinSolderButton, INPUT);
  pinMode(pinHotAirOut, OUTPUT);
  pinMode(pinHotAirButton, INPUT);
  lcd.begin(16, 2);
  lcd.createChar(1, char_cel);

  // Вывод приветствия
  lcd.setCursor(0, 0);
  lcd.print("SOLDER STATION");
  lcd.setCursor(0, 1);
  lcd.print("ver. 0.5");
  // задержка
  delay (3000);
  lcd.clear();
```

```
}

void loop()
{
  // Преобразовываем значения
  int setSolderTemp = map(analogRead(pinSolderIn), 0, 1023, 0, 480);
  int solderTCouple = map(analogRead(pinSolderTCouple), 0, 750, 0, 480);
  int setHotAirTemp = map(analogRead(pinHotAirIn), 0, 1023, 0, 480);
  int hotAirTCouple = map(analogRead(pinHotAirTCouple), 0, 750, 0, 480);
  int setHotAirCooler = map(analogRead(pinHotAirCoolerIn), 0, 1023, 130, 255);
  int displayHotAirCooler = map(analogRead(pinHotAirCoolerIn), 0, 1023, 0, 99);

  // Защита, если не работает термопара
  if (solderTCouple > 480) {
    setSolderTemp = 0;
  }
  if (hotAirTCouple > 480) {
    setHotAirTemp = 0;
  }

  // Поддержка установленной температуры паяльника
  if (setSolderTemp >= solderTCouple && digitalRead(pinSolderButton) == HIGH)
  {
    digitalWrite(pinSolderOut, LOW);
    // delay(100);
    digitalWrite(pinSolderOut, HIGH);
  }
  else {
    digitalWrite(pinSolderOut, LOW);
  }

  // Поддержка установленной температуры фена
  if (setHotAirTemp >= hotAirTCouple && digitalRead(pinHotAirButton) == HIGH)
```

```
{
  digitalWrite(pinHotAirOut, HIGH);
  delay(100);
  digitalWrite(pinHotAirOut, LOW);
}
else {
  digitalWrite(pinHotAirOut, LOW);
}

// Установка оборотов вентилятора фена
if (hotAirTCouple < 85 && digitalRead(pinHotAirButton) == LOW) {
  analogWrite(pinHotAirCoolerOut, 0);
}
else {
  analogWrite(pinHotAirCoolerOut, setHotAirCooler);
}

// Убираем прыганье цифр
if ((setSolderTemp + 10) > solderTCouple && (setSolderTemp - 10) < solderTCouple) {
  solderTCouple = setSolderTemp;
}
if ((setHotAirTemp + 10) > hotAirTCouple && (setHotAirTemp - 10) < hotAirTCouple) {
  hotAirTCouple = setHotAirTemp;
}

// Данные паяльника на дисплей
lcd.clear();
lcd.print("Solder:");
lcd.setCursor(7, 0);
if (digitalRead(pinSolderButton) == HIGH) {
  if (solderTCouple < 480) {
    lcd.print(setSolderTemp);
    lcd.print("\1");
```

```
    lcd.setCursor(12, 0);
    lcd.print(solderTCouple);
    lcd.print("\1");
  }
  else {
    lcd.print(" Error");
  }
}
else {
  lcd.print(" Off");
}
// Данные фена на дисплей
lcd.setCursor(0, 1);
lcd.print("Smd:");
lcd.setCursor(4, 1);
if (digitalRead(pinHotAirButton) == HIGH) {
  if (hotAirTCouple < 480) {
    lcd.print(setHotAirTemp);
    lcd.print("\1");
    lcd.setCursor(9, 1);
    lcd.print(hotAirTCouple);
    lcd.print("\1");
    // Вентилятор фена на дисплей
    lcd.setCursor(14, 1);
    lcd.print(displayHotAirCooler);
  }
  else {
    lcd.print(" Error");
  }
}
else {
  lcd.print(" Off");
}
```

```
  delay(100);
}
```

Список использованной литературы

Основные источники:

1. Микропроцессорная техника А.В. Кузин, М.А. Жаворонков. М: Академия 2013г 7изд стер

Дополнительные источники:

2. Цифровые устройства и микропроцессорные системы Б. А. Калабеков М.: Просвещение, 2008.

Российские журналы:

Прикладная информатика: научно-практический журнал.- ЭБС znanium.com Договор № 4220 эбс от 09.01.2020 г.

Компоненты и технологии. - ЭбС Университетская библиотека Договор ЭБС znanium.com Договор № 4220 эбс от 09.01.2020 г.

Мир ПК. - ЭБС znanium.com Договор № 4220 эбс от 09.01.2020 г.

1.