МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное
учреждение высшего образования
«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ РАДИОЭЛЕКТРОНИКИ И ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

ПРАКТИКУМ ДИСЦИПЛИНЕ «ПРАКТИЧЕСКИЕ ОСНОВЫ РАБОТЫ С ИНТЕЛЛЕКТУАЛЬНЫМИ ПРОГРАММИРУЕМЫМИ СИСТЕМАМИ (AVR)»

Часть 1

Учебно-методическое пособие

Севастополь
СевГУ
2020

УДК 621.396.6
П69

Р е ц е н з е н т ы:
Д.В. Начаров – доцент кафедры «Электронная техника»,
кандидат технических наук
Ю.Н. Тыщук – ст. преподаватель кафедры
«Радиоэлектроника и телекоммуникации»

Ответственный за выпуск:
Ю.П. Михайлюк – канд. техн. наук, доцент,
заведующий кафедрой «Электронная техника»

Составители: И.В. Скорик, М.А. Дурманов

П69 Практикум по дисциплине «Практические основы работы с интеллектуальными программируемыми системами (AVR)»: учеб.-метод. пособие. Ч. 1 / Сост. И.В. Скорик, М.А. Дурманов ; Министерство науки и высшего образования Российской Федерации, Севастопольский государственный университет, Институт радиоэлектроники и информационной безопасности. – Севастополь: СевГУ, 2020. – 47 с. – Текст : электронный.

Содержит теоретическую и практическую информацию необходимую для выполнения практических занятий, а также получения дополнительных знаний по изучаемой дисциплине.

УДК 621.396.6

Учебно-методическое пособие рассмотрено и утверждено на заседании кафедры «Электронная техника», протокол № 08 от 17 марта 2020 г.

Учебно-методическое пособие рассмотрено и рекомендовано к изданию на заседании ученого совета Института радиоэлектроники и информационной безопасности, протокол № 7 от 26 февраля 2020 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цикл практических работ, представленных в настоящем учебнометодическом пособии, позволяет изучить структуру и основы программирования микроконтроллеров семейства ATMegaxxx фирмы «Microchip». А также освоить и закрепить компьютерный язык программирования высокого уровня С, на ряду с построением цифровых и аналоговых схем.

Выполнение практических работ требует от обучающихся начальных знаний какого-либо языка компьютерного программирования и понимание логики его работы, а также базовых знаний электронной компонентной базы. Выполнение каждой практической работы заключается в реализации программного обеспечения «firmware — неизменяемое программное обеспечение, загружаемое в ROM память», которое должно выполнять поставленные задачи в зависимости от решаемой проблемы и построения требуемой цифровой схемы (схема может быть собрана в любом из доступных симуляторов или при использовании реальных компонентов на беспаечной макетной плате). Каждая практическая работа считается выполненной после демонстрации работоспособной схемы и предоставления написанного программного кода.

Отчет по практической работе должен быть представлен в рукописном исполнении (отчет может также быть оформлен в печатной форме). В отчет должны быть включены: цель, электрическая схема, программный код с комментариями и подробный вывод. В выводе важно описать особенности реализации программного кода или нестандартные способы программных решений.

Защита отчета практической работы состоит из двух этапов: в написанный код программы, вносятся изменения, которые нарушают работу программы, для дальнейшего поиска и устранения внесенного изменения обучающимся, далее обучающемуся следует найти и исправить внесенные изменения для получения работоспособного программного кода. Далее обучающийся отвечает на контрольные вопросы, после чего защита практической работы считается завершенной.

Практические работы составлены таким образом, что разработка программы для каждой последующей лабораторной работы основывается на наработках, выполненных в предыдущих практических работах. Таким образом, все практические работы взаимосвязаны и поэтому необходимо сохранять программный код на съемных носителях информации или в интернет облаке.

Практическая работа № 1.
ПОСТРОЕНИЕ ЦИФРОВОЙ СХЕМЫ И ИЗУЧЕНИЕ ОСНОВ ЯЗЫКА С

1.1. Цель работы

Изучить онлайн симулятор «TinkerCAD» для симуляции схем с использованием 8 разрядного микроконтроллера AVR на платформе «Arduino Uno».

Ознакомиться с интегрированной средой разработки «Arduino» и основами языка С.

1.2. Теоретические сведения

В настоящее время большинство радиоэлектронных и электронных схем проектируется с использованием микроконтроллеров (embedded systems — встраиваемые системы). Такие схемы встречают повсеместно начиная от простейших устройств бытовой техники, карманных мобильных устройств, медицинских приборов и переходя к самым сложным телекоммуникационным космическим системам. Микроконтроллер позволяет построить электронную схему достаточно гибкой и удобной в отладке, в большинстве случаев путем корректировки кода программы достигается требуемый результат без изменения схемы устройства.

На сегодняшний день на мировом рынке присутствуют десятки компаний занимающихся производством микроконтроллеров, некоторые из них: STMicroelectronics, Microchip, TexasInstruments, Infineon, Atmel, Motorola, Simens и другие.

Микроконтроллеры можно разделить на несколько групп в соответствии с их разрядностью: 8, 16 и 32 разрядными архитектурами. Наиболее удобными для изучения являются микроконтроллеры с 8 разрядной архитектурой, так как они имеют минимальный набор периферийных модулей и требуют написания относительно простого кода для получения первой работоспособной программы.

1.3. Основная часть

1.3.1. Среда онлайн симулятора

Для настоящей и последующих практических работ необходимо использовать доступный онлайн симулятор «TinkerCAD», предоставляемый компанией AutoDesk. Для использования указанного симулятора требуется произвести регистрацию для доступа в личный кабинет по ссылке [1]. После регистрации и входа в учетную запись онлайн среды «TinkerCAD», на панели слева следует выбрать вкладку «Circuits», затем «Создать цепь», далее откроется рабочее пространство симулятора (рис.1.1). Панель справа содержит необходимые компоненты, где следует выбрать: плату «Arduino Uno», светодиод и резистор (установив номинал в 200 Ом, для этого следует по выбранному резистору

осуществить щелчок любой кнопкой мыши для открытия панели параметров и задать требуемый номинал) и собрать схему показанную на рис. 1.1 (цветовая маркировка резисторов доступна в приложении А).

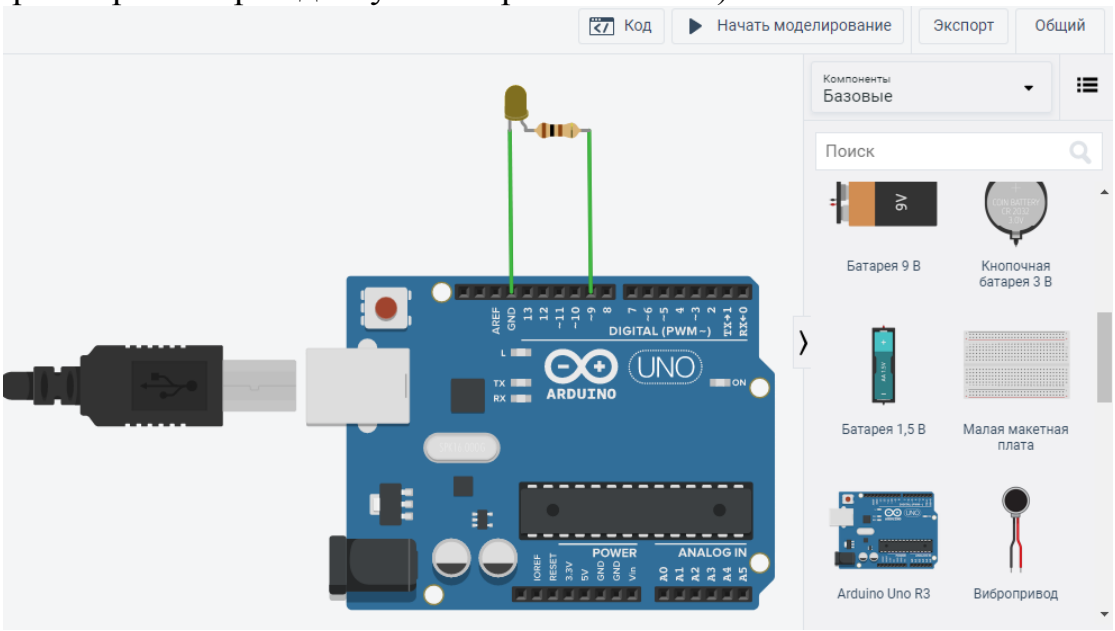


Рис. 1.1. Рабочее пространство симулятора TinkerCAD с простейшей схемой

После сборки схемы (см. рис. 1.1) необходимо составить программный код для проведения симуляции. Для доступа к полю программного кода требуется открыть вкладку «Код», слева в верхней части из выпадающего меню выбрать способ представления кода «Текст» (рис. 1.2).

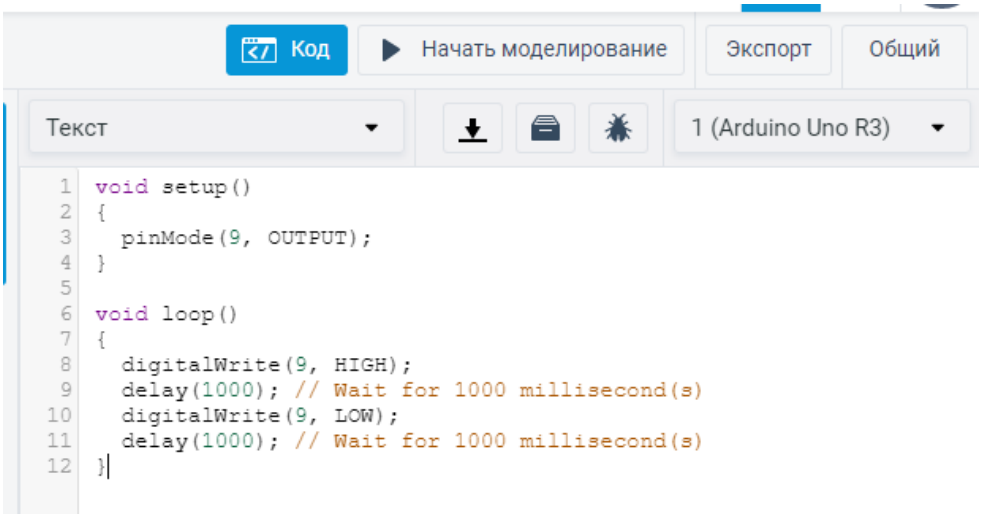


Рис. 1.2. Поле программного кода онлайн симулятора

Программный код (см. рис.1.2) необходимо задействовать в режиме симуляции, нажав кнопку «Начать моделирование», после чего светодиод в собранной схеме (см. рис. 1.1) должен начать мигать с интервалами по 1 с на каждое состояние.

1.3.2. Использование монитора последовательного порта

Монитор последовательны порта в онлайн симуляторе полностью аналогичен интегрированной среде «Ardunio» при работе с реальным микроконтроллером. Монитор последовательного порта позволяет контролировать программу и выводить в терминал монитора последовательного порта необходимую информацию, значения переменных, текстовую информацию и другое. Настройка и инициализация последовательно порта осуществляется функцией Serial.begin(9600), где 9600 это скорость бит/с. Далее используя функцию Serial.println(“Hello world”) можно выводить текстовую информацию или значение переменной Serial.println(variable) (рис. 1.3).

```
1  void setup()
2  {
3  Serial.begin(9600);
4  Serial.println("Hellow world");
5  }
6  int a=0;
7  void loop()
8  {
9  delay(500);
10 a++;
11   if(a == 10) {
12   a = 0;
13   }
14 Serial.println(a);
15
16 }
```

Монитор последовательного интерфейса

```
Hellow world
1
2
3
4
5
6
7
```

Рис. 1.3. Пример программы с использованием монитора последовательного порта

1.3.3. Описание программы

Языки прикладного программирования (микроконтроллеров или встраиваемых систем) делятся на языки высокого и низкого уровней. Язык «С» представляет собой язык высокого уровня позволяющий быстро и удобно составлять код большого объема со сложными логическими конструкциями, не требующий глубокого и детального изучения периферии микроконтроллера, а позволяющего работать на уровне «черный ящик».

Необходимо подробно рассмотреть программный код представленный на рис. 1.2. Программа состоит из двух основных и базовых функций: **void** setup() и **void** loop().

Назначение каждой из функций следующее:

— функция **setup()** предназначена для начальной конфигурации (настройки) периферийных модулей микроконтроллера и выполняется только один раз при подаче питания на микроконтроллер;

— функция **loop()** является основной частью программы и выполняется циклически на протяжении всего времени подключенного питания микроконтроллера.

Помимо основных функций в среде симулятора используется встроенные функции позволяющие удобно настраивать и управлять периферийными модулями микроконтроллера (данные функции доступны только при работе в интегрированной среде «Arduino»):

pinMode(номер вывода, функция) — функция конфигурации направленности цифрового вывода микроконтроллера;

digitalWirte(номер вывода, логический уровень) — функция управления логическим уровнем подаваемого на вывод микроконтроллера;

delay() — функция программной задержки в миллисекундах.

Описание основных функций приведено в приложении В.

Анализ логики работы программы. В программном коде осуществляется настройка вывода в функции **setup()**, а затем в основной функции **loop()** осуществляется поочередное переключение логических уровней вывода 9 с задержкой каждого состояния на 1 секунду для визуального восприятия переключения светодиода.

1.3.4. Сборка и программирование реальной схемы

После получения успешных результатов выполнения вышеизложенных пунктов следует приступить к практической части сборки схемы и загрузки программного кода из интегрированной среды «Arduino».

На беспаечной макетной плате следует собрать схему идентичную схеме собранной в симуляторе (см. рис. 1.1). При сборке схемы на беспаечной макетной плате следует учитывать особенности соединения контактов (рис. 1.4).

