ОТЧЕТ

ПО ЭКЗАМЕНАЦИОННОЙ РАБОТЕ

ПО ДИСЦИПЛИНЕ

«ПРОЕКТИРОВАНИЕ ВСТРОЕННЫХ СИСТЕМ»

Работу выполнили:

Студенты 61 группы

3 курса

Вафин А.Р.

Халтурина О.П.

Шевцова А.Д.

Воронеж, 2025

СОДЕРЖАНИЕ

Введение…………………………………………………………………………3

Задача 1…………………………………………………………………………..3

Постановка задачи 1…………………………………………………………….3

Техническое задание задачи 1………………………………………………….3

Схема задачи 1…………………………………………………………………..5

Протокол задачи 1………………………………………………………………5

Экономическая выдержка задачи 1…………………………………………....6

Задача 2………………………………………………………………………….7

Постановка задачи 2……………………………………………………………7

Техническое задание задачи 2…………………………………………………8

Схема задачи 2……………………………………………………………...…10

Протокол задачи 2…………………………………………………………….10

Экономическая выдержка задачи 2………………………………….……….11

Заключение……………………………………………………………….……12

Приложение……………………………………………………………………14

**Введение**

В современном мире встроенные системы играют ключевую роль в различных областях, от бытовой электроники до промышленной автоматизации. Данная работа посвящена проектированию и реализации двух практических задач, демонстрирующих навыки работы с микроконтроллерами, такими как ESP32 и Arduino. Цель работы — освоить принципы взаимодействия устройств, обработки сигналов, а также экономические аспекты выбора компонентов для реализации проектов.

Первая задача направлена на изучение возможностей ESP32 в части работы с Bluetooth и таймерами. Вторая задача предполагает создание системы взаимодействия между ESP32 и Arduino, имитирующей работу замка с защитным механизмом. Обе задачи включают в себя не только программную реализацию, но и анализ стоимости компонентов, что позволяет оценить экономическую целесообразность их использования в реальных проектах.

**Задача 1**

1. **Постановка задачи 1**

Исходная формулировка задачи:

Плата ESP32 имеет в себе уже wi-fi и Bluetooth.

При включении плата по Bluetooth раздает широковещательно заданное значение, например, 100.

По сигналу (датчика движения в оригинале) на плате запускается таймер на заданный отрезок времени (10 сек, например) (выводить отчет времени не обязательно) и пока таймер отсчитывает время, плата вещает иное значение, например, 101. Если в процессе работы таймера еще раз придет сигнал, таймер начинает отсчёт заново, а в эфир транслируется иное значение, например, 102. И так далее. Последовательность значений должна быть зациклена и количество шагов цикла должно быть прописано в config (101,102,103,104,105). Если время, отсчитываемое таймером, кончилось, то плата вновь вещает базовое значение (в примере - 100). Все значения и константы должны быть прописаны в config или с помощью #define.

1. **Техническое задание задачи 1**
   1. **Устройства**

Для реализации данной задачи потребуются следующие устройства:

* + плата ESP32-WROOM-32 на базе CP2102
  + кнопка
  + Dupont-провода (П-М, 2 штуки)
  + макетная плата для соединения компонентов без пайки
  1. **Входные и выходные данные**

На вход программы подается сигналы – нажатия кнопки

На выход программы подаются значения, в примере – 1000, 1001, 1002, 1003, 1004 и 1005 в зависимости от количества нажатий кнопки.

* 1. **Требования**

Программа, выполняющая данный функционал должна быть разбита на config и main (с помощью которого создается MakeFile).

Программа должна работать в штатном режиме без сбоев и внештатных ситуаций.



Рис. 1. Диаграмма вариантов использования задачи 1.

