ОГЛАВЛЕНИЕ

От авторов **6**
Введение **7**
1. Технические характеристики Arduino UNO R3 ..... **8**
2. Программирование контроллера Arduino UNO R3 ..... **11**
2.1. Среда разработки программ для контроллера 11
2.2. Общие сведения о программировании контроллера 11
2.3. Программная обработка прерываний 12
2.4. Подключение программных модулей и библиотек 13
3. Порядок выполнения лабораторных и практических работ **14**
4. Состав отчёта о выполненной работе **16**
5. Подключение к контроллеру Arduino UNO двух светодиодов **17**
5.1. Цель выполнения работы 17
5.2. Необходимые комплектующие и инструменты 17
5.3. Теоретический материал 17
5.4. Задание на выполнение лабораторной работы 19
5.5. Вопросы для самопроверки 20
6. Подключение к контроллеру Arduino UNO светодиода и кнопки **21**
6.1. Цель выполнения работы 21
6.2. Необходимые комплектующие и инструменты 21
6.3. Теоретический материал 21
6.4. Задание на выполнение лабораторной работы 25
6.5. Вопросы для самопроверки 25
7. Подключение к контроллеру Arduino UNO семисегментного светодиодного индикатора **26**
7.1. Цель выполнения работы 26
7.2. Необходимые комплектующие и инструменты 26
7.3. Теоретический материал 26
7.4. Задание на выполнение лабораторной работы 28
7.5. Вопросы для самопроверки 29
8. Подключение к контроллеру Arduino UNO датчика температуры и индикация измеренного значения **30**
8.1. Цель выполнения работы 30
8.2. Необходимые комплектующие и инструменты 30
8.3. Теоретический материал 30
8.3.1. Датчики температуры 30
8.3.2. Простое скользящее среднее 33
8.3.3. Светодиодные многоразрядные семисегментные индикаторы 33
8.3.4. Последовательный периферийный интерфейс SPI 35

8.3.5. Библиотека класса LedControl ... 36
8.3.6. Алгоритм кодирования индикации сегментов ... 36
8.3.7. Дисплей LCD 1602 ... 37
8.3.8. Библиотека класса LiquidCrystal ... 38
8.3.9. Дисплей LCD 1602 с интерфейсом I²C ... 39
8.3.10. Последовательная асимметричная шина I²C ... 40
8.3.11. Библиотека класса LiquidCrystal_I2C ... 40
8.4. Задание на выполнение лабораторной работы ... 41
8.5. Вопросы для самопроверки ... 43
9. Подключение к контроллеру источника питания и индикация измеренного напряжения ... 44
9.1. Цель выполнения работы ... 44
9.2. Необходимые комплектующие и инструменты ... 44
9.3. Теоретический материал ... 44
9.3.1. Резистивный делитель напряжения ... 44
9.3.2. Экспоненциальное скользящее среднее ... 45
9.3.3. Многоразрядный модуль семисегментных светодиодных индикаторов на базе сдвиговых регистров TM74HC595 ... 45
9.3.4. Источники постоянного тока ... 46
9.4. Задание на выполнение лабораторной работы ... 47
9.5. Вопросы для самопроверки ... 47
10. Подключение к контроллеру электродвигателя постоянного тока, переменного резистора и индикация оборотов на TFT-дисплее ... 48
10.1. Цель выполнения работы ... 48
10.2. Необходимые комплектующие и инструменты ... 48
10.3. Теоретический материал ... 48
10.3.1. Электродвигатели постоянного тока ... 48
10.3.2. Переменные резисторы ... 50
10.3.3. Тахогенератор ... 51
10.3.4. Цветной дисплей 1.44" TFT LCD ... 52
10.3.5. Библиотека класса TFT ... 53
10.3.6. Широтно-импульсная модуляция ... 54
10.3.7. Транзисторы ... 55
10.3.8. Диоды ... 56
10.3.9. Конденсаторы ... 57
10.3.10. Дроссели ... 58
10.4. Задание на выполнение лабораторной работы ... 59
10.5. Вопросы для самопроверки ... 60
Приложение А. Примеры программ управления Arduino UNO ... 61
А.1. Программа управления индикацией двух светодиодов ... 61
А.2. Программа управления индикацией светодиода нормально разомкнутой кнопкой ... 61
А.3. Программа управления индикацией семисегментного индикатора ... 62
А.4. Программы управления индикацией измеренной температуры ... 64

