УДК 004.032 **DOI 10.52928/2070-1624-2022-39-11-16-20**

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ БЛОКА УПРАВЛЕНИЯ ЛАБОРАТОРНОГО СТЕНДА ПО МОДЕЛИРОВАНИЮ ТЕПЛОВЫХ РЕЖИМОВ РАДИОЭЛЕКТРОННЫХ СРЕДСТВ

Т. С. КАПАЧ, М. И. ХАМИЧЕНОК, канд. техн. наук, доц. Т. В. МОЛОДЕЧКИНА
(Полоцкий государственный университет имени Евфросинии Полоцкой)

Представлен блок управления, входящий в лабораторный стенд для моделирования тепловых режимов радиоэлектронных средств. Приведены основные функции, выполняемые блоком управления лабораторного стенда, описаны особенности его конструкции. Обоснован выбор программной среды Arduino IDE для разработки программного обеспечения. Проведен выбор библиотек, использующихся для организации взаимодействия между платой Arduino и компонентами стенда. Разработано и детально рассмотрено программное обеспечение для блока управления лабораторного стенда.

Ключевые слова: *тепловой режим, блок управления, программное обеспечение, среда Arduino IDE, программный код.*

Введение. В настоящее время большинство электронной аппаратуры представляет собой программно-управляемые электронные средства (ПУЭС), которые сконструированы на основе микроконтроллеров и микропроцессоров. Такое конструкторское решение позволяет не только уменьшить размеры и вес аппаратуры, но и расширить область ее применения, сделать ее работу более точной и надежной. Однако таким ПУЭС необходимо встроенное программное обеспечение (ПО) – список инструкций, в соответствии с которыми ПУЭС будет функционировать и осуществлять необходимые действия [1].

В рамках данной статьи будут рассмотрены особенности разработки ПО для блока управления лабораторного стенда по моделированию тепловых режимов радиоэлектронных средств.

Функции, выполняемые блоком управления лабораторного стенда. Блок управления должен измерять температуру исследуемых резисторов, расположенных непосредственно в стенде, посредством датчиков измерения температуры на каждом из 6 резисторов и выводить показания температуры по каждому из них на дисплей в течение всего времени работы лабораторного стенда. Вывод показаний температуры должен производиться циклически, т. е. после вывода температуры, измеряемой датчиком № 6, следующими должны вновь выводиться температуры датчиков № 1, № 2 и т. д. [2].

Особенности конструкции блока управления лабораторного стенда. Блок управления представляет собой классическое ПУЭС, спроектированное на базе платформы Arduino, в котором роль программируемого микроконтроллера (МК) выполняет плата Arduino Nano v.3. Выбор платформы Arduino обусловлен не только широкими возможностями и гибкостью применения, но и простотой разработки и отладки программного обеспечения. Прошивка платы осуществляется через разъем USB Type Mini, присутствующий на плате по умолчанию, посредством подключения платы соответствующим кабелем к компьютеру. Запуск прошивки платы можно инициировать нажатием одной кнопки непосредственно в среде для разработки и отладки ПО.

Для измерения температуры используются датчики DS18B20[1]. Выбор данного датчика обусловлен его относительной дешевизной, возможностью измерения температур в широком диапазоне: от минус 55 до 125 °С при погрешности измерения не более 0,5 °С, что является приемлемым в случае данного ПУЭС.

Для вывода информации используется жидкокристаллический двухстрочный дисплей LCD1602[2]. Этот дисплей часто использовался в старых кассовых аппаратах, калькуляторах благодаря низкому энергопотреблению, четкому отображению выводимых символов. Он имеет параллельный интерфейс, поэтому с целью упрощения организации процесса взаимодействия дисплея с МК в данном устройстве LCD1602 используется совместно с переходником I2C – конвертером FC-113. Подобное решение позволит также уменьшить количество задействованных выводов МК (рисунок 1).

Выбор среды для разработки ПО. В качестве среды для разработки ПО используется среда Arduino IDE, рекомендуемая разработчиком платформы Arduino. В этой среде программное обеспечение составляется в виде последовательности команд – программного кода на языках С и С++.

Осуществлять разработку ПО для устройства на базе платы Arduino Nano можно различными методами и средствами.

---

[1] Датчик температуры Arduino DS18B20. URL: https://arduinomaster.ru/datchiki-arduino/arduino-ds18b20/.

