

Федеральное государственное образовательное бюджетное учреждение
высшего образования
«Финансовый университет при Правительстве Российской Федерации»
Канашский филиал Финуниверситета

Методические рекомендации для студентов
по выполнению самостоятельной работы
по ПМ.01 Участие в проектировании архитектуры интеллектуальных
интегрированных систем
по специальности 09.02.08 Интеллектуальные интегрированные системы

2025 г.

Организация-разработчик: ФГОБУ ВО «Финансовый университет при Правительстве Российской Федерации» Канашский филиал Финуниверситета

Разработчик(и):

Николаева И.В.- преподаватель ВКК Канашского филиала Финуниверситета

Рабочая программа дисциплины рассмотрена и рекомендована к утверждению на заседании предметной (цикловой) комиссии интеллектуальных интегрированных систем

Протокол от «25» мар 20 25 г. № 1

Председатель предметно (цикловой) комиссии:  /Славкина А.И./

Пояснительная записка

Методические указания (рекомендации) для студентов по выполнению самостоятельной работы по дисциплине разработаны на основе требований Федерального государственного образовательного стандарта по специальности (специальностям) среднего профессионального образования по специальности 09.02.08 Интеллектуальные интегрированные системы.

Самостоятельная работа по ПМ.01 Участие в проектировании архитектуры интеллектуальных интегрированных систем проводится с целью:

- систематизации и закрепления полученных теоретических знаний и практических умений;
- формирования общих и профессиональных компетенций;
- углубления и расширения теоретических знаний;
- формирования умений использовать нормативную, правовую, справочную документацию и специальную литературу;
- развития познавательных способностей и активности: творческой инициативы, самостоятельности, ответственности и организованности;
- формирования самостоятельности мышления, способностей к саморазвитию, самосовершенствованию и самореализации;
- развития исследовательских умений.

Самостоятельная работа по ПМ.01 Участие в проектировании архитектуры интеллектуальных интегрированных систем включает задания по выполнению практических заданий, составлению конспектов, решению ситуационных задач и др.

Самостоятельная работа по ПМ.01 Участие в проектировании архитектуры интеллектуальных интегрированных систем является внеаудиторной и обязательна для всех студентов. Внеаудиторная самостоятельная работа - планируемая учебная, учебно-исследовательская работа студентов, выполняемая вне занятий по заданию и при управлении преподавателем, но без его непосредственного участия.

Критериями оценки результатов внеаудиторной самостоятельной работы являются:

- уровень освоения учебного материала;
- умения использовать теоретические знания при выполнении практических задач;
- сформированность общеучебных умений;
- обоснованность и четкость изложения ответа;
- оформление материала в соответствии с требованиями.

**САМОСТОЯТЕЛЬНАЯ РАБОТА СТУДЕНТА
ПО ПМ.01 УЧАСТИЕ В ПРОЕКТИРОВАНИИ АРХИТЕКТУРЫ ИНТЕЛЛЕКТУАЛЬНЫХ
ИНТЕГРИРОВАННЫХ СИСТЕМ
СПЕЦИАЛЬНОСТИ 09.02.08 ИНТЕЛЛЕКТУАЛЬНЫЕ ИНТЕГРИРОВАННЫЕ
СИСТЕМЫ**

Тема	Кол-во часов	Вид работы
2	3	4
МДК 01.01 Цифровая схемотехника	2	Подготовка сообщений (по выбору студента)
МДК. 01.02 Микроконтроллерные системы	4	Изучение основ архитектуры микроконтроллера. Работа с интерфейсами микроконтроллера. Взаимодействие микроконтроллера с аналоговыми датчиками. Взаимодействие микроконтроллера с цифровыми датчиками

Самостоятельная работа 1

МДК 01.01 Цифровая схемотехника Задание 1. Подготовка сообщений (по выбору студента):

1. Цифровые логические элементы
2. Арифметико-логические устройства
3. Параллельно-последовательные регистры
4. Анализ и синтез последовательностных устройств
5. Запоминающие устройства
6. Цифровые интегральные схемы
7. Современные технологии изготовления печатных плат

Самостоятельная работа 2

МДК. 01.02 Микроконтроллерные системы

Задание 1. Изучение основ архитектуры микроконтроллера

Описание задачи: Изучите архитектуру популярного микроконтроллера семейства AVR — ATmega328P. Ваша цель — разобраться в ключевых аспектах его архитектуры, таких как регистры, память, периферия и режимы работы.