А.4.1. Модуль измерения температуры (readTempr.ino) 64
А.4.2. Программа измерения температуры и индикация на многоразрядный модуль семисегментных светодиодных индикаторов (восемь) на базе микросхемы MAX7219 65
А.4.3. Программа измерения температуры и индикация измеренного значения на индикаторе LCD 1602 67
А.4.4. Программа измерения температуры и индикация измеренного значения на индикаторе LCD 1602 по I^2C-интерфейсу 68
А.5. Программы управления индикацией измеренного напряжения источника питания 70
А.5.1. Заголовочный файл библиотеки управления сборкой светодиодных индикаторов на базе сдвиговых регистров TM74HC595 (SevenSegment_TM74HC595.h) 70
А.5.2. Файл реализации класса библиотеки управления сборкой светодиодных индикаторов на базе сдвиговых регистров TM74HC595 (SevenSegment_TM74HC595.cpp) 71
А.5.3. Программа измерения напряжения питания и индикация на индикаторе на базе сдвиговых регистров TM74HC595 73
А.6. Программы управления электродвигателем постоянного тока 74
А.6.1. Модуль индикации оборотов электродвигателя на TFT-дисплее (graf.ino) 74
А.6.2. Программа управления электродвигателем постоянного тока 76
Приложение Б. Программа расчёта номинала токоограничивающего резистора на языке Python 79
Приложение В. Программа расчёта напряжения на выводах конденсатора в зависимости от времени на языке Python 81
Приложение Г. Программа расчёта номиналов резисторов делителя напряжений на языке Python 82

ОТ АВТОРОВ

При выполнении лабораторных работ необходимы базовые знания языка программирования C/C++, а также знания основных характеристик линейки контроллеров Arduino и начальные знания об электронных компонентах. Для этого нужно ознакомиться с материалами.

1. *Керниган, Б. У.* Язык программирования C / Б. У. Керниган, Д. М. Ритчи ; пер. с англ. — 2-е изд., перераб. и доп. — М. : ИД «Вильямс», 2019. — 288 с.

2. *Страуструп, Б.* Язык программирования C++ : пер. с англ. — М. : Бином, 2012. — 1136 с.

3. *Платт, Ч.* Электроника для начинающих : пер. с англ. — 2-е изд. — СПб. : БХВ-Петербург, 2017. — 416 с.

4. Блум, Д. Изучаем Arduino. Инструменты и методы технического волшебства : пер. с англ. — 2-е изд. — СПб. : БХВ-Петербург, 2020. — 544 с.

5. Аппаратная платформа Arduino [Электронный ресурс]. — URL: https://arduino.ru/ (дата обращения 01.09.2023).

ВВЕДЕНИЕ

Arduino — это удобная платформа быстрой разработки электронных устройств для обучения и практического применения. Устройство программируется непосредственно из среды разработки Arduino, установленной на персональный компьютер через интерфейс USB без использования программаторов.

Устройства на базе Arduino могут получать информацию посредством различных датчиков, пультов управления и электрических сигналов, информировать о состоянии технологического объекта управления с помощью различного вида индикаторов, табло, а также могут управлять различными исполнительными устройствами. К контроллеру могут быть подключены платы расширения, например плата расширения сети Ethernet Shield W5100.

Существует несколько версий платформ Arduino. Версия Leonardo базируется на микроконтроллере ATmega32u4, UNO и Nano построены на микроконтроллере ATmega328, Mega2560 построена на микроконтроллере ATmega2560, а также имеются другие.

Для выполнения практических работ будет использоваться платформа Arduino UNO R3. На рисунке 1 приведён общий вид контроллера Arduino UNO R3.

Рис. 1

Общий вид контроллера Arduino UNO R3

1. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ ARDUINO UNO R3

Технические характеристики контроллера Arduino UNO R3 представлены в таблице 1.