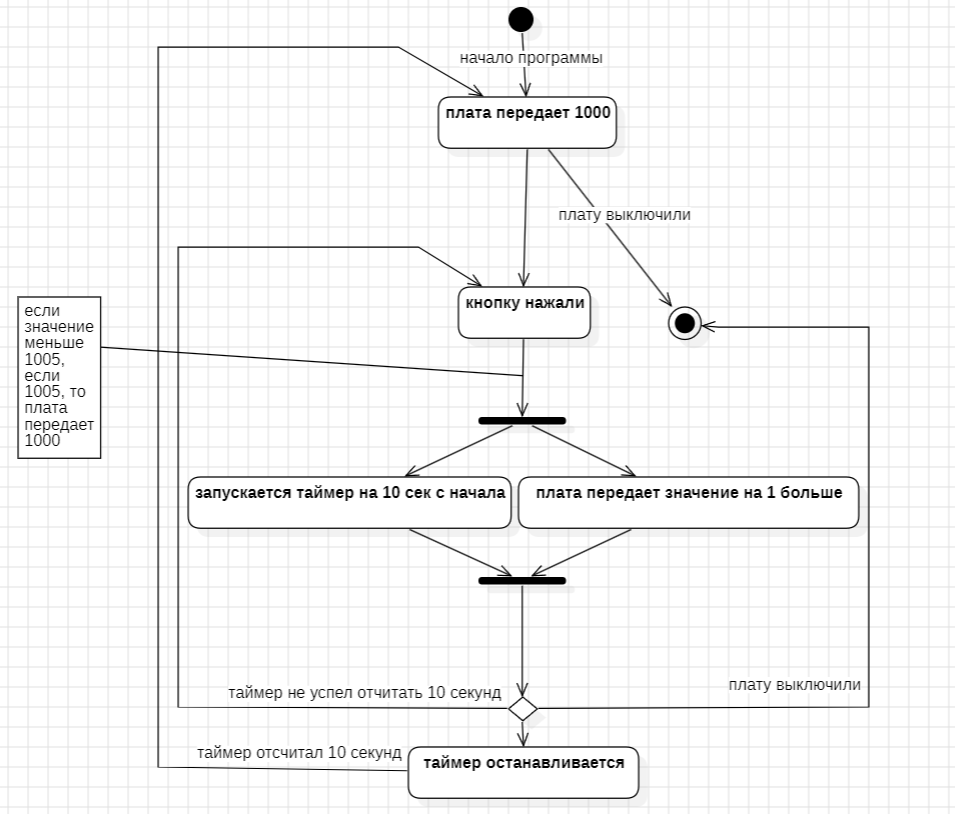


Рис. 2. Диаграмма действий задачи 1

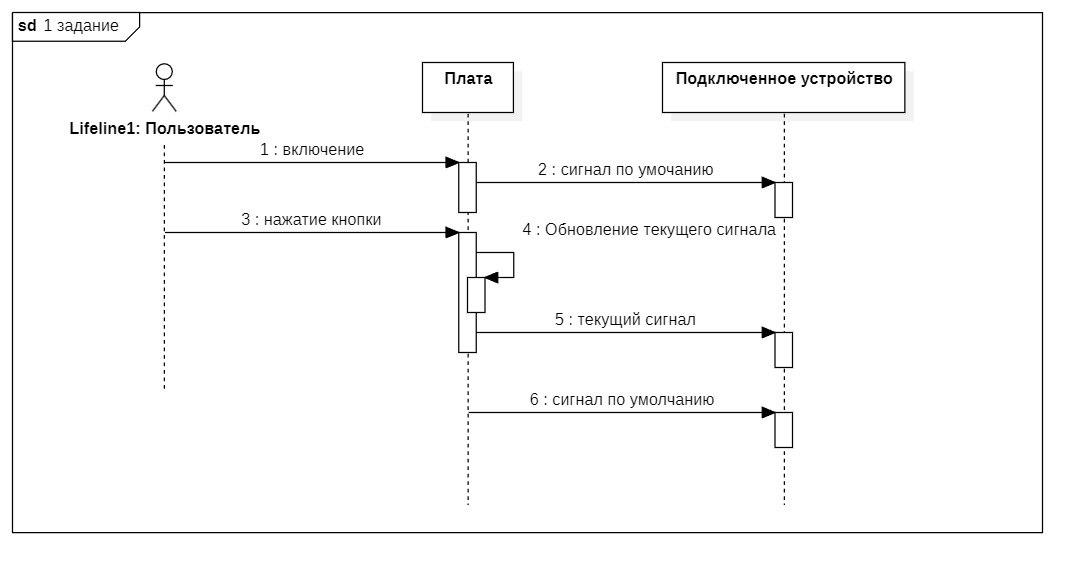


Рис. 3. Диаграмма последовательности задачи 1

1. **Схема задачи 1**

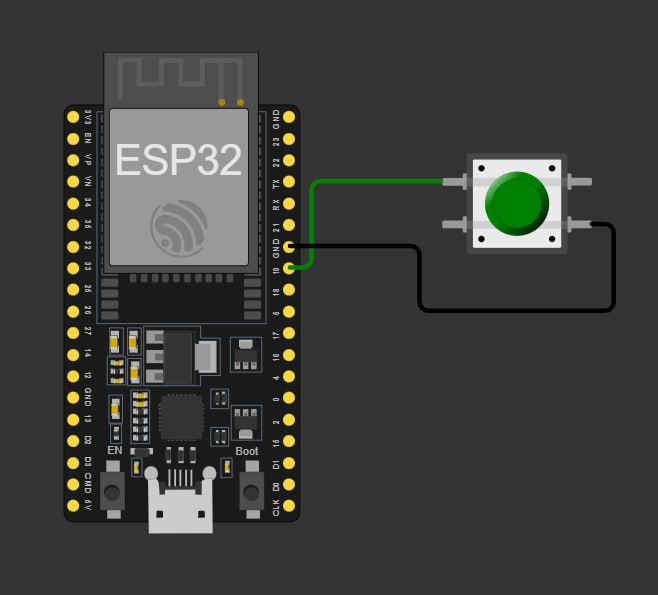
****

Рис. 4. Схема задачи 1

1. **Протокол задачи 1**

Плата ESP32 использует Bluetooth-модуль для широковещательной передачи информации по принципу работы последовательного порта (SerialBT). Также возможно использование последовательного порта самой платы для отладки платы и проверки работы Bluetooth-модуля и самой платы.