[2] Подключение дисплея LCD 1602 к arduino по i2c / IIC. URL: https://arduinomaster.ru/datchiki-arduino/lcd-i2c-arduino-displey-ekran/.

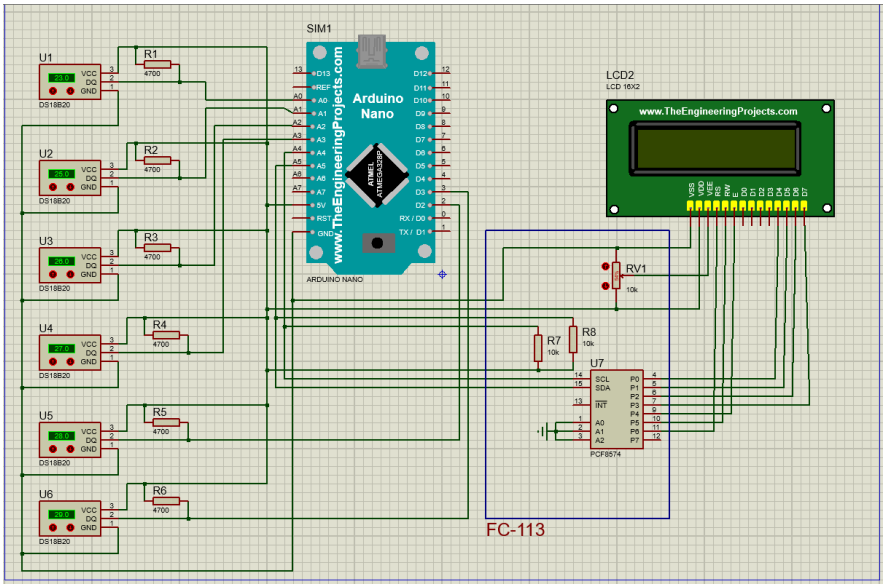

Рисунок 1. - Схема подключения электрорадиоэлемента в блоке управления лабораторного стенда

Классический метод программирования любой платы Arduino – написание программного кода в виде последовательного списка команд на языках C и C++ в среде Arduino IDE[3].

Большим преимуществом данной среды является поддержка всех плат Arduino, а также удобное и быстрое подключение сторонних библиотек.

Единственный субъективный недостаток – необходимость знания базовых возможностей языков C и C++, так как именно на последнем пишется код программы в среде, а пользовательские библиотеки нередко написаны и на чистом C.

Одной из альтернативных сред разработки является FLProg[4], позволяющая создавать прошивки для плат Arduino с помощью графических языков FBD и LAD, которые являются стандартом в области программирования промышленных контроллеров.

Достоинства FLProg заключаются в том, что при работе с программой пользователю нет необходимости заниматься написанием кода в виде последовательности команд, контролем за использованием входов – выходов, проверкой уникальности имен и согласованностью типов данных. За всем этим следит программа. Также она проверяет корректность проекта целиком и указывает на наличие ошибок.

Однако в этой среде существует весомый недостаток: далеко не для всех компонентов существует поддержка прямо в среде, следовательно, каждый неподдерживаемый «из коробки» компонент придется программировать с нуля. Не выйдет, как в Arduino IDE, просто добавить стороннюю библиотеку и работать с неподдерживаемым по стандарту компонентом с помощью удобных и понятных команд.

Еще одной достойной внимания средой разработки является XOD[5]. Эта среда, подобно FLProg, позволяет выстраивать программу с помощью визуальных блоков и создавать связи между ними, однако графический язык отличается от аналога.

Достоинства XOD: возможность графического программирования без какого-либо знания C и C++, понятный интерфейс. Эта среда распространяется абсолютно бесплатно, как и Arduino IDE, может работать в браузере, без установки.

Недостаток XOD аналогичен недостатку FLProg: сложность программирования отдельных неподдерживаемых компонентов.

Для разработки программы рассматриваемого устройства будет использоваться классический метод программирования МК путем написания программного кода в среде Arduino IDE. Главный аргумент в обоснование подобного выбора – возможность использования сторонних библиотек для взаимодействия платы с компонентами программно, что упрощает и ускоряет процесс программирования.

Разработка ПО. Стоит отметить, что компоненты, используемые в стенде, являются достаточно популярными среди разработчиков, поэтому для организации взаимодействия между платой Arduino и компонентами написано множество библиотек. Библиотеки, используемые в данном проекте, представляют собой программное обеспечение с открытым исходным кодом и могут быть использованы для некоммерческой деятельности с указанием разработчика библиотеки.