Таблица 1

Характеристики контроллера Arduino UNO R3

№ п/п	Характеристика	Значение
1	Микроконтроллер:	ATmega328P
1.1	Ядро	8-битный AVR
1.2	Тактовая частота	16 МГц
1.3	Флеш-память	32 Кб, из которых 0,5 Кб используется для загрузчика
1.4	RAM-память	2 Кб
1.5	EEPROM-память	1 Кб
2	Порты входов/выходов общего назначения	20
2.1	Цифровые входы/выходы	14
2.1.1	– из них цифровые пины с ШИМ	6
2.2	Аналоговые входные порты (порты с АЦП)	6
2.3	Порты с прерыванием	2
3	Разрядность АЦП	10 бит
4	Разрядность ШИМ	8 бит
5	Аппаратные интерфейсы	1× UART, 1× I²C, 1× SPI
6	Напряжение логических уровней	5 В
7	Входное напряжение питания:	
7.1	через USB	5 В
7.2	через DC-разъём или пин VIN	7,5–12 В
8	Максимальный выходной ток пина 3.3V	150 мА
9	Максимальный выходной ток пина 5V	1 А
10	Размеры	69×53 мм

На рисунке 2 приведены контакты контроллера Arduino UNO R3.

Контакты питания.

• **VIN**: входной контакт для подключения внешнего источника напряжения в диапазоне от 7 до 12 В.

• **5V**: выходные контакты от стабилизатора напряжения с выходом 5 В и максимальным током 1 А. Регулятор обеспечивает питание микроконтроллера и другой обвязки платы.

• **3.3V**: выходной контакт от стабилизатора напряжения с выходом 3,3 В и максимальным током 150 мА.

• **IOREF**: вывод предоставляет платам расширения информацию о рабочем напряжении микроконтроллера. В нашем случае рабочее напряжение платформы 5 В.

• **AREF**: контакт для подключения внешнего опорного напряжения АЦП, относительно которого происходят аналоговые измерения при использовании функции analogReference() с параметром EXTERNAL.

• **GND**: выводы общей точки отсчёта напряжения. Электрический потенциал, который используется для обеспечения среды с нулевым электрическим напряжением.

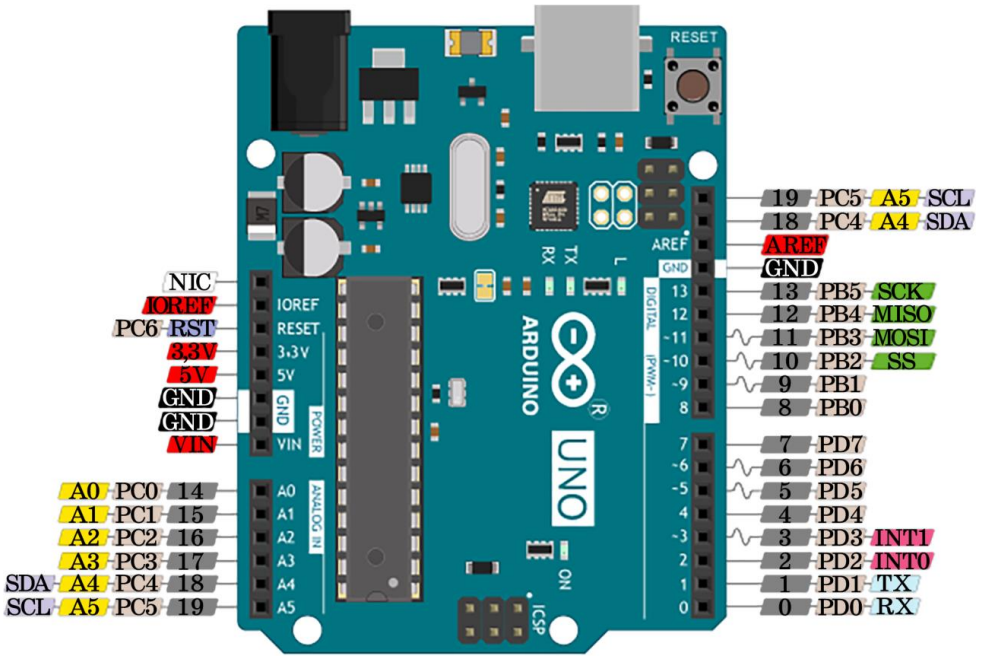

Рис. 2

Контакты контроллера Arduino UNO R3

Порты (пины) ввода/вывода.

• **Пины общего назначения:** 20 пинов — 0–19. Логический уровень единицы — +5 В, нуля — 0 В. К контактам подключены подтягивающие резисторы, которые по умолчанию выключены, но могут быть включены программно.