Что вам предстоит сделать:

Ознакомьтесь с документацией:

Найдите и изучите даташит (datasheet) на микроконтроллер ATmega328P.

Обратите внимание на общие характеристики микроконтроллера, такие как тактовая частота, объем памяти и поддерживаемые напряжения питания.

Исследуйте блок-схему:

Найдите и изучите блок-схему архитектуры ATmega328P.

Опишите основные блоки микроконтроллера: ядро процессора, память (Flash, EEPROM, SRAM), таймеры/счетчики, АЦП, USART, SPI, I²C и другие периферийные модули.

Подробнее о ядре процессора:

Исследуйте, как устроена архитектура ядра процессора ATmega328P.

Объясните, как происходит исполнение инструкций, что такое конвейер команд и как работает ALU (арифметико-логическое устройство).

Память:

Изучите организацию памяти ATmega328P: Flash-память для хранения программы, EEPROM для долговременного хранения данных и SRAM для временного хранения переменных.

Опишите, как осуществляется доступ к различным видам памяти и каковы их особенности.

Таймеры и прерывания:

Подробно разберитесь с работой таймеров/счетчиков микроконтроллера.

Объясните, как используются прерывания для обработки внешних событий и управления временем.

Аналого-цифровой преобразователь (АЦП):

Изучите принцип работы встроенного АЦП.

Опишите, как производится аналого-цифровое преобразование и как можно управлять параметрами АЦП через регистры.

Последовательные интерфейсы:

Исследуйте работу последовательных интерфейсов UART, SPI и I²C.

Объясните различия между этими интерфейсами и приведите примеры их использования.

Энергосберегающие режимы:

Ознакомьтесь с различными энергосберегающими режимами ATmega328P.

Объясните, как можно снизить энергопотребление микроконтроллера в зависимости от режима работы.

Реальное применение:

Создайте простейшую схему на макетной плате с использованием ATmega328P.

Напишите короткую программу на языке C или ассемблере, демонстрирующую работу одного из периферийных модулей (например, управление светодиодом через ШИМ или считывание значения с аналогового входа).

Итоги:

Подготовьте отчет, в котором обобщите полученную информацию об архитектуре ATmega328P.

Ответьте на вопросы, касающиеся особенностей данной архитектуры и её преимуществ перед другими микроконтроллерами.

Задание 2. Работа с интерфейсами микроконтроллера

Описание задачи: Вам необходимо написать программу для микроконтроллера (например, Arduino Uno с микроконтроллером ATmega328P), которая управляет светодиодами через интерфейс UART. Программа должна принимать команды через последовательный порт и в зависимости от полученной команды включать или выключать светодиоды.

Что вам предстоит сделать:

Подготовка оборудования:

Подключите микроконтроллер к компьютеру через USB-кабель.

Подключите один или несколько светодиодов к цифровым пинам микроконтроллера (не забудьте про ограничивающие резисторы).

Написание программы:

Создайте программу на языке C/C++ или Arduino IDE, которая будет принимать команды через UART-интерфейс.

Команды могут быть простыми строками, такими как "ON" для включения светодиода и "OFF" для его выключения.

Используйте функцию Serial.read() для чтения данных из последовательного порта.

Обработайте введенные команды и управляйте состоянием светодиодов.

Тестирование:

Загрузите программу в микроконтроллер.

