МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Кафедра Комп'ютерних систем та мереж

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
комп'ютерних систем та
мереж

_____(Жуков І.А.)

« ____» _____2022 р.

ДИПЛОМНИЙ ПРОЕКТ

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ"МАГІСТР"

Тема: Методи та засоби для функційного програмування мікроконтролерів

Виконавець: Багрійчук О.В.

Керівник: Антонов В.К.

Нормоконтролер: Андреєв О.В.

Київ 2022

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп'ютерної та програмної інженерії
Кафедра комп'ютерних систем та мереж
Напрям (спеціальність) 123 "Комп'ютерна інженерія"
(шифр, найменування)

ЗАТВЕРДЖУЮ
Завідувач кафедри
комп'ютерних систем та
мереж

_____(Жуков І.А.)

« ___» _____2022 р.

.

З А В Д А Н Н Я
на виконання дипломного проекту
Багрійчуку Олександру Васильовичу
(прізвище, ім'я, по батькові випускника в родовому відмінку)

1. Тема проекту (роботи): Методи та засоби для функційного програмування мікроконтролерів

затверджена наказом ректора від " 06 " вересня 2022 року № 1266/ст.

2. Термін виконання проекту (роботи): з 05 вересня 2022 року по 30 лістопада 2022 року

3.Вихідні дані до проекту (роботи): вимоги до змісту методів та засобів для функційного програмування мікроконтролерів, дослідження основних парадигм програмування

4. Зміст пояснювальної записки (перелік питань, що підлягають розробці):
Літературний огляд використання мікроконтролерів, методи та засоби вирішення, задачі з проектування системи керування звуковідтворюючими пристроями, практична реалізація системи керування звуковідтворюючими пристроями на базі мікроконтролера ESP-8266/ESP32

5. Перелік обов'язкового графічного матеріалу:
Презентація *PowerPoint*

6. Календарний план

№ п/п	Етапи виконання дипломного проекту	Термін виконання етапів	Примітка
1	Вивчити довідникову літературу	05.09.2022-10.09.2022	
2	Проаналізувати основні компоненти мікроконтролеру та його можливості	10.09.2022-16.09.2022	
3	Побудувати алгоритм обробки інформації для технології дистанційного керування ЗВП	16.09.2022-23.09.2022	
4	Розробити структурну схему та архітектуру прототипу	23.09.2022-30.09.2022	
5	Сформувати загальні вимоги	30.09.2022-03.10.2022	
6	Розробити принципову схему	03.10.2022-20.10.2022	
7	Розробити програмну частину	20.10.2022-04.11.2022	
8	Оформити пояснювальну записку	04.11.2022-08.11.2022	
9	Підготувати графічний демонстраційний матеріал	08.11.2022-15.11.2022	
10	Захистити дипломну роботу	25.11.2022-26.11.2022	

7. Консультанти з окремих розділів

Розділ	Консультатнт (посада, ПІБ)	Дата, підпис	
		Завдання видав	Завдання прийняв

8. Дата отримання завдання «05» вересня 2022 р.

Керівник дипломного проекту Антонов В.К.
(підпис)

Завдання прийняв до виконання Багрійчук О.В.
(підпис студента)

РЕФЕРАТ

Пояснювальна записка до дипломного проекту «Методи та засоби для функційного програмування мікроконтролерів», 83 сторінки, 26 рисунків, 35 джерел літератури.

ПРОГРАМУВАННЯ, ДИСТАНЦІЙНЕ КЕРУВАННЯ, АВТОМАТИЗОВАНА СИСТЕМА, МІКРОКОНТРОЛЕР, ПРОГРАМНО-АПАРАТНИЙ КОМПЛЕКС, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ШТУЧНИЙ ІНТЕЛЕКТ.

Мета дипломного проекту – розробити та застосувати ефективний алгоритм обробки інформації для технології дистанційного керування ЗВП.

Об'єкт дослідження – процес програмування мікроконтролеру для реалізації системи керування звуковідтворюючими пристроями.

Предмет дослідження – особливості розробки та впровадження системи дистанційного керування на базі ESP-8266/ESP32.

Наукова новизна одержаних результатів: результати дослідження пропонують альтернативний метод використання засобів системного програмування в процесі розробки алгоритму обробки інформації систем дистанційного керування на базі мікроконтролерів.

Практична значимість дослідження – полягає в аналізі особливостей створення та реалізації ефективного алгоритму обробки інформації для технології дистанційного керування ЗВП на базі ESP-8266/ESP32, а також розробці практичних рекомендацій щодо підключення та експлуатації пристрою.