Формат команд SerialBT:

- Value: <number> - передача значения из цикла

Формат команд последовательного порта имеет такую же структуру, но с добавлением сообщения об успешной инициализации Bluetooth-модуля: ESP32 Buletooth broadcaster started.

1. **Экономическая выдержка задачи 1**

Для решения данной задачи необходимы следующие аппаратные средства:

* плата ESP32-WROOM-32 на базе CP2102
* кнопка
* Dupont-провода (П-М, 2 штуки)
* макетная плата для соединения компонентов без пайки

Если мы говорим о постановке данных товаров на поток или о регулярном использовании приспособлений, стоит отказаться от макетной платы и использовать спайку, таким образом для создания продукта необходимы:

* плата ESP32-WROOM-32 на базе CP2102
* кнопка
* Dupont-провода (П-М, 2 штуки)

Также будет необходим источник питания данной системы, и здесь есть 2 пути:

1. Использовать USB-провод и портативное зарядное устройство с соответствующим разъемом
2. Перепаивать плату и встраивать в него возможность питания от батареек того или иного типа

Так как мы не обладаем информацией об условиях использования и масштабах производства данных схем, рассчитывать в данном отчете затраты на устройства питания не будем.

Цены указаны актуальные на момент 24.06.2025.

**Плата ESP32-WROOM-32 на базе CP2102**

|  |  |  |
| --- | --- | --- |
| **магазин** | **цена** | **примечания** |
| Ozon | 598 руб | - |
| Wildberries | 517руб | Регулярные задержки в поставках |
| avrobot.ru | 559 руб | - |
| compacttool.ru | 520 руб | - |
| voltiq.ru | 827 руб | Опт. Необходима доставка |

Выбрана именно эта плата по нижеперечисленным причинам:

1. Средняя цена
2. продается в нескольких магазинах (в случае осложнений в поставках, можно приобретать в другом магазине)
3. Есть возможность работать с Bluetooth
4. Есть возможность связи с Arduino (необходимо для второй задачи), о чем говорит производитель (т.е. он гарантирует коннект с Arduino)

**Провода Dupont папа-мама, 2 шт**

|  |  |  |
| --- | --- | --- |
| **магазин** | **цена** | **замечания** |
| Оzon | 164-194руб | 40 шт пачка |
| duino.ru | 110руб | 30 шт пачка, по 10 шт П-П, М-П, М-М |
| ЯндексМаркет | 267-1440руб | Различная длина и комплектация |
| amperkot.ru | 70-130руб | По 40 шт в наборе, цена разнится в зависимости от длины проводов |
| chipdip.ru | 180руб | 40 шт |
| AliExpress | 58-91руб | 10 шт в упаковке |

**Кнопка**

|  |  |  |
| --- | --- | --- |
| **магазин** | **цена** | **замечания** |
| AliExpress | 87руб | 1 шт |
| ЯндексМаркет | 292руб | 20 шт |

**Задача 2**

1. **Постановка задачи 2**

Исходная формулировка задачи:

Плата ESP32 играет роль замка, а arduino – декодера.

Arduino включает в себя:

* Провод, который подсоединяется к плате ESP32
* Экран
* Клавиатура (не обязательно, можут быть ручки крутящиеся или что-то еще)
* Дополнительные органы управления

Должна быть реализовано передача данных с arduino на ESP32.

Что видит пользователь:

Условный пользователь подходит к замку (ESP32) со своей отмычкой (arduino). Подключается arduino к замку. Arduino получает от ESP32 код-ответ. Пользователь вводит на arduino набор чисел. Подбор кода только на Arduino. Arduino посылает эти значения на плату ESP32 и, только когда пользователь ввел правильное значение или пользователь превысил количество попыток, Arduino дает команду ESP32 уйти в сон на заданный отрезок времени. В идеале динамически генерировать правильный код. В ESP32 в config время сна и код-ответ.

1. Введенный код верен – «замок» симулирует открытие замка, например, светодиодом
2. Введенный код не верен – «замок» светодиодом сигнализирует о том, что значение не верно и уходит на какой-то промежуток времени в спячку, т.е. не принимает значения-ответы.

Arduino же, зная ключ-ответ к ESP32, дает подсказки, чтобы можно было угадать код от замка. Подсказку необходимо придумать – это творческий элемент экзаменационной работы. Был пример с подсвечиванием верного числа на верной позиции.

1. **Техническое задание задачи 2**
   1. **Устройства**

Используются:

* плата ESP32-WROOM-32 на базе CP2102
* плата Arduino Uno R3
* 0,96'' OLED-дисплей SSD1306 с I2C-интерфейсом
* кнопочная матрица 4х4
* светодиод
* резистор на 220 Ом
* Dupont-провода (П-М, 17 штук)
* макетная плата для соединения компонентов без пайки

**Входные и выходные данные**

На вход ESP32 должны подаваться код-ответ, количество попыток, время сна.

На выходе должны гореть все светодиоды – 4 на плате Arduino и один на плате ESP32.

