Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
(повна назва)

Кафедра Автоматизації проектування обчислювальної техніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Бездротова система світломузики з керуванням Андроїд застосунком
(тема)

Виконав:
студент _4_ курсу, групи КІУКІ-17-7
Малишев М. О.
(прізвище, ініціали)

Спеціальність 123 - Комп'ютерна інженерія
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Комп'ютерна інженерія
(повна назва освітньої програми)

Керівник: доцент каф. АПОТ Ларченко Л. В.
(посада, прізвище,ініціали)

Допускається до захисту

Зав. кафедри _____ Чумаченко С.В.
(підпис) (прізвище, ініціали)

2021 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління

Кафедра Автоматизації проектування обчислювальної техніки

Рівень вищої освіти перший (бакалаврський)

Спеціальність 123 - Комп'ютерна інженерія
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Комп'ютерна інженерія
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Малишеву Михайлу Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Бездротова система світломузики з керуванням Андроїд застосунком

затверджена наказом університету від 13.05.2021 р. № 607Ст

2. Термін подання студентом роботи до екзаменаційної комісії 29. 06. 2021 р.

3. Вихідні дані до роботи
Плата NodeMCU v3
Адресна світлодіодна стрічка на основі WS2812b
Смартфон з операційною системою Андроїд

4. Перелік питань, що потрібно опрацювати в роботі
Сучасні системи світломузики
Програмно-апаратний комплекс
Апаратна реалізація проекту
Програмна реалізація проекту

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) _____
_____13 слайдів_____
_____
_____
_____
_____
_____
_____
_____
_____

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Видача теми роботи, її узгодження та затвердження.	21.04.2021– 23.04.2020	
2	Аналіз проблемної області, постановка задачі, вибір засобів реалізації.	24.04.2021– 01.05.2021	
3	Придбання необхідних компонентів	02.05.2021– 05.05.2021	
4	Збірка пристрою	06.05.2021– 09.05.2021	
5	Написання коду для мікроконтролера	10.05.2021– 15.05.2021	
6	Написання Андроїд застосунку.	16.05.2021– 24.05.2021	
7	Оформлення пояснювальної записки	25.05.2021– 31.05.2021	
8	Оформлення графічного матеріалу	01.06.2021– 07.06.2021	
9	Перевірка виконаного проекту керівником	08.06.2021– 14.06.2021	
10	Захист роботи	29.06.2021	

Дата видачі завдання __21_ _квітня_ 2021р.

Студент _____
(підпис)

Керівник роботи _____ __доц. каф. АПОТ Ларченко Л. В._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 42 с., 1 табл., 14 рис., 4 розділи, 3 додатки, 10 джерел.

СВІТЛОМУЗИКА, АДРЕСНА СВТЛОДІОДНА СТРІЧКА , ПРОГРАМА, ПРОШИВКА, ІНТЕРФЕЙС, ВІДЖЕТ, СКЕТЧ, ПІН.

Метою кваліфікаційної роботи є розробка бездротової системи світломузики з керуванням Андроїд застосунком на основі мікроконтролера та адресної світлодіодної стрічки.

В розробленій системі здійснюється перетворення звукового сигналу в світлове миготіння з функцією підсвічення. Розроблено програму-скетч для керування прототипом та прототип бездротової системи світломузики з керуванням Андроїд застосунком, що дозволяє керувати адресною світлодіодною стрічкою за допомогою запрограмованих режимів. Було розглянуто принцип роботи та процес створення даного приладу, виконано підбір компонентів згідно з необхідними параметрами і режимами роботи. Скетч створено за допомогою мов C/C++ у середовищі програмування ArduinoIDE. Андроїд застосунок створено за допомогою мови програмування Kotlin.

ABSTRACT

Explanatory note of the qualification work: 42 pp., 1 tables, 14 figures, 4 sections, 3 appendices, 10 sources.

LIGHT MUSIC, ADDRESSED LED STRIP, PROGRAM, FIRMWARE, INTERFACE, WIDET, SKETCH, PIN.

The purpose of the qualification work is to develop a wireless light music system with Android application control based on a microcontroller and an addressable LED strip.

During the qualification work, a sketch program, an Android application for prototyping and a prototype of a wireless light music system with Android application control were developed, which allows to control the address LED strip using programmed modes. The principle of operation and the process of creating this device were considered, the selection of components in accordance with the required parameters and modes of operation. The sketch was created using C \ C ++ languages in the ArduinoIDE programming environment. Android application was created using the Kotlin programming language.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

АЦП – аналого-цифровий перетворювач

ШІМ – широтно-імпульсна модуляція

GND (GROUND) – точка нульового потенціалу мікросхеми

Скетч (Scetch) – програма-прошивка мікроконтролера

URL (Uniform Resource Locator) – схема створення унікальних адрес електронних ресурсів

SPI (Serial Peripheral Interface) – послідовний синхронний повнодуплексний стандарт передачі даних

GPIO (General-purpose input/output) – інтерфейс для зв'язку між компонентами комп'ютерної системи

HTTP (HyperText Transfer Protocol) – протокол прикладного рівня передачі даних

TCP/IP (Transmission Control Protocol/Internet Protocol) – мережева модель, що описує процес передачі цифрових даних

ВСТУП

В даний час світлодіоди та світлодіодні системи завдяки інноваційним розробкам стали дійсно універсальним рішенням і застосовуються як індикатори в побутовій техніці або самостійно у вигляді енергозберігаючого освітлювального пристрою, в космічній галузі, а також в сфері спецефектів. До останньої можна віднести і світломузику, яка робить наше життя набагато цікавіше.

Системи світломузики автоматично перетворюють звукові сигнали у світлові ефекти, які співпадають з гучністю або частотою сигналів.

Сучасні системи світломузики володіють повним набором функцій для забезпечення різних світлових ефектів, що реагують на звукові хвилі, в яких звук і світло стають єдиним цілим. Конструктивні особливості, а також функціонал систем світломузики залежать від конкретного обладнання.

Сьогодні актуальним напрямком розвитку систем спецефектів стають бездротові системи світломузики з віддаленим керуванням за допомогою мобільних пристроїв. При використанні бездротових технологій покращується зручність керування та зовнішній вигляд систем світломузики в цілому.

Метою кваліфікаційної роботи є розробка бездротової системи світломузики з керуванням Андроїд застосунком на основі мікроконтролера та адресної світлодіодної стрічки.

Проект створений для перетворення аналогового звукового сигналу у світлові хвилі, які залежать від гучності (амплітуди звукового сигналу) або частоти. Проект реалізує три режима роботи світломузики, яка залежить від гучності, частоти та режиму різнокольорового підсвічування.

1 СУЧАСНІ СИСТЕМИ СВІТЛОМУЗИКИ

У розділі розглянуто поняття світломузики, загальні відомості про системи світломузики, визначено мету та постановку завдання.

1.1 Загальні відомості систем світломузики

Світломузика – це вид мистецької та науково-технічної діяльності, заснований на синтезі музики і світла. Системи світломузики автоматично перетворюють звукові сигнали у світлові ефекти, які співпадають з гучністю або частотою сигналів. Сучасні системи світломузики володіють повним набором функцій для забезпечення різних світлових ефектів, що реагують на звукові хвилі, в яких звук і світло стають єдиним цілим. Конструктивні особливості, а також функціонал систем світломузики залежать від конкретного обладнання. Сучасні системи світломузики у своєму складі містять світлодіодні стрічки. Перевагами світлодіодів в системах світломузики є широка кольорова гамма і більш насичене світло; різні варіанти виконання, висока швидкість спрацьовування; низьке енергоспоживання. В основі систем світломузики використовується спосіб частотного перетворення музики і його передачі, за допомогою окремих каналів, для управління джерелами світла. В результаті в залежності від основних музичних параметрів, робота системи світломузики буде їй відповідати.

В системах світломузики керуючим пристроєм є мікроконтролер – пристрій створений на основі інтегральної схеми, призначений для виконання запрограмованих алгоритмів. Типовий мікроконтролер включає в себе мікропроцесор, пам’ять та периферію введення/виведення на одному чіпі, тим самим дуже схожий на систему на кристалі.

У обчислювальній техніці для представлення та обчислення кольору використовуються кольорові моделі. Кольорова модель – метод представлення

кольору в числовому значенні. У обчислювальній техніці для передачі даних використовуються середовища передачі даних.

Середовище передачі даних – фізична субстанція, що є посередником розповсюдження сигналів. Дані можуть модулювати звук, і середовищем передачі звуків може бути повітря, але тверді речовини та рідини також можуть виступати в якості середовища передачі. Вакуум або повітря є хорошим середовищем для пропускання електромагнітних хвиль, таких як світлові та радіохвилі. Хоча матеріальна речовина не потрібна для поширення електромагнітних хвиль, на такі хвилі зазвичай впливають середовища передачі, через які вони проходять, наприклад, поглинанням, відбиттям або заломленням на межі розділу середовищ.

1.2 Мета та постановка завдання

Метою кваліфікаційної роботи є розробка бездротової системи світломузики з керуванням Андроїд застосунком на основі мікроконтролера та адресної світлодіодної стрічки.

Поставлена мета визначила наступні завдання проектування:

– аналіз та вибір мікроконтролеру і компонентів, що входять складовими до бездротової системи світломузики;

– розробка структурної та функціональної схем проекту;

– створення прототипу бездротової системи для перетворення звукового сигналу в світлове миготіння, з функцією підсвічення та керуванням Андроїд застосунком;

– аналіз, вибір мов програмування та середовищ розробки клієнтської та серверної частин проекту;

– створення скетчу мовою C++ в Arduino IDE;

– створення програмного Android застосунку і налаштування Kotlin проекту; тестування спроектованої системи світломузики.

2 ПРОГРАМНО-АПАРАТНИЙ КОМПЛЕКС

У другому розділі здійснено вибір мікроконтролера для проекту та приведено опис обраних компонентів, що входять складовими до бездротової системи світломузики.

2.1 Вибір мікроконтролера для проекту

Для створення бездротової системи світломузики з керуванням Андроїд застосунком було відібрано чотири мікроконтролери ESP8266, ESP32, STM32 та Arduino MKR1000. Для проекту було обрано мікроконтролер ESP8266 (NodeMCU) на основі мікрокопроцесора Tensilica 32-bit RISC CPU Xtensa LX106, основними факторами були та функціональні можливості та ціна.

NodeMCU (Node MicroController Unit) – середовище для розробки програмного та апаратного забезпечення з відкритим кодом, побудоване на системі на чіпі (SoC) під назвою ESP8266. ESP8266, розроблений та виготовлений компанією Espressif Systems, містить найважливіші елементи комп'ютера: процесор, оперативну пам'ять, мережу (WiFi) і сучасну операційну систему та SDK, що може бути використано для для проектів Інтернету речей (IoT)[2].

Подібно до NodeMCU, апаратне забезпечення Arduino - це плата мікроконтролера з роз’ємом USB, світлодіодами та стандартними пінами для передачі даних. Він також визначає стандартні інтерфейси для взаємодії з датчиками або іншими платами. Але на відміну від NodeMCU, плата Arduino може мати різні типи мікросхем процесора з чіпами пам'яті та різноманітні середовища програмування. Для чіпа ESP8266 також є еталонний дизайн Arduino.

2.2 Програмно-апаратна частина для використання Андроїд застосунку

Для реалізації даного проекту було використано смартфон з операційною системою Андроїд.

Андроїд – операційна система для смартфонів, планшетів, електронних книг, цифрових програвачів, наручних годинників, фітнес-браслетів, ігрових приставок, ноутбуків, нетбуків, смартбуків, окулярів Google Glass, телевізорів, проекторів та інших пристроїв. Створена на основі ядра операційної системи Linux, хоча основою операційної системи є Dalvik – регістрова віртуальна машина, що написана на мові програмування Java.

2.3 Опис обраних компонентів системи світломузики

Опис обраних компонентів системи світломузики наведено нижче.

Плата для розробки NodeMCU V3 – плата для розробки на базі чіпа ESP8266(ESP12E), який представляє собою UART-WiFi модуль з низьким споживанням електроенергії (додаток Б). Чіп проектувався для створення приладів інтернету речей, на платі вже реалізовано підключення по USB, встановлений стабілізатор напруги для живлення мікроконтролеру.