Методи дослідження: методи системного аналізу, аналіз наукової літератури, спостереження, абстрагування, узагальнення, математичного моделювання.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

АС – автоматизована система;

БД – база даних;

ЗВП – звуковідтворюючі пристрої;

ІБД – інтегрована база даних;

ІІТ – інтелектуальні інформаційні технології;

НМ – нейронна мережа;

МК – мікроконтролер;

ПАК – програмно-апаратний комплекс;

СДК – система дистанційного керування;

СУБД – система управління базою даних;

ФП – функційне програмування;

ШНМ – штучні нейронні мережі.

ВСТУП

Актуальність дослідження. На сьогоднішній стадії розвитку соціуму комп'ютерні технології є невід'ємним атрибутом швидкого зростання актуальності галузей науки, які пов'язані з математичним проектуванням явищ та процесів.

Наразі важливе місце в автоматизаційних процесах належить системам дистанційного керування (СДК), які дозволяють за допомогою засобів штучного інтелекту здійснювати контроль за різноманітними пристроями. Такими засобами програмування є мікроконтролери (МК), з яких складається велика кількість сучасної побутової та виробничо-промислової електронної техніки. Звичайний мікроконтролер представляє собою мікросхему для керування електронними пристроями будь-якої архітектури та складності виконання.

За своєю архітектурою МК - це однокристальний цифровий обчислювальний пристрій, що може мати на одному кристалі функціонал процесора – обчислювального модуля, різних допоміжних переферійних пристроїв, наприклад, таких як універсальні цифрові порти , вводу-виводу інформації, різні інтерфейси комунікації такі як UART, USB, Ethernet , різні запам'ятовуючі пристрої. Слід зауважити, що такого набору компонентів цілком достатньо для виконання задач простого – середнього рівня складності.

Отже, завдяки використанню сучасних МК користувачеві доступні функції дистанційного керування різною виконавчою технікою через використання інтерфейсів керування таких як: радіоканали, Wi-fi , Bluetooth та ZigBee. В цьому випадку виконавчими пристроями можуть виступати:

– освітлювальні пристрої;

– системи сигналізації та відеоконтролю, в тому числі різноманітні звуковідтворюючі пристрої (ЗВП);

– підйомні механізми або ворота;

– електронні комплексні системи безпеки;

– будь які пристрої що мають інтерфейс дистанційного зв’язку та знаходяться в важкодоступних місцях з відсутністю можливості організації дротового каналу зв’язку.

Слід зазначити, шо останнім часом все популярнішою стає функційна парадигма програмування МК. Сучасні найбільш популярні об’єктно-орієнтовані мови програмування для повноцінних комп’ютерів, такі як C# та Java, уже підтримують багато корисних особливостей функційного програмування.

Виходячи з вищенаведеного, наше дослідження особливостей програмування мікроконтролеру для реалізації системи керування звуковідтворюючими пристроями є актуальним.

Мета та завдання дослідження.

Мета дослідження - розробити та застосувати ефективний алгоритм обробки інформації для технології дистанційного керування ЗВП.

Завдання:

1. розглянути теоретичні засади використання мікроконтролерів;
2. визначити основні компоненти мікроконтролеру та його можливості;
3. проаналізувати методи та засоби проектування СДК за допомогою МК ;

4. представити реалізацію розробленої СДК для ЗВП на базі ESP-8266/ESP32 із застосуванням функційної парадигми програмування.

Об’єкт дослідження – процес програмування мікроконтролеру для реалізації системи керування звуковідтворюючими пристроями.

Предмет дослідження – особливості розробки та впровадження системи дистанційного керування на базі ESP-8266/ESP32.

Методи дослідження: методи системного аналізу, аналіз наукової літератури, спостереження, абстрагування, узагальнення, математичного моделювання.

Теоретично - інформаційна база дослідження представлена науковими роботами таких дослідників, як С. Ардатський, О. Басманов, А. Бодрова, А. Вуд, А. Дейч, О. Голощапов, Б. Джошуа, В. Квашнін, Г. Поспєлов, Г. Смірнова, В. Соколов, В. Сташин, Дж. Уокерлі, П. Хоровіц, М. Щєлкунов та інших.

Практичне значення одержаних результатів полягає в дослідженні особливостей створення та реалізації ефективного алгоритму обробки інформації для технології дистанційного керування ЗВП на базі ESP-8266/ESP32, а також розробці практичних рекомендацій щодо підключення та експлуатації пристрою.

Апробація результатів дослідження. Результати досліджень були апробовані на засіданні кафедри комп'ютерної інженерії Національного авіаційного університету.

Структура роботи. Дипломна робота складається зі вступу, 3 розділів, висновків, списку використаних джерел, додатків. Зміст роботи викладено на 83 сторінках. Перелік джерел посилання складається із 35 найменувань.