• **АЦП:** 6 пинов — 14–19 / A0–A5. Позволяет представить аналоговое напряжение в цифровом виде. Разрядность АЦП не меняется и установлена в 10 бит. Диапазон входного напряжения от 0 до 5 В, при подаче большего напряжения микроконтроллер может выйти из строя.

• **ШИМ:** 6 пинов — 3, 5, 6 и 9–11. Позволяет выводить аналоговое напряжение в виде ШИМ-сигнала из цифровых значений. Разрядность ШИМ не меняется и установлена в 8 бит.

• **I²C:** пины SDA/18/A4 и SCL/19/A5. I²C предназначен для общения контроллера с платами расширения и сенсорами по интерфейсу I²C.

• **SPI:** пины MOSI/11, MISO/12 и SCK/13. SPI предназначен для общения контроллера с платами расширения и сенсорами по интерфейсу SPI.

• **Serial**: пины TX1/1 и RX1/0. Serial/UART предназначен для общения контроллера с платами расширения и сенсорами по интерфейсу UART. Контакты также соединены с соответствующими выводами сопроцессора ATmega16U2 для общения платы по USB. Во время прошивки и отладки программы через персональный компьютер нельзя использовать эти пины в проекте.

2. ПРОГРАММИРОВАНИЕ КОНТРОЛЛЕРА ARDUINO UNO R3

2.1. Среда разработки программ для контроллера

Микроконтроллер на плате программируется при помощи языка программирования Arduino (основанного на платформе Wiring и языка, синтаксически сходного с языком программирования C/C++). С описанием языка программирования Arduino можно ознакомиться здесь: https://arduino.ru/Reference. Среда разработки Arduino IDE доступна для свободной загрузки с сайта https://www.arduino.cc/en/software. Среда разработки может быть установлена на ПЭВМ под управлением операционных систем Microsoft Windows, Linux, Mac OS X.

2.2. Общие сведения о программировании контроллера

Программу управления контроллером можно представить в виде схемы (рис. 3).

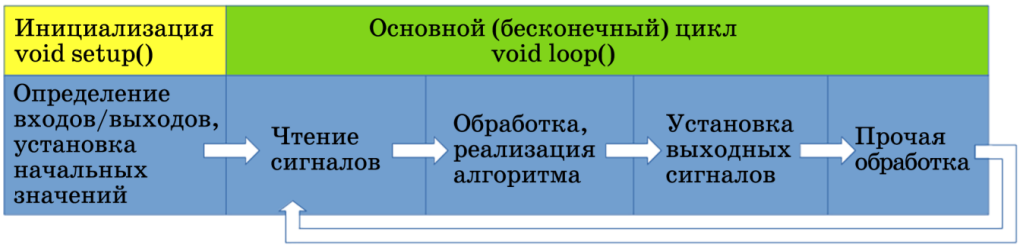

Рис. 3

Схема управления контроллером

После включения питания контроллера Arduino загружается программа управления контроллером и, выполнив все необходимые проверки, вызывает на выполнение функцию void setup(). Функция setup() выполняется один раз. В ней должно быть реализовано определение всех входов и выходов путём вызова системной функций pinMode(), параметры которой определяют номер пина и режим (INPUT/OUTPUT). Кроме этого, здесь устанавливаются начальные значения переменных и выходных сигналов.

Далее программа управления контроллером вызывает функцию основного цикла управления контроллером void loop(), в которой на первом этапе должно быть реализовано чтение сигналов с использованием системных функций digitalRead() и analogRead(), параметры которых задают номер порта. А после обработки и реализации алгоритма — целевой функции контроллера, если необходимо, то осуществляется установка выходных сигналов — для дискретных (цифровых) с использованием системной функции digitalWrite(), и для сигналов ШИМ analogWrite(), параметры которых задают номер порта и соответ-

ствующее значение (HIGH/LOW для цифровых сигналов, от 0 до 255 для сигналов ШИМ).

Для вывода одного байта побитно используйте системную функцию shiftOut(). Вывод может осуществляться как с первого (левого), так и с последнего (правого) бита. Каждый бит последовательно подается на заданный порт (пин), после чего подается сигнал на синхронизирующий порт «вход/выход», информируя о доступности к считыванию бита. Параметры определяют номер пина выхода, на который выводятся биты (int), номер пина синхронизации, используемую последовательность вывода бит (MSBFIRST — слева или LSBFIRST — справа), байт вывода.

Примеры программ управления Arduino UNO приведены в Приложении А.