Технічні характеристики NodeMCU:

- напруга: 3,3 В;
- Wi-Fi Direct (P2P), soft-AP;
- споживання струму: 10uA ~ 170mA;
- приєднана флеш-пам’ять: макс. 16 Мб ;
- вбудований стек протоколів TCP / IP;
- процесор: Tensilica L106 32-біт;
- швидкість процесора: 80 ~ 160 МГц;
- оперативна пам'ять: 32K + 80K;
- GPIO: 17 (мультиплексовано з іншими функціями) ;
- аналоговий вхід: 1 вхід з роздільною здатністю 1024 кроки;

– вихідна потужність + 19,5 дБ у режимі 802.11b;
– підтримка 802.11: b / g / n;
– максимальне одночасне з'єднання TCP: 5.

Модуль ESP8266 – це модуль Wi-Fi, який належить до сімейства ESP, його можна використовувати для управління проектами в галузі електроніки в будь-якій точці світу. Він має вбудований мікроконтролер та Flash пам’ять розміром 1 Мб, що дозволяє йому підключатися до Wi-Fi. Стек протоколів TCP / IP дозволяє модулю взаємодіяти з сигналами WiFi. Максимальна робоча напруга модуля становить 3.3 В[3].

Технічні характеристики мікроконтролеру ESP8266 (ESP-12E) наведено у таблиці 2.1.

Таблиця 2.1 – Технічні характеристики мікроконтролеру ESP8266 (ESP-12E)

Categories	Items	Values
WiFi Paramters	WiFi Protocles	802.11 b/g/n
	Frequency Range	2.4GHz-2.5GHz (2400M-2483.5M)
Hardware Paramaters	Peripheral Bus	UART/HSPI/I2C/I2S/Ir Remote Contorl
		GPIO/PWM
	Operating Voltage	3.0~3.6V
	Operating Current	Average value: 80mA
	Operating Temperature Range	-40°~125°
	Ambient Temperature Range	Normal temperature
	Package Size	16mm*24mm*3mm
	External Interface	N/A
Software Parameters	Wi-Fi mode	station/softAP/SoftAP+station
	Security	WPA/WPA2
	Encryption	WEP/TKIP/AES
	Firmware Upgrade	UART Download / OTA (via network) / download and write firmware via host
	Ssoftware Development	Supports Cloud Server Development / SDK for custom firmware development
	Network Protocols	IPv4, TCP/UDP/HTTP/FTP
	User Configuration	AT Instruction Set, Cloud Server, Android/iOS App

Зовнішній вигляд та опис пінів ESP8266 (ESP-12E) приведено на рисунку 2.2.

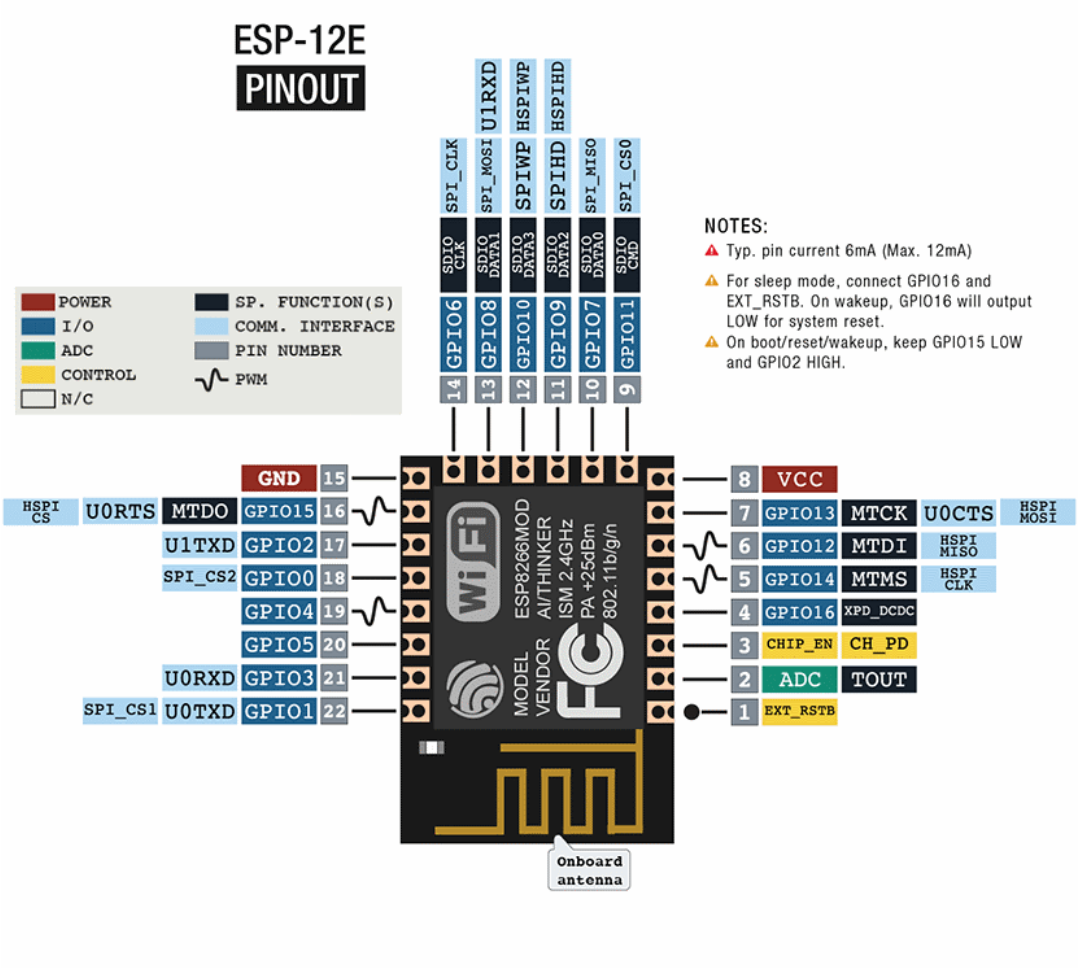


Рисунок 2.2 – Зовнішній вигляд та опис пінів ESP8266 (ESP-12E)

Адресна світлодіодна стрічка на основі WS2812b. Для реалізації проекту обрано адресну світлодіодну стрічку на основі WS2812b (рисунок 2.3). Адресна світлодіодна стрічка – світлодіодна стрічка у якої є можливість керування кольором та яскравістю кожного світлодіоду.

Рисунок 2.3 – Світлодіодна стрічка на основі WS2812b

Всі пікселі у адресній світлодіодній стрічці на основі WS2812b з'єднуються між собою послідовно. Вхід Din кожного з них підключається до виходу Do наступного. Схему чіпу WS2812b приведено на рисунку 2.4.

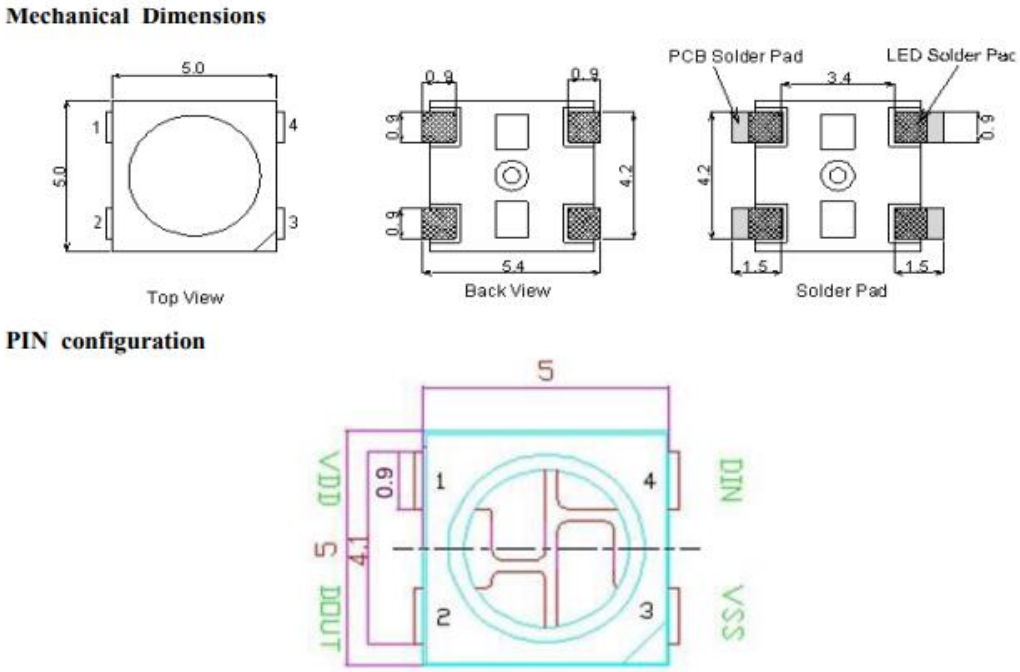


Рисунок 2.4 – Схема чіпу WS2812b

Сигнал управління повинен надходити на перший з них.

Команди управління подаються пакетами по 3 байта, по одному для кожного з трьох кольорів.

Між пакетами йде пауза тривалістю 50 мкс, пауза більше 100 мкс означає кінець передачі.

Тривалість будь-якого біта - 1,25 мкс. Біт "1" кодується імпульсом тривалістю 0,8 мкс і паузою в 0,45 мкс. Біт "0" - 0,4 і 0,85 мкс.

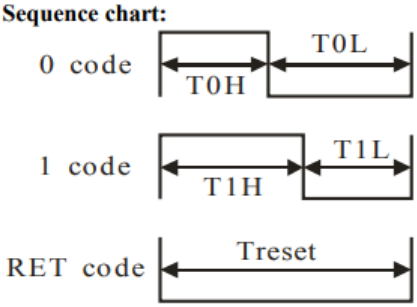
Можливі розбіжності за часом до 150 нс. Такий пакет повинен бути відправлений для кожного пікселя в світлодіодним стрічці[5].

На рисунку 2.5 приведено кодування сигналу для передачі у WS2812b.

Data transfer time(TH+TL=1.25μs±600ns)

T0H	0 code ,high voltage time	0.35us	±150ns
T1H	1 code ,high voltage time	0.7us	±150ns
T0L	0 code , low voltage time	0.8us	±150ns
T1L	1 code ,low voltage time	0.6us	±150ns
RES	low voltage time	Above 50μs	

Sequence chart:



Cascade method:

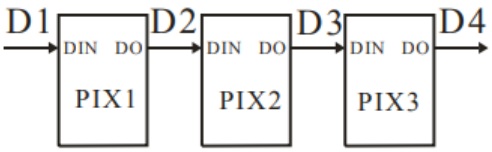


Рисунок 2.5 – Кодування сигналу для передачі у WS2812b

Bluetooth модуль VHM-314-V2.0 – аудіо Bluetooth-приймач з аналоговим AUX виходом (рисунок 2.6).

На платі присутній перемикач режимів живлення.

У положенні VBAT живлення поступає від акумулятора, при підключенні зовнішнього живлення через microUSB здійснюється зарядка, при цьому модуль працює.

У положенні MICRO основне живлення здійснюється від зовнішнього джерела, але при цьому акумулятор заряджається. Якщо відключити зовнішнє живлення модуль відключається. При роботі модуль споживає 20 -30mA.

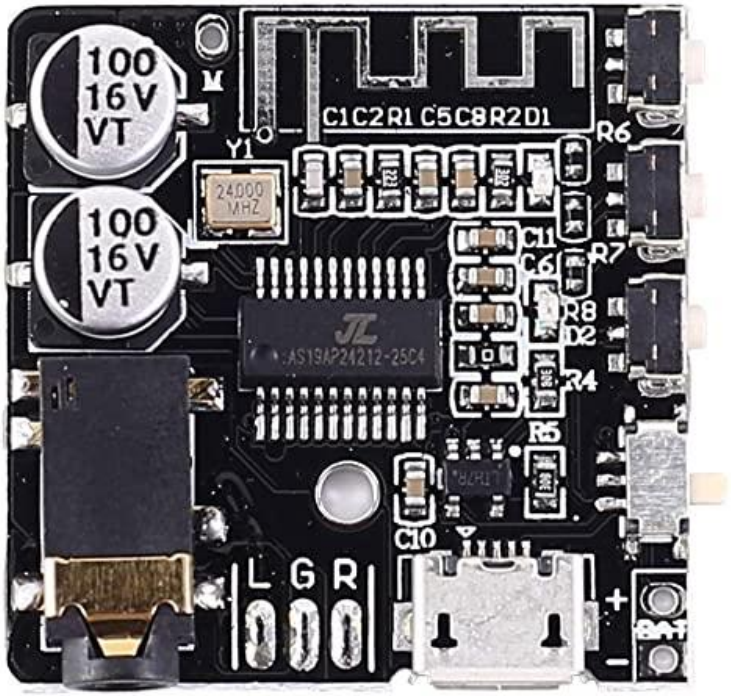


Рисунок 2.6 – Зовнішній вигляд Bluetooth модуля VHM-314-V2.0

Після включення живлення загоряється синій індикатор і модуль входить в режим Bluetooth.