РОЗДІЛ 1.
ЛІТЕРАТУРНИЙ ОГЛЯД ВИКОРИСТАННЯ МІКРОКОНТРОЛЕРІВ

1.1. Загальна характеристика об’єкту дослідження

Мікроконтролер (МК) являє собою невеликий комп’ютер на одному чіпі інтегральної схеми (ІС) НВІС метал-оксид-напівпровідник (МОП). Мікроконтролер містить один або кілька процесорів (процесорних ядер), а також пам’ять та програмовані периферійні пристрої вводу/виводу.

Програмна пам’ять у вигляді фероелектричної ОЗУ, флеш-пам’яті NOR або OTP ROM також часто включається до мікросхеми, а також невеликий обсяг ОЗУ.

Мікроконтролери призначені для додатків, що вбудовуються, на відміну від мікропроцесорів, що використовуються в персональних комп’ютерах або інших додатках загального призначення, що складаються з різних дискретних мікросхем.

У сучасній термінології мікроконтролер нагадує систему на кристалі (SoC) , але менш складний. SoC може підключати зовнішні мікросхеми мікроконтролера як компоненти материнської плати, але SoC зазвичай поєднує передові периферійні пристрої, такі як графічний процесор (GPU) і контролер інтерфейсу Wi-Fi, як внутрішні схеми блоку мікроконтролера [7].

Мікроконтролери використовуються в продуктах і пристроях з автоматичним керуванням, таких як системи керування автомобільними двигунами, медичні пристрої, що імплантуються, пульти дистанційного керування, офісні машини, побутова техніка, електроінструменти, іграшки та інші вбудовувані системи.

Завдяки зменшенню розміру та вартості в порівнянні з конструкцією, в якій використовується окремий мікропроцесор, пам’ять та пристрої введення/виводу, мікроконтролери роблять цифрове управління ще більшою кількістю пристроїв та процесів економічним.

Широко поширені мікроконтролери зі змішаними сигналами, що поєднують аналогові компоненти, необхідні управління нецифровими електронними системами.

Принцип дії мікроконтролерів.

Мікроконтролер вбудований у систему для керування окремою функцією пристрою. Він робить це, інтерпретуючи дані, які він отримує від своїх периферійних пристроїв введення-виведення, використовуючи свій центральний процесор. Тимчасова інформація, яку отримує мікроконтролер, зберігається в пам'яті даних, де процесор отримує до неї доступ і використовує інструкції, що зберігаються в його пам'яті програм, для розшифровки та застосування вхідних даних. Потім він використовує свої периферійні пристрої введення-виведення для зв'язку та виконання відповідних дій [7].

Мікроконтролери використовуються в різних системах і пристроях. Пристрої часто використовують кілька мікроконтролерів, які працюють разом усередині пристрою для виконання відповідних завдань.

Наприклад, в автомобілі може бути багато мікроконтролерів, які керують різними окремими системами всередині, такими як антиблокувальна гальмівна система, контроль тяги, упорскування палива або керування підвіскою.

Усі мікроконтролери взаємодіють один з одним для інформування про правильні дії. Деякі можуть зв'язуватися з більш складним центральним комп'ютером в автомобілі, інші можуть зв'язуватися тільки з іншими мікроконтролерами. Вони відправляють і одержують дані, використовуючи свої периферійні пристрої вводу-виводу, та обробляють ці дані для виконання призначених їм завдань.

У контексті Інтернету речей мікроконтролери є економічним і популярним засобом збору даних. Сприйняття та приведення в дію фізичного світу як прикордонних пристроїв (рис. 1.1).

Рис. 1.1 – МК PIC 18F8720
у 80-контактному корпусі TQFP

Деякі мікроконтролери можуть використовувати чотирибітні слова та працювати на частотах до 4 кГц для низького енергоспоживання (однорозрядні мілліват або мікроват).

Як правило, вони можуть зберігати функціональність в очікуванні події, такого як натискання кнопки або інше переривання; енергоспоживання під час сну (тактова частота процесора і більшість периферійних пристроїв вимкнені) може становити всього нановати, що робить багато з них добре підходящими для програм з тривалим терміном служби батареї.

Інші мікроконтролери можуть виконувати ролі, критично важливі для продуктивності, де їм може знадобитися діяти як процесор цифрових сигналів (DSP) з вищими тактовими частотами та енергоспоживанням (рис. 1.2).