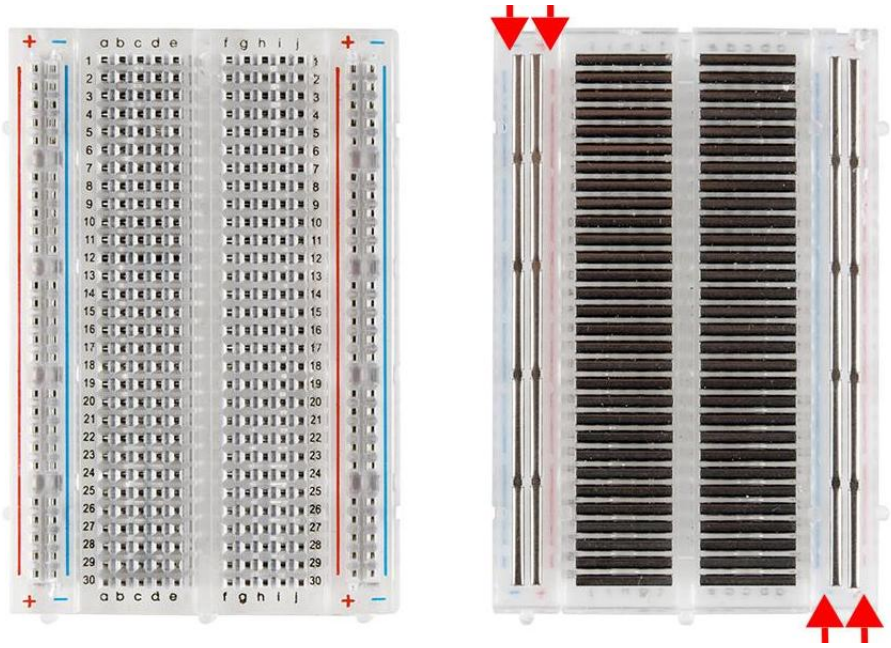


Рис. 1.4. Направления контактов беспаечной макетной платы

Программный код из онлайн симулятора требуется перенести в среду «Arduino Uno» (рис. 1.5), затем подключить платформу «Arduino Uno» к доступному «USB» порту предварительно собрав схему на макетной плате, после этого в среде Arduino назначается порт (Tools -> Port -> Port с надписью Arduino Uno) на котором система Windows обнаружила подключение платы «Arduino Uno».

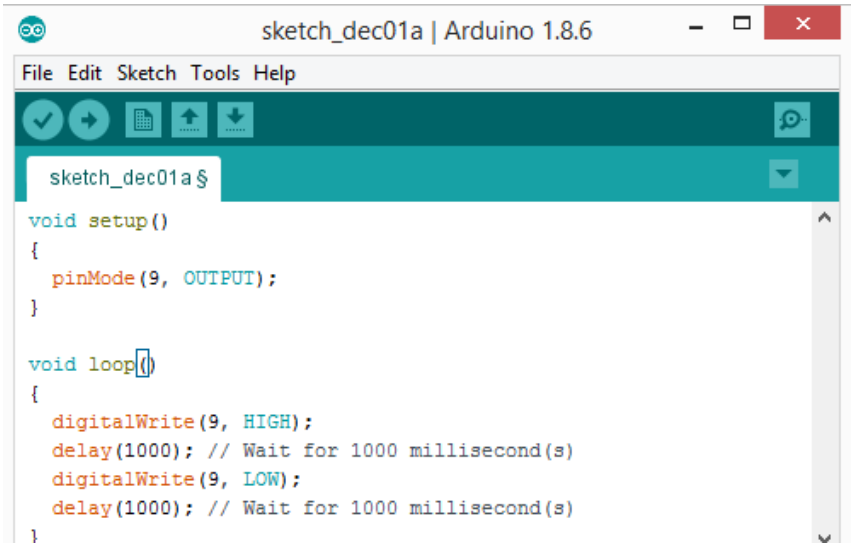


Рис. 1.5. Программа в среде Arduino

После выполнения вышеописанной настройки, необходимо загрузить программу в микроконтроллер нажав кнопку со стрелкой вправо (левый верхний угол окна среды см. рис. 1.5). По завершению загрузки программы светодиод начинает сразу мигать.

1.4. Порядок выполнения практической работы

1.4.1. Подробно ознакомьтесь с материалом, изложенным в основной части практической работы.

1.4.2. Последовательно выполните все пункты как описано в тексте с сопутствующими снимками экрана для составления отчета.

1.4.3. В качестве самостоятельного задания, необходимо добавить количество светодиодов n (где n определяется последней цифрой номера зачетной книги, для четных цифр n = 2, для нечетных n =3) для нечетных номеров переключение светодиодов должно линейно ускоряться или замедляться, а для четных номеров обеспечить стабильное переключение светодиодов (программу необходимо представить в отчете) с циклическим повторением.

1.4.4. Используйте возможность монитора порта для отображения номера зажигающегося светодиода (следует условно задать номера светодиодов после их добавления согласно пункту 1.4.3).

1.4.5. Приведите расчет токоограничивающего резистора в цепи светодиода (параметры светодиода: номинальная величина тока 10 мА при напряжении питания 3 В).

1.4.6. Собранную схему в онлайн симуляторе следует реализовать на практике, осуществить программирование микроконтроллера для проверки работоспособности собранной схемы.

1.5. Содержание отчета

Цель работы.

Снимок экрана со схемой окна онлайн симулятора.

Программа с описанием.

Программа самостоятельного задания.

Подробные выводы.

Приложение с электрической схемой, начерченной в соответствии с ГОСТом и чертежной рамкой (условно-графические обозначения радиоэлектронных компонентов приведены в приложении Б).

1.6. Контрольные вопросы

1.6.1. Основные разрядности архитектур микроконтроллеров.

1.6.2. Структура программы, основные функции и их назначение.

1.6.3. Перечислите встроенные функции, использованные в настоящей работе.

1.6.4. Последовательность действий для симуляции схемы в онлайн симуляторе.

1.6.5. Последовательность действий при работе с реальной схемой из интегрированной среды «Arduino».

Практическая работа № 2.
ПРЕРЫВАНИЯ И УСТРОЙСТВА ВВОДА

2.1. Цель работы

Изучить и научиться применять простейшие устройства ввода с использованием кнопочных контактов с аппаратным прерыванием и программной обработкой, а также резистивного элемента в качестве устройства ввода.

Изучить основные прерывания микроконтроллера.

2.2. Теоретические сведения

Взаимодействие пользователя с программой микроконтроллера осуществляется через внешние органы оперативного взаимодействия схемы (кнопочные контакты подключенные индивидуально или в виде матрицы, сенсорные датчики, энкодеры, датчики наклона, резистивные элементы и другие). Основная задача устройств ввода — это взаимодействие с программой и управление другими элементами и параметрами схемы.

Устройства ввода могут быть подключены к цифровым выводам микроконтроллера с обязательным условием их настройки на вход и задействованием внутреннего подтягивающего резистора (в основном при использовании кнопочных контактов). Внутренний подтягивающий резистор устраняет эффект «антенны» вывода возникающего в силу кондуктивных наводок и обеспечивает условие устойчивого потенциала логической «единицы» или «нуля».

Другой способ подключения устройств ввода, через аналоговые каналы, в таких случаях применяются переменные резисторы или резистивные делители.

Схема работы с некоторыми устройствами ввода показана на рис. 2.1.

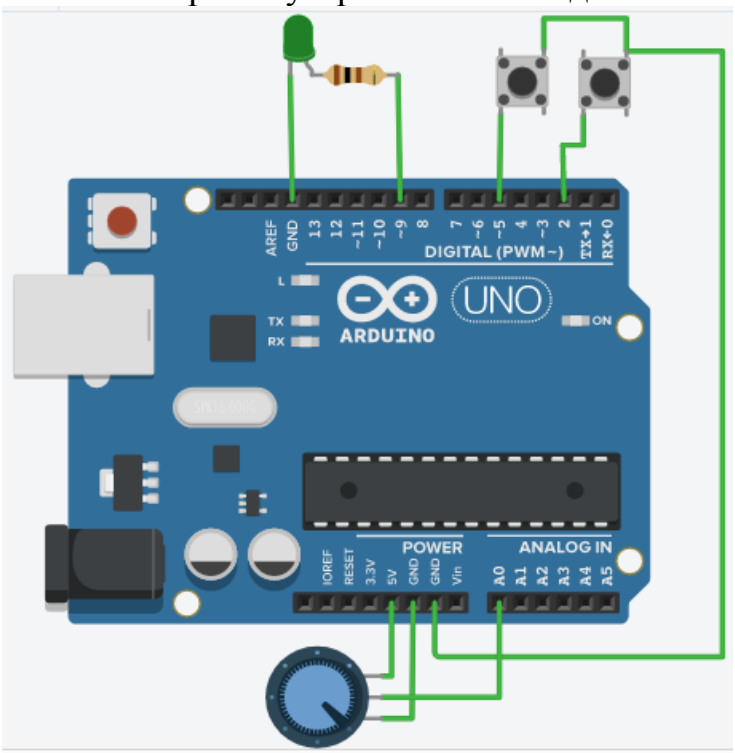


Рис. 2.1. Схема с использованием нескольких устройств ввода

Описание схемы. Пара кнопочных контактов подключена к выводам «5» и «2» микроконтроллера, которые настроены на вход с использованием внутренних подтягивающих резисторов (функция настройки pinMode(Dx, INPUT_PULLUP)). Вывод «5» работает в режиме цифрового входа общего назначения, а вывод «2» работает как цифровой вход как источник внешнего прерывания. Такое подключение кнопочных контактов позволит сравнить реакцию программы и микроконтроллера на используемые кнопочные контакты с применением прерываний и без использования прерываний.

Переменный резистор подключается к аналоговому каналу A0.

Описание логики работы схемы. В начальный момент времени светодиод мигает с интервалами в 1 секунду, при нажатии на пару кнопочных контактов (вывод «5») светодиод остается в погашенном состоянии, а при повторном нажатии мигание светодиода возобновляется (однако программное считывание состояния кнопочных контактов имеет значительный недостаток, заключающего в пропуске нажатий). Другая пара кнопочных контактов (вывод «2») выполняет тоже действие, что первая пара кнопочных контактов, но с использованием обработчика прерываний (реакция программы на нажатие кнопки всегда своевременна).

Переменный резистор работающий как устройство ввода блокирует или разрешает работу кнопочных контактов. Если уровень напряжения на втором (ползунок) выводе переменного резистора превышает 3 В, кнопочные контакты блокированы и если напряжение на втором выводе переменного резистора ниже 3 В, тогда кнопочные контакты разблокированы на программном уровне.

Программный код демонстрирующий работу схемы представлен на рис. 2.2.

Вспомогательные функции используемые в программе:

attachInterrupt(0, button, FALLING) — функция включения внешнего прерывания (параметры функции: «0» указывает номер внешнего прерывания, «button» имя функции обработчика прерывания, «FALLING» задает реакцию прерывания на фронт);

digitalRead(5) — функция считывания логического уровня поданного на цифровой вывод микроконтроллера настроенного на вход, из внешней цепи (параметр «5» указывает номер цифрового вывода);

analogRead(A0) — функция аналого-цифрового преобразования напряжения поданного на аналоговый канал (параметр функции «A0» номер аналогового канала), данная функция формирует значение от 0 до 1023 пропорциональное напряжению входному напряжению от 0 В до 5 В;

void button() — функция обработчика внешнего прерывания.

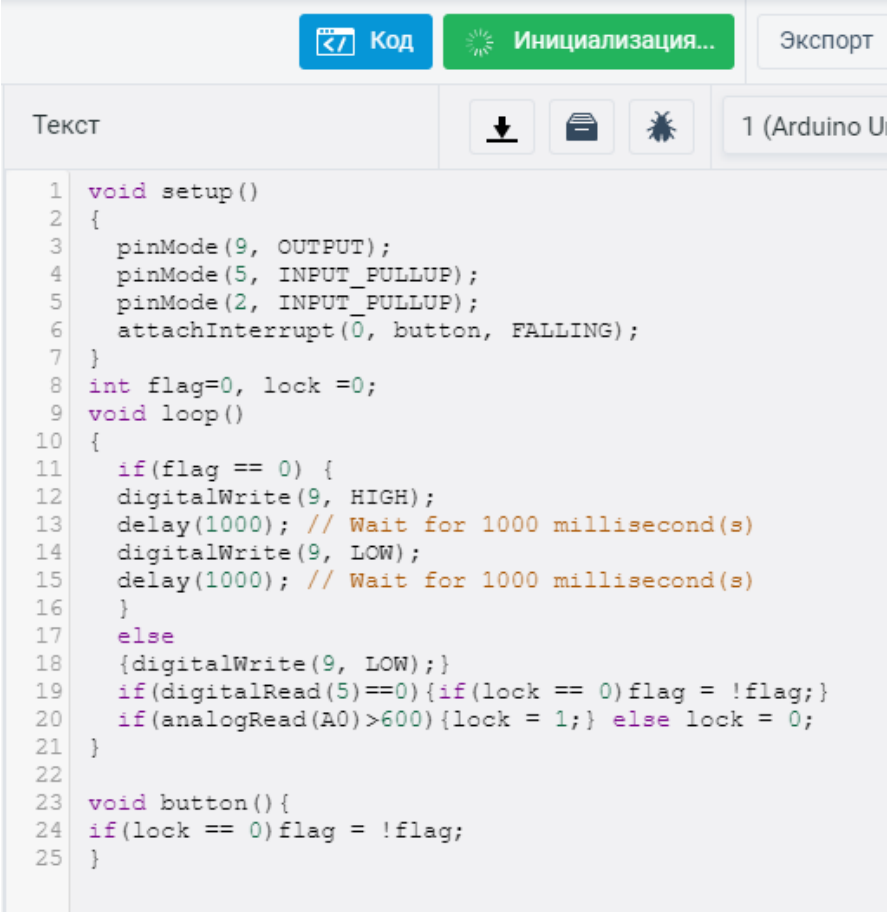


Рис. 2.2. Программный код с использованием устройств ввода

Прерывания. Прерывание — это сигнал, сообщающий центральному процессору микроконтроллера, о необходимости временной приостановки выполнения основной программы и перехода в обработчик прерывания. Прерывания используются в микроконтроллерах для своевременного реагирования на асинхронные события незапланированные в программном коде. В момент регистрации прерывания, выполнение основного кода программы (функция loop()) приостанавливается и управление передается функции обработки прерываний (button(), см. рис. 2.2). Регистрация прерывания осуществляется на аппаратном уровне без участия программы. Прерывания могут быть внешними или внутренними, которые генерируются периферийными модулями микроконтроллера.