Откройте монитор последовательного порта (например, в Arduino IDE) и отправьте команды для управления светодиодами.

Проверьте, что светодиоды реагируют на команды должным образом.

Расширение функционала:

Добавьте возможность регулировки яркости светодиодов с помощью PWM (широтно-импульсной модуляции).

Реализуйте поддержку нескольких светодиодов, каждый из которых управляется отдельной командой.

Документирование:

Подготовьте отчет, в котором опишите архитектуру программы, использованные функции и интерфейсы микроконтроллера.

Приведите примеры команд и реакции светодиодов на них.

Пример кода на Arduino IDE:

```
#define LED_PIN 13 // Пин, к которому подключен светодиод

void setup() {
  Serial.begin(9600); // Инициализация последовательного порта
  pinMode(LED_PIN, OUTPUT); // Установка пина как выход
}

void loop() {
  if (Serial.available()) { // Проверяем, есть ли данные в буфере
    String command = Serial.readStringUntil('\n'); // Читаем строку до символа новой строки

    if (command == "ON") {
      digitalWrite(LED_PIN, HIGH); // Включаем светодиод
      Serial.println("Светодиод включен");
    } else if (command == "OFF") {
      digitalWrite(LED_PIN, LOW); // Выключаем светодиод
      Serial.println("Светодиод выключен");
    } else {
      Serial.println("Неизвестная команда");
    }
  }
}
```

Оборудование и материалы:

Микроконтроллер (например, Arduino Uno).

Светодиоды.

Резисторы (например, 220 Ом).

USB-кабель для подключения микроконтроллера к компьютеру.

Компьютер с установленным ПО для программирования микроконтроллера (например, Arduino IDE).

Требуется продемонстрировать умение работать с UART-интерфейсом микроконтроллера, написав программу, которая принимает команды через последовательный порт и управляет светодиодами.

Задание 3. Взаимодействие микроконтроллера с аналоговыми датчиками.

Описание задачи: Вам необходимо собрать схему, состоящую из микроконтроллера (например, Arduino Uno) и аналогового датчика температуры (например, LM35). Задача заключается в том, чтобы считывать показания температуры с датчика и выводить их на экран компьютера через последовательный порт.

Что вам предстоит сделать:

Подготовка оборудования:

Подключите датчик температуры LM35 к микроконтроллеру Arduino.

Подключите микроконтроллер к компьютеру через USB-кабель.

Написание программы:

Создайте программу на языке C/C++ или Arduino IDE, которая будет считывать значение с аналогового входа микроконтроллера, соответствующего выводу датчика LM35.

Преобразуйте считанное значение в градусы Цельсия.

Отправьте данные через последовательный порт на компьютер.

Тестирование:

Загрузите программу в микроконтроллер.

Откройте монитор последовательного порта (например, в Arduino IDE) и убедитесь, что температура выводится корректно.

Проверьте работу схемы при различных температурах окружающей среды.

Расширение функционала:

Добавьте возможность вывода температуры на дисплей (например, LCD-экран).

Реализуйте сигнализацию при достижении определенной температуры (например, включение светодиода при превышении установленного порога).

Документирование:

Подготовьте отчет, в котором опишите архитектуру программы, использованные функции и интерфейсы микроконтроллера.

Приведите примеры показаний температуры и реакцию системы на изменение температуры.

Пример кода на Arduino IDE:

```
int sensorPin = A0; // Пин, к которому подключен датчик LM35
float temperature;

void setup() {
  Serial.begin(9600); // Инициализация последовательного порта
}

void loop() {
  int sensorValue = analogRead(sensorPin); // Считываем значение с аналогового входа
  float voltage = (sensorValue / 1023.0) * 5.0; // Преобразуем значение в напряжение
  temperature = voltage * 100.0; // Преобразуем напряжение в температуру (LM35
  // выдает 10 мВ на градус Цельсия)

  Serial.print("Температура: ");
  Serial.print(temperature);
  Serial.println(" °C");

  delay(1000); // Пауза в 1 секунду
}
```

Оборудование и материалы:

Микроконтроллер (например, Arduino Uno).