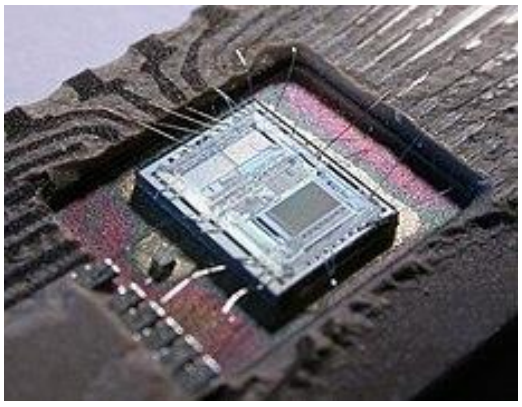

Рис. 1.2 - Кристал від Intel 8742, 8-розрядного МК, який включає ЦП, що працює на частоті 12 МГц, 128 байт ОЗУ, 2048 байт СППЗУ і введення - виведення в одному чіпі

Мікроконтролер можна розглядати як автономну систему з процесором, пам'яттю та периферійними пристроями та використовувати як вбудовану систему [26].

Більшість мікроконтролерів, що використовуються сьогодні, вбудовані в інше обладнання, таке як автомобілі, телефони, побутові прилади та периферійні пристрої для комп'ютерних систем.

Хоча деякі вбудовані системи дуже складні, багато з них мають мінімальні вимоги до пам'яті та довжини програми, не мають операційної системи та мають низьку складність програмного забезпечення (рис. 1.3).

Типові пристрої введення та виведення включають перемикачі, реле, соленоїди, світлодіоди, невеликі або спеціальні рідкокристалічні дисплеї, радіочастотні пристрої та датчики для таких даних, як температура, вологість, рівень освітленості.

Вбудовані системи зазвичай не мають клавіатури, екрану, диски, принтери або інші пристрої введення-виводу ПК, що розпізнаються, і можуть не мати жодних пристроїв взаємодії з людиною (рис. 1.4).

Рис. 1.3 - Два МК ATmega

Типові пристрої введення та виведення включають перемикачі, реле, соленоїди, світлодіоди, невеликі або спеціальні рідкокристалічні дисплеї, радіочастотні пристрої та датчики для таких даних, як температура, вологість, рівень освітленості [29].

Рис. 1.4 - МКPIC 18F8720
у 80-контактному корпусі TQFP

Вбудовані системи зазвичай не мають клавіатури, екрану, диски, принтери або інші пристрої введення-виводу персонального комп'ютера, що розпізнаються, і можуть не мати жодних пристроїв взаємодії з людиною.

Мікроконтролери повинні забезпечувати реагування в реальному часі (передбачуване, хоч і не обов'язково швидке) на події у вбудованій системі, якою вони керують.

Коли відбуваються певні події, система переривань може сигналізувати процесору про припинення обробки поточної послідовності команд та запуску процедури обслуговування переривання. (ISR або «обробник переривань»), який виконає будь-яку обробку, необхідну залежно від джерела переривання, перш ніж повернутися до вихідної послідовності інструкцій (рис. 1.5).

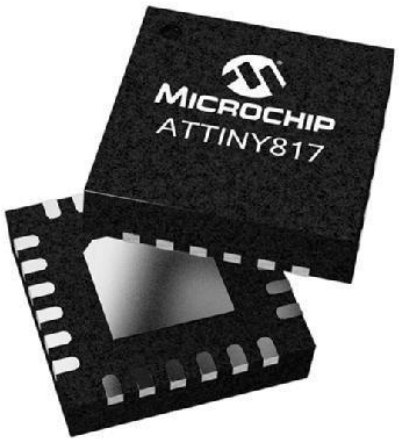

Рис. 1.5 - МК Microchip Technology ATtiny817

Можливі джерела переривання залежать від пристрою і часто включають такі події, як переповнення внутрішнього таймера, завершення аналого-цифрового перетворення, зміна логічного рівня на вході, наприклад при натисканні кнопки, і дані, отримані по каналу зв’язку.

Там, де енергоспоживання важливе, як у пристроях з батареями, переривання також можуть вивести мікроконтролер зі сну з низьким енергоспоживанням, коли процесор зупиняється до тих пір, поки не потрібно щось зробити через периферійну подію [29].

Зазвичай програми мікроконтролера повинні поміщатися в доступну вбудовану пам’ять, оскільки було б дорого постачати систему зовнішньої пам’яті, що розширюється. Компілятори та асемблери використовуються для перетворення високорівневого коду та коду мовою асемблера на компактний машинний код для зберігання в пам’яті мікроконтролера. Залежно від пристрою програмна пам’ять може бути постійною пам’яттю тільки для читання, яка може бути запрограмована тільки на заводі, або це може бути змінена на місці флеш-пам’ять або пам’ять, що стирається, тільки для читання.