* 1. **Требования**

Программы, выполняющие данный функционал должны быть разбиты на config и main (с помощью которого создается MakeFile).

Программа должна работать в штатном режиме без сбоев и внештатных ситуаций.

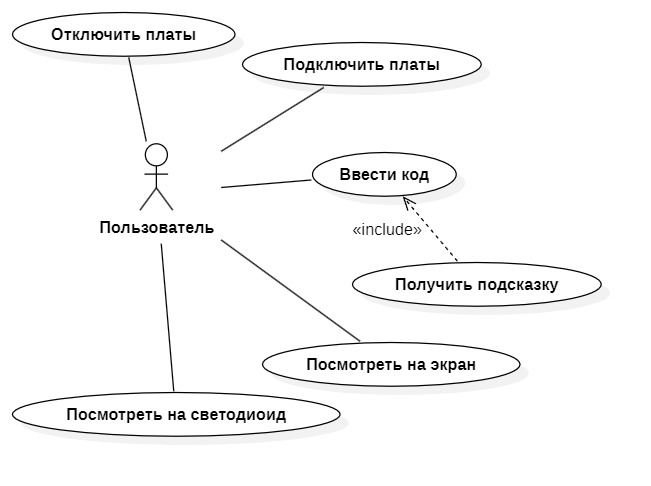


Рис. 4. Диаграмма вариантов использования задачи 2.

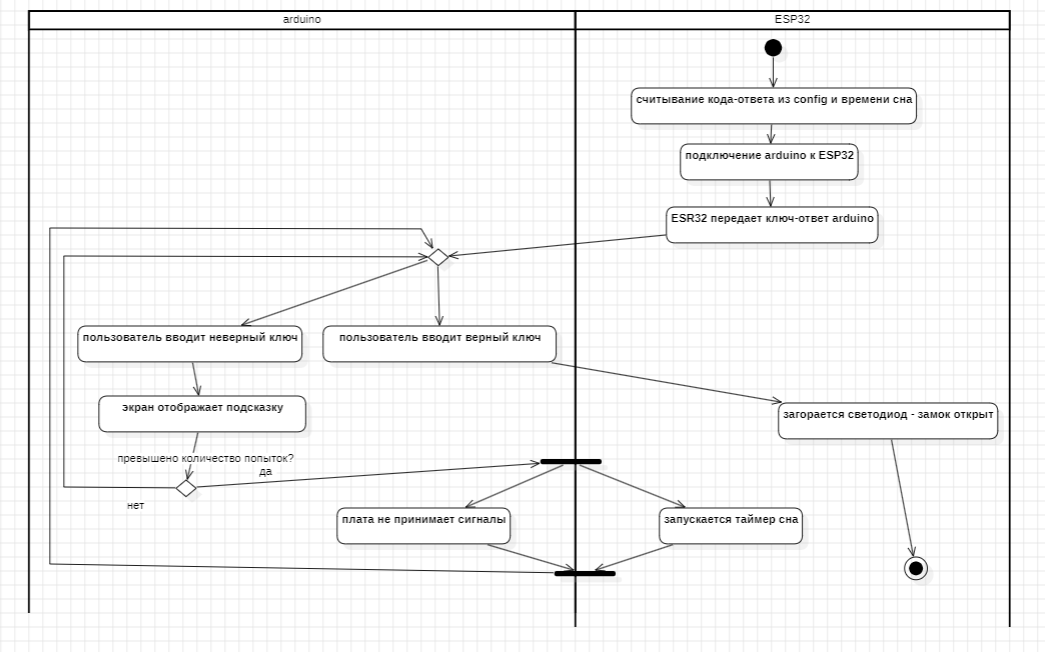


Рис. 5. Диаграмма действий задачи 2

К плате ESP32 будет подсоединены таймер и светодиод, обозначающий открытие замка.

Команду о запуске таймера сна передает Arduino.

Светодиод открытия замка загорается на ESP32 только когда Arduino передала сигнал о том, что введен верный ключ.