Аналоговый датчик температуры (например, LM35).

Соединительные провода.

USB-кабель для подключения микроконтроллера к компьютеру.

Компьютер с установленным ПО для программирования микроконтроллера (например, Arduino IDE).

Требуется продемонстрировать умение работать с аналоговыми датчиками, написав программу, которая считывает показания температуры и выводит их на экран компьютера через последовательный порт.

Задание 4. Взаимодействие микроконтроллера с цифровыми датчиками

Описание задачи: Вам необходимо собрать схему, состоящую из микроконтроллера (например, Arduino Uno), кнопки и реле. Задача заключается в том, чтобы управлять реле с помощью кнопки, подключенной к микроконтроллеру. Когда кнопка нажата, реле должно замыкать цепь, включая нагрузку (например, лампочку).

Что вам предстоит сделать:

Подготовка оборудования:

Подключите кнопку к одному из цифровых входов микроконтроллера.

Подключите реле к другому цифровому выходу микроконтроллера.

Подключите микроконтроллер к компьютеру через USB-кабель.

Написание программы:

Создайте программу на языке C/C++ или Arduino IDE, которая будет считывать состояние кнопки и управлять реле.

Используйте функцию `digitalRead()` для считывания состояния кнопки.

Используйте функцию `digitalWrite()` для управления реле.

Тестирование:

Загрузите программу в микроконтроллер.

Нажмите кнопку и убедитесь, что реле срабатывает, замыкая цепь и включая нагрузку.

Расширение функционала:

Добавьте возможность управления нагрузкой через последовательность нажатий кнопки (например, одно короткое нажатие включает нагрузку, второе — выключает).

Реализуйте защиту отдребезга контактов кнопки.

Документирование:

Подготовьте отчет, в котором опишите архитектуру программы, использованные функции и интерфейсы микроконтроллера.

Приведите примеры работы схемы и реакцию нагрузки на нажатия кнопки.

Пример кода на Arduino IDE:

```
const int buttonPin = 2; // Пин, к которому подключена кнопка
const int relayPin = 13; // Пин, к которому подключено реле

bool lastButtonState = false; // Хранит последнее состояние кнопки
bool relayState = false; // Хранит текущее состояние реле

void setup() {
    pinMode(buttonPin, INPUT_PULLUP); // Устанавливаем кнопку как вход с подтягивающим резистором
    pinMode(relayPin, OUTPUT); // Устанавливаем реле как выход
}

void loop() {
    bool currentButtonState = digitalRead(buttonPin); // Считываем текущее состояние кнопки

    if (currentButtonState != lastButtonState && !currentButtonState) { //
        Обрабатываем нажатие кнопки
        relayState = !relayState; // Меняем состояние реле
        digitalWrite(relayPin, relayState); // Управляем реле
    }

    lastButtonState = currentButtonState; // Сохраняем текущее состояние кнопки
}
```

Оборудование и материалы:

Микроконтроллер (например, Arduino Uno).
Кнопка.
Реле.
Соединительные провода.
Нагрузка (например, лампочка).
USB-кабель для подключения микроконтроллера к компьютеру.
Компьютер с установленным ПО для программирования микроконтроллера (например, Arduino IDE).
Требуется продемонстрировать умение работать с цифровыми датчиками, написав программу, которая считывает состояние кнопки и управляет реле, включая и выключая нагрузку.

Список использованной литературы

1.Чекмарев, А. А. Инженерная графика: учебник для среднего профессионального образования / А. А. Чекмарев. — 13-е изд., испр. и доп. — Москва: Издательство Юрайт, 2023. — 389 с. — (Профессиональное образование). — ISBN 978-5-534-07112-2. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/511680>