2.3. Программная обработка прерываний

Прерывание возникает во время работы контроллера при наступлении какого-либо события (тайм-аут, изменение сигнала на пине микроконтроллера и др.), при этом выполнение программы управления прерывается, состояние регистров, сигналов, стека и т. п. сохраняется. Для обработки прерывания может быть определена функция — обработчик прерываний. После выполнения необходимых действий в обработчике прерываний, как правило, восстанавливается сохранённое состояние и возвращается управление прерванной программы.

External hardware interrupt (EHI) — это прерывание, вызванное изменением напряжения на пине микроконтроллера. Как только напряжение на пине меняется (дискретный сигнал) — микроконтроллер получает сигнал, обрабатывает прерывание и возвращается к работе.

Чаще всего прерывания используются для детектирования коротких событий — импульсов для подсчёта их количества, например импульсов, генерируемых тахогенератором, подсчёт которых и обеспечивает определение количества оборотов в единицу времени.

Arduino UNO обеспечивает обработку и других прерываний через дополнительные библиотеки.

Для Arduino UNO R3 EHI-прерывания доступны только для 2-го и 3-го дискретных пинов: INT0 — для пина 2, INT1 — для пина 3. Таким образом, прерывания имеют **свой номер**, который отличается от номера пина. Имеется стандартная функция digitalPinToInterrupt(), которая возвращает номер прерывания, параметр — номер пина.

Для управления обработкой прерываний для Arduino необходимо в блоке инициализации определить функцию «обработчик прерываний» с использованием системной функции attachInterrupt(), параметры которой определяют номер прерывания, указатель на функцию — обработчик прерываний и режим обработки прерываний:

- RISING (рост) — срабатывает при изменении сигнала с LOW на HIGH;
- FALLING (падение) — срабатывает при изменении сигнала с HIGH на LOW;

• CHANGE (изменение) — срабатывает при изменении сигнала (с LOW на HIGH и наоборот);

• LOW (низкий) — срабатывает постоянно при сигнале LOW.

Прерывание можно отключить при помощи функции detachInterrupt(), параметр которой определяет номер прерывания. Чтобы избежать проблем несовместимости, рекомендуется использовать следующую команду:

```
detachInterrupt(digitalPinToInterrupt(pin));
```

Внимание! Внешние переменные, которые будут изменяться в функции «обработчик прерываний», должны иметь атрибут volatile, который информирует компилятор о том, что значение переменной может меняться в обработчике прерываний.

2.4. Подключение программных модулей и библиотек

При разработке программ управления контроллером Arduino иногда требуется выделить некоторые функции (например, расчёт контрольной суммы или среднего) в отдельные модули. Для этого необходимо в каталоге, в который сохраняется основная программа, создать файл с расширением .ino. Arduino IDE автоматически откроет этот файл. Обратите внимание на то, что такие файлы будут включены после основного файла и все функции, глобальные переменные и определения в дополнительных модулях будут доступны.

Для управления подключаемыми индикаторами, средствами управления, двигателями и т. п. для Arduino разработано большое количество библиотек. При необходимости их можно подключать средствами Arduino IDE. В некоторых случаях такие библиотеки могут быть разработаны самостоятельно. В этом случае такая библиотека должна содержать заголовочный файл описания класса (файл с расширением .h) и файл реализации класса (файл с расширением .cpp).

Файлы библиотеки с именем, совпадающим с именем класса, помещаются в каталог с разрабатываемыми проектами, обычно это каталог Arduino\libraries в домашнем каталоге.

3. ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНЫХ И ПРАКТИЧЕСКИХ РАБОТ

При выполнении лабораторных и практических работ студент должен создать модель с использованием системы моделирования, такой как Tinkercad Circuits, Wokwi или любой другой.

Tinkercad Circuits — это бесплатный онлайн-сервис от фирмы Autodesk, запущенный в 2017 г. Используя этот сервис, можно легко проектировать свои собственные схемы, создавать программы в блочном или текстовом формате, а затем отлаживать их. Для использования сервиса Tinkercad Circuits необходимо зарегистрироваться на сайте https://www.tinkercad.com/circuits, выбрав учётную запись студента или личную учётную запись.

Wokwi — это онлайн-симулятор электроники. Он может использоваться для моделирования Arduino, ESP32, STM32 и других плат, радиоэлементов, датчиков. Сервис позволяет в графическом виде создавать схемы, которыми можно управлять разрабатываемыми программами. Для использования сервиса Wokwi необходимо зарегистрироваться на сайте https://wokwi.com/.