Управління трьома кнопками: центральна стоп/пуск, крайні при короткочасному натисканні гортають треки, при тривалому змінюють гучність.

Технічні характеристики модуля VHM-314-V2.0:

- живлення 3,7-5 V;
- відношення сигнал / шум 90dB;
- THD + N -70dB;
- живлення -86dB;
- ДНР 91dB;
- підтримка профіль A2DP / AVCTP / AVDTP / AVRCP / HFP;
- LOS> 15 м.

Смартфон з операційною системою Андроїд. Було використвно смарфон OnePus 6t з операційною системою Андроїд (рисунок 2.7).

Рисунок 2.7 – Зовнішній вигляд смарфону OnePus 6t

Технічні характеристики OnePus 6t:

– матеріали корпусу: метал, скло;
– операційна система: Android 10 + OxygenOS;
– екран: діагональ 6,418 ", AMOLED, 2340x1080 точок, ppi 402;
– процесор: Qualcomm Snapdragon 845;
– графіка: Adreno 630;
– оперативна пам'ять: 8 ГБ;
– пам'ять для зберігання даних: 128 ГБ;
– основна камера: два модуля 16 + 20 Мп, гібридний автофокус, подвійний LED спалах;
– фронтальна камера: 16 Мп;
– мережі: 2G, 3G, 4G;
– інтерфейси: Wi-Fi, Bluetooth 5.0, USB Type-C;
– навігація: GPS, ГЛОНАСС, BeiDou;
– додатково: акселерометр, датчик освітленості, датчик наближення, гіроскоп, компас, датчик Холла, сканер відбитку пальців;
– батарея 3700 мАг;
– габарити: 157,5 × 74,8 × 8,2 мм, вага: 185 г.

3 АПАРАТНА РЕАЛІЗАЦІЯ ПРОЕКТУ

У розділі розглянуто розроблений прототип бездротової системи світломузики з керуванням Андроїд застосунком, який містить обрані компоненти обладнання. Приведено структурну та функціональну схеми взаємодії компонентів пристрою та результати його тестування.

3.1 Структурна та функціональна схеми з’єднання

Під час розробки даного проекту було створено структурну та функціональну схеми.

Структурна схема описує методи зв’язку компонентів, а функціональна описує фізичне з’єднання.

Для з’єднання мікроконтролера, адресної світлодіодної стрічки, Bluetooth приймача та смартфону використовуються такі середовища передачі даних як радіохвилі та дроти.

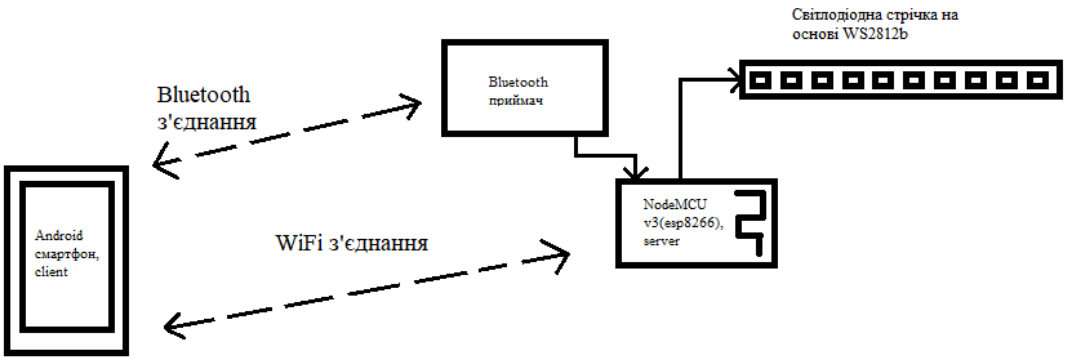


Рисунок 3.1 – Структурна схема проекту

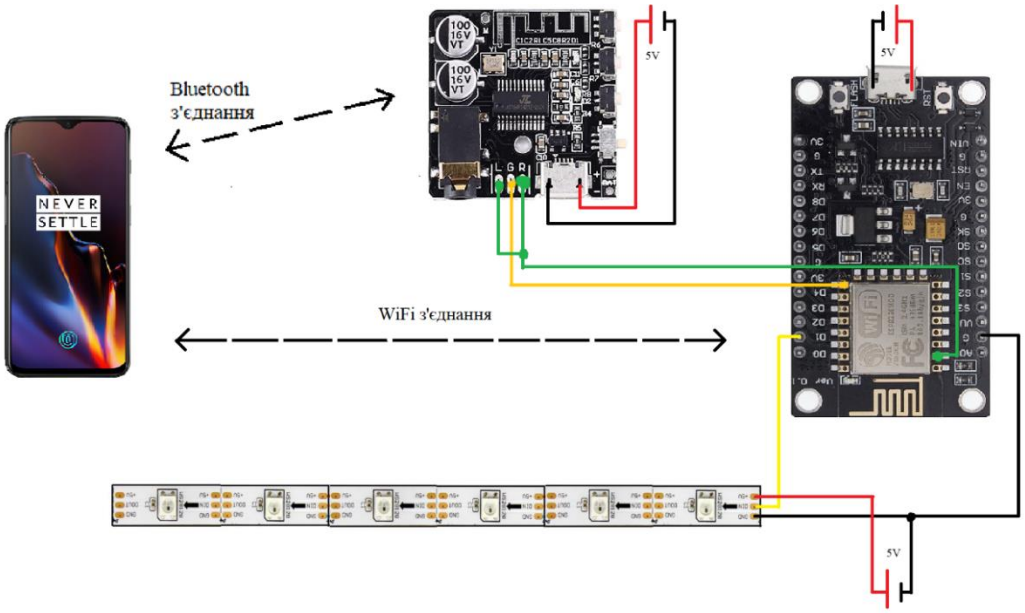


Рисунок 3.2 – Функціональна схема проекту

3.2 Схеми з`єднання компонентів

Розглянемо підключення елементів проекту детальніше.

Для підключення адресної світлодіодної стрічки на основі чіпу WS2812b до мікроконтролеру знадобилося три дроти. Перед підключенням потрібно дізнатися направлення адресної світлодіодної стрічки, через те, що дані для керування адресною світлодіодною стрічкою надсилаються послідовно, з початку і до кінця, через кожній світлодіод. Направлення повинно бути вказано стрілкою на самій світлодіодній стрічці, якщо воно не вказано то на початку світлодіодної стрічки повинен бути пін Din, а на кінці стрічки повинен бути пін Do, як показано на рисунку 3.3.

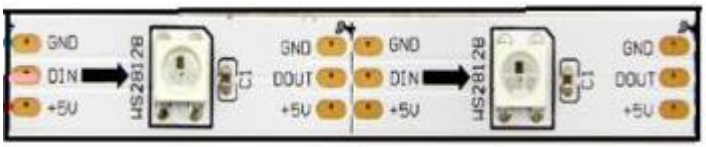


Рисунок 3.3 – Направлення адресної світлодіодної стрічки

Піни +5V, GND адресної світлодіодної стрічки підключаються до блоку живлення з вихідною напругою 5V відповідно. Також мікроконтролер потрібно підключити піном (D1), який підтримує широтно-імпульсна модуляція (ШІМ), до піну Din адресної світлодіодної стрічки. ШІМ – метод зменшення середньої потужності, переданої електричним сигналом, шляхом поділу її на дискретні частини. Середнє значення напруги, що подається на навантаження, регулюється шляхом швидкого вмикання та вимикання перемикача між живленням та навантаженням. Чим довше перемикач знаходиться у включеному стані в порівнянні з періодами вимкнення, тим вище загальна потужність, що подається на навантаження.

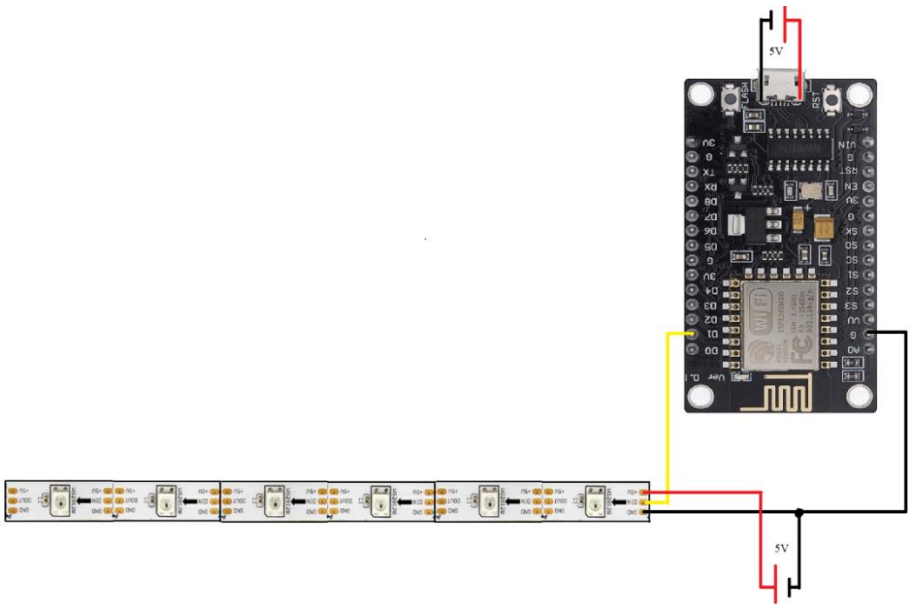


Рисунок 3.4 – Підключення адресної світлодіодної стрічки до мікроконтролеру

Для підключення Bluetooth приймача VHM-314-V2.0 до мікроконтролеру знадобилося три дроти. Піни L, R Bluetooth приймача підключаються до аналогового піну самого мікроконтролера, що використовує аналого-цифровий перетворювач (АЦП), як показано на рисунку 3.5 АЦП – пристрій, що перетворює вхідний аналоговий у цифровий сигнал за

допомогою дискретизації. Характеристиками аналого-цифрового перетворювача є його розрядність (максимальна кількість дискретних значень, яку АЦП може вивести при максимальній амплітуді аналогового сигналу) та частота дискретизації (частота, з якою отримують дискретні значення з аналогового сигналу).

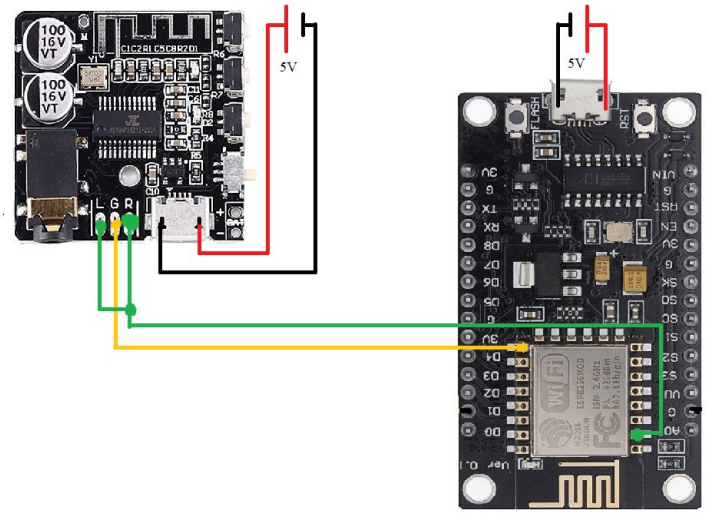


Рисунок 3.4 – Підключення Bluetooth приймача до мікроконтролеру

Підключення смартфону до мікроконтролера та Bluetooth приймача здійснюється за допомогою радіочастот.