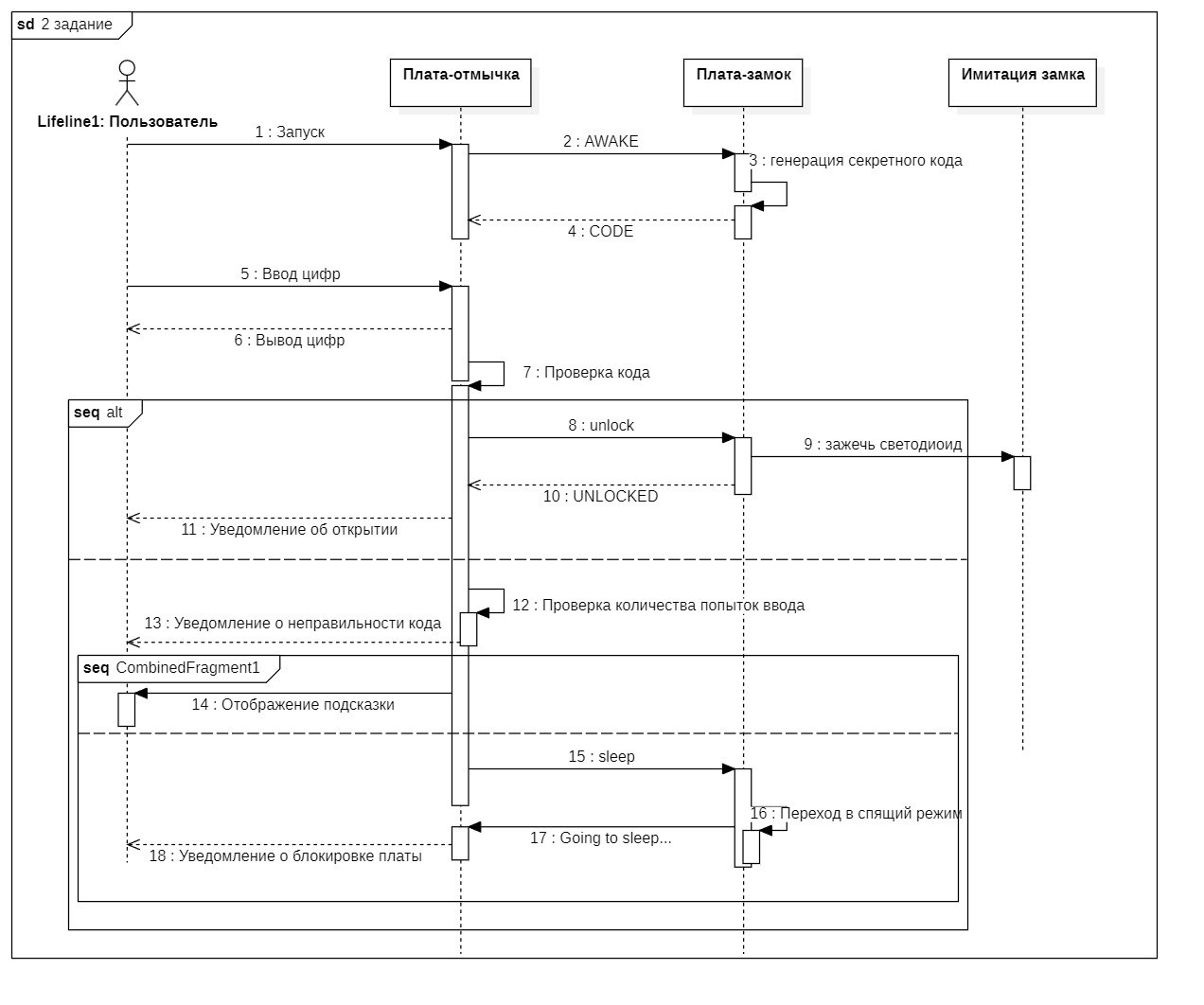


Рис. 6. Диаграмма последовательности задачи 2

1. **Схема задачи 2**

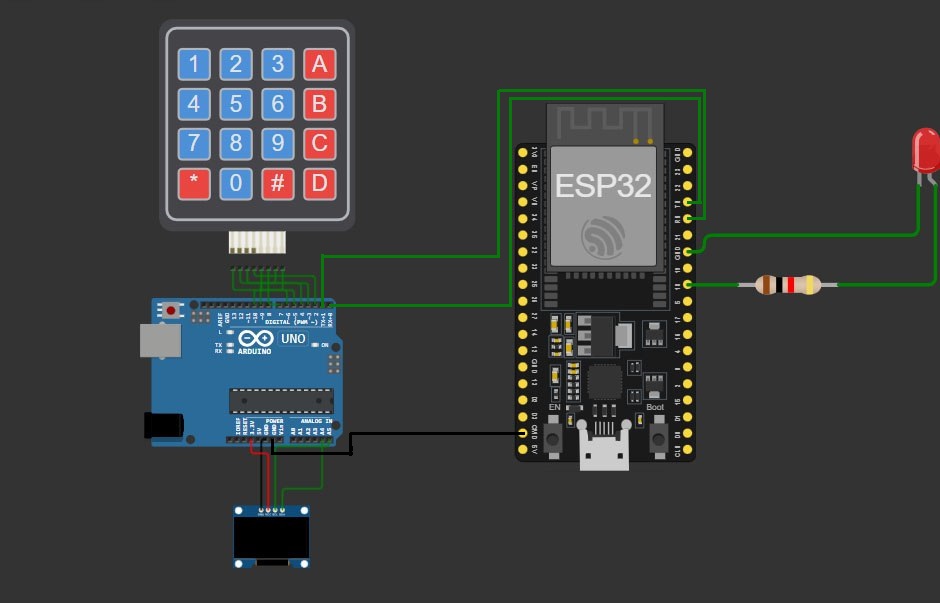
****

Рис. 7. Схема задачи 2

1. **Протокол задачи 2**

Платы взаимодействуют между собой через универсальный последовательный приемопередатчик (UART).

Формат команд платы, исполняющей роль замка (ESP32):

- AWAKE: сообщение для отмычки о том, что замок готов к работе. Посылается при запуске замка и выходе его из спячки.