Виробники часто випускають спеціальні версії своїх мікроконтролерів, щоб допомогти в розробці апаратного та програмного забезпечення цільової системи. Спочатку вони включали версії EPROM , які мають вікно у верхній частині пристрою,

через яке пам’ять програм може бути стерта ультрафіолетовим світлом, готові до перепрограмування після програмування (пропалювання) і циклу випробувань. З 1998 року версії EPROM зустрічаються рідко та були замінені EEPROM та флеш-пам’яттю, які простіше у використанні (можна стерти електронним способом) та дешевшими у виробництві [30].

Можуть бути доступні інші версії, в яких доступ до ПЗУ здійснюється як до зовнішнього пристрою, а не як до внутрішньої пам’яті, однак вони стають рідкістю через широку доступність дешевих програматорів для мікроконтролерів.

Використання програмованих пристроїв на мікроконтроллері може дозволити оновлення вбудованого програмного забезпечення на місці або дозволити пізні заводські модифікації продуктів, які були зібрані, але ще не відправлені. Програмована пам’ять також скорочує час, необхідний розгортання нового продукту.

Там, де потрібні сотні тисяч однакових пристроїв, використання деталей, запрограмованих під час виробництва, може бути економічним.

Ці «запрограмовані маскою» частини мають програму, закладену так само, як і логіка чіпа, водночас.

Індивідуальний мікроконтролер включає блок цифрової логіки, який можна персоналізувати для додаткових можливостей обробки, периферійних пристроїв і інтерфейсів, адаптованих до вимог програми. Одним із прикладів є AT91CAP від Atmel [19].

Мікроконтролери зазвичай містять від декількох до десятків контактів введення/виводу загального призначення (GPIO). Висновки GPIO програмно налаштовуються на вхід, або на вихід. Коли контакти GPIO налаштовані на вхідний стан, вони часто використовуються для читання датчиків або зовнішніх сигналів. Налаштовані на вихідний стан контакти GPIO можуть керувати зовнішніми пристроями, такими як світлодіоди або двигуни, часто побічно через зовнішню силову електроніку.

Багатьом системам, що вбудовуються, необхідно зчитувати показання датчиків, що видають аналогові сигнали. Це призначення аналого-цифрового перетворювача (АЦП).

Оскільки процесори призначені для інтерпретації та обробки цифрових даних, тобто одиниць та нулів, вони не можуть нічого робити з аналоговими сигналами, які можуть бути надіслані йому пристроєм. Таким чином, аналого-цифровий перетворювач використовується для перетворення даних, що надходять у форму, яку може розпізнати процесор.

Менш поширеною функцією деяких мікроконтролерів є цифроаналоговий перетворювач (ЦАП), який дозволяє процесору виводити аналогові сигнали або рівні напруги.

На додаток до перетворювачів багато вбудованих мікропроцесорів також включають різні таймери. Одним із найбільш поширених типів таймерів є програмований інтервальний таймер (PIT). PIT може вести зворотний відлік від деякого значення до нуля або до ємності регістра лічильника, переповнюючись до нуля. Як тільки він досягає нуля, він відправляє переривання процесору, що вказує, що закінчив підрахунок [22].

Це корисно для таких пристроїв, як термостати, які періодично перевіряють температуру навколо себе, щоб побачити, чи потрібно включати кондиціонер, нагрівач.

Спеціальний блок широтно-імпульсної модуляції (ШІМ) дозволяє ЦП управляти силовими перетворювачами, резистивними навантаженнями, двигунами тощо без використання великої кількості ресурсів ЦП у вузьких циклах таймера.

Універсальний блок асинхронного приймача/передавача (UART) дозволяє приймати та передавати дані по послідовній лінії з дуже невеликим навантаженням на ЦП. Виділене апаратне забезпечення на кристалі також часто включає можливості для зв’язку з іншими пристроями (мікросхемами) у цифрових форматах, таких як Inter-Integrated Circuit (I^2C), послідовний периферійний інтерфейс (SPI), універсальна послідовна шина (USB) та Ethernet.

Інтеграція пам’яті та інших периферійних пристроїв в один чіп та тестування їх як єдиного цілого збільшує вартість цього чіпа, але часто призводить до зниження собівартості вбудованої системи загалом.

Навіть якщо вартість ЦП із вбудованими периферійними пристроями трохи перевищує вартість ЦП та зовнішніх периферійних пристроїв, менша кількість мікросхем зазвичай дозволяє використовувати меншу та дешевшу друковану плату, а також знижує трудомісткі витрати, необхідні для складання та тестування друкованої плати. крім тенденції до зниження рівня шлюбу для готової збірки.