На примере настоящей практической работы используются две пары кнопочных контактов. Одна пара кнопочных контактов проверяется через условие в основном цикле программы, а другая пара кнопочных контактов обрабатывается в специальной функции (обработчике прерываний). Такой способ использования кнопочных контактов позволяет сравнить «отклик» микроконтроллера на нажатие каждой пары кнопочных контактов по отдельности. Пара кнопочных контактов, которая обрабатывается при помощи оператора условия, реагирует на нажатия нерегулярно, так как, если нажатие происходит в момент выполнения программной задержки, тогда реакция микроконтроллера на нажатие отсутствует. В свою очередь нажатие другой пары кнопочных контактов

позволяет реагировать своевременно без пропусков, так как каждое нажатие генерирует прерывание на аппаратном уровне микроконтроллера.

Разработчику программы необходимо учитывать особенности работы устройства перед принятием решения о необходимости использования прерываний. На практике существуют случаи, когда использование прерываний не имеет смысла и обработка определенного события должна осуществляться только программным путем.

Модуль аналого-цифрового преобразования (АЦП).

Модуль АЦП предназначен для преобразования величины напряжения в заданном динамическом диапазоне в пропорциональный двоичный код (позиционный) или пропорциональную величину, выраженную в единицах других систем счисления.

Существует несколько типов АЦП: последовательного приближения, параллельного преобразования, дельта-сигма и другие. В настоящем микроконтроллере используется АЦП последовательного преобразования с 10 разрядным регистром последовательного приближения.

В результате аналого-цифрового преобразования регистр последовательного приближения модуля АЦП формирует число в диапазоне от 0 до 1023 (в случае 10 разрядного регистра последовательного приближения), данный результат можно удобно пересчитать в величину соответствующую входному напряжению по следующей формуле

$$U_{in} = \frac{U_{+ref} - U_{-ref}}{2^n} N, \qquad (2.1)$$

где U_{+ref} — опорное напряжение, определяющее верхнюю границу входного динамического диапазона (по умолчанию положительное напряжение питания микроконтроллера);

U_{-ref} — опорное напряжение, определяющее нижнюю границу входного диапазона (по умолчанию общая шина микроконтроллера);

N — результат аналого-цифрового преобразования (значение регистра последовательного приближения);

n — разрядность модуля АЦП (разрядность регистра последовательного приближения).

Пользуясь формулой (2.1) удобно получить величину входного напряжения по значению, которое возвращает функция analogRead().

В некоторых случая модуль АЦП может использоваться в качестве устройства ввода при помощи потенциометра (см. рис. 2.1) или кнопочных контактов с резистивными делителями.

Применение модуля АЦП для работы с аналоговыми датчиками.

В большом своем количестве аналоговые датчики представляют собой резистивный компонент, в котором величина сопротивления изменяется пропорционально измеряемому параметру. Поэтому, аналоговые датчики включаются по схеме резистивного делителя напряжения, где одно из плеч делителя изменяет величину сопротивления в зависимости от влияющего фактора. Вы-

ходное напряжение делителя подается на вход модуля АЦП для преобразования измеряемого напряжения в пропорциональный двоичный код.

Точность измерения параметров аналоговыми датчиками зависит от напряжения питания датчика и разрядности модуля АЦП.

2.3. Порядок выполнения практической работы

2.3.1. Подробно ознакомьтесь с материалом, изложенным в основной части практической работы.

2.3.2. Последовательно выполните все пункты как описано в тексте с сопутствующими снимками экрана для составления отчета.

2.3.3. Проанализируйте реакцию микроконтроллера на пару кнопочных контактов с использование условия и обработчика прерывания.

2.3.4. В качестве самостоятельного задания, следует произвольно выбрать доступные устройства ввода, составить схему и программу.

2.3.6. Схему, собранную в онлайн симуляторе необходимо собрать на практике, осуществить ее программирование и проверку работоспособности.

2.4. Содержание отчета

Цель работы.

Снимок экрана со схемой окна онлайн симулятора.

Код программы с описанием.

Код программы части задания с самостоятельным выполнением.

Подробные выводы.

Приложение схемы, начерченной в соответствии с ГОСТом и чертежной рамкой.

2.5. Контрольные вопросы

2.5.1. Назначение устройств ввода.

2.5.2. Предложите схемотехнические решения устройств ввода для микроконтроллеров.

2.5.3. Прерывания и их использование.

2.5.4. Способы программной регистрации нажатия пары кнопочных контактов.

2.5.5. Способы аппаратно-программной регистрации нажатия кнопки.

2.5.6. Отличие отклика микроконтроллера на нажатие пары кнопочных контактов с использованием прерывания и без использования прерывания.

2.5.7. Модуль АЦП.

2.5.8. Расчет величины входного напряжения по значению, возвращаемому функцией аналого-цифрового преобразования.

2.5.9. Использование АЦП для получения значений измеряемых параметров аналоговыми датчиками.

Практическая работа № 3.
ШИМ СИГНАЛ И УПРАВЛЕНИЕ СИЛОВОЙ НАГРУЗКОЙ

3.1. Цель работы

Изучить основные способы применения ШИМ сигнала.
Освоить основу схемотехники управления силовыми нагрузками.

3.2. Теоретические сведения

Сигнал с широтно-импульсной модуляцией (ШИМ) широко применяется в цифровой и аналоговой схемотехнике. ШИМ сигналы применяются для управления силовыми транзисторными ключами в импульсных преобразователях напряжения, генерации аналоговых сигналов средствами цифровых устройств, управление мощностью силовых нагрузок и другие.

ШИМ сигнал характеризуется следующими основными параметрами:

— частота следования импульсов;

— коэффициент заполнения (выражается в процентах или коэффициентом от 0 до 1);

— амплитуда импульсов.

Микроконтроллер платформы «Arduino» способен генерировать ШИМ сигнал с фиксированной частотой следования импульсов. Все цифровые выводы имеющие символ «~» могут использоваться для генерации ШИМ сигнала. В программе используется функция digitalWrite(Dx, duty cycle), где Dx — это номер цифрового вывода со знаком «~» и «duty cycle» — параметр указывающий коэффициент заполнения в диапазоне от 0 до 255.

Для наглядной демонстрации ШИМ сигнала в онлайн симуляторе необходимо собрать схему состоящую из платформы «Arduino» и осциллографа (рис. 3.1).

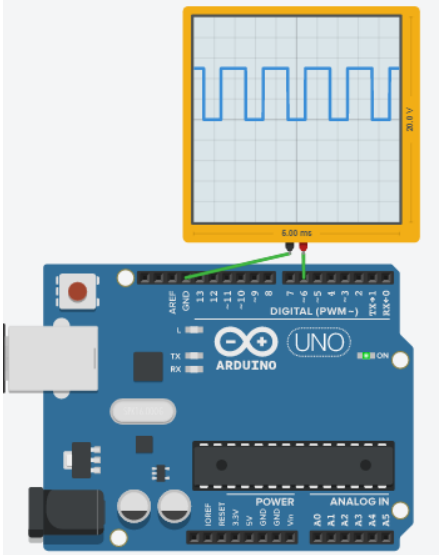


Рис. 3.1. Осциллограмма ШИМ сигнала

Программа для генерации ШИМ сигнала с заданным коэффициентом заполнения показана на рис. 3.2.

```
void setup()
{
  pinMode(6, OUTPUT);
}

void loop()
{
 analogWrite(6,150);
}
```

Рис. 3.2. Программа генерации ШИМ сигнала
с указанием коэффициента заполнения

Управление коэффициентом заполнения ШИМ сигнала можно реализовать программными средствами (алгоритмическое управления и циклическое) или при помощи внешних устройств ввода и датчиков. Программный код с реализацией одного из доступных способов управления коэффициентом заполнения при помощи потенциометра показан на рис. 3.3. Схема подключения потенциометра (см. рис. 2.1).

```
void setup()
{
  pinMode(6, OUTPUT);
}

void loop()
{
 analogWrite(6,analogRead(A0)/4);
}
```

Рис. 3.3. Программа генерации ШИМ сигнала
с управляемым коэффициентом заполнения

Управление силовой нагрузкой. Силовой нагрузкой можно считать нагрузку потребляющую ток (мощность) больший, чем может обеспечить вывод порта микроконтроллера. Управление мощностью силовой нагрузки может осуществляться при помощи транзистора, включенного по схеме с общим коллектором (рис. 3.4, б) или эмиттером (рис. 3.4, а) и работающем в ключевом режиме под управлением ШИМ сигнала. Такой способ управления силовым транзисторным ключом позволяет управлять мощностью нагрузки с высоким значением КПД, однако такое управление осуществляется без гальванической развязки.

Управление силовой нагрузкой может также осуществляться с гальванической развязкой. Гальваническая развязка позволяет управлять током нагрузки без электрического контакта, например, при помощи оптрона (рис. 3.4). Такой способ управления позволяет защитить выводы порта микроконтроллера от короткого замыкания, превышения допустимого значения тока или напряжения. В оптроне, световой поток излучаемый светодиодом управляет сопро-

тивлением фототранзистора, данный процесс управления реализован без электрического контакта.

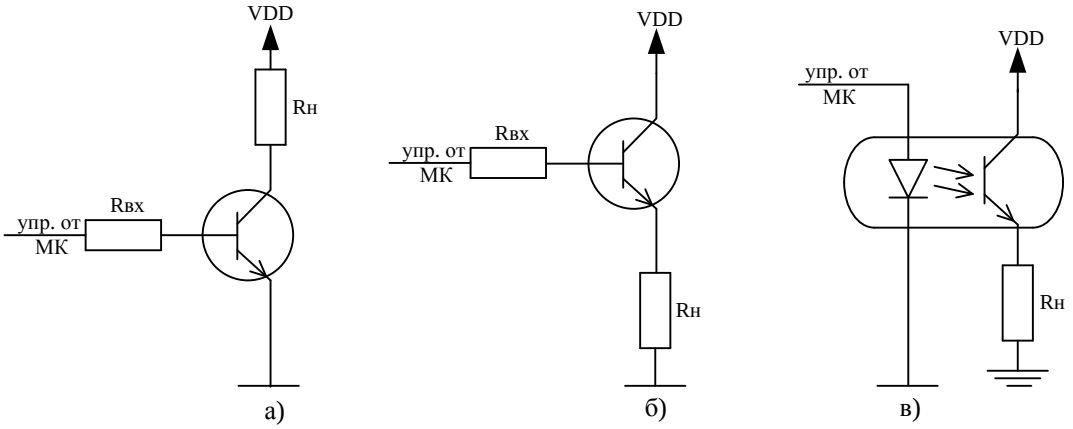


Рис. 3.4. Схемы включения силовых нагрузок
без гальванической развязки (а, б) и с гальванической развязкой (в)

Схема управления некоторыми силовыми нагрузками в онлайн симуляторе показана на рис. 3.5.

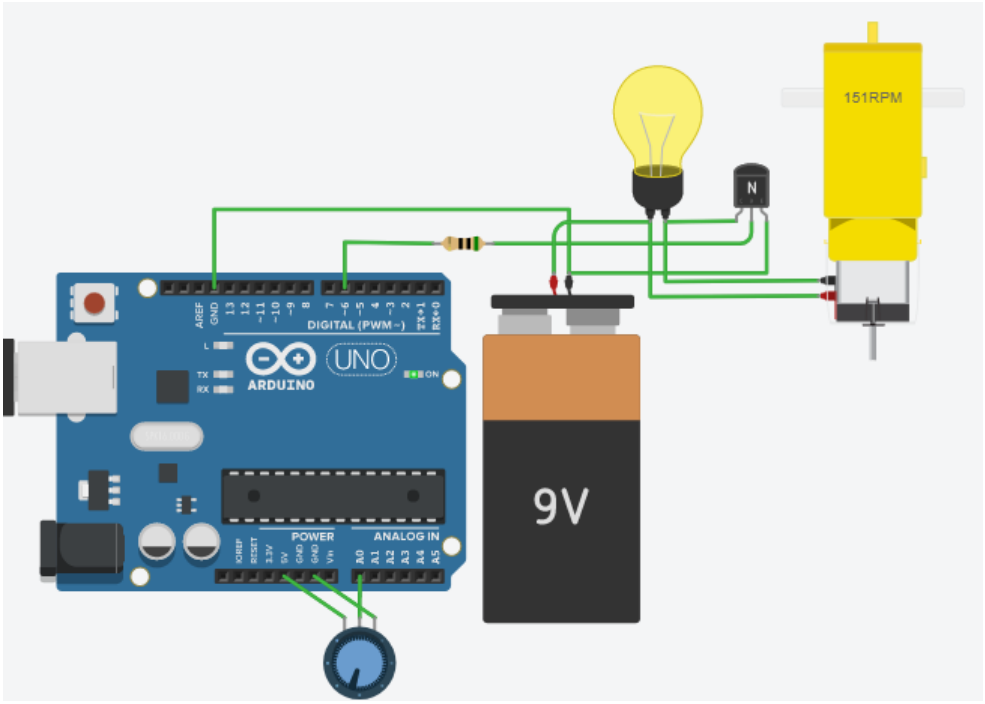


Рис. 3.5. Схема управления несколькими нагрузками при помощи ШИМ сигнала

Вращение потенциометра (см. рис. 3.5) позволяет плавно управлять яркостью накала и скоростью вращения электромотора.

3.4. Порядок выполнения практической работы

3.4.1. Подробно ознакомьтесь с материалом, изложенным в основной части практической работы.

3.4.2. Последовательно выполните все пункты как описано в тексте с сопутствующими снимками экрана для составления отчета.

3.4.3. В качестве самостоятельного задания, следует произвольно выбрать доступные устройства ввода, составить схему и программу.

3.4.4. Осуществите симуляцию схемы регулировки мощности любой из доступных нагрузок при помощи транзистора. Способ регулировки следует выбрать произвольно.