- CODE:1234 : сообщение плате-отмычке корректного кода, который генерируется случайным образом в момент запуска замка и хранится в EEPROM (благодаря этому достигается сохранность кода при выходе из спячки)

- Going to sleep... - сообщение отмычке об уходе замка в спячку

- UNLOCKED - сообщение отмычке о том, что замок открыт

Формат команд платы, исполняющей роль отмычки (Arduino Uno):

- unlock - команда "открыть замок", отправляется в том случае, если пользователь ввел правильный код

- sleep - команда замку "уйти в сон", отправляется в случае исчерпания пользователем попыток ввода

1. **Экономическая выдержка задачи 2**

В данной задаче использовались следующие элементы:

* плата ESP32-WROOM-32 на базе CP2102 (выкладку о ней см в задаче1)
* плата Arduino Uno R3
* 0,96'' OLED-дисплей SSD1306 с I2C-интерфейсом
* кнопочная матрица 4х4
* светодиод
* резистор на 220 Ом
* Dupont-провода (П-М, 17 штук) (выкладку о них см в задаче1)
* макетная плата для соединения компонентов без пайки

Тут стоит отметить, что плату, дисплей, кнопочную матрицу, провода, светодиод и резистор можно взять из набора, а можно закупить отдельно в зависимости от задач. Рассмотрим оба варианта.

**Набор с Arduino Uno R3**

|  |  |  |
| --- | --- | --- |
| **магазин** | **цена** | **примечания** |
| Ozon | 2359-2366руб | - |
| ЯндексМаркет | 3136-2003руб | - |

**Arduino Uno R3**

|  |  |  |
| --- | --- | --- |
| **магазин** | **цена** | **примечания** |
| Ozon | 372 - 1513 руб | - |
| AliExpress | 167-299руб | - |
| ЯндексМаркет | 560-4013руб | - |
| amperkot.ru | 870руб | Только Москва и Санкт-Петербург |
| procontact74.ru | 598руб | Доставка из Китая, ожидание 4-8 недель, опт 5+ - 402.03руб, 25+ - 328.93руб |
| duino.ru | 700руб | - |
| iarduino.ru | 890руб | Мало на складе |

**0,96'' OLED-дисплей SSD1306 с I2C-интерфейсом**

|  |  |  |
| --- | --- | --- |
| **магазин** | **цена** | **примечания** |
| Ozon | 348 руб | - |
| AliExpress | 100руб | - |
| ЯндексМаркет | 448руб | - |
| avrobot.ru | 239руб | - |
| chipdip.ru | 990руб | - |

**Кнопочная матрица 4х4**

|  |  |  |
| --- | --- | --- |
| **магазин** | **цена** | **примечания** |
| Ozon | 160-242 руб | - |
| AliExpress | 58-189руб | - |
| ЯндексМаркет | 217-1044руб | - |
| arduino-kit.ru | 95руб | - |

**Светодиод**

|  |  |  |
| --- | --- | --- |
| **магазин** | **цена** | **примечания** |
| Ozon | 513-626 руб | 50 шт |
| AliExpress | 74-262руб | 20-50шт |
| chipdip.ru | 9руб | - |
| ledpremium.ru | 0.88-12руб | - |

**резистор на 220 Ом**

|  |  |  |
| --- | --- | --- |
| **магазин** | **цена** | **примечания** |
| Ozon | 103-315 руб | 100 шт |
| AliExpress | 45-54руб | 100шт |
| chipdip.ru | 5руб | 1 шт |
| iarduino.ru | 35руб | 10шт |

**Заключение**

В ходе выполнения работы были успешно реализованы две задачи, каждая из которых позволила углубить понимание принципов работы встроенных систем. Первая задача продемонстрировала возможности ESP32 в части Bluetooth-коммуникации и управления таймерами, а вторая — взаимодействие между ESP32 и Arduino для создания системы с элементами защиты.

Практическая часть работы сопровождалась анализом стоимости компонентов, что помогло оценить экономическую эффективность предложенных решений. Были рассмотрены различные варианты закупки оборудования, что может быть полезно для будущих проектов.

Выполнение работы способствовало развитию навыков программирования микроконтроллеров, работы с периферийными устройствами, а также анализа технико-экономических аспектов проектов. Полученные знания и опыт будут полезны для дальнейшей профессиональной деятельности в области проектирования встроенных систем.