3.3 Інструкція користувача

Для використання приладу в режимі точки доступу, потрібно під’єднати живлення до плати NodeMCU, адресної світлодіодної стрічки та Bluetooth приймача, як показано на рисунку 3.5. Після під’єднання живлення до приладу почне працювати режим «райдуга». На смартфоні з операційною системою Андроїд відкрити заздалегідь встановлений застосунок для керування приладом, попередньо підключившись до точки доступу яка створена мікроконтролером. Тепер за допомогою Андроїд застосунку можна керувати приладом.

Якщо використовувати прилад у локальній мережі, спочатку потрібно під’єднати живлення до плати NodeMCU, адресної світлодіодної стрічки та Bluetooth приймача, як показано на рисунку 3.5. Після під’єднання живлення до приладу почне працювати режим «райдуга». далі потрібно відкрити налаштування маршрутизатора та присвоїти статичну IP адресу пристрою, та запам’ятати її. На смартфоні з операційною системою Андроїд відкрити заздалегідь встановлений застосунок для керування приладом, попередньо підключившись до вашої локальної мережі. Останнім кроком буде зміна IP адресу у верхньому рядку в Андроїд застосунку на ту, яку було встановлено в маршрутизаторі. Тепер за допомогою Андроїд застосунку можна керувати приладом.

Керування пристроєм. Для керування пристроєм є чотири поля вводу чисел, та шість кнопок. Поле вводу чисел, над яким є напис «Enter brightness from 0 to 255» відповідає за яскравість світлодіодної стрічки. Інші поля вводу чисел, над якими є написи «Red», «Green» та «Blue» відповідають за колір у форматі RGB. При натисканні на кнопки змінюється режим роботи та оновлюються налаштування кольору та яскравості.

Режими роботи адресної світлодіодної стрічки:

а) кнопка «OFF» – світлодіодна стрічка вимкнена;

б) кнопка «BR LIGHT» – режим підсвічування, є можливість змінювати колір та яскравість;

в) кнопка «RAINBOW» – режим відображення веселки, є можливість змінювати тільки яскравість;

г) кнопка «MAX BRIGHT» – режим відображення максимальної гучності за допомогою яскравості при всіх ввімкнених світлодіодах, можна змінювати тільки колір;

ґ) кнопка «MAX NUMBER» – режим відображення максимальної гучності за допомогою кількості ввімкнених світлодіодів від центру світлодіодної стрічки, можна змінювати колір та яскравість;

д) кнопка «FREQUENCY» – режим відображення максимальної гучності певної частоти звуку за допомогою яскравості у відповідності до номеру світлодіоду на адресній світлодіодній стрічці.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ ПРОЕКТУ

У розділі проведено аналіз та вибір мов програмування та середовищ розробки, дано опис налаштування середовища розробки та підключення бібліотек Arduino IDE, створення скетчу мовою C++ в Arduino IDE , створення та налаштування Kotlin проекту

4.1 Аналіз, вибір мов програмування та середовищ розробки

Для написання скетчу для мікроконтролера ESP8266 можна використовувати такі середовища програмування, як Arduino IDE, Visual Studio Code, Microsoft Visual Studio, Eclipse C++, ESPlorer, MicroPython, текстовий редактор та такі мови програмування, як Assambly, C/C++, Lua, Python.

Мова Assambly – низькорівнева мова програмування, яка перетворює лексеми машинного коду в байт код. Ця мова досить складна для вивчення, тому що потрібно знати архітектуру пристрою для якого має бути написана програма. Також за допомогою цієї мови важко писати великі програми через те, що програміст повинен слідкувати за усіма процесами, які відбуваються.

Якщо порівнювати мови C/C++, Lua та Python то найменший розмір зібраного проекту буде, написаний на мовах C/C++. Крім того на мовах C/C++ написано величезну кількість бібліотек, які іноді використовуються в програмах, що написані на інших мовах та програми, які виконані на мовах C/C++ можуть швидше працювати, ніж такі ж самі програми, написані на мовах Lua та Python. Через це були обрані мови C/C++.

З приводу середовищ програмування було обрано Arduion IDE, що обумовлено простотою налаштування як самого середовища, так і проекту, просте підключення бібліотек та зручний інтерфейс[1].

Для розробки Андроїд застосунку було обрано мову програмування

Kotlin через те, що мову розроблено спеціально для створення Андроїд застосунків, що має велику кількість бібліотек для Андроїд, навіть створених на інших мовах програмування, яка має простий синтаксис та відсутність потреби у підключенні та налаштуванні додаткових плагінів.

При виконанні програмної реалізації проекту було обрано середовище програмування android studio, що обумовлено єдиним розробником, компанією Google, як середовища програмування android studio, так і операційної системи Андроїд.

4.2 Налаштування середовища розробки Arduino IDE

Програма/скетч була розроблена в середовищі програмування Arduino IDE, за допомогою мови програмування C++ та з використанням операційної системи Windows 10.

Перед початком роботи виконується перевірка встановлення драйверів для плати Arduino. Для цього необхідно відкрити «Диспетчер пристроїв» на локальному комп’ютері та під’єднати плату Arduino до локального комп’ютера за допомогою USB. Якщо в назві пристрою присутнє «CH340», це означає, що драйвер встановлено, але якщо пристрій відображається як «Невідомий пристрій» потрібно встановити драйвер, який знаходиться в папці з файлами Arduino IDE «\drivers\dpinst-x86.exe».

Для роботи в Arduino IDE з мікроконтролером ESP8266 потрібно запустити середовище розробки і відкрити Файл / Налаштування, вставити посилання (http://arduino.esp8266.com/stable/package_esp8266com_index.json) в поле «Додаткові посилання для Менеджера плат:" і натиснути "OK".

Далі у пункті меню Інструменти / Плата / Менеджер плат необхідно вибрати останню версію та натиснути Встановити. Тепер в панелі Інструменти має з’явитись ESP8266[4].

4.3 Підключення бібліотек в Arduino IDE

При створенні скетчу було використано бібліотеки: ESP8266WiFi, ESP8266WebServer, FastLED та arduinoFFT. Для подальшого використання спочатку потрібно перевірити, чи встановлені необхідні бібліотеки. Для цього потрібно у Arduino IDE, що запущений, відкрити пункт меню Скетч / Підключити бібліотеку та перевірити наявність вище вказаних бібліотек. Якщо наявність бібліотек відсутня, необхідно завантажити бібліотеки і надалі додати до середовища розробки Arduino IDE, відкривши Скетч / Підключити бібліотеку / Додати .ZIP бібліотеку, вибравши на локальному пристрої заздалегідь завантажену бібліотеку у форматі архіву .ZIP.

У мові C++ перед використанням бібліотеки, її потрібно підключити у файлі «WierlessLED». Для цього використовується препроцесорна директива include.

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
#include <FastLED.h>
#include <arduinoFFT.h>
```

ESP8266WiFi – бібліотека створена на основі WiFi.h з бібліотеки Arduino WiFi shield. За допомогою бібліотеки існує можливість підключати мікроконтролер до WiFi мережі, створювати власну точку доступу та обмінюватися даними з іншими користувачами даної WiFi мережі.

ESP8266WebServer – бібліотека, за допомогою якої можна створити Web сервер з одночасною роботою одного клієнта. Дана бібліотека підтримує HTTP GET та POST запити.

HTTP (HyperText Transfer Protocol) – протокол передачі даних в комп’ютерних мережах який в основному використовується для передачі веб-сторінок.

FastLED – бібліотека для простого та ефективного управління різноманітними світлодіодними чіпами. Для керування світлодіодами можуть використовуватися кольорові моделі RGB та HSV. Окрім керування світлодіодами бібліотека включає ряд функцій для 8-бітної математики для маніпулювання значеннями RGB, а також низькорівневі класи для абстрагування доступу до пінів та апаратного забезпечення SPI(протокол передачі даних), зберігаючи при цьому якомога більшу швидкість[6].

RGB (абревіатура англійських слів red, green, blue) – кольорова модель, яка описує спосіб кодування кольору за допомогою трьох кольорів: червоного, зеленого та синього. Дана кольорова модель працює завдяки додаванню кольорів до чорного кольору. Якщо червоний, зелений та синій дорівнюють нулю, відображається чорний колір, але якщо червоний, зелений та синій мають максимальне значення, відображається білий колір.

HSV (абревіатура англійських слів hue, saturation, value) – кольорова модель, яка описує спосіб кодування кольору за допомогою трьох значень: тон кольору, насиченість та яскравість як вказано в назві. У даній кольоровій моделі тон кольору знаходиться в діапазоні 0-360°, для зручності зберігання даних іноді використовується інші діапазони: 0-100, 0-1, 0-255 та інші. Також тон кольору циклічний, тобто якщо після 360° перейти доразу до 0° різкої зміни тону кольору не буде.

Бібліотека підтримується на різних сумісних платформах Arduino. Підтримуються ARM та AVR при використанні програмного забезпечення Arduino або його модифікованої версії. На даний момент підтримуються лише компілятори, які постачаються з програмним забезпеченням Arduino.

Плати/мікроконтролери, що підтримуються: Arduino та сумісні пристрої, Arduino Yún, Adafruit Trinket, Adafruit Gemma, Teensy 2, Teensy++ 2, Teensy 3.0, Teensy 3.1/3.2, Teensy LC, Arduino Due, RFDuino, SparkCore, Arduino Zero, ESP8266 та The wino board.

Світлодіодні чіпи, що підтримуються: DotStars Adafruit(APA102), Neopixel Adafruit(WS2812B, WS2811, WS2812), TM1809, TM1803, UCS1903,

GW6205, LPD8806, WS2801, SM16716, APA102 та P9813(Cool Neon's Total Control Lighting).

Панелі SmartMatrix - потрібна бібліотека SmartMatrix - https://github.com/pixelmatix/SmartMatrix.

Швидке перетворення Фур’є – це математична функція прискореного обчислення дискретного перетворення Фур'є, що дозволяє отримати з тимчасової залежності сигналу його частотні компоненти, тобто проводити спектральний аналіз сигналів. Велика частина бібліотек для платформи Arduino з реалізацією швидкого перетворення Фур’є, написані з використанням асемблерних вставок, а мікроконтролери Arduino та ESP8266 мають різну архітектуру, через це неможливо використовувати дані бібліотеки. В результаті було обрано бібліотеку arduinoFFT, яка написана на мовах C/C++, і реалізує швидке перетворення Фур’є. В даному проекті arduinoFFT використовується для розкладання аудіо сигналу на частотний діапазон[7].

4.4 Створення скетчу мовою C++ в Arduino IDE

Файл «WierlessLED». Створюємо об’єкт класу IPAddress. Цей об’єкт створений для зберігання IP адреси, яка використовується для створення та підключення до точки доступу:

```
IPAddress apIP(192, 168, 4, 1);
```

Створюємо об’єкт класу ESP8266WebServer. Цей об’єкт створений для зберігання номеру мережевого порту, який надалі буде використовувати Web сервер:

```
ESP8266WebServer HTTP(80);
```

Директива define створює макрос, який здійснює заміну імені макросу (його ідентифікатор) на його значення. Дана директива використовується як заміна константним змінним, так як макрос не займає пам’ять пристрою. У даній частині коду, що приведено нижче, визначено MIC_PIN – пін, що використовується для отримання аудіо сигналу, NUM_LEDS – кількість світлодіодів на адресній світлодіодній стрічці, PWM_PIN – пін, що повинен мати ШІМ та використовується для керування адресною світлодіодною стрічкою, LED_MODEL – назва моделі світлодіоду має співпадати з переліком підтримуваних світлодіодів FastLED, SAMPLES – кількість вимірів звуку для швидкого перетворення Фур’є, RAINBOW_DELAY – затримка для відображення режиму райдуги, AUDIO_DELAY – затримка для відображення трьох режимів, SATURATION – контроль насиченості кольорів світлодіодів (лише для режиму веселки):

```
#define MIC_PIN A0
#define NUM_LEDS 60
#define PWM_PIN 1
#define LED_MODEL WS2812B
#define SAMPLES 128            //Must be a power of 2
#define RAINBOW_DELAY 50
#define AUDIO_DELAY 50
#define SATURATION 255   // Control the saturation of your leds (only for rainbow mode)
```

Створення та ініціалізація об’єкту FFT класу arduinoFFT. Даний об’єкт створений для подальшої роботи зі швидким перетворенням Фур’є:

```
arduinoFFT FFT = arduinoFFT();
```

Створення масивів vReal, vImag з кількістю елементів, що дорівнює значенню макросу SAMPLES. Ці масиви використовуються для введення та отримання даних з методів швидкого перетворення Фур’є:

```
double vReal[SAMPLES];
double vImag[SAMPLES];
```

Створення масиву для керування адресною світлодіодною стрічкою:

```
CRGB leds[NUM_LEDS];
```

Глобальні змінні, які зберігають ім’я та паролі як для підключення мережі, так і для створення точки доступу:

```
const String _ssid = "home",
_password = "i12345678",
_ssidAP = "WiFi",
_passwordAP = "";
```

Далі наведені змінні, які зберігають режим роботи світлодіодної стрічки, попередній режим роботи світлодіодної стрічки, яскравість, та кольори: червоний, зелений, блакитний. На початку роботи або після перезавантаження мікроконтролера буде включений другий режим роботи – райдуга адресної світлодіодної стрічки та її яскравість буде дорівнювати 50:

```
unsigned int current_mode = 2,
previous_mode;
uint8_t brightness = 50,
red, green, blue;
```

Глобальна змінна яка зберігає максимальну гучність аудіо сигналу:

```
int max_vol;
```

Стандартна функція скетчу в Arduino IDE. Ця функція виконується один раз на початку роботи скетчу та слугує для початкового налаштування та ініціалізації об’єктів. В даному випадку у цій функції викликаються такі функції: WIFIinit, HTTP_init та fastLed_init:

```
void setup() {
WIFIinit();
HTTP_init();
fastLed_init();}
```

Глобальна змінна для реалізації затримки за допомогою стандартної функції millis(функція для отримання внутрішнього часу мікроконтролеру):

```
int start_t,
t_dif;
```

Loop – стандартна функція скетчу в Arduino IDE (додаток А). Дана функція реалізує нескінченний цикл. У цій функції описується основний функціонал скетчу. Спочатку викликається метод handleClient, який перевіряє та оброблює HTTP запити до web серверу. В коректному HTTP запиті повинен міститися номер режиму роботи адресної світлодіодної стрічки, що зберігається у глобальній змінній current_mode.

Режими роботи адресної світлодіодної стрічки:

а) current_mode = 0 (світлодіодна стрічка вимкнена);

б) current_mode = 1 (режим підсвічування, є можливість змінювати колір та яскравість);

в) current_mode = 2 (режим відображення веселки, є можливість змінювати тільки яскравість);

г) current_mode = 3 (режим відображення максимальної гучності за допомогою яскравості при всіх ввімкнених світлодіодах, можна змінювати тільки колір);

ґ) current_mode = 4 (режим відображення максимальної гучності за допомогою кількості ввімкнених світлодіодів від центру світлодіодної стрічки, можна змінювати колір та яскравість);

д) current_mode = 5 (режим відображення максимальної гучності певної частоти звуку за допомогою яскравості у відповідності до номеру світлодіоду на адресній світлодіодній стрічці).

Режим обертається за допомогою оператора switch. У кожному case обраного режиму викликається функція відповідно до обраного режиму.

Якщо режим обраний неправильно, відтворюється попередній.

Для режимів current_mode = 2, 3, 4 та 5 встановлена затримка в 50 мілісекунд за допомогою стандартної функції millis(функція для отримання внутрішнього часу мікроконтролеру).

Файл «HTTP_init». У файлі HTTP_init (додаток А) описані усі налаштування для HTTP серверу (додаток А).

У функції HTTP_init додаються HTTP GET запити за допомогою методу on, які будуть використовуватись в подальшому та за допомогою методу begin запускається HTTP сервер.

Функція handleRoot викликається HTTP сервером, якщо була введена коректна IP адреса, після чого відправляється HTTP POST запит з текстом correct IP.

Функція led_settings викликається HTTP сервером, якщо був введений коректний HTTP GET запит (після IP адреси він має містити /led) для керування адресною світлодіодною стрічкою. Після /led для керування адресною світлодіодною стрічкою, можуть бути використані наступні аргументи: mode, brightness, red, green та blue. Дані, що отримані з аргументів зберігаються у змінних: current_mode, brightness, red, green та blue, після чого відправляється HTTP POST запит з текстом «OK».

Файл «LED». У цьому файлі (додаток А) зберігаються функції для ініціалізації адресної світлодіодної стрічки та її всі режими роботи. Відображення світлодіодами відбувається за рахунок кольорового простору RGB (додаток А).

Функція fastLed_init ініціалізує об’єкт FastLED для роботи з адресною світлодіодною стрічкою.

Функція mode_off вимикає усі світлодіоди на стрічці.

Функція mode_backlight реалізує режим підсвічування за допомогою адресної світлодіодної стрічки. Підсвічення залежить від змінних red, green, blue та brightness.

Програмний код, що написаний у функції rainbow реалізує режим райдуги за допомогою адресної світлодіодної стрічки. В програмном коді використовуються затримки з використанням стандартної функції millis (функція для отримання внутрішнього часу мікроконтролеру), перетворення кольорового простору HSV (використовується для реалізації «райдуги») у RGB. Режим відображення райдуги залежить від змінної brightness. Також була створена додаткова глобальна змінна яка зберігає крок, на якому зупинилося відображення райдуги для роботи.

Функція display_max_vol_br відображає максимальну гучність яскравістю усіх світлодіодів на адресній світлодіодній стрічці. Даний режим залежить від змінних red, green та blue. Також викликається функція get_max_vol для отримання гучності в конкретний проміжок часу.

Функція display_max_vol_num відображає гучність за допомогою кількості ввімкнених світлодіодів від центру світлодіодної стрічки. Викликається функція get_max_vol для отримання гучності в конкретний проміжок часу. Режим залежить від змінних red, green, blue та brightness.

Для того, щоб мікроконтролер не перераховував зазначення кожного виклику функції friqency_mode було створено константну глобальну змінну для збереження значення, яке потрібно для відображення роботи швидкого перетворення Фур’є. Функція friqency_mode спочатку викликає функцію get_fft_from_audio, для отримання аудіо даних після швидкого перетворення Фур’є. Далі ці дані перетворюються та виводяться на адресну світлодіодну стрічку.

Файл «WIFI». Файл WIFI (додаток А) слугує для ініціалізації роботи технології WIFI та для вибору режиму роботи: точка доступу або підключення до маршрутизатору.

Функція WIFIinit створена для ініціалізації, запуску точки доступу або підключення до WIFI мережі. Підчас роботи даної функції мікроконтролер намагається підключитися до WIFI мережі за допомогою імені мережі та паролю, які знаходяться у змінних _ssid та _password відповідно. Для того, щоб мікроконтролер встиг підключитися до WIFI мережі використовується змінна tries та стандартна функція затримки delay (в даному випадку її можна використовувати, незважаючи на те, що вона зупиняє роботу мікроконтролера в даному місці на ту кількість мілісекунд, яка вказана аргументом, через те що функція WIFIinit виконується лише раз на початку роботи скетчу). Якщо підключення було здійснено, на цьому моменті завершується робота функції, якщо ні – викликається функція StartAPMode.

Функція StartAPMode слугує для створення мікроконтролером точки доступу. Для створення точки доступу потрібно встановити маску IP мережі, що в даному прикладі дорівнює 255, 255, 255, 0. Також потрібно встановити ім’я, пароль для точки доступу які зберігаються у змінних: _ssidAP та _passwordAP відповідно.

Файл «audio_preparation» (додаток А) – файл у якому міститься дві невеликі функції обробки аудіо сигналу.

Функція get_max_vol створена для отримання гучності в конкретний проміжок часу. Для цього послідовно здійснюється зчитування сто значень аудіо сигналу та вибирається найбільше з них та записується у змінну max_vol.

У функції get_fft_from_audio аудіо сигнал обробляється за допомогою швидкого перетворення Фур’є.

4.5 Створення та налаштування Kotlin проекту

Kotlin – об'єктно-орієнтована, статично типізована мова програмування, завдяки Java Virtual Machine, що забезпечує повну підтримку мови програмування Java та її бібліотек[10].

При створенні проекту було обрано Empty Activity. На рисунку 4.1 приведено створення Kotlin проекту.

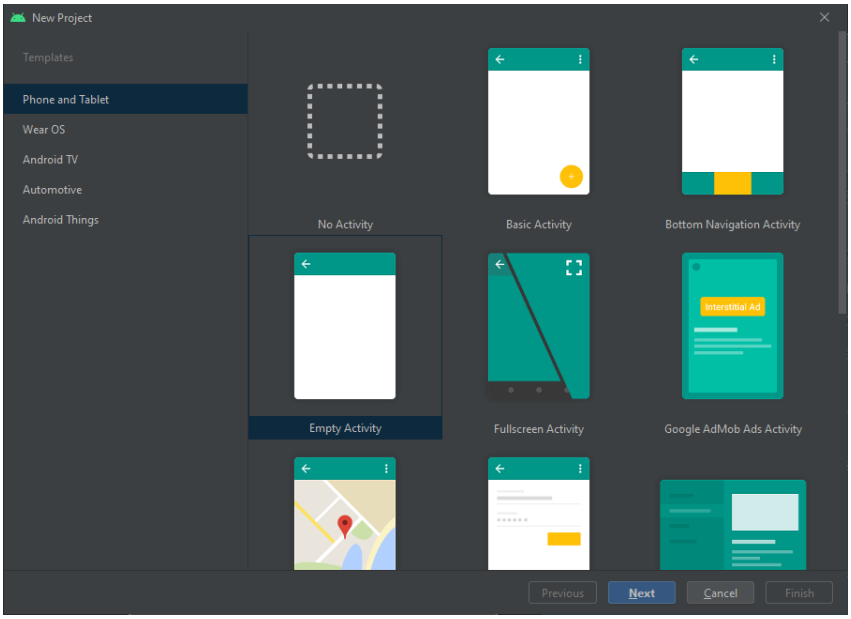


Рисунок 4.1 – Створення Kotlin проекту

Після створення проекту в файл AndroidManifest.xml було додано запит на доступ до Інтернету:

```
<uses-permission android:name="android.permission.INTERNET"/>
```

Також у файлі AndroidManifest.xml, у тегу <application> було додано дозвіл на усі запити без шифрування:

```
android:usesCleartextTraffic="true"
```

4.6 Створення застосунку мовою Kotlin

При розробці програмного застосунку було створено користувацький інтерфейс за допомогою графічного редактору (рисунок 4.2). При цьому було додано шість елементів типу button, шість елементів типу TextView, чотири

елементи типу Number, один елемент типу Plain Text та один елемент типу WebView.

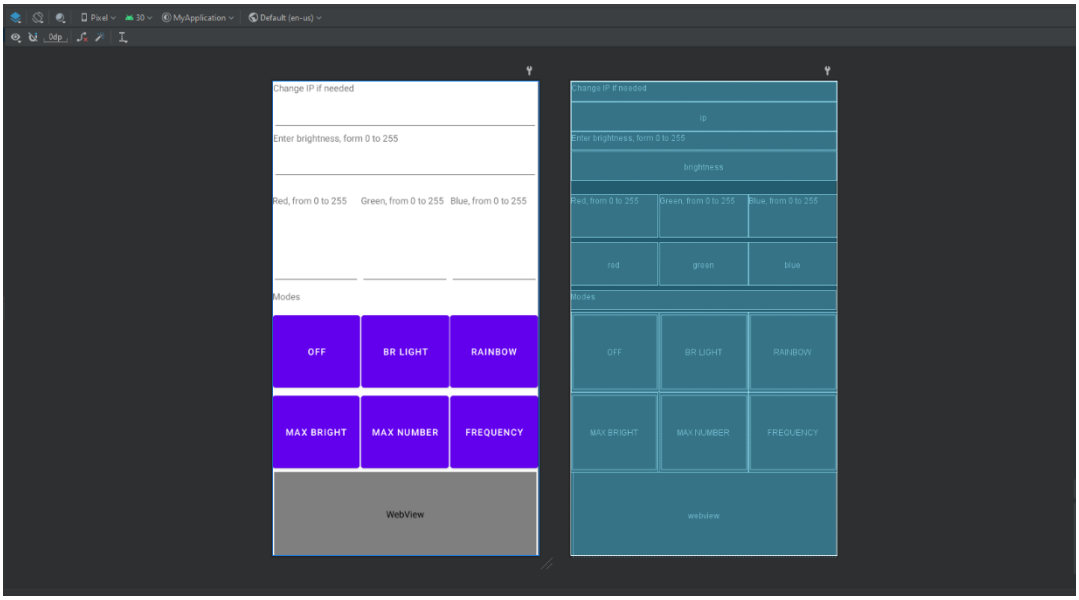


Рисунок 4.2 – Створення користувацького інтерфейсу

Для кожного елементу було встановлено віджет обмеження (рисунок 4.3).

У файлі MainActivity.kt (додаток А) функція onCreate викликається при запуску застосунку. Підчас роботи цієї функції запускається Web сервер, встановлюються значення IP адреси в поле Plain Text, встановлюється значення яскравості у полі Number та приховується елемент WebView який використовується для роботи Web серверу.

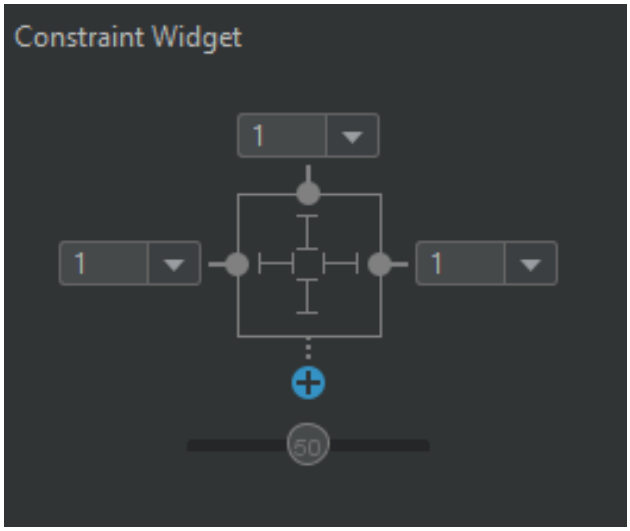


Рисунок 4.3 – Віджет обмеження

Кожен елемент типу button відповідає за певний режим відображення на адресній світлодіодній стрічці. Для кожного елементу типу button було створено функцію для обробки натискання.

Усі функції обробки натискання на елементи button схожі, у функціях створюється URL адреса натиснутої кнопки, IP адреси (знаходиться в полі вводу елементу Plain Text), яскравості (знаходиться в полі вводу елементу Number) та кольору (знаходиться в полях вводу елементів Number). Далі відправляється HTTP запит за допомогою URL адреси.

ВИСНОВКИ

Підчас виконання кваліфікаційної роботи було розроблено бездротову систему світломузики з керуванням Андроїд застосунком на основі мікроконтролера, адресної світлодіодної стрічки та смартфону з операційною системою Андроїд, де мікроконтролер виконує серверну частину, а смартфон є клієнтом. У ході виконання роботи було виконано підбір компонентів приладу з необхідними технічними характеристиками та функціями, створено структурну та функціональну схеми, створено програмний код мікроконтролерної частини та частини Андроїд застосунку.

Основою розробленої системи є плата NodeMCU v3, що має надзвичайно великий потенціал. Основними переваги даної системи є швидкість роботи, застосунок для керування за допомогою смартфону з найпопулярнішою операційною системою та зручність монтажу. Даний пристрій може бути прикладом використання мікроконтролрів, локальних комп’ютерних мереж, технології WiFi, технології SPI, стеку протоколів TCP/IP, протоколу HTTP та технології Bluetooth. Прототип бездротової системи світломузики може бути вдосконалений. По-перше можна покращити користувацький інтерфейс Андроїд застосунку та додати більше режимів роботи. По-друге використати Li-ion акумулятор разом з платою для контроля рівня напруги Li-ion акумуляторів для того щоб пристрій став портативним. І нарешті поєднати усі компоненти в невеликому корпусі для зручності використання.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Arduino [Електронний ресурс]. – Режим доступу: https://www.arduino.cc/. – Дата доступу: 15.02.2021 р.

2. NodeMCU V3 [Електронний ресурс]. – Режим доступу: https://curtocircuito.com.br/datasheet/esp8266.pdf. – Дата доступу: 15.02.2021 р.

3. ESP8266 (ESP-12E) [Електронний ресурс]. – Режим доступу: https://www.alldatasheet.com/datasheet-pdf/pdf/1179099/ETC2/ESP12E.html. – Дата доступу: 16.02.2021 р.

4. Arduino on ESP8266 [Електронний ресурс]. – Режим доступу: https://github.com/esp8266/Arduino#readme. – Дата доступу: 17.02.2021 р.

5. WS2812b [Електронний ресурс]. – Режим доступу: https://cdn-shop.adafruit.com/datasheets/WS2812B.pdf. – Дата доступу: 20.02.2021 р.

6. FastLED [Електронний ресурс]. – Режим доступу: http://fastled.io/docs/3.1/md__r_e_a_d_m_e.html. – Дата доступу: 23.02.2021 р.

7. ArduinoFFT [Електронний ресурс]. – Режим доступу: https://github.com/kosme/arduinoFFT. – Дата доступу: 03.03.2021 р.

8. Фрунзе А.В. Микроконтроллеры? Это же просто! — ДДЭКА 2015.

9. Таненбаум Э. С. Уэзеролл Д. Компьютерные сети. 5-е изд. — СПб.: Питер, 2012.

10. Kotlin [Електронний ресурс]. – Режим доступу: https://kotlinlang.org/docs/home.html. – Дата доступу: 21.03.2021 р.

ДОДАТОК А

Лістинг програмного коду

Лістинг А.1 – Програмний код мікроконтролеру, файл «WierlessLED»

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
#include <FastLED.h>
#include <arduinoFFT.h>

IPAddress apIP(192, 168, 4, 1);

ESP8266WebServer HTTP(80);

#define MIC_PIN A0

#define NUM_LEDS 60
#define PWM_PIN 1
#define LED_MODEL WS2812B

arduinoFFT FFT = arduinoFFT();

#define SAMPLES 128             //Must be a power of 2
double vReal[SAMPLES];
double vImag[SAMPLES];

CRGB leds[NUM_LEDS];

#define RAINBOW_DELAY 50
#define AUDIO_DELAY 50

const String _ssid = "home", // SSID
  _password = "i12345678",
  _ssidAP = "WiFi",   // SSID AP
  _passwordAP = "";

unsigned int current_mode = 2,
  previous_mode;
uint8_t brightness = 50,
  red, green, blue;

#define SATURATION 255   // Control the saturation of your leds (only for rainbow
mode)

int max_vol;

void setup() {
  Serial.begin(115200);
  Serial.println("");
  Serial.println("Start WIFI");
  WIFIinit();
  Serial.println("Start WebServer");
```

```
  HTTP_init();
  fastLed_init();}

 int start_t,
  t_dif;

 void loop() {
  HTTP.handleClient();

  switch(current_mode)  {
    case 0:
    mode_off();
    previous_mode = current_mode;
    break;

    case 1:
    mode_backlight();
    previous_mode = current_mode;
    break;

    case 2:
    rainbow();
    previous_mode = current_mode;
    break;

    case 3:
    t_dif = millis() - start_t;
    if(t_dif>=AUDIO_DELAY && t_dif)    {
      display_max_vol_br();
      previous_mode = current_mode;    }
    break;

    case 4:
    t_dif = millis() - start_t;
    if(t_dif>=AUDIO_DELAY && t_dif)    {
      display_max_vol_num();
      previous_mode = current_mode;    }
    break;

    case 5:
    t_dif = millis() - start_t;
    if(t_dif>=AUDIO_DELAY && t_dif)    {
      friqency_mode();
      previous_mode = current_mode;    }
    break;

    default:
    current_mode = previous_mode;
    break;  }
 }
```

Лістинг А.2 – Програмний код мікроконтролеру, файл «HTTP_init»

```
 void HTTP_init(void) {
  HTTP.on("/", handleRoot);
```

```
  HTTP.on("/led", led_settings);
  HTTP.begin();}

void handleRoot() {
  HTTP.send(200, "text/plain", "correct IP");}

void led_settings(){
  current_mode = atoi(HTTP.arg("mode").c_str());
  brightness = atoi(HTTP.arg("brightness").c_str());
  red = atoi(HTTP.arg("red").c_str());
  green = atoi(HTTP.arg("green").c_str());
  blue = atoi(HTTP.arg("blue").c_str());
  HTTP.send(200, "text/plain", "OK");  }
```

Лістинг А.3 – Програмний код мікроконтролеру, файл «LED»

```
void fastLed_init(){
   FastLED.addLeds <LED_MODEL, PWM_PIN, GRB>(leds,
NUM_LEDS).setCorrection(TypicalLEDStrip);}

void mode_off(){
   for(int i = 0; i < NUM_LEDS; i++)   {
    leds[i].setRGB(0, 0, 0);
    leds[i].fadeLightBy( 255 );   }
   FastLED.show();}

void mode_backlight(){
   for(int i = 0; i < NUM_LEDS; i++)   {
    leds[i].setRGB(red, green, blue);
    leds[i].fadeLightBy( 255 - brightness);   }
   FastLED.show();}

byte for_loop;

inline void rainbow(){
  t_dif = millis() - start_t;
  if(t_dif>=RAINBOW_DELAY && t_dif)  {
   for (int i = 0; i < NUM_LEDS; i++) {
    hsv2rgb_rainbow( CHSV(i - (for_loop * 2), SATURATION, brightness), leds[i]);   }
   ++for_loop;
   FastLED.show();
   start_t=millis();  }}

void display_max_vol_br(){
   get_max_vol();
   for(int i = 0; i < NUM_LEDS; i++)   {
    leds[i].setRGB(red, green, blue);
    leds[i].fadeLightBy( 255 - max_vol);   }
   FastLED.show();}

void display_max_vol_num(){
  get_max_vol();
  int num = (max_vol * NUM_LEDS)/1024;
   for(int i = 0; i < NUM_LEDS; i++)   {
```

```
          if((NUM_LEDS/2 - num/2) <= i && i <=((NUM_LEDS/2 - 1) + num/2))    {
            leds[i].setRGB(red, green, blue);
            leds[i].fadeLightBy( 255 - brightness);    }
          else    {
            leds[i].setRGB(0, 0, 0);
            leds[i].fadeLightBy( 255 - brightness);    }   }
         FastLED.show();}

      const int v = (SAMPLES/2 - 3)/NUM_LEDS;

      void friqency_mode(){
        get_fft_from_audio();
        int max_val = 250;
        for (int i = 0; i < NUM_LEDS; ++i)  {
          int value = int(vReal[i * v + 3]);
          if(value>max_val)    {
            max_val = value;    }
          else if(value < 250)    {
            vReal[i * v + 3] = 0;    }  }

        for (int i = 0; i < NUM_LEDS; ++i)  {
          leds[i].setRGB(red * int(vReal[i * v + 3]) / max_val, green * int(vReal[i * v + 3]) /
max_val, blue * int(vReal[i * v + 3]) / max_val);  }
        LEDS.show();}
```

Лістинг А.4 – Програмний код мікроконтролеру, файл «WIFI»

```
      void WIFIinit() {
        WiFi.mode(WIFI_STA);
        byte tries = 11;
        WiFi.begin(_ssid.c_str(), _password.c_str());
        while (--tries && WiFi.status() != WL_CONNECTED)  {
          delay(1000);  }
        if (WiFi.status() != WL_CONNECTED)  {
          StartAPMode();  } }

      bool StartAPMode(){
        WiFi.disconnect();
        WiFi.mode(WIFI_AP);
        WiFi.softAPConfig(apIP, apIP, IPAddress(255, 255, 255, 0));
        WiFi.softAP(_ssidAP.c_str(), _passwordAP.c_str());
        return true;}
```

Лістинг А.5 – Програмний код мікроконтролеру, файл «audio_preparation»

```
      void get_max_vol()
      {
        max_vol = 0;
        int x;
        for(uint8_t i = 0; i < 100; ++i)
        {
          x = analogRead(MIC_PIN);
```

```
            if(x > max_vol)
            {
             max_vol=x;
            }
          }
        }

        void get_fft_from_audio()
        {
          for (int i = 0; i < SAMPLES; i++)
          {
            vReal[i] = analogRead(A0);
            vImag[i] = 0;
          }
          FFT.Windowing(vReal, SAMPLES, FFT_WIN_TYP_HAMMING, FFT_FORWARD);
          FFT.Compute(vReal, vImag, SAMPLES, FFT_FORWARD);
          FFT.ComplexToMagnitude(vReal, vImag, SAMPLES);
        }
```