3.5. Содержание отчета

Цель работы.

Электрическая принципиальная схема задействованной части макета.

Часть программы, где реализована обработка прерываний от кнопочных контактов с программой индивидуального задания.

Подробные выводы.

Приложение со схемой, начерченной в соответствии с ГОСТом и чертежной рамкой.

3.6. Контрольные вопросы

3.6.1. Общие сведения о ШИМ сигнале.

3.6.2. Применение ШИМ сигнала.

3.6.3. Схемы включения транзистора для управления силовой нагрузкой.

3.6.4. Способы управления мощностью или током силовых нагрузок.

3.6.5. Оценка КПД электронной ШИМ управляемой схемы.

3.6.6. Предложите другие способы управления мощностью силовой нагрузки.

3.6.7. Гальваническая развязка. Схемная реализация управления нагрузкой с гальванической развязкой.

3.6.8. Критерий разделения нагрузки на нагрузку малой мощности и высокой мощности.

Практическая работа № 4.
ВЫВОД ЧИСЛОВЫХ ДАННЫХ НА СЕМИСЕГМЕНТНЫЙ ИНДИКАТОР

4.1. Цель работы

Изучить схемотехнику использования семисегментного светодиодного индикатора.

Составить программу для вывода цифр и чисел.

4.2. Теоретическая часть

Семисегментный индикатор состоит из семи элементов индикации (сегментов), включающихся и выключающихся по отдельности. Сегменты обозначаются буквами: A, B, C, D, E, F, G и DP. Обозначения сегментов одного разряда индикатора показаны на рис. 4.1.

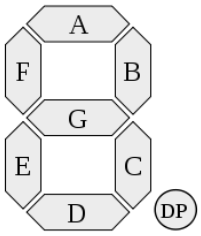


Рис. 4.1. Обозначения сегментов семисегментного индикатора

Семисегментные светодиодные индикаторы получили широкое применение в радиоэлектронных и электронных схемах для вывода буквенно-цифровой информации, так как являются одними из простейших устройств вывода и отображения информации. Светодиоды всех сегментов таких индикаторов могут быть соединены общим анодом или катодом, где включение сегментов обеспечивается за счет подачи низких или высоких уровней соответственно (рис. 4.2).

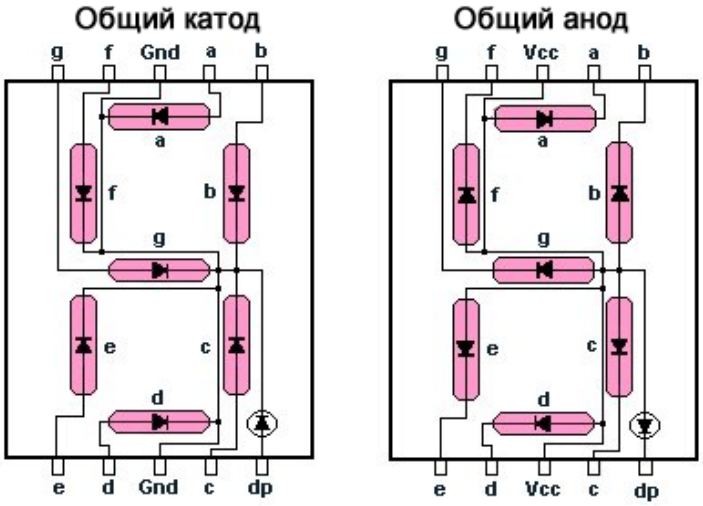


Рис. 4.2. Схемы четырехразрядных семисегментных индикаторов

4.3. Принцип работы динамической и статической индикации

Вывод информации на семисегментные индикаторы может быть организован при помощи статической и динамической индикации.

Динамическая индикация — это вид индикации, при котором вывод информации на разряды индикатора осуществляется поочередно. При этом частота вывода сигналов на индикатор выбирается такой, чтобы переключение разрядов индикатора не было ощутимо глазу человека. Данный вид индикации может быть использован при использовании более двух разрядов. Данный вид индикации требует наименьшее количество выводов микроконтроллера, но требуется относительно сложный алгоритм управления.

Статическая индикация — это вид индикации, при котором вывод информации осуществляется без переключения индикаторов. Такой вид индикации более прост в программной реализации, но требует большое количество выводов микроконтроллера. Данный вид индикации используется для относительно малого количества разрядов.

Использование специализированных микросхем регистров хранения с дешифратором (декодером) двоичного кода в семисегментный (CD4511BCN) или сдвиговых регистров (74HC164) позволяет уменьшить количество используемых выводов микроконтроллера и добиться более компактной компоновки печатной платы конечного устройства и упрощения программного кода.

4.4. Реализация схем индикации

4.4.1. Статическая индикация

Рассмотрим схему подключения семисегментного индикатора (статический режим индикатора) для получения простейшего счетчика от 0 до 9 (рис. 4.1).

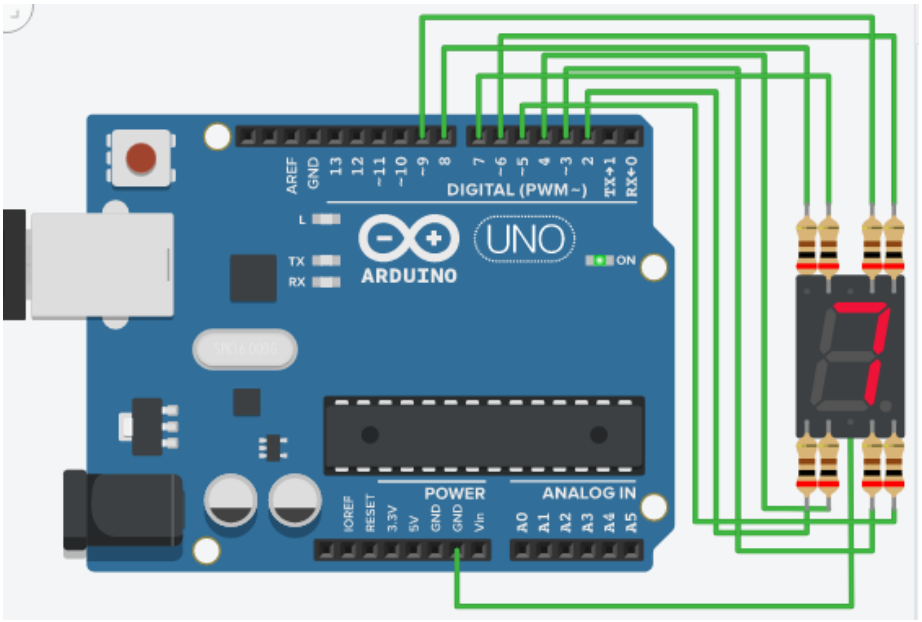


Рис. 4.1. Схема подключения семисегментного индикатора (статический режим индикации)

Семисегментный индикатор необходимо настроить на общий катод (см. рис. 4.1), резисторы установить номиналом 200 Ом. Подключения семисегментного индикатора к цифровым выводам отладочной платы «Arduino» в соответсвии с рис. 4.1.

Анализ схемы статической индикацией. Схема (см. рис. 4.1) позволяет выводить только цифры на один разряд в режиме статической индикации. Увеличение разрядности, для вывода двухзначных чисел и более, требует использования большего количества семисегментных индикаторов, в свою очередь необходимо задействовать 8 цифровых выводов на каждый отдельный индикатор (разряд). Со схемотехнической точки зрения, такой способ использования доступных цифровых выводов микроконтроллера только для индикаторов не рационален. Этим и обусловлен значительный недостаток реализации многоразрядной индикации в статическом режиме.

Полный программный код для получения счетчика от 0 до 9 приведен на рис. 4.2. Приведенный код соответствует схеме (см. рис. 4.1.).

Текст

```
byte num[10] = {
  0b10110111, //0
  0b00010010, //1
  0b11010101, //2
  0b11010110, //3
  0b01110010, //4
  0b11100110, //5
  0b11100111, //6
  0b10010010, //7
  0b11110111, //8
  0b11110110, //9
};

void setup()
{
  for (int i = 0; i < 8; i++){
    pinMode(i + 2, OUTPUT);}
}

void loop()
{
  int number = (millis()/100)%10;
  int mask = num[number];
  for (int i = 0; i < 8; ++i) {
  digitalWrite(i + 2, bitRead(mask, i));
  }
 }
```

Рис. 4.2. Полный программный код для получения счетчика от 0 до 9

Анализ и описание работы программного кода.

В начале программы объявлен массив num типа byte содержащий 10 элементов 8 разрядных двоичных чисел. Данные числа представляют собой транскодировочный код предназначенный для преобразования адреса элемента массива в позиционный двоичный код, который далее выводится на цифровые выводы порта микроконтроллера для включения светодиодов семисегментного индикатора, чтоб отобразить требуемую цифру или символ.

В функции setup() использует цикл настройки 8 требуемых цифровых выводов микроконтроллера на выход, через которые осуществляется управление сегментами индикатора.

В основном цикле программы loop() используется функция таймера millis(), которая возвращает количество миллисекунд с момента начала выполнения текущей программы на плате Arduino. Это количество сбрасывается на ноль, в следствие переполнения, приблизительно через 50 дней. С целью замедлить счет, значение возвращаемое функцией millis() делится на 100, а затем при помощи оператора модуля «%» извлекается целая часть от деления и присваивается переменной number, где number в свою очередь используется для выбора элемента массива num. Выбранный элемент массива загружается в переменную mask, а затем в цикле из переменной mask функция bitRead извлекает поочередно биты для формирования высокого или низкого логического уровней на цифровых выводах микроконтроллера. В результате зажигающиеся сегменты отображающие цифру соответствующую значению переменной number.

Данная программа предназначена для работы с семисегментным индикатором включенным по схеме общего катода.

Программный код можно реализовать без использования функции bitRead, применяя двухмерный массив.

4.4.2. Динамическая индикация

Динамическая индикация может быть реализована используя микроконтроллер и транзисторы в качестве электронно-управляемых ключей (рис. 4.4).

Как упоминалось ранее, использование специализированных микросхем регистров хранения с дешифратором (декодером) двоичного кода в семисегментный (CD4511BCN) или сдвиговых регистров (74HC164) позволяет уменьшить количество используемых выводов микроконтроллера и добиться более компактной компоновки печатной платы конечного устройства и упрощения программного кода, в тех случаях когда используется два и более разряда семисегментных индикаторов.

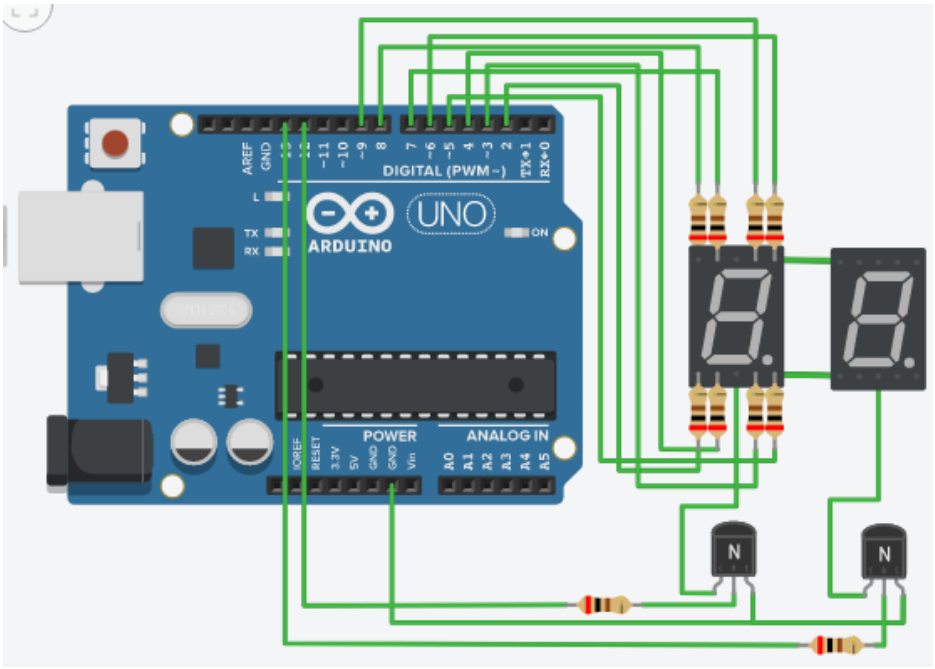


Рис. 4.4. Схема двух семисегментных индикаторов работающих в режиме динамической индикации

Программный код с реализацией циклического счета от 0 до 99 (рис. 4.5) при помощи функции millis().

```
15  void setup(){
16    for (int i = 0; i < 8; i++){
17      pinMode(i + 2, OUTPUT);}
18      pinMode(12, OUTPUT);
19      pinMode(13, OUTPUT);
20  }
21  void loop()
22  {
23    int ones = (millis()/1000)%10;
24    int tens = (millis()/100)%10;
25
26    int mask = num[ones];
27    digitalWrite(13, LOW);
28    for (int i = 0; i < 8; ++i) {
29    digitalWrite(i + 2, bitRead(mask, i));
30    }
31    digitalWrite(12, HIGH);
32    delay(10);
33    mask = num[tens];
34    digitalWrite(12, LOW);
35    for (int i = 0; i < 8; ++i) {
36    digitalWrite(i + 2, bitRead(mask, i));
37    }
38    digitalWrite(13, HIGH);
39    delay(10);
40   }
```

Рис. 4.5. Программный код динамической индикации для двух разрядов семисегментных индикаторов

По индивидуальному заданию преподавателя требуется реализовать схему с использованием специализированных микросхем и требуемым количеством разрядов с добавлением различных устройств ввода.

4.5. Порядок выполнения практической работы

4.5.1. Выполните последовательно все пункты теоретической части работы.

4.5.2. Реализуйте счетчик от 0 до 99 с использованием динамической индикации применяя одну из доступных специализированных микросхемы дешифраторов. Осуществите проверку схемы и программы в онлайн симуляторе.

4.5.3. Реализуйте схему на практике и убедитесь в ее работоспособности.