[3] Arduino IDE. URL: https://ru.wikipedia.org/wiki/Arduino_IDE.
[4] URL: https://flprog.ru.
[5] A visual programming language for microcontrollers. URL: https://xod.io.

Таким образом, прежде чем приступить к разработке программного обеспечения для измерительной установки стенда, следует подобрать подходящие библиотеки для используемых компонентов.

Для организации взаимодействия между платой Arduino и датчиками DS18B20 удобно использовать библиотеки «OneWire»[6] и «DallasTemperature»[7]. Библиотека «OneWire» позволяет организовать работу платы Arduino с компонентами, поддерживающими протокол OneWire. Для использования данной библиотеки также необходимо подключить к скетчу библиотеку «Wire», которая уже присутствует в Arduino IDE. Библиотека «DallasTemperature» значительно упрощает разработку программного кода для взаимодействия платы Arduino с датчиками DS18B20 благодаря готовым функциям для инициализации датчиков, чтения измеряемых температур, настройки точности датчика и др.

Для организации взаимодействия между платой Arduino и жидкокристаллическим дисплеем LCD1602 с I2C-адаптером удобно использовать библиотеку «LiquidCrystal_I2C»[8]. Данная библиотека имеет готовые функции для инициализации дисплея, его программной настройки и вывода на него необходимой информации.

Для начала необходимо установить вышеупомянутые библиотеки в Arduino IDE, а затем подключить их к скетчу следующим образом:

```
#include <OneWire.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <DallasTemperature.h>
```

Секция предварительной конфигурации и подготовки для работы с бибилиотеками

Далее необходимо создать объект для работы с дисплеем, в который сразу передаются адрес и параметры дисплея. Этот объект далее будет использоваться в функциях работы с дисплеем:

```
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

Затем следует создать объект для работы с датчиками, куда передаются обозначения выводов (пинов) платы Arduino, к которым подключены соответствующие выводы датчиков, по которым передаются данные посредством интерфейса OneWire:

```
OneWire ds18x20[] = { A0, A1, A2, A3, 2, 3 };
```

После этого создается переменная для учета датчиков. Она потребуется для определения датчиков:

```
const int oneWireCount = sizeof(ds18x20)/sizeof(OneWire);
```

И наконец определяем датчики в библиотеке:

```
DallasTemperature sensor[oneWireCount];
```

Секция подготовки и инициализации

Весь код, помещенный внутрь функции «setup», выполнится один раз при запуске измерительной установки. Таким образом, в рамках данной функции необходимо разместить код для инициализации компонентов, а также вывода тестовой (стандартной) информации для проверки работоспособности измерительной установки. В соответствии со всем перечисленным функция «setup» будет реализована следующим образом:

```
void setup(void) {
  //Инициализация дисплея LCD1602
  lcd.init();//Инициализация
  lcd.backlight();//Активация подсветки
  //Вывод стандартного сообщения для проверки работоспособности
  lcd.setCursor(0,0);//Перестановка курсора на нулевой символ 1-й строки
  lcd.println("Prog by Tikhon");//Вывод тестовой фразы на 1-ю строку
  lcd.setCursor(0,1);//Перестановка курсора на нулевой символ 2-й строки
  lcd.print("for ");//Вывод слова на 2-ю строку
  lcd.print(oneWireCount);//Вывод переменной, содержащей кол-во подключенных датчиков на 2-ю строку
  lcd.println(" sensors");//Вывод слова на 2-ю строку
  delay(1501);//Пауза

  // Запуск библиотеки циклом инициализации всех датчиков на указанных портах
  DeviceAddress deviceAddress;//Создание объекта для адресов подключенных датчиков
//Инициализация всех подключенных датчиков
  for (int i = 0; i < oneWireCount; i++) {;
```

[6] URL: https://github.com/PaulStoffregen/OneWire.
[7] URL: https://github.com/jmchiappa/DallasTemperature.
[8] URL: https://github.com/mrkaleArduinoLib/LiquidCrystal_I2C.