МК є єдиною інтегральною схемою, зазвичай з наступними характеристиками [27]:

– центральний процесор - від невеликих і простих 4-бітних процесорів до складних 32-бітних або 64-бітних процесорів

– енергозалежна пам’ять (RAM) для зберігання даних

– ROM, EPROM, EEPROM або флеш-пам’ять для зберігання програм та робочих параметрів

– дискретні вхідні та вихідні біти, що дозволяють контролювати або виявляти логічний стан окремого виведення корпусу

– послідовне введення/виведення , таке як послідовні порти (UART)

– інші інтерфейси послідовного зв’язку, такі як I²C, послідовний периферійний інтерфейс та мережа контролерів для міжсистемного з’єднання

– периферійні пристрої, такі як таймери, лічильники подій, генератори ШІМ та сторожовий таймер

– тактовий генератор - часто генератор для кварцового синхронізуючого кристала, резонатора або RC-ланцюга

– багато включають аналого-цифрові перетворювачі, деякі включають цифро-аналогові перетворювачі

– внутрішньосхемне програмування та підтримка внутрішньосхемного налагодження.

Ця інтеграція різко зменшує кількість мікросхем, а також обсяг проводки та місця на друкованій платі, які потрібні для створення еквівалентних систем з використанням окремих мікросхем.

Крім того, на пристроях з невеликою кількістю контактів кожен контакт може взаємодіяти з декількома внутрішніми периферійними пристроями, при цьому функція контакту вибирається програмним забезпеченням. Це дозволяє використовувати деталь у ширшому спектрі додатків, ніж якби контакти мали спеціальні функції.

Архітектури мікроконтролерів дуже різняться. Деякі конструкції включають ядра мікропроцесора загального призначення з однією або декількома функціями ПЗП, ОЗП або вводу-виводу, інтегрованими в корпус [4].

Інші конструкції спеціально створені для програм управління.

Набір інструкцій мікроконтролера зазвичай містить безліч інструкцій, призначених для маніпулювання бітами (побітових операцій), щоб зробити керуючі програми компактнішими.

Наприклад, процесору загального призначення може знадобитися кілька інструкцій для перевірки біта в регістрі та переходу, якщо біт встановлений, тоді як мікроконтролер може бути одна інструкція для виконання цієї звичайно необхідної функції.

Мікроконтролери зазвичай не мають математичного співпроцесора, тому арифметичні операції з плаваючою комою виконуються програмним забезпеченням. Однак деякі останні розробки включають оптимізовані функції FPU і DSP. Прикладом може бути лінійка PIC32 MIPS від Microchip.

Середовище програмування.

Спочатку мікроконтролери програмувалися тільки мовою асемблера, але різні мови програмування високого рівня, такі як C, Python і JavaScript, тепер також широко використовуються для цільових мікроконтролерів та вбудованих систем.

Компілятори для мов загального призначення зазвичай мають деякі обмеження, а також покращення для кращої підтримки унікальних характеристик мікроконтролерів. Деякі мікроконтролери мають середовище, що допомагає розробляти певні типи

програм. Постачальники мікроконтролерів часто надають безкоштовні інструменти, щоб спростити використання їх устаткування.

Для мікроконтролерів зі спеціальним обладнанням можуть знадобитися власні нестандартні діалекти C, такі як SDCC для 8051 , які не дозволяють використовувати стандартні інструменти (такі як бібліотеки коду або статичного аналізу) навіть для коду, не пов'язаного з апаратними функціями. Інтерпретатори також можуть містити нестандартні функції, такі як MicroPython, хоча форк CircuitPython прагнув перемістити апаратні залежності в бібліотеки та змусити мову відповідати більшому стандарту C Python [35].

Для деяких мікроконтролерів також є прошивка інтерпретатора. Наприклад, BASIC на ранніх мікроконтролерах Intel 8052; BASIC і FORTH на Zilog Z8, а також на деяких сучасних пристроях. Зазвичай ці інтерпретатори підтримують інтерактивне програмування.

Останні мікроконтролери часто інтегруються із вбудованою схемою налагодження, яка при доступі до внутрішньосхемного емулятора (ICE) через JTAG дозволяє налагоджувати прошивку за допомогою наладчика. ICE у реальному часі може дозволяти переглядати та/або керувати внутрішніми станами під час роботи. Трасуючий ICE може записувати програму і стани MCU до/після точки спрацьовування.

У мікроконтролерах зазвичай використовуються два різні типи пам'яті: енергонезалежна пам'ять для зберігання прошивки та пам'ять для читання та запису для тимчасових даних.