4.5.4. По индивидуальному заданию преподавателя синтезируйте схему и программный код для динамической индикации с применением сдвигового регистра или дешифратора (количество разрядов и тип микросхемы задается преоподавателем). Справочная информация в приложении А.

4.6. Содержание отчета

Цель работы.

Краткая теоретическая часть.

Снимок экрана индивидуально собранной схемы.

Код программы с комментариями.

Подробные выводы.

Приложение схемы начерченной в соответствии с ГОСТом и чертежной рамкой (схема из пункта 4.4.2).

4.7. Контрольные вопросы

4.7.1. Необходимость устройств вывода.

4.7.2. Внутреннее устройство семисегментного светодиодного индикатора.

4.7.3. Сравнение статической и динамической индикаций.

4.7.4. Устройство транскодировочной таблицы.

4.7.5. Применение специализированных микросхем для подключения семисегментных индикаторов.

4.7.6. Способы извлечения битов из числа байтового типа.

4.7.7. Фрагмент программного кода для формирования

Практическая работа № 5.
СЕКУНДОМЕР

5.1. Цель работы

Составить схему и программу секундомера.

5.2. Теоретическая часть

В настоящей практической работе необходимо реализовать динамическую индикацию с использованием семисегментных светодиодных индикаторов для вывода секунд. Дополнительно задействовать устройства ввода для возможности управления секундомером.

Для отсчета секунд могут использоваться доступные модули таймеров микроконтроллера ATMega328p на платформе «Arduino», функция millis() или модуль внешних часов реального времени.

Использование функции millis() позволяет получать значение миллисекунд с момента начала работы микроконтроллера, однако существенный недостаток использования данной функции заключается в том, что возвращаемое значение может происходить в разные моменты времени, потому возможны некоторое искажение счета секунд.

5.3. Составлением программного кода секундомера

Программный код для реализации динамической индикации в настоящей работе не приводится, так как рассматривался в предыдущих практических работах.

Реализация счета секунд с использованием функции millis(). Функция millis() возвращает целочисленное значение в формате long int. Данное значение имеет величину миллисекунд, которые отсчитываются с момента включения питания микроконтроллера. Наиболее важным моментом является получение значения секунд несколько раз в секунду для корректного счета самих секунд. Несложным математическим преобразованием извлекается значение секунд из возвращаемого значения функцией

$$T_S = millis()/1000\%10. \qquad (5.1)$$

В программной реализации данный код представлен на рис. 5.1.

Использование монитора последовательного порта показывает, что с использованием функции delay(500) переменная **a** принимает новое значение через два отображения в мониторе последовательного порта. Если задержку увеличить в два раза, тогда переменная **a** будет изменяться каждую секунду.

```
1  void setup()
2  {
3    pinMode(13, OUTPUT);
4    Serial.begin(9600);
5  }
6  int a=0;
7  void loop()
8  {
9    a = millis()/1000%10;
10   Serial.print("a = ");
11   Serial.println(a);
12   delay(500);
13 }
```

Монитор последовательного интерфейса

```
a = 0
a = 0
a = 1
a = 1
a = 2
a = 2
a = 3
```

Рис. 5.1. Программная реализация извлечения значения секунд

Одним из важнейших недостатков использования функции millis() состоит в том, что увеличение программного кода или использование дополнительных программных задержек может нарушить точность отсчета секунд, которое проявляется в виде отставания счета секундомера от реальных секунд. Другими словами счет секунд становится зависимым от объема программного кода, так как в реальном микроконтроллере увеличение программного кода требует большего времени для его выполнения, а функция millis() непрерывно увеличивает значение миллисекунд.

Реализация счета секунд с использованием прерываний от модуля таймера.

Недостаток использования функции millis() устраняется при использовании аппаратного модуля таймера, который генерирует прерывания со строго заданным интервалом времени. Программный код с использованием модуля таймера приведен на рис. 5.2.

```
volatile unsigned long int i;
int cnt=0;
ISR(TIMER0_COMPA_vect){
cnt++;
if (cnt == 1000) i++;
if (i == 59) i =0;
}

void setup()
{
  Serial.begin(9600);

  TCCR0A |= (1<WGM01);
  OCR0A = 0xF9;
  TIMSK0 |= (1<OCIE0A);
  TCCR0B |= (1 << CS01) | (1 << CS00);
  sei();

}

void loop()
{
  delay(500);
  Serial.println(i);

}
```

Рис. 5.2. Программа счета секунд с использованием прерываний от модуля таймера

Примечание — Программный код (рис. 5.2) не работает должным образом в симуляторе, поэтому приведенный код следует использовать с релаьным макетом «Arduino».

5.4. Порядок выполнения практической работы

5.4.1. Используя схемы предыдущих лабораторных работ, реализуйте схему динамической индикации. Реализуйте вывод секунд на семисегментный индикатор дополнив код, приведенный на рис. 5.1.

5.4.2. Следует выполнить реализация секундомера с использованием функции millis() и аппаратных прерываний от модуля таймера.

5.4.3. Дополните схему необходимым количеством устройств ввода для управления секундомером (старт, стоп, сброс).

5.4.4. После проверки схемы и программы в онлайн симуляторе, осуществите практическую сборку схемы, программирование и убедитесь в корректности работы схемы.

5.5. Содержание отчета

Цель работы.
Краткая теоретическая часть.
Снимок экрана симулятора с индивидуально собранной схемы.
Код программы с комментариями.
Подробные выводы.

Приложение схемы начерченной в соответствии с ГОСТом и чертежной рамкой (схема из пункта 5.5.3).

5.6. Контрольные вопросы

5.6.1. Способы реализации отсчета секунд.

5.6.2. Извлечение разрядов из переменной секунд для отображения на индикаторе.

5.6.3. Сравните программный метод и функцию millis() для отсчета секунд.

5.6.4. Предложите возможные способы вывода значений секунд.

5.6.5. Извлечение секунд из возвращаемого значения функцией millis() и при использовании обработчика прерываний по таймеру.

5.6.6. Сравнительный анализ использования таймера с аппаратным прерыванием и функции millis().

5.6.7. Фрагмент программы для формирования значений секунд, минут и часов.

5.6.8. Предложите способ формирования данных и отображения на устройстве вывода календаря.

Практическая работа № 6.
ЦИФРОВЫЕ ИЗМЕРИТЕЛИ

6.1. Цель работы

Изучить основы построения цифровых измерительных приборов с использованием микроконтроллера.

6.2. Теоретическая часть

При помощи микроконтроллеров стало возможным построение компактных измерительных приборов, таких как вольтметры, амперметры, ваттметры, частотомеры, термометры, измерители освещенности и многие другие. Цифровые измерительные приборы достаточно интенсивно замещают подобные аналоговые приборы. При этом цифровые измерительные приборы позволяют проводить измерения с большей точностью по отношению к аналоговым измерительным приборам.

Датчики можно разделить на две группы аналоговые и цифровые. Наиболее простыми в работе, однако наименее точными, являются аналоговые датчики. В большинстве случаев аналоговый датчик представляет собой резистивный элемент, сопротивление которого изменяется в зависимости от воздействующего фактора. Считывание аналоговых датчиков осуществляется при помощи модуля АЦП, где полученное напряжение выражается в виде пропорционального числа. Цифровые датчики представляют собой отдельный микроконтроллер, который осуществляет необходимые преобразования и передает по цифровому интерфейсу величину измеренного параметра в виде двоичного кода, далее МК принимает цифровой код и преобразовывает его нужный числовой формат. Для работы с цифровыми интерфейсами используется встроенные цифровые модули I2C, SPI или 1-wire.

В настоящей практической работе рассматривается построение простейших цифровых приборов, таких как вольтметр, амперметр и измеритель освещенности.

6.3. Построение и анализ схем

Для отображения измеряемых параметров необходимо реализовать схему индикации любым удобным способом, как описывалось в предыдущих практических работах.

Измерение напряжения. Измерение напряжения осуществляется при помощи параллельного подключения к измеряемому участку цепи. В случае если максимальное напряжение на измеряемом участке цепи не превышает напряжение питания микроконтроллера (опорное напряжение модуля АЦП определяющее верхнюю границу динамического диапазона), тогда дополнительная цепь делителя напряжения не нужна. В тех случаях, когда напряжение на измеряемом участке цепи превышает опорное напряжение, необходимо применять

делитель напряжения, с таким коэффициент деления, который бы позволил делить максимальное напряжение на измеряемом участке цепи до напряжения равного или меньшего, чем опорное напряжение. Схемы регулируемого и нерегулируемого делителей напряжение показаны на рис. 6.1.

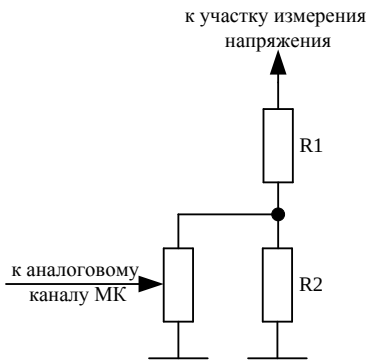


Рис. 6.1. Схема измерения напряжения нагрузки

Переменный резистор (см. рис. 6.1) позволяет точно настроить соответствие показания цифрового вольтметра с измеряемым напряжением.

Измерение тока нагрузки. Измерение величины тока осуществляется за счет измерения напряжения на низкоомном резисторе включенного последовательно с нагрузкой. В большинстве случаев сопротивление такого резистора выбирается как можно меньше с целью уменьшения тепловыделения и потери электрической мощности на нагревание резистора. Из-за малого значения напряжения протекающий ток создает относительно малое падение напряжения, которое следует предварительно усилить перед подачей на вход модуля АЦП. Усиление напряжения позволяет повысить точность измерения величины тока. Однако следует учитывать, что при усилении напряжения падения на токовом резисторе, усиленное максимальное напряжение не должно превышать опорного напряжения модуля АЦП. Схема цепи усиления напряжения на токовом резисторе показана на рис. 6.2.

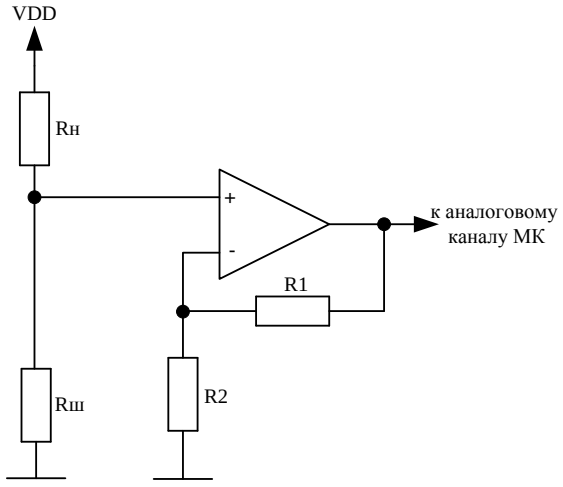


Рис. 6.2. Схема для измерения тока нагрузки

Номиналы резисторов обратной связи R_1 и R_2 задаются в зависимости от требуемого коэффициента усиления, таким образом, чтобы при заданном коэффициенте усиления максимальное напряжение на выходе операционного усилителя не превышало значения верхнего опорного напряжения модуля АЦП при наибольшем падении напряжения на токовом резисторе $R_Ш$ (шунте). Расчет резисторов обратной связи операционного усилителя включенного по не инвертирующей схеме осуществляется по следующей формуле:

$$K = 1 + \frac{R_1}{R_2}, \qquad (6.1)$$

где К — коэффициент усиления;

R_1 и R_2 — резисторы формирующие обратную связь и задающие коэффициент усиления.

6.4. Анализ измерительной части схемы

В качестве примера рассматривается схема измерения тока (рис. 6.3) в нагрузке при помощи резистора – шунта с сопротивлением 0,1 Ома и коэффициентом усиления операционного усилителя 10 раз (см. формулу 6.1).

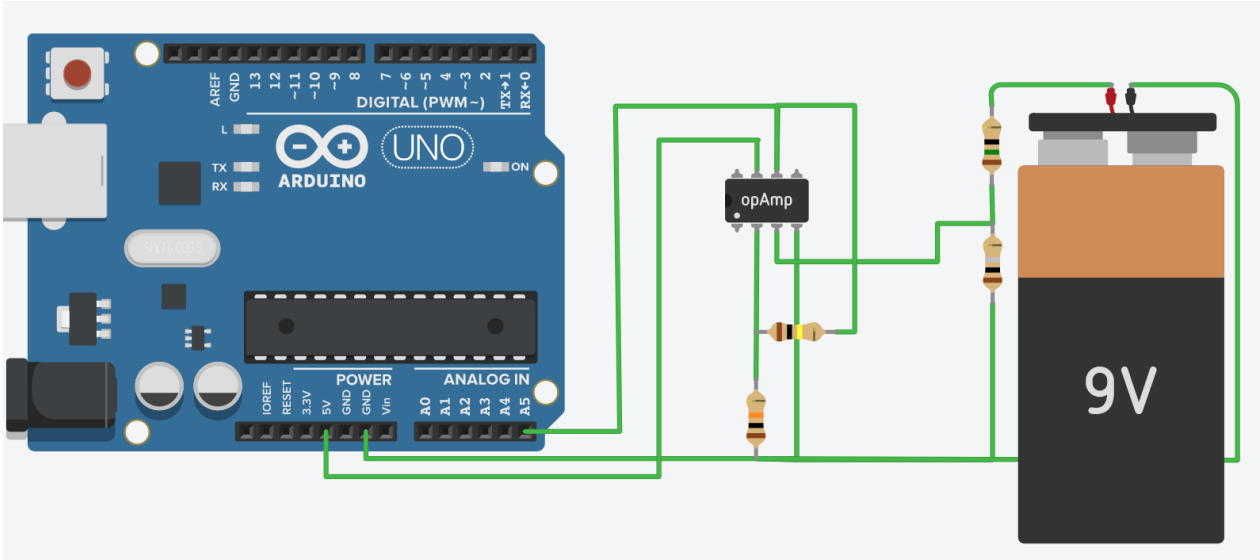


Рис. 6.3. Схема измерения тока нагрузки в симуляторе

Математическое преобразование.