Лістинг А.6 – Програмний код Андроїд застосунку, файл «MainActivity.kt»

```
package com.example.myapplication

import android.annotation.SuppressLint
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.webkit.WebView
import android.webkit.WebViewClient
import android.widget.TextView
import java.io.BufferedInputStream
import java.net.HttpURLConnection
import java.net.URL

class MainActivity : AppCompatActivity() {

    @SuppressLint("SetJavaScriptEnabled", "SetTextI18n")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        findViewById<WebView>(R.id.webview).webViewClient = WebViewClient()

        findViewById<TextView>(R.id.ip).text = "192.168.4.1"
        findViewById<TextView>(R.id.brightness).text = "100"

        findViewById<WebView>(R.id.webview).visibility = View.GONE    }

    fun offClicked(view : View) {
        var url = "http://" + findViewById<TextView>(R.id.ip).text
        url += "/led?mode=0&brightness=" + findViewById<TextView>(R.id.brightness).text
        url += "&red=" + findViewById<TextView>(R.id.red).text
        url += "&green=" + findViewById<TextView>(R.id.green).text
        url += "&blue=" + findViewById<TextView>(R.id.blue).text
```

```
    findViewById<WebView>(R.id.webview).apply {
        loadUrl(url)        }    }

fun bgClicked(view : View)  {
    var url = "http://" + findViewById<TextView>(R.id.ip).text
    url += "/led?mode=1&brightness=" + findViewById<TextView>(R.id.brightness).text
    url += "&red=" + findViewById<TextView>(R.id.red).text
    url += "&green=" + findViewById<TextView>(R.id.green).text
    url += "&blue=" + findViewById<TextView>(R.id.blue).text

    findViewById<WebView>(R.id.webview).apply {
        loadUrl(url)        }    }

fun rainbowClicked(view : View)  {
    var url = "http://" + findViewById<TextView>(R.id.ip).text
    url += "/led?mode=2&brightness=" + findViewById<TextView>(R.id.brightness).text
    url += "&red=" + findViewById<TextView>(R.id.red).text
    url += "&green=" + findViewById<TextView>(R.id.green).text
    url += "&blue=" + findViewById<TextView>(R.id.blue).text

    findViewById<WebView>(R.id.webview).apply {
        loadUrl(url)        }    }

fun maxbClicked(view : View)  {
    var url = "http://" + findViewById<TextView>(R.id.ip).text
    url += "/led?mode=3&brightness=" + findViewById<TextView>(R.id.brightness).text
    url += "&red=" + findViewById<TextView>(R.id.red).text
    url += "&green=" + findViewById<TextView>(R.id.green).text
    url += "&blue=" + findViewById<TextView>(R.id.blue).text

    findViewById<WebView>(R.id.webview).apply {
        loadUrl(url)        }    }

fun maxnClicked(view : View)  {
    var url = "http://" + findViewById<TextView>(R.id.ip).text
    url += "/led?mode=4&brightness=" + findViewById<TextView>(R.id.brightness).text
    url += "&red=" + findViewById<TextView>(R.id.red).text
    url += "&green=" + findViewById<TextView>(R.id.green).text
    url += "&blue=" + findViewById<TextView>(R.id.blue).text

    findViewById<WebView>(R.id.webview).apply {
        loadUrl(url)        }    }

fun fClicked(view : View) {
    var url = "http://" + findViewById<TextView>(R.id.ip).text
    url += "/led?mode=5&brightness=" + findViewById<TextView>(R.id.brightness).text
    url += "&red=" + findViewById<TextView>(R.id.red).text
    url += "&green=" + findViewById<TextView>(R.id.green).text
    url += "&blue=" + findViewById<TextView>(R.id.blue).text

    findViewById<WebView>(R.id.webview).apply {
        loadUrl(url)        }    }}
```

Лістинг А.7 – Програмний код Андроїд застосунку, файл «AndroidManifest.xml»

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.myapplication">

    <uses-permission android:name="android.permission.INTERNET"/>

    <application
        android:usesCleartextTraffic="true"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MyApplication">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Лістинг А.8 – Програмний код Андроїд застосунку, файл «activity_main.xml»

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/ipTextView"
        android:layout_width="409dp"
        android:layout_height="30dp"
        android:layout_marginStart="1dp"
        android:layout_marginTop="1dp"
        android:layout_marginEnd="1dp"
        android:text="Change IP if needed"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```

```
<TextView
    android:id="@+id/brightnessTextView"
    android:layout_width="409dp"
    android:layout_height="28dp"
    android:layout_marginStart="1dp"
    android:layout_marginTop="1dp"
    android:layout_marginEnd="1dp"
    android:text="Enter brightness, form 0 to 255"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/ip" />

<Button
    android:id="@+id/button4"
    android:layout_width="135dp"
    android:layout_height="123dp"
    android:layout_marginStart="1dp"
    android:layout_marginTop="4dp"
    android:layout_marginEnd="1dp"
    android:onClick="offClicked"
    android:text="off"
    app:layout_constraintEnd_toStartOf="@+id/button5"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView" />

<Button
    android:id="@+id/button5"
    android:layout_width="136dp"
    android:layout_height="123dp"
    android:layout_marginStart="1dp"
    android:layout_marginTop="4dp"
    android:layout_marginEnd="1dp"
    android:onClick="bgClicked"
    android:text="br light"
    app:layout_constraintEnd_toStartOf="@+id/button6"
    app:layout_constraintStart_toEndOf="@+id/button4"
    app:layout_constraintTop_toBottomOf="@+id/textView" />

<Button
    android:id="@+id/button8"
    android:layout_width="136dp"
    android:layout_height="123dp"
    android:layout_marginStart="1dp"
    android:layout_marginTop="1dp"
    android:layout_marginEnd="1dp"
    android:onClick="maxnClicked"
    android:text="max number"
    app:layout_constraintEnd_toStartOf="@+id/button9"
    app:layout_constraintStart_toEndOf="@+id/button7"
    app:layout_constraintTop_toBottomOf="@+id/button5" />

<Button
    android:id="@+id/button6"
    android:layout_width="136dp"
    android:layout_height="123dp"
```