**Приложение**

**Код задачи 1**

**Config**

#ifndef CONFIG\_H

#define CONFIG\_H

// Пин GPIO, к которому подключена кнопка (для ESP32)

const int BUTTON\_PIN = 5;

// Базовое значение, которое передается при старте и после таймаута

const int DEFAULT\_VALUE = 100;

// Время в миллисекундах, после которого система сбрасывается (10 секунд)

const unsigned long TIMER\_DURATION = 10000;

#endif

**Main**

#include <Arduino.h>

#include "BluetoothSerial.h"

#include "config.h"

// Инициализация Bluetooth-интерфейса

BluetoothSerial SerialBT;

// Переменные состояния системы:

unsigned long timerStartMillis = 0; // Время старта таймера

bool timerRunning = false; // Флаг активен ли таймер

int lastBroadcastValue = DEFAULT\_VALUE; // Последнее отправленное значение

bool lastButtonState = HIGH; // Предыдущее состояние кнопки

int value = 1; // Текущее значение для отправки (начинается с 1)

// Функция отправки значения по Bluetooth и в Serial

void broadcastValue(int value) {

if (value != lastBroadcastValue) { // Проверка изменения значения

lastBroadcastValue = value; // Обновление последнего значения

String msg = String("Value: ") + String(value);

Serial.println(msg); // Вывод в монитор порта

SerialBT.println(msg); // Отправка по Bluetooth

}

}

void setup() {

// Настройка кнопки с внутренней подтяжкой к питанию

pinMode(BUTTON\_PIN, INPUT\_PULLUP);

// Инициализация последовательного порта

Serial.begin(115200);

// Запуск Bluetooth с именем устройства

SerialBT.begin("ESP32\_Broadcaster");

// Стартовые сообщения

Serial.println("ESP32 Bluetooth broadcaster started.");

SerialBT.println("Value: " + String(DEFAULT\_VALUE));

}

void loop() {

// Чтение текущего состояния кнопки

bool buttonState = digitalRead(BUTTON\_PIN);

// Обработка нажатия (переход из HIGH в LOW)

if (buttonState == LOW && lastButtonState == HIGH) {

timerStartMillis = millis(); // Старт таймера

timerRunning = true; // Активация флага таймера

// Отправка текущего значения и инкремент

broadcastValue(value);

value += 1; // Увеличение значения на 1

}

lastButtonState = buttonState; // Сохранение состояния кнопки

// Проверка таймера

if (timerRunning) {

// Если таймер истек (10 секунд)

if (millis() - timerStartMillis > TIMER\_DURATION) {

timerRunning = false; // Сброс флага таймера

value = 1; // Сброс значения на начальное

broadcastValue(DEFAULT\_VALUE); // Отправка значения по умолчанию

}

}

delay(50); // Задержка для подавления дребезга кнопки

}

**Код задачи 2**

**ESP**

**Config**

#ifndef CONFIG\_H

#define CONFIG\_H

const int LED\_PIN = 2;

const int MAX\_SLEEP\_TIME\_SEC = 30;

#endif

**Main**

#include <Arduino.h>

#include "config.h"

#include <EEPROM.h>

// Размер выделяемой EEPROM (10 байт)

#define EEPROM\_SIZE 10

// Глобальные переменные состояния

String correctCode = ""; // Хранит правильный код

bool unlocked = false; // Флаг состояния замка

// Функция записи строки в EEPROM

void writeToEEPROM(int addrOffset, String& code) {

int len = code.length();

EEPROM.write(addrOffset, len); // Записываем длину строки

for (int i = 0; i < len; i++) {

EEPROM.write(addrOffset + 1 + i, code[i]); // Посимвольно записываем код

}

EEPROM.commit(); // Сохраняем изменения

}

// Функция чтения строки из EEPROM

String readFromEEPROM(int addrOffset) {

int len = EEPROM.read(addrOffset); // Читаем длину строки

char data[len + 1]; // Буфер для данных

for (int i = 0; i < len; i++) {

data[i] = EEPROM.read(addrOffset + 1 + i); // Читаем посимвольно

}

data[len] = '\0'; // Добавляем терминатор строки

return String(data);

}

// Функция перехода в режим глубокого сна

void goToSleep() {

Serial.println("Going to sleep..."); // Уведомление Arduino

digitalWrite(LED\_PIN, LOW); // Выключаем светодиод

// Настраиваем пробуждение по таймеру

esp\_sleep\_enable\_timer\_wakeup(MAX\_SLEEP\_TIME\_SEC \* 1000000ULL);

delay(100); // Короткая задержка для завершения операций

esp\_deep\_sleep\_start(); // Переход в глубокий сон

}

// Генерация случайного кода

String generateCode(int length = 4) {

String code = "";

for (int i = 0; i < length; i++) {

code += String(esp\_random() % 10); // Цифра от 0 до 9

}

return code;

}

void setup() {

// Настройка вывода для светодиода

pinMode(LED\_PIN, OUTPUT);

digitalWrite(LED\_PIN, LOW);

// Инициализация последовательного порта и EEPROM

Serial.begin(9600);

EEPROM.begin(EEPROM\_SIZE);

delay(1000); // Стабилизация

// Генерация/чтение кода в зависимости от причины пробуждения

if(esp\_sleep\_get\_wakeup\_cause() != ESP\_SLEEP\_WAKEUP\_TIMER) {

// Первый запуск - генерируем новый код

correctCode = generateCode();

writeToEEPROM(0, correctCode);

} else {

// Пробуждение после сна - читаем сохраненный код

correctCode = readFromEEPROM(0);

}

// Отправка статуса Arduino

Serial.println("AWAKE"); // Уведомление о готовности

delay(100);