Для правильного определения и отображения значения тока, протекающего в нагрузке следует учитывать коэффициент усиления операционного усилителя и сопротивление токового резистора (шунта).

Необходимо принять условное значение тока протекающего в цепи нагрузки 1 А, соответственно в результате преобразований на дисплее должно отобразиться фактическое значение тока нагрузки в 1 А, в свою очередь для этого необходимо выполнить корректные математические преобразования.

Если сопротивление токового резистора 0,1 Ома, тогда при токе в 1 А напряжение падения составит 0,1 В. Если выбрать коэффициент усиления операционного усилителя равным 10, значение напряжения, подаваемого на аналоговый вход микроконтроллера, составит 1 В. По известному пропорциональному значению, модуль АЦП сформирует значение, пропорциональное подаваемому напряжению, с некоторым приближением равным 204. Далее, следует в программе осуществить преобразование полученного результата аналого-цифрового преобразования к величине измеряемого значения тока нагрузки путем деления получаемого значения результата преобразования на 204 и извлечением разрядов единиц, десятков и сотен из результата деления для отображения на дисплее. Также следует учитывать степенной показатель величины измеряемого тока (10^{-3} мА).

Аналоговые датчики.

В настоящей работе рассматриваются способы использования модуля АЦП для измерения напряжения падения на резистивном элементе. Измерение параметров при помощи других аналоговых датчиков в большинстве случаев сводится к задаче эквивалентной измерению напряжения на центральном выводе потенциометра, где напряжение на выходе датчика изменяется в зависимости от воздействующего параметра. В случаях измерения относительно малых величин напряжений может отрицательно сказываться на точности измеряемых параметров, поэтому для повышения точности измеряемых параметров может использоваться усилитель напряжения на операционном усилителе.

6.5. Порядок выполнения практической работы

6.5.1. Реализуйте схему динамической индикации с использованием микросхем дешифраторов для трех разрядного дисплея (приложение Г).

6.5.2. Проведите симуляцию трех схем цифровых измерителей вольтметра, амперметра и измерителя освещенности.

6.5.3. Реализуйте одну из схем на макетной плате, осуществите программирование и убедитесь в корректности работы схемы.

6.6. Содержание отчета

Цель работы.

Краткая теоретическая часть.

Снимок экрана индивидуально собранной схемы.

Код программы с комментариями.

Подробные выводы.

Приложение схемы начерченной в соответствии с ГОСТом и чертежной рамкой (схема из пункта 6.4.3).

6.7. Контрольные вопросы

6.7.1. Способы измерения напряжения на участке цепи.

6.7.2. Способы измерения величины тока, протекающего в нагрузке.

6.7.3. Схема усиления постоянного напряжение.

6.7.4. Опорное напряжение модуля АЦП.

6.7.5. Расчет коэффициента пропорциональности между величиной измеряемого напряжения и результатом аналого-цифрового преобразования.

6.7.6. Способ повышения точности измеряемых параметров при использовании аналоговых датчиков.

Практическая работа № 7.
ЖИДКОКРИСТАЛЛИЧЕСКИЙ СИМВОЛЬНЫЙ LCD ДИСПЛЕЙ

7.1. Цель работы

Изучить схему подключения буквенно-цифрового (символьного) жидкокристаллического LCD дисплея и функций для вывода информации на дисплей.

7.2. Теоретические сведения

Микроконтроллер представляет собой микроаналог системного блока персонального компьютера, реализованного на одном кристалле, однако у него нет устройств ввода и вывода информации. В большинстве случаев взаимодействие с программой микроконтроллера без использования устройств вывода невозможно. В качестве устройств вывода могут использоваться светодиоды для вывода логического состояния или чисел в двоичном формате, семисегментные светодиодные индикаторы для вывода цифровых и некоторых буквенных символов и для вывода более полной и сложной информации применяются жидкокристаллические дисплеи буквенно-цифровые или графические (как цветные, так и монохромные).

В настоящей практической работе рассматривается применение буквенно-цифрового 16 символьного двухстрочного дисплея, подключенного к микроконтроллеру и работающего в параллельном полубайтовом режиме передачи данных.

7.3. Схема включения

Схема подключения дисплея в симуляторе представлена на рис. 6.1.

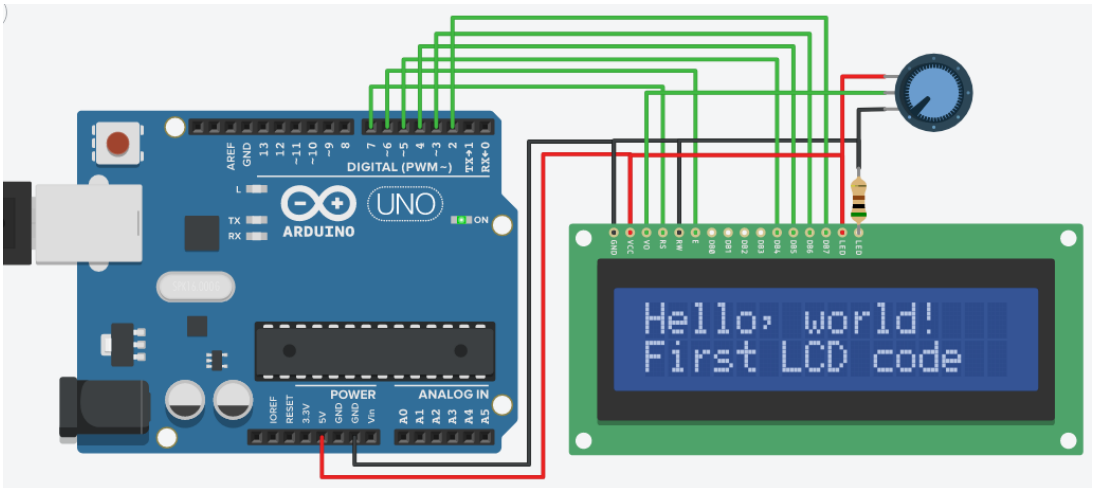


Рис. 7.1. Схема подключения буквенно-цифрового дисплея

Жидкокристаллический LCD дисплей подключен в полубайтовом параллельном режиме (данный режим удобен меньшим количеством используемых выводов) с использованием дополнительного резистора в цепи светодиодной

подсветки дисплея номиналом в 500 Ом и подстроечным резистором для регулировки контрастности изображения на дисплее.

Для подключения выводов дисплея и платформы «Arduino» приведена таблица 7.1.

Таблица 7.1 Таблица подключения выводов платформы «Arduino» и LCD дисплея

Выводы платформы Arduino	Выводы LCD дисплея
D2	DB7
D3	DB6
D4	DB5
D5	DB4
D6	E
GND	RW
D7	RS
5V	VCC
GND	GND
5V	LED+
GND	LED- (резистор 500 Ом)

7.4. Анализ программы

Пример программного кода с инициализацией дисплея и выводом символьной информации показан на рис. 7.2.

```
#include <LiquidCrystal.h>
// Библиотека для работы с LCD
LiquidCrystal lcd(7, 6, 5, 4, 3, 2);
// (RS, E, DB4, DB5, DB6, DB7)

void setup(){
  lcd.begin(16, 2);                 // Размерность экрана

  lcd.setCursor(0, 0);              // Установка курсора(x,y)
  lcd.print("Hello, world!");       // Вывод текста
  lcd.setCursor(0, 1);
  lcd.print("First LCD code");
}

void loop(){
}
```

Рис. 7.2. Пример программного кода инициализации и вывода данных на дисплей

В примере программы (рис. 7.2) приводится минимальный программный код необходимый для инициализации жидкокристаллического дисплея, позиционирования курсора и вывода простой строковой информации. Все используемые функции для инициализации, управления и вывода информации на дисплей содержатся в библиотечном файле «LiquidCrystal.h», данный библио-

течный файл требуется подключить при помощи директивы «include» в самом начале программного кода в месте с функцией назначения выводов для работы с дисплеем «LiquidCrystal lcd()». Описание функций приводится в таблице 6.2.

Таблица 6.2 Описание функций для работы с дисплеем

Функции библиотеки LiquidCrystal.h для работы с дисплеем	Описании функции
LiquidCrystal()	Инициализации выводов дисплея (используется вне шаблонных функций среды Arduino)
begin()	Определение количества символов в строке и количества строк дисплея
clear()	Очистка всего дисплея
home()	Установка курсора в крайнее верхнее левое положение
setCursor()	Установка курсора в нужном месте, параметры x и y.
write()	Вывод информации на дисплей
print()	Вывод данных на дисплей
cursor()	Отображение положения курсора
noCursor()	Отключения отображения положения курсора
blink()	Мигание курсора
noBlink()	Выключение мигания курсора
display()	Включение дисплея
noDisplay()	Выключение дисплея
scrollDisplayLeft()	Прокрутка информации изображения на один символ влево
scrollDisplayRight()	Прокрутка информации изображения на один символ вправо
autoscroll()	Включение смещения экрана на один символ влево при добавлении символа
noAutoscroll()	Отключение смещения экрана
leftToRight()	Порядок вывода символов на экран слева на право
rightToLeft()	Порядок вывода символов на экран справа на лево
createChar()	Создание пользовательских символов
LiquidCrystal()	Функция назначения выводов для взаимодействия дисплея с микроконтроллером
begin()	Инициализация дисплея с указанием количества строк и символов в строке

7.5. Порядок выполнения практической работы

7.5.1. Осуществите повторную сборку схемы и написание программного кода, показанного на рис. 7.1 и рис. 7.2.

7.5.2. Внесите в имеющуюся схем изменения в виде устройств ввода: двух кнопочных контактов и потенциометра. При помощи кнопочных контактов необходимо одной парой кнопочных контактов инкрементировать, а другой декрементировать число от 0 до 100, данное число необходимо отображать в верхней строке LCD дисплея по центру. При помощи потенциометра требуется изменять напряжение на одном из аналоговых каналов и выводить результат аналого-цифрового преобразования во второй строке LCD дисплея в крайнем правом положении.

7.6. Содержание отчета

Цель работы
Снимок экрана симулятора со схемой
Код программы с подробными объяснениями.
Сборка реальной схемы с проверкой работоспособности.
Подробные выводы.

7.7. Контрольные вопросы

7.7.1. Виды средств вывода информации при помощи МК.

7.7.2. Виды жидкокристаллических дисплеев.

7.7.3. Способы подключения LCD дисплея, рассматриваемого в практической работе.

7.7.4. Основные функции для настройки и вывода информации на LCD дисплей.

7.7.5. Сравнительный анализ буквенно-цифрового LCD дисплея и семисегментного светодиодного индикатора.

7.7.6. Основные функции для работы с курсором и выводом информации на буквенно-цифровой LCD дисплей.

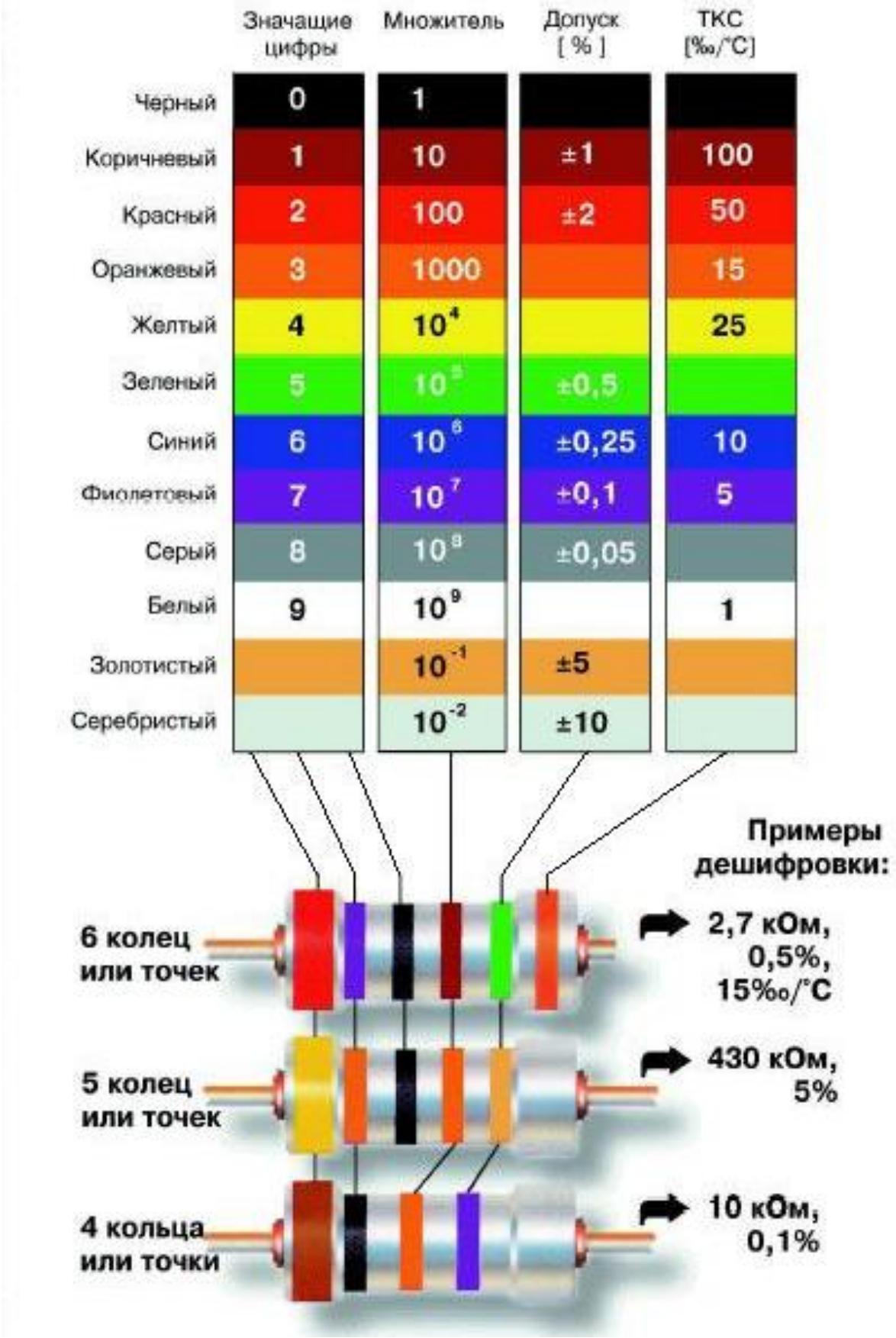
БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Бокселл Д. Изучаем Arduino. 65 проектов своими руками. — СПБ.: Питер, 2017. — 400 с.: ил. — (Серия «Вы и ваш ребенок»). ISBN 978-5-496-02421-1