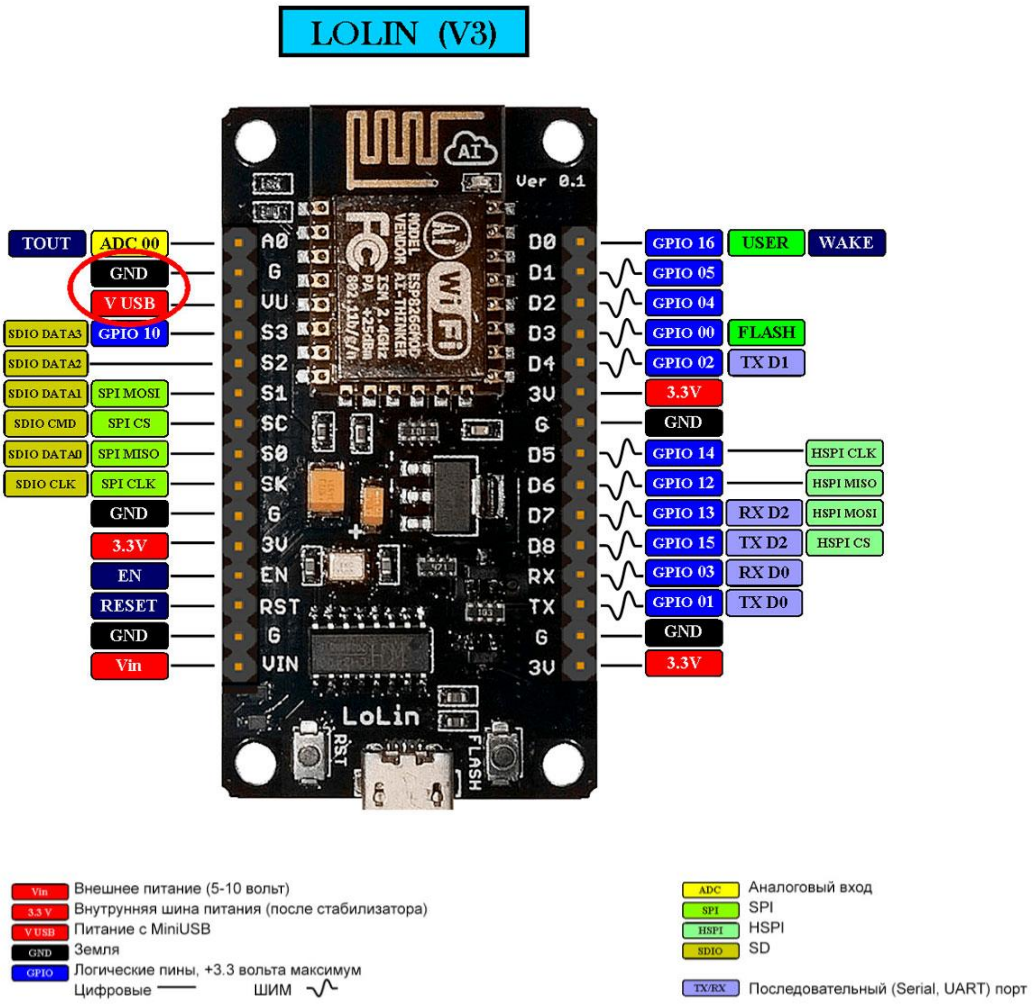
```
    android:layout_marginTop="4dp"
    android:layout_marginEnd="1dp"
    android:onClick="rainbowClicked"
    android:text="rainbow"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@+id/button5"
    app:layout_constraintTop_toBottomOf="@+id/textView" />

<TextView
    android:id="@+id/textView"
    android:layout_width="410dp"
    android:layout_height="29dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="1dp"
    android:text="Modes"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/green" />

<TextView
    android:id="@+id/textView6"
    android:layout_width="136dp"
    android:layout_height="66dp"
    android:layout_marginStart="1dp"
    android:layout_marginTop="22dp"
    android:layout_marginEnd="1dp"
    android:text="Green, from 0 to 255"
    app:layout_constraintEnd_toStartOf="@+id/textView7"
    app:layout_constraintStart_toEndOf="@+id/textView5"
    app:layout_constraintTop_toBottomOf="@+id/brightness" />

<EditText
    android:id="@+id/red"
    android:layout_width="135dp"
    android:layout_height="66dp"
    android:layout_marginStart="1dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="1dp"
    android:ems="10"
    android:inputType="number"
    app:layout_constraintEnd_toStartOf="@+id/green"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView5" />

<EditText
    android:id="@+id/blue"
    android:layout_width="136dp"
    android:layout_height="66dp"
    android:layout_marginStart="1dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="1dp"
    android:ems="10"
    android:inputType="number"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@+id/green"
    app:layout_constraintTop_toBottomOf="@+id/textView7" />
```

```
<TextView
    android:id="@+id/textView7"
    android:layout_width="136dp"
    android:layout_height="66dp"
    android:layout_marginStart="1dp"
    android:layout_marginTop="22dp"
    android:layout_marginEnd="1dp"
    android:text="Blue, from 0 to 255"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@+id/textView6"
    app:layout_constraintTop_toBottomOf="@+id/brightness" />

<EditText
    android:id="@+id/green"
    android:layout_width="136dp"
    android:layout_height="66dp"
    android:layout_marginStart="1dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="1dp"
    android:ems="10"
    android:inputType="number"
    app:layout_constraintEnd_toStartOf="@+id/blue"
    app:layout_constraintStart_toEndOf="@+id/red"
    app:layout_constraintTop_toBottomOf="@+id/textView6" />

<TextView
    android:id="@+id/textView5"
    android:layout_width="135dp"
    android:layout_height="66dp"
    android:layout_marginStart="1dp"
    android:layout_marginTop="22dp"
    android:layout_marginEnd="1dp"
    android:text="Red, from 0 to 255"
    app:layout_constraintEnd_toStartOf="@+id/textView6"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/brightness" />

<Button
    android:id="@+id/button7"
    android:layout_width="135dp"
    android:layout_height="123dp"
    android:layout_marginStart="1dp"
    android:layout_marginTop="1dp"
    android:layout_marginEnd="1dp"
    android:onClick="maxbClicked"
    android:text="max bright"
    app:layout_constraintEnd_toStartOf="@+id/button8"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/button4" />

<Button
    android:id="@+id/button9"
    android:layout_width="136dp"
    android:layout_height="123dp"
    android:layout_marginStart="1dp"
    android:layout_marginTop="1dp"
```

```
        android:layout_marginEnd="1dp"
        android:onClick="fClicked"
        android:text="frequency"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toEndOf="@+id/button8"
        app:layout_constraintTop_toBottomOf="@+id/button6" />

    <EditText
        android:id="@+id/ip"
        android:layout_width="409dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="1dp"
        android:layout_marginTop="1dp"
        android:layout_marginEnd="1dp"
        android:ems="10"
        android:inputType="textPersonName"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/ipTextView" />

    <EditText
        android:id="@+id/brightness"
        android:layout_width="409dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="1dp"
        android:layout_marginTop="1dp"
        android:layout_marginEnd="1dp"
        android:ems="10"
        android:inputType="number"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/brightnessTextView" />

    <WebView
        android:id="@+id/webview"
        android:layout_width="406dp"
        android:layout_height="130dp"
        android:layout_marginStart="1dp"
        android:layout_marginTop="1dp"
        android:layout_marginEnd="1dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/button7" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

ДОДАТОК Б

Зовнішній вигляд та опис пінів NodeMCU V3



ДОДАТОК В

Графічний матеріал кваліфікаційної роботи

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ
кафедраАПОТ

Кваліфікаційна робота бакалавра

Бездротова система світломузики з керуванням Андроїд застосунком

Виконав:
Студент гр. КІУКІ-17-7 Малишев М.О.
Керівник:
к.т.н., доцент каф. АПОТ Ларченко Л.В.

Харків 2021 р.

Актуальність проекту

В даний час **світлодіоди** та **світлодіодні системи** завдяки інноваційним розробкам стали дійсно універсальним рішенням і застосовуються як індикатори в побутовій техніці, або самостійно у вигляді енергозберігаючого освітлювального пристрою, в космічній галузі, а також в сфері спецефектів. До останньої можна віднести і **світломузику**, яка робить наше життя набагато цікавіше.

Сучасні системи світломузики володіють повним набором функцій для забезпечення різних світлових ефектів, що реагують на звукові хвилі, в яких звук і світло стають єдиним цілим. Конструктивні особливості, а також функціонал систем світломузики залежать від конкретного обладнання .

Сьогодні актуальним напрямком розвитку систем спецефектів стають **бездротові системи світломузики з віддаленим керуванням** за допомогою мобільних пристроїв. При використанні бездротових технологій покращується зручність керування та зовнішній вигляд систем світломузики в цілому.

Мета і постановка завдання

Метою кваліфікаційної роботи є розробка бездротової системи світломузики з керуванням Андроїд застосунком на основі мікроконтролера та адресної світлодіодної стрічки.

Об'єктом розробки є бездротова програмно-апаратна мікроконтролерна система світломузики.

Постановка завдання

Поставлена мета визначила наступні завдання проектування:

- аналіз та вибір мікроконтролеру і компонентів, що входять складовими до бездротової системи світломузики;
- розробка структурної схеми проекту;
- розробка функціональної схеми взаємодії компонентів бездротової системи світломузики;
- створення прототипу бездротової системи для перетворення звукового сигналу в світлове миготіння, з функцією підсвічення та керуванням Андроїд застосунком;
- аналіз, вибір мов програмування та середовищ розробки клієнтської та серверної частин проекту;
- створення скетчу мовою C++ в Arduino IDE;
- створення програмного Android застосунку і налаштування Kotlin проекту;
- тестування спроектованої системи світломузики.

Апаратна база проекту

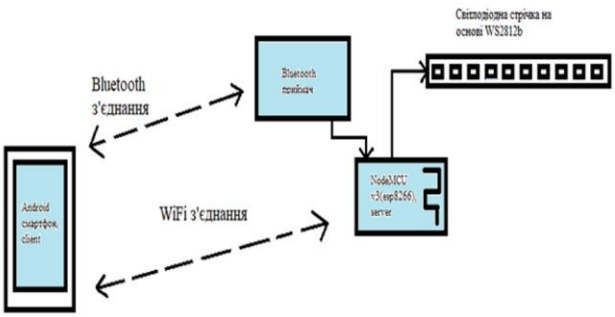
NodeMCU V3 – плата для розробки на базі чіпу ESP8266(ESP12E), який представляє собою UART-WiFi модуль.

Адресна світлодіодна стрічка на основі **WS2812b** – світлодіодна стрічка у якої є можливість керування кольором та яскравістю кожного світлодіоду

Bluetooth модуль VHM-314-V2.0 – аудіо Bluetooth -приймач з аналоговим AUX виходом.

Смарфон OnePus 6t з операційною системою Андроїд.

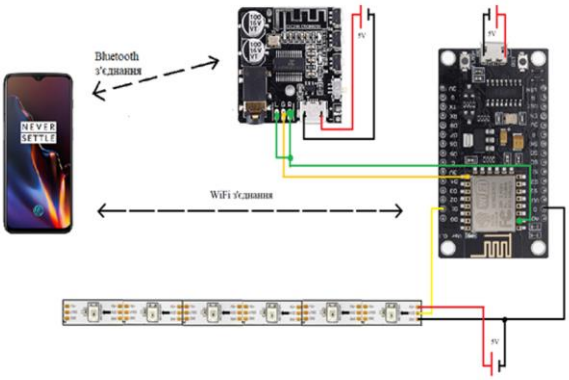
Структурна схема проекту



- Смартфон відправляє дані, які використовуються для керування пристроєм, мікроконтролеру за допомогою радіочастот.
- Також смартфон відправляє аудіо сигнал в цифровому вигляді Bluetooth приймачу за допомогою радіочастот.
- Після того, як Bluetooth приймач отримав аудіо сигнал, він відправляє сигнал мікроконтролеру.
- Мікроконтролер в залежності від отриманих даних зі смартфону та Bluetooth передавача керує адресною світлодіодною стрічкою.

Функціональна схема проекту

- На даній схемі смартфон слугує для одночасного керування світлодіодною стрічкою за допомогою технології WiFi та протоколу HTTP та відправлення аудіо сигналу за допомогою Bluetooth технології.
- Bluetooth приймач отримує цифровий аудіо сигнал, перетворює його в аналоговий сигнал за допомогою цифро-аналогового перетворювача.
- Мікроконтролер працює у режимі веб-серверу та отримує HTTP пакети відправлені з смартфону в залежності від отриманих даних, у яких описано режим, яскравість та колір, далі зчитує дані з Bluetooth приймача, якщо потрібно, обробляє їх та перетворює дані для виводу на світлодіодній стрічці.



Програмно-апаратна платформа мікроконтролера

- Було обрано плату NodeMCU V3 на базі мікроконтролера ESP8266 через можливість використання бездротового зв'язку та низьку ціну.

- Також було обрано середовище розробки Arduino IDE через простий інтерфейс та можливість програмування для багатьох мікроконтролерів.

7

Програмно-апаратна платформа Андроїд

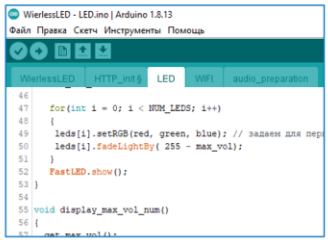
- Вибір операційної системи для смартфонів був здійснений на користь Андроїд, так як Андроїд це найпопулярніша операційна система для смартфонів, для якої досить просто розповсюджувати програмне забезпечення.

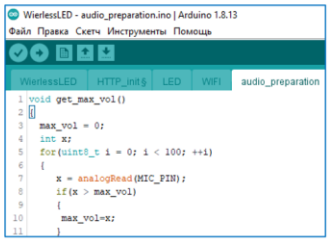
- Для написання програми для керування приладом було обрано мову програмування Kotlin, так як вона розроблялася виключно для Андроїд пристроїв, також вона досить легка для освоєння.

8

Короткий опис програмної частини мікроконтролера

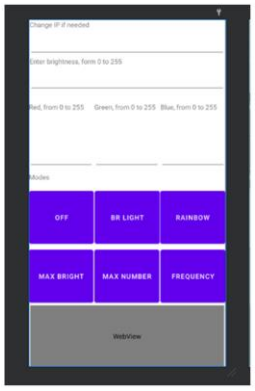
- У файлі WierlessLED прописані першочергові налаштування, ініціалізація деяких змінних та основний цикл.
- В HTTP_init описані усі налаштування для HTTP серверу.
- LED у цьому файлі зберігаються функції для ініціалізації світлодіодної стрічки та всі режими роботи.
- Файл WIFI слугує для ініціалізації роботи технології WIFI та для вибору режиму роботи: точка доступу або підключення до маршрутизатору.
- audio_preparation - файл у якому міститься дві невеликі функції обробки аудіо сигналу.

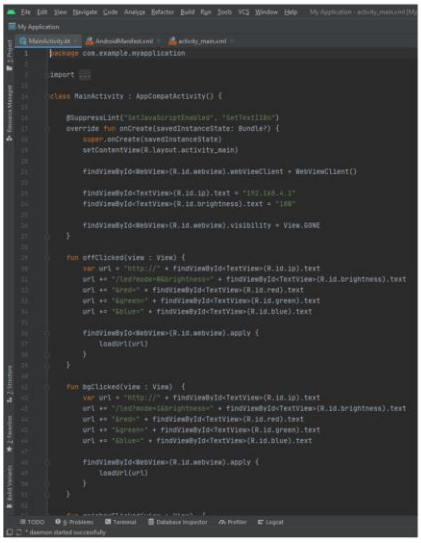




Короткий опис програмної частини смартфона

В Андроїд застосунку обробляється натискання на клавіші, отримуються дані з полів введення, формуються URL адреси, в залежності від режиму та налаштувань світлодіодної стрічки, та відправляються HTTP GET протоколи.





Опис режимів роботи прототипу безпровідної системи світломузики

У прототипу безпровідної системи світломузики є **п’ять режимів** роботи:

- **Підсвічення**

Для цього режиму в Андроїд застосунку можна обрати один з 16 777 216-ти кольорів та обрати яскравість від 0 до 255.

- **Райдуга**

Для режиму «райдуга» в Андроїд застосунку можна обрати лише яскравість. Цей режим відтворює кольори райдуги які повторюються.

- **Режим відображення гучності через кількість працюючих світлодіодів від центру**

Для цього режиму в Андроїд застосунку можна обрати один з 16 777 216-ти кольорів яким буде відображена гучність звукового сигналу, також є можливість змінювати яскравість від 0 до 255.

- **Режим відображення гучності яскравістю світлодіодів**

Для цього режиму в Андроїд застосунку можна обрати один з 16 777 216-ти кольорів яким буде відображена гучність звукового сигналу, через те що гучність відображається яскравістю світлодіодів неможливо змінювати яскравість.

- **Режим відображення гучності частоти**

Можливо змінювати колір, кожний світлодіод своєю яскравістю відображає гучність частоти чутного звуку.

11

Демонстрація роботи приладу

12

Висновки

- Під час виконання кваліфікаційної роботи було розроблено **бездротову систему світломузики з керуванням Андроїд застосунком** на основі мікроконтролера, адресної світлодіодної стрічки та смартфону з операційною системою Андроїд, де мікроконтролер виконує серверну частину, а смартфон є клієнтом.
- У ході виконання роботи було здійснено **вибір компонентів приладу** з необхідними технічними характеристиками та функціями, створено **структурну** та **функціональну схеми**, створено **програмний код мікроконтролерної частини та частини Андроїд застосунку**.
- **Перевагами** розробленого **пристрою** є зручність управління, простота використання, гнучкість налаштування та невеликі габарити.
- **Даний прототип системи світломузики** може бути використаний для покращення настрою та для отримання естетичного задоволення різних вікових та соціальних груп. Також цей пристрій можливо покращити шляхом вдосконалення користувацького інтерфейсу на мобільних пристроях, створення можливості керуванням частотою оновлення світлодіодної стрічки та додавання нових режимів керуванням світлодіодною стрічкою.