Від ранніх мікроконтролерів до наших днів шеститранзисторна SRAM майже завжди використовується як робоча пам'ять для читання/запису, а в регістровому файлі використовується ще кілька транзисторів на біт.

Крім SRAM деякі мікроконтролери також мають внутрішню EEPROM для зберігання даних; і навіть ті, у яких їх немає (або недостатньо), часто підключаються до зовнішньої послідовної мікросхеми EEPROM (наприклад, BASIC Stamp) або зовнішньої послідовної мікросхеми флеш-пам'яті.

1.2. Основні компоненти мікроконтролеру та його можливості

Основними елементами мікроконтролера є [27]:

1. Процесор (ЦП) - процесор можна розглядати як мозок пристрою. Він обробляє і відповідає різні інструкції, управляючі роботою мікроконтролера. Це включає виконання основних арифметичних, логічних операцій і операцій введення-виведення. Він також виконує операції передачі даних, які передають команди іншим компонентам більшої вбудованої системи.

2. Пам’ять. Пам’ять мікроконтролера використовується для зберігання даних, які процесор отримує та використовує для відповіді на інструкції, виконання яких запрограмовано. Мікроконтролер має два основні типи пам’яті:

– Пам’ять програм, в якій зберігається довготривала інформація про інструкції, що виконуються ЦП. Пам’ять програм є незалежною пам’яттю, тобто вона зберігає інформацію з часом, не вимагаючи джерела живлення.

– Пам’ять даних, потрібна для тимчасового зберігання даних під час виконання інструкцій. Пам’ять даних є енергозалежною, тобто дані, що зберігаються в ній, є тимчасовими і зберігаються тільки в тому випадку, якщо пристрій підключено до джерела живлення.

3. Периферійні пристрої вводу/виводу. Пристрої введення та виведення являють собою інтерфейс процесора із зовнішнім світом. Вхідні порти отримують інформацію та відправляють її процесору у вигляді двійкових даних. Процесор отримує ці дані та відправляє необхідні інструкції на пристрої виводу, які виконують завдання, зовнішні стосовно мікроконтролера.

Хоча процесор, пам’ять та периферійні пристрої вводу-виводу є визначальними елементами мікропроцесора, часто в нього включаються інші елементи. Сам термін периферійні пристрої введення-виведення просто відноситься до допоміжних компонентів, що взаємодіють з пам’яттю та процесором.

Існує безліч допоміжних компонентів, які можна зарахувати до периферійних пристроїв. Наявність деяких периферійних пристроїв введення-виведення є

елементарним для мікропроцесора, тому що вони є механізмом, за допомогою якого застосовується процесор.

Інші допоміжні елементи мікроконтролера включають [29]:

- Аналого-цифровий перетворювач (АЦП) - АЦП є схемою, яка перетворює аналогові сигнали в цифрові сигнали. Це дозволяє процесору в центрі мікроконтролера взаємодіяти із зовнішніми аналоговими пристроями, такими як датчики.
- Цифро-аналоговий перетворювач (ЦАП) - ЦАП виконує функцію, зворотну АЦП, і дозволяє процесору в центрі мікроконтролера передавати свої вихідні сигнали зовнішнім аналоговим компонентам.
- Системна шина. Системна шина - це з'єднувальний провід, що з'єднує всі компоненти мікроконтролера разом.
- Послідовний порт. Послідовний порт є одним із прикладів порту введення-виводу, який дозволяє мікроконтролеру підключатися до зовнішніх компонентів. Він виконує функції, аналогічні USB або паралельному порту, але відрізняється способом обміну бітами.

Можливості мікроконтролера.

Процесор мікроконтролера залежить від програми. Варіанти варіюються від простих 4-бітних, 8-бітових або 16-бітових процесорів до складніших 32-бітних або 64-бітових процесорів. Мікроконтролери можуть використовувати типи енергозалежної пам'яті, такі як оперативна пам'ять (RAM) і типи енергонезалежної пам'яті, включаючи флеш-пам'ять, стирається програмовану постійну пам'ять (EPROM) і електрично стирається програмовану постійну пам'ять (EEPROM).

Як правило, МК розробляються так, щоб їх можна було легко використовувати без додаткових обчислювальних компонентів, оскільки вони мають достатньо вбудованої пам'яті, а також пропонують висновки для загальних операцій виводу-введення-виводу, тому вони можуть безпосередньо взаємодіяти з датчиками та іншими компонентами.

Архітектура МК може бути заснована на гарвардській архітектурі або архітектурі фон Неймана, обидві з яких пропонують різні методи обміну даними між процесором

та пам’яттю. У гарвардській архітектурі шина даних та інструкція розділені, що дозволяє здійснювати одночасну передачу. В архітектурі фон Неймана одна шина використовується як для даних, так інструкцій.