2. Блум Д. Изучаем Arduino. Инструменты и методы технического волшебства: Пер. с англ. — СПб.: БХВ-Петербург, 2015. — 336 с. ISBN 978-5-9775-3585-4

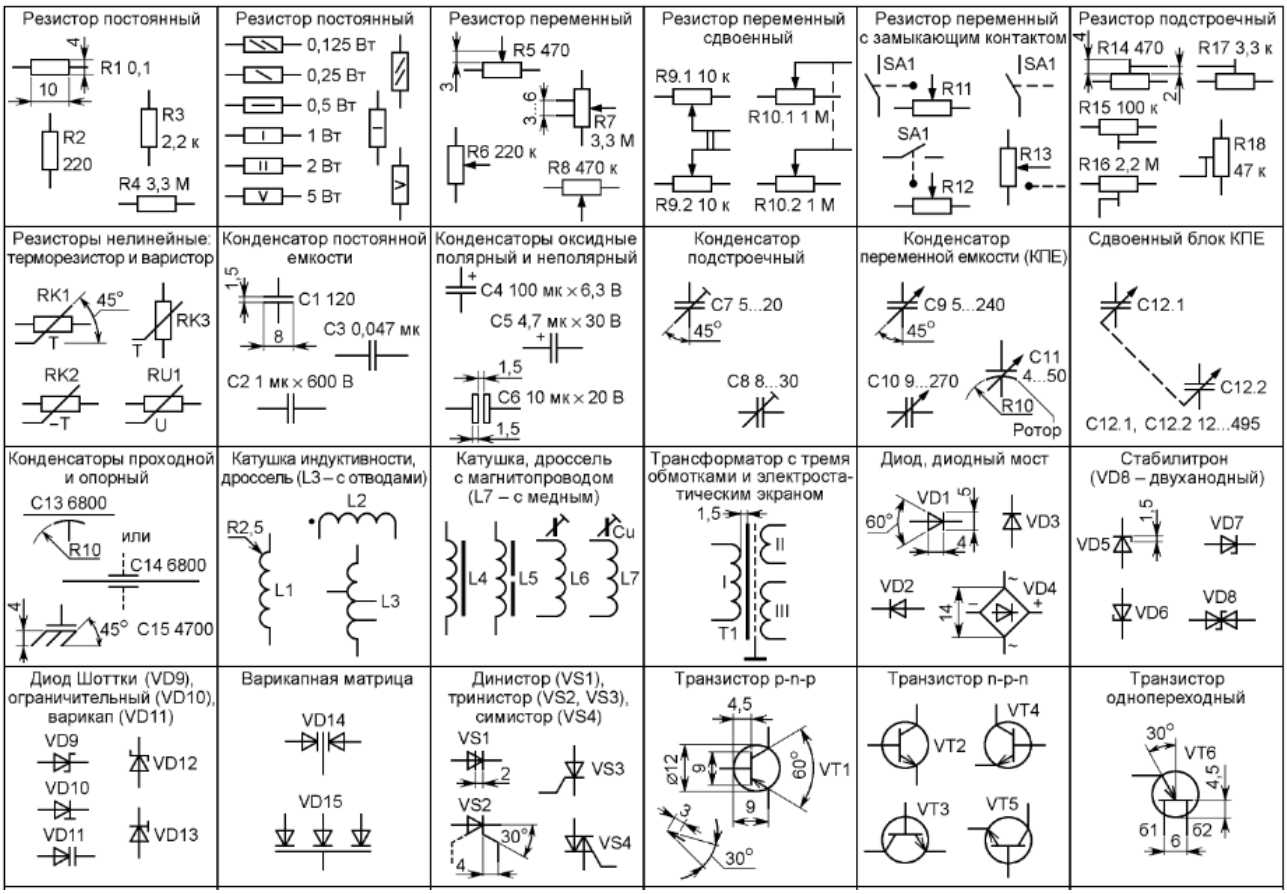
ПРИЛОЖЕНИЕ А

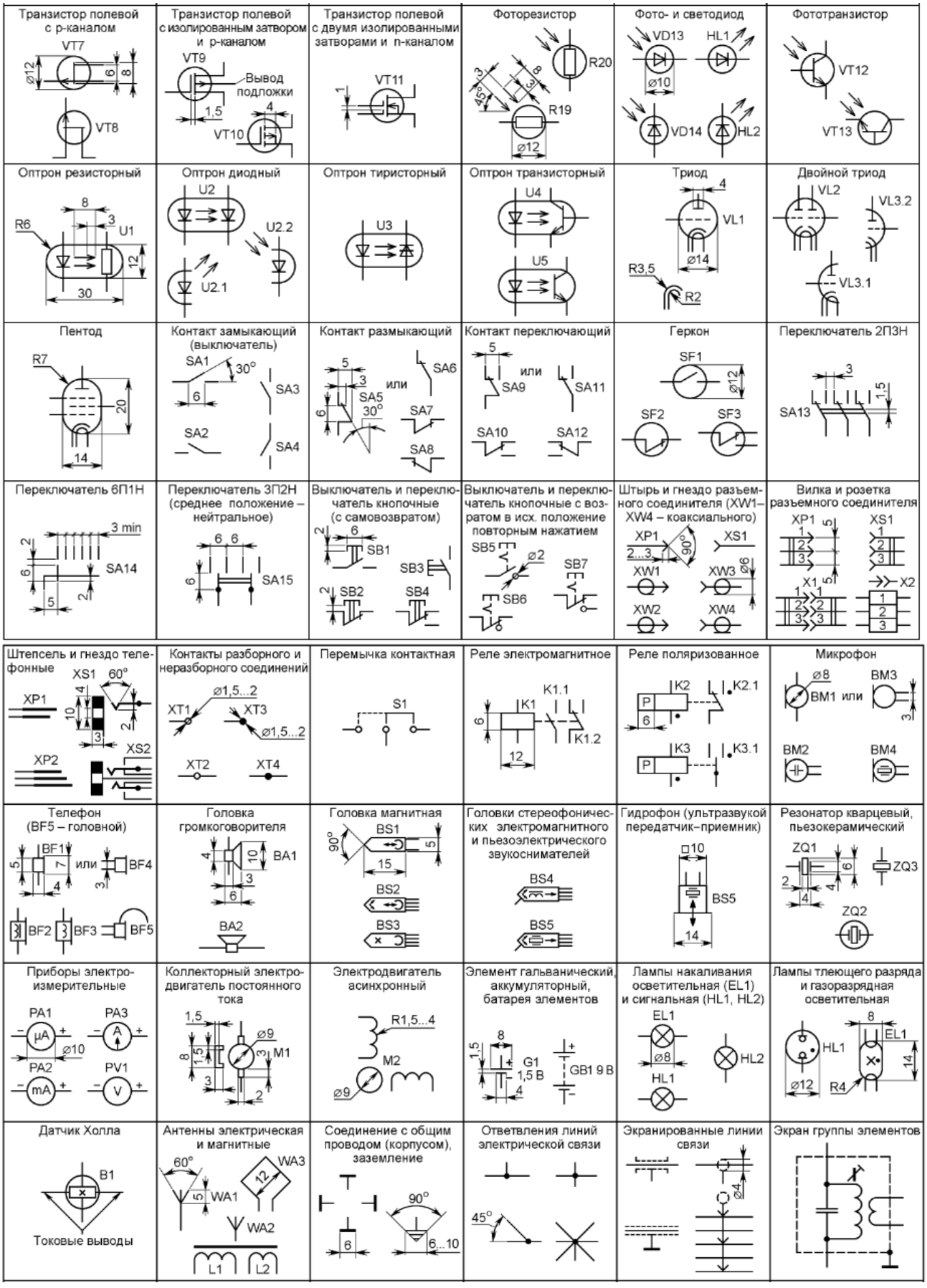
ЦВЕТОВАЯ МАРКИРОВКА РЕЗИСТОРОВ

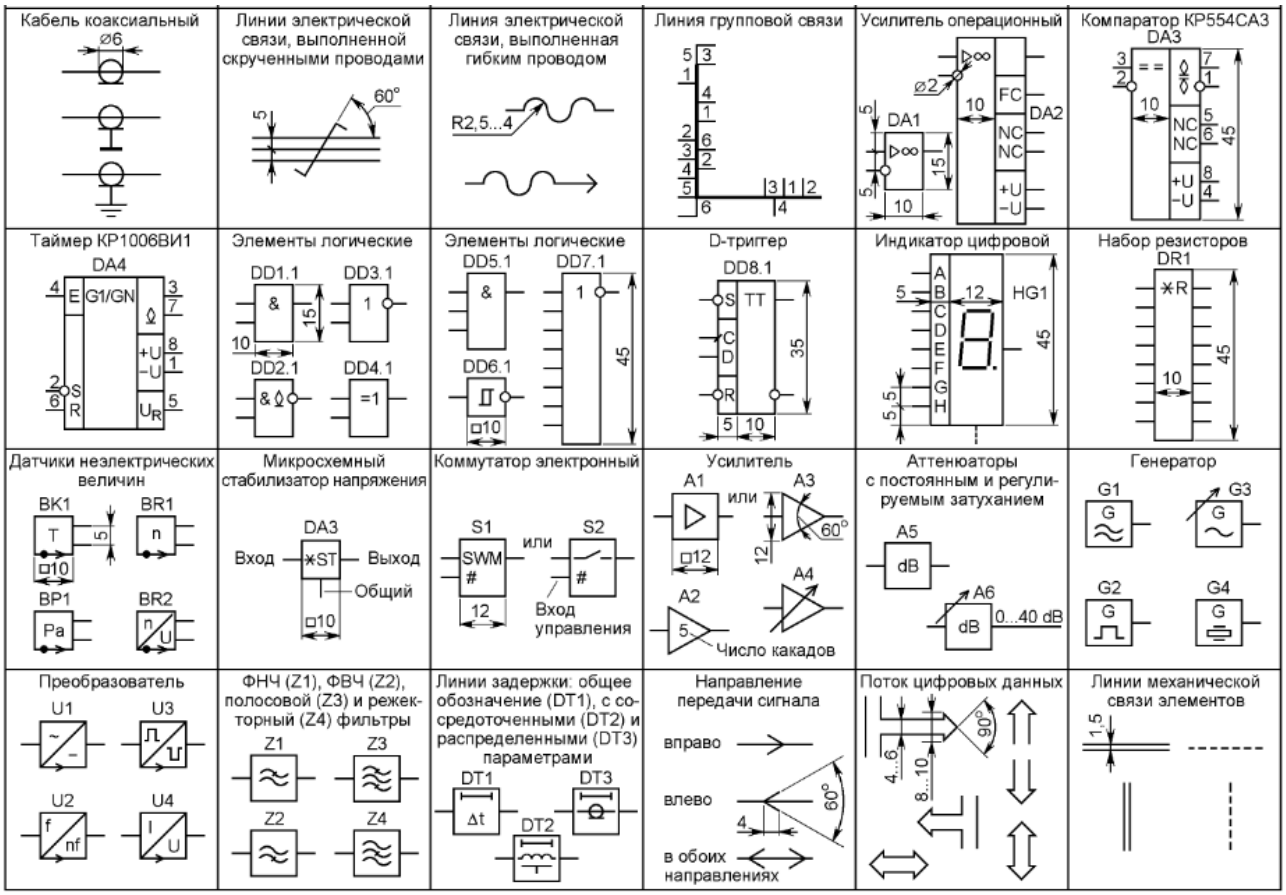


ПРИЛОЖЕНИЕ Б

ТРЕБОВАНИЯ К УСЛОВНО-ГРАФИЧЕСКИМ ОБОЗНАЧЕНИЯМ КОМПОНЕНТОВ СОГЛАСНО ЕСКД




Транзистор полевой с р-каналом
VT7
VT8
Транзистор полевой с изолированным затвором и р-каналом
VT9
Вывод подложки
VT10
Транзистор полевой с двумя изолированными затворами и n-каналом
VT11
Фоторезистор
R20
R19
Фото- и светодиод
VD13
HL1
VD14
HL2
Фототранзистор
VT12
VT13
Оптрон резисторный
R6
U1
Оптрон диодный
U2
U2.1
U2.2
Оптрон тиристорный
U3
Оптрон транзисторный
U4
U5
Триод
VL1
R3,5
R2
Двойной триод
VL2
VL3.2
VL3.1
Пентод
R7
Контакт замыкающий (выключатель)
SA1
SA2
SA3
SA4
Контакт размыкающий
SA5
SA6
SA7
SA8
Контакт переключающий
SA9
SA10
SA11
SA12
Геркон
SF1
SF2
SF3
Переключатель 2П3Н
SA13
Переключатель 6П1Н
SA14
Переключатель 3П2Н (среднее положение – нейтральное)
SA15
Выключатель и переключатель кнопочные (с самовозвратом)
SB1
SB2
SB3
SB4
Выключатель и переключатель кнопочные с возвратом в исх. положение повторным нажатием
SB5
SB6
SB7
Штырь и гнездо разъемного соединителя (XW1–XW4 – коаксиального)
XP1
XS1
XW1
XW2
XW3
XW4
Вилка и розетка разъемного соединителя
XP1
XS1
X1
X2
Штепсель и гнездо телефонные
XP1
XS1
XP2
XS2
Контакты разборного и неразборного соединений
XT1
XT2
XT3
XT4
Перемычка контактная
S1
Реле электромагнитное
K1.1
K1
K1.2
Реле поляризованное
K2
K2.1
K3
K3.1
Микрофон
BM1
BM2
BM3
BM4
Телефон (BF5 – головной)
BF1
BF2
BF3
BF4
BF5
Головка громкоговорителя
BA1
BA2
Головка магнитная
BS1
BS2
BS3
Головки стереофонических электромагнитного и пьезоэлектрического звукоснимателей
BS4
BS5
Гидрофон (ультразвукой передатчик–приемник)
BS5
Резонатор кварцевый, пьезокерамический
ZQ1
ZQ2
ZQ3
Приборы электроизмерительные
PA1
PA2
PA3
PV1
Коллекторный электродвигатель постоянного тока
M1
Электродвигатель асинхронный
M2
R1,5...4
Элемент гальванический, аккумуляторный, батарея элементов
G1
1,5 В
GB1 9 В
Лампы накаливания осветительная (EL1) и сигнальная (HL1, HL2)
EL1
HL1
HL2
Лампы тлеющего разряда и газоразрядная осветительная
EL1
HL1
R4
Датчик Холла
B1
Токовые выводы
Антенны электрическая и магнитные
WA1
WA2
WA3
L1
L2
Соединение с общим проводом (корпусом), заземление
Ответвления линий электрической связи
Экранированные линии связи
Экран группы элементов