Serial.println("CODE:" + correctCode); // Отправка кода

}

void loop() {

// Обработка входящих команд от Arduino

if (Serial.available()) {

String input = Serial.readStringUntil('\n');

input.trim();

if (input == "unlock") { // Команда разблокировки

digitalWrite(LED\_PIN, HIGH); // Включаем светодиод

Serial.println("UNLOCKED"); // Подтверждение

unlocked = true;

} else if (input == "sleep") { // Команда сна

goToSleep();

}

}

}

**Arduino**

**Config**

#ifndef CONFIG\_H

#define CONFIG\_H

#include <Arduino.h>

const int SCREEN\_WIDTH = 128;

const int SCREEN\_HEIGHT = 64;

const int OLED\_RESET = -1;

const int ROWS = 4;

const int COLS = 4;

char keys[ROWS][COLS] = {

  {'1','2','3','A'},

  {'4','5','6','B'},

  {'7','8','9','C'},

  {'\*','0','#','D'}

};

const byte colPins[COLS] = {6,7,8,9};

const byte rowPins[ROWS] = {5,4,3,2};

const int maxAttempts = 3;

#endif

**Main**

#include <Arduino.h>

#include <Keypad.h>

#include <Wire.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

#include "config.h"

// Инициализация OLED-дисплея

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, OLED\_RESET);

// Инициализация клавиатуры

Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

// Переменные состояния системы

String correctCode = ""; // Правильный код от замка

String inputCode = ""; // Вводимый пользователем код

int attemptCount = 0; // Счетчик попыток ввода

bool codeReceived = false; // Флаг получения кода от замка

// Функция вывода сообщения на дисплей

void showMessage(const String& msg, int y = 0, bool clear = true) {

if (clear) {

display.clearDisplay(); // Очистка дисплея при необходимости

display.setCursor(0, y); // Установка курсора

display.print(msg); // Вывод текста

display.display(); // Обновление дисплея

} else {

display.setCursor(0, y);

display.print(msg);

display.display();

}

}

// Отправка команды на замок через Serial

void sendCommand(String cmd) {

Serial.println(cmd); // Протоколирование команды

}

// Подсказка - показывает правильно введенные цифры

void giveHint() {

display.setCursor(0, 40); // Позиция для подсказки

for (int i = 0; i < inputCode.length(); i++) {

// Сравниваем каждый символ с правильным кодом

if (i < correctCode.length() && inputCode[i] == correctCode[i]) {

display.print(inputCode[i]); // Показываем правильную цифру

} else {

display.print('\_'); // Маскируем неправильные

}

}

display.display();

}

void setup() {

Serial.begin(9600); // Инициализация последовательного порта

// Инициализация дисплея

if (!display.begin(SSD1306\_SWITCHCAPVCC, 0x3C, false, false)) {

while (true); // Бесконечный цикл при ошибке инициализации

}

// Настройки отображения текста

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(SSD1306\_WHITE);

showMessage("Waiting for lock..."); // Стартовое сообщение

}

void loop() {

// Обработка входящих сообщений от замка

if (Serial.available()) {

String msg = Serial.readStringUntil('\n');

msg.trim();

if (msg.startsWith("CODE:")) { // Получен новый код

correctCode = msg.substring(5); // Извлекаем код из сообщения

inputCode = ""; // Сброс введенного кода

attemptCount = 0; // Сброс счетчика попыток

codeReceived = true; // Флаг получения кода

showMessage("PIN:"); // Приглашение для ввода

}

else if (msg == "AWAKE") { // Замок готов к работе

showMessage("Connected!");

delay(1500);

if (codeReceived) {

showMessage("PIN: "); // Продолжаем запрос кода

} else {

showMessage("Waiting..."); // Ожидаем код

}

}

}

// Обработка нажатий клавиш

char key = keypad.getKey();

if (key && codeReceived) { // Обработка только после получения кода

if (key == '#') { // Подтверждение ввода

attemptCount++;

// Проверка кода

if (inputCode.length() == correctCode.length() && inputCode == correctCode) {

showMessage("Unlocked!"); // Успешное открытие

sendCommand("unlock"); // Команда разблокировки

delay(2000);

showMessage("PIN:"); // Возврат к вводу

}

else if (attemptCount >= maxAttempts) { // Превышены попытки

showMessage("Lock blocked!"); // Блокировка системы

sendCommand("sleep"); // Команда блокировки

codeReceived = false; // Сброс флага

}

else { // Неверный код, но есть попытки

showMessage("Oops! Try " + String(attemptCount) + "/" + String(maxAttempts));

giveHint(); // Показ подсказки

delay(2000);

showMessage("PIN:"); // Повторный запрос

}

inputCode = ""; // Сброс ввода

}

else if (key == '\*') { // Сброс ввода

inputCode = "";

showMessage("PIN:");

}

else if (isdigit(key)) { // Ввод цифр

if (inputCode.length() < correctCode.length()) {

inputCode += key; // Добавление цифры

display.clearDisplay();

showMessage("PIN:", 0, false);

showMessage(inputCode, 20, false); // Показ вводимого кода

}

}

}

}