```
    sensor[i].setOneWire(&ds18x20[i]);
    sensor[i].begin();
    if (sensor[i].getAddress(deviceAddress, 0)) sensor[i].setResolution(deviceAddress, 12);
   }
  }
```

Выполнение данной функции во время работы устройства будет заметно по выводу тестового сообщения.

Рисунок 2. – Вывод тестового сообщения на дисплей в процессе моделирования

Тестовое сообщение выводит количество успешно определившихся и инициализировавшихся датчиков DS18B20. В случае выхода из строя одного или нескольких датчиков цифра будет отличной от 6.

Секция исполнительной части программы

Весь код, помещенный внутрь функции «loop», будет выполняться циклически в течение всего периода работы измерительной установки. Таким образом, внутрь данной функции логично поместить код основной части программы. В соответствии с перечисленным функция «loop» будет реализована следующим образом:

```
void loop(void) {
  lcd.clear();//Очистка дисплея от всех символов
  lcd.setCursor(0,0);//Перестановка курсора на нулевой символ 1-й строки
  lcd.print("Requesting");//Вывод слова «запрашивание» на 1-ю строку
  //Цикл запроса показаний датчиков и вывода информации на экран
  for (int i = 0; i < oneWireCount; i++) {
    sensor[i].requestTemperatures();//Считывание данных i-го датчика
    delay(1000);//Пауза
    float temperature = sensor[i].getTempCByIndex(0);//Конвертирование полученной информации в градусы Цельсия и передача
     lcd.setCursor(0,0);//Перестановка курсора на нулевой символ 1-й строки
     lcd.print("Temperature - ");//Вывод слова «температура»
     lcd.print(i + 1);//Вывод номера датчика
     lcd.print(" is ");
     lcd.setCursor(0,1);//Перестановка курсора на нулевой символ 2-й строки
     lcd.println(temperature);//Вывод температуры в градусах Цельсия
     delay(1500);//Пауза
    }
    lcd.clear();//Очистка дисплея
    lcd.setCursor(0,0);//Перестановка курсора на нулевой символ 1-й строки
    lcd.println("DONE");//Вывод слова «сделано», указывающего на успешное завершение текущего цикла измерений
    lcd.setCursor(0,1); //Перестановка курсора на нулевой символ 2-й строки
    lcd.println("Next loop");//Вывод фразы «следующий цикл», указывающей на переход к следующему циклу измерений
    delay(1500);//Пауза
  }
```

Заключение. В данной статье был подробно рассмотрен процесс разработки ПО для блока управления лабораторного стенда. Корректность и работоспособность предлагаемого ПО была подтверждена как в ходе моделирования, так и при использовании в реальном устройстве.

ЛИТЕРАТУРА

1. Дульнев, Г. Н. Методы расчета тепловых режимов прибора / Г. Н. Дульнев, В. Г. Парфенов, А. В. Сигалов. – М. : Радио и связь, 1990. – 312 с.
2. Конструкторско-технологическое проектирование электронной аппаратуры : учеб. для вузов / К. И. Билибин [и др.] ; под общ. ред. В. А. Шахнова. – М. : Изд-во МГТУ им. Н. Э. Баумана, 2002. – 528 с.

REFERENCES

1. Dul'nev, G. N., Parfenov, V. G., Sigalov, A. V. (1990). *Metody rascheta teplovykh rezhimov pribora. [Methods of calculation of thermal modes of the device]*. Moscow: Radio i svyaz'. (In Russ.).
2. Bilibin, K. I. (2002). *Konstruktorsko-tekhnologicheskoe proektirovanie elektronnoi apparatury [Design and technological design of electronic equipment]*. Moscow: Izd-vo MGTU im. N. E. Baumana. (In Russ.).

Поступила 31.10.2022

THE DEVELOPMENT OF THE CONTROL UNIT OF A LABORATORY STAND FOR SIMULATION THE THERMAL MODS OF RADIO-ELECTRONIC DEVICES

T. KAPACH, M. KHAMICHONAK, T. MALADZECHKINA
(Euphrosyne Polotskaya State University of Polotsk)

The control unit included in the laboratory stand for modeling thermal modes of radio-electronic means is presented. The main functions performed by the control unit of the laboratory stand are given, the features of its design are described. The rationale for choosing the Arduino IDE software environment for software development is given. A selection of libraries used to organize the interaction between the Arduino board and the components of the stand was carried out. The software for the control unit of the laboratory stand has been developed and reviewed in detail.

Keywords: *thermal mode, control unit, software, Arduino IDE environment, program code.*