Кабель коаксиальный
Линии электрической связи, выполненной скрученными проводами
Линия электрической связи, выполненная гибким проводом
Линия групповой связи
Усилитель операционный
Компаратор КР554СА3
Таймер КР1006ВИ1
Элементы логические
Элементы логические
D-триггер
Индикатор цифровой
Набор резисторов
Датчики неэлектрических величин
Микросхемный стабилизатор напряжения
Коммутатор электронный
Усилитель
Аттенюаторы с постоянным и регулируемым затуханием
Генератор
Преобразователь
ФНЧ (Z1), ФВЧ (Z2), полосовой (Z3) и режекторный (Z4) фильтры
Линии задержки: общее обозначение (DT1), с сосредоточенными (DT2) и распределенными (DT3) параметрами
Направление передачи сигнала
Поток цифровых данных
Линии механической связи элементов

(По основным надписям: http://gk-drawing.ru/plotting/inscription.php)

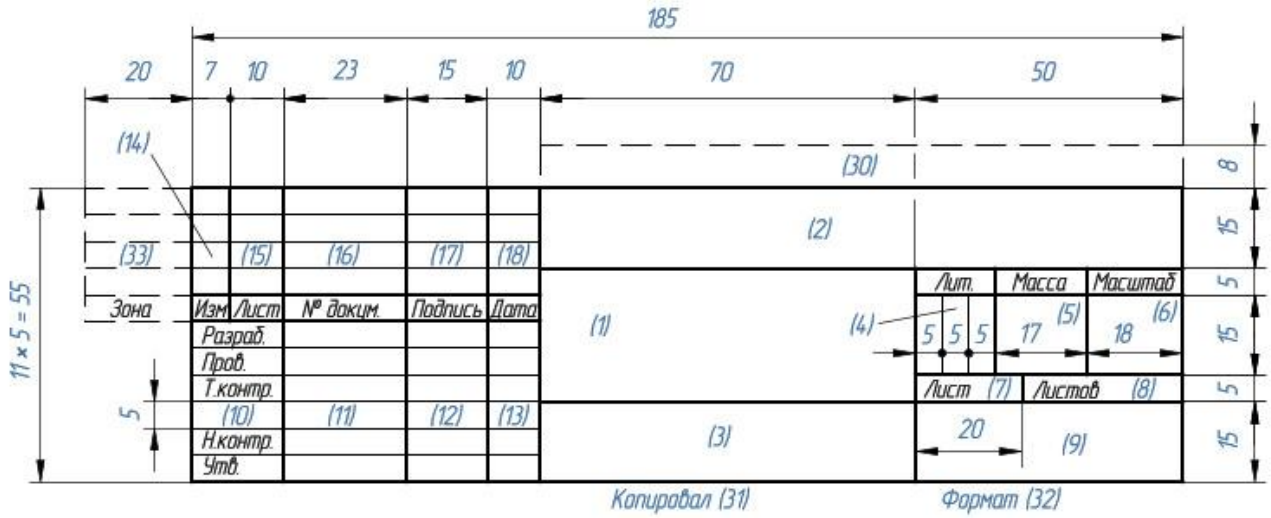


Рис.Б.1 — Основная надпись для чертежей и схем

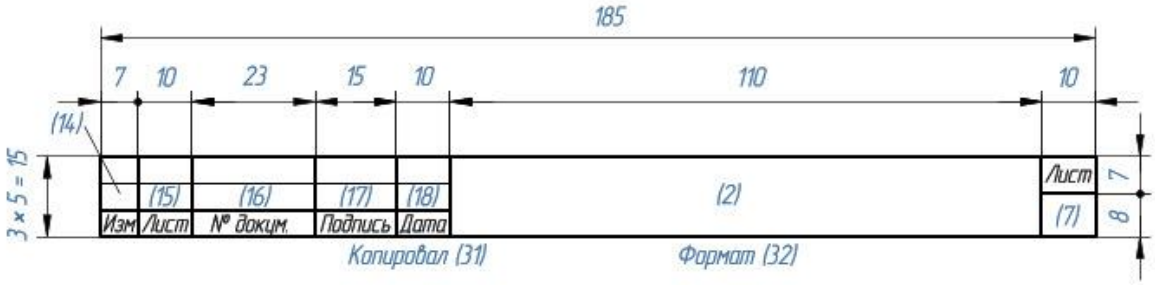


Рис. Б.2 — Основная надпись для последующих листов чертежей, схем и текстовых документов

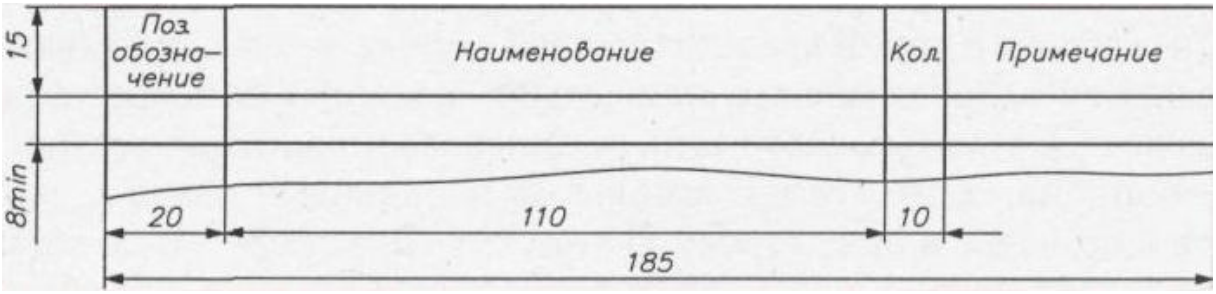


Рис. Б.3 — Таблица для перечня элементов

ПРИЛОЖЕНИЕ В

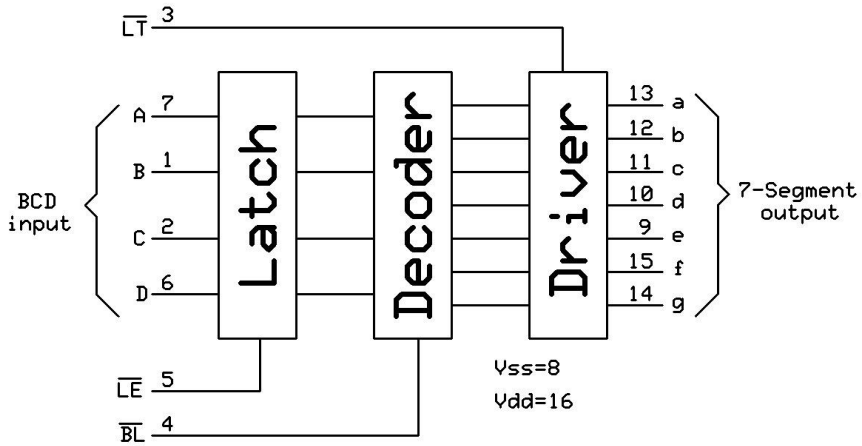
ОСНОВНЫЕ ФУНКЦИИ ЯЗЫКА С ДЛЯ ПЛАТФОРМЫ ARDUINO

Основные функции Arduino среды	Описании функции
pinMode()	Настройка направленности вывода
digitalWirte()	Формирование логического уровня на выводе настроенном как выход
digitalRead()	Считывание логического уровня с вывода настроенного как вход
analogRead()	Запуск аналого-цифрового преобразования
analogWirte()	Управление коэффициентом заполнения ШИМ сигнала
Serial.begin()	Инициализация последовательного порта
Serial.println()	Вывод данных в последовательный порт
delay()	Функция генерации программной задержки в миллисекундах

ПРИЛОЖЕНИЕ Г

СПРАВОЧНАЯ ИНФОРМАЦИЯ МИКРОСХЕМ ДЛЯ РАБОТЫ С СЕМИСЕГМЕНТНЫМ ИНДИКАТОРОМ

Внутреннее устройство и описание дешифратора 74HC4511



Назначение выводов:

LE — при подаче активного уровня (0) содержимое буфера передается на декодер.

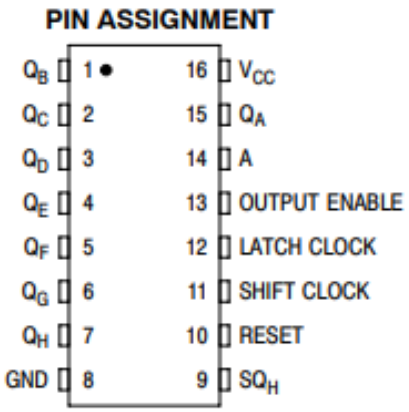
BL — включение выходов декодера

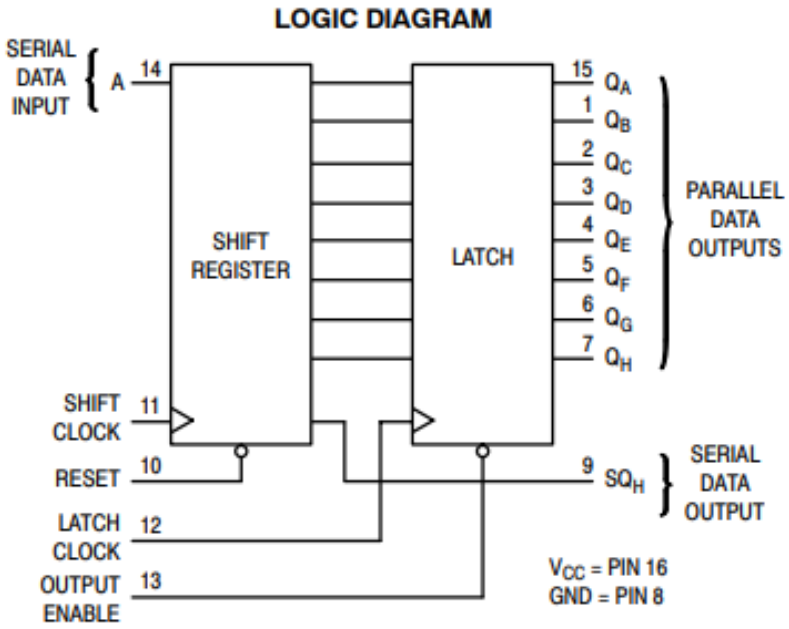
LT (lamp test) — при подаче активного уровня (0) входные уровни драйвера передаются на выходы драйвера.

Функциональная таблица дешифратора

INPUTS							OUTPUTS							
$\overline{LE}$	$\overline{BL}$	$\overline{LT}$	D_3	D_2	D_1	D_0	a	b	c	d	e	f	g	DISPLAY
X	X	L	X	X	X	X	H	H	H	H	H	H	H	8
X	L	H	X	X	X	X	L	L	L	L	L	L	L	Blank
L	H	H	L	L	L	L	H	H	H	H	H	H	L	0
L	H	H	L	L	L	H	L	H	H	L	L	L	L	1
L	H	H	L	L	H	L	H	H	L	H	H	L	H	2
L	H	H	L	L	H	H	H	H	H	H	L	L	H	3
L	H	H	L	H	L	L	L	H	H	L	L	H	H	4
L	H	H	L	H	L	H	H	L	H	H	L	H	H	5
L	H	H	L	H	H	L	L	L	H	H	H	H	H	6
L	H	H	L	H	H	H	H	H	H	L	L	L	L	7
L	H	H	H	L	L	L	H	H	H	H	H	H	H	8
L	H	H	H	L	L	H	H	H	H	L	L	H	H	9
L	H	H	H	L	H	L	L	L	L	L	L	L	L	Blank
L	H	H	H	L	H	H	L	L	L	L	L	L	L	Blank
L	H	H	H	H	L	L	L	L	L	L	L	L	L	Blank
L	H	H	H	H	L	H	L	L	L	L	L	L	L	Blank
L	H	H	H	H	H	L	L	L	L	L	L	L	L	Blank
L	H	H	H	H	H	H	L	L	L	L	L	L	L	Blank
H	H	H	X	X	X	X	†	†	†	†	†	†	†	†

Внутреннее устройство и описание сдвигового регистра 74HC595





Функции выводов:

Serial Data Input — вход последовательных данных.

Shift Clock — тактовый сигнал управления сдвиговым регистром.

Reset — сигнал сброса.

Latch Clock — тактовый сигнал выходного буфера.

Output Enable — разрешение вывода логических уровней на выводы буфера.

Учебное издание

ПРАКТИКУМ ДИСЦИПЛИНЕ «ПРАКТИЧЕСКИЕ ОСНОВЫ РАБОТЫ С ИНТЕЛЛЕКТУАЛЬНЫМИ ПРОГРАММИРУЕМЫМИ СИСТЕМАМИ (AVR)»

Часть 1

Учебно-методическое пособие

Составители: **Скорик** Иван Викторович,
Дурманов Максим Анатольевич

В авторской редакции

Изд. № 88/2020. Объем 3 п.л.
РИИЦМ ФГАОУ ВО «Севастопольский государственный университет»