Проверив на модели функционирование схемы, необходимо практически осуществить сборку электрической схемы, подключив необходимые элементы к контроллеру с использованием макетной платы (рис. 4). На макетной плате каждый вертикальный ряд имеет несколько контактов для подключения, так же как и горизонтальные ряды. Горизонтальные ряды обычно используются для подключения питания (+5V) и общей точки отсчёта напряжения (GND).

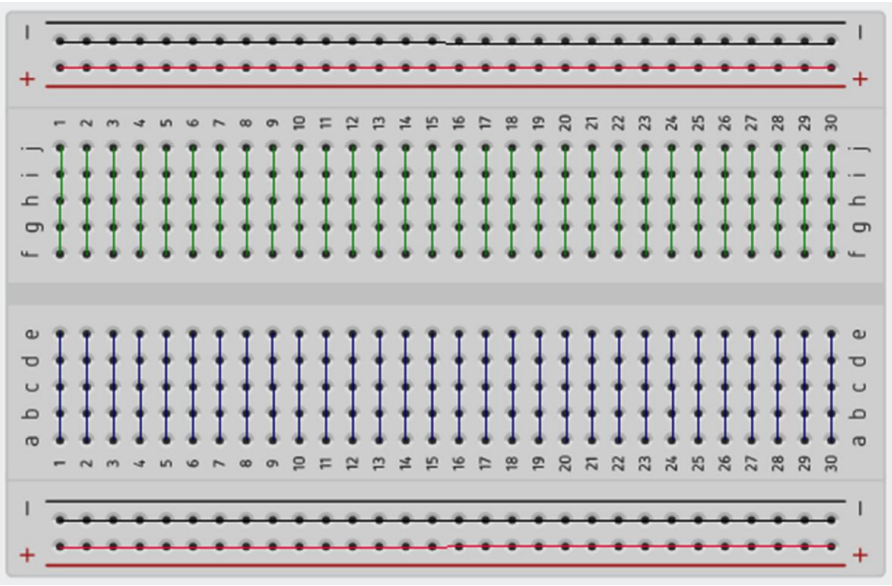

Рис. 4

Макетная плата

Установка элементов на макетную плату должна осуществляться в соответствии с разработанной в симуляторе схемой подключения. После установки на макетную плату элементов и соединительных проводов необходимо тщательно проверить надёжность и правильность соединения, проверить отсутствие замыкания элементов.

В программную среду Arduino IDE необходимо перенести программу управления контроллером, разработанную в системе моделирования. И только после тщательной проверки подключить кабель USB к компьютеру и контроллеру Arduino и загрузить в контроллер программу управления.

Внимание! Изменения в собранной схеме выполнять только при отключенном электропитании контроллера (USB-кабеле).

В соответствии с заданием на выполнение лабораторной работы проверить функционирование схемы и при необходимости выполнить фотофиксацию результата работы.

Для уточнения электрических характеристик подключаемых элементов нужно воспользоваться мультиметром.

4. СОСТАВ ОТЧЁТА О ВЫПОЛНЕННОЙ РАБОТЕ

Отчёт о выполненной лабораторной и практической работе должен содержать следующие разделы:

- титульный лист;
- задание;
- теоретические сведения;
- схема электрическая принципиальная;
- модель собранной схемы;
- программа управления контроллером;
- результат;
- выводы;
- используемые источники.

В раздел «Теоретические сведения» должна быть включена информация об элементах, которые будут использованы в ходе выполнения работы, а также об особенностях применения некоторых решений. Например, о светодиодах, кнопках, индикаторах, применяемых методах и т. п.

В раздел «Схема электрическая принципиальная» включается чертёж схемы и необходимые пояснения.

В раздел «Модель собранной схемы» включается скриншот модели и необходимые пояснения. Для некоторых заданий модель собранной схемы не включается, так как в системе моделирования отсутствуют необходимые для реализации радиокомпоненты.

В раздел «Программа управления контроллером» включается исходный текст программы управления контроллером. Программа должна содержать необходимые комментарии.

В раздел «Результат» включаются результаты фотофиксации работы собранной схемы.

В раздел «Выводы» включается описание того, что было студентом освоено в ходе выполнения лабораторной и практической работы.