Процесори МК можуть бути засновані на обчисленнях зі складним набором команд (CISC) або обчисленнях зі скороченим набором команд (RISC). CISC зазвичай має близько 80 інструкцій, а RISC - близько 30, і навіть більше режимів адресації, 12-24 проти RISC 3-5. Хоча CISC може бути простіше в реалізації і більш ефективно використовує пам’ять, його продуктивність може знижуватися через більшу кількість тактів, необхідних виконання інструкцій.

RISC, який приділяє більше уваги програмному забезпеченню, часто забезпечує більш високу продуктивність, ніж процесори CISC, які приділяють більше уваги апаратному забезпеченню завдяки спрощеному набору команд і, отже, більшій простоті конструкції, але через те, що він наголошує на програмному забезпечення, програмне забезпечення може бути складнішим. Який ISC використовується, залежить від програми [28].

Коли вони вперше стали доступними, мікроконтролери використовували виключно мову асемблера. Сьогодні мова програмування С є найпопулярнішим варіантом. Інші поширені мови мікропроцесорів включають Python та Java Script.

МК мають вхідні та вихідні контакти для реалізації периферійних функцій. До таких функцій відносяться аналого-цифрові перетворювачі, контролери рідкокристалічних дисплеїв (LCD), годинник реального часу (RTC), універсальний синхронний/асинхронний приймач-передавач (USART), таймери, універсальний асинхронний приймач-передавач (UART) та універсальна). USB) можливість підключення. Датчики, що збирають дані, пов’язані, зокрема, з вологістю та температурою, також часто підключаються до мікроконтролерів.

МК бувають таких типів [33]:

1. Загальні мікроконтролери включають Intel MCS-51, який часто називають мікроконтролером 8051, який був вперше розроблений в 1985 році;
2. МК AVR, розроблений Atmel у 1996 році;

3. Програмовані контролери інтерфейсу (PIC) від Microchip Technology;
4. Різні ліцензовані МК Advanced RISC Machines (ARM).

Ряд компаній виробляють та продають МК, у тому числі NXP Semiconductors, Renesas Electronics, Silicon Labs та Texas Instruments.

Програми для МК.

МК використовуються в багатьох галузях і додатках, у тому числі вдома та на підприємстві, автоматизації будівель, виробництві, робототехніці, автомобілебудуванні, освітленні, інтелектуальній енергетиці, промисловій автоматизації, зв'язку та розгортанні Інтернету речей (IoT).

Одним із дуже специфічних застосувань мікроконтролера є його використання як процесор цифрових сигналів. Часті аналогові сигнали мають певний рівень шуму. Шум у цьому контексті означає неоднозначні значення, які не можуть бути легко переведені у стандартні цифрові значення. МКможе використовувати свої АЦП і ЦАП для перетворення вхідного зашумленого аналогового сигналу на вихідний цифровий сигнал.

Найпростіші МК полегшують роботу електромеханічних систем, що використовуються у повсякденних предметах повсякденного побуту, таких як духовки, холодильники, тостери, мобільні пристрої, брелоки, системи відеоігор, телевізори та системи поливу газонів. Вони також поширені в офісних машинах, таких як копіювальні апарати, сканери, факсимільні апарати та принтери, а також в інтелектуальних лічильниках, банкоматах та системах безпеки. Більш складні МК виконують критично важливі функції у літаках, космічних кораблях, океанських суднах, транспортних засобах, медичних системах та системах життєзабезпечення, а також у роботах. У медичних сценаріях мікроконтролери можуть регулювати роботу штучного серця, нирки та інших органів. Вони також можуть відігравати важливу роль у функціонуванні протезів.

1.3. Постановка задачі дослідження

Метою є розробка та застосування ефективний алгоритму обробки інформації для технології дистанційного керування ЗВП на базі МК ESP-8266/ESP32.

Відповідно поставленій меті потрібно вирішити такі завдання:

1. Розглянути теоретичні засади використання мікроконтролерів;
2. Визначити основні компоненти мікроконтролеру та його можливості;
3. Проаналізувати методи та засоби проектування СДК за допомогою МК ;
4. Представити реалізацію розробленої СДК для ЗВП на базі ESP-8266/ESP32 із застосуванням функційної парадигми праграмування;

1.4. Висновки за розділом

Підсумовуючи перший розділ, можемо зробити такі висновки:

1. Визначено, що мікроконтролер можна розглядати як автономну систему з процесором, пам’яттю та периферійними пристроями та використовувати як вбудовану систему. Більшість мікроконтролерів, що використовуються сьогодні, вбудовані в інше обладнання, таке як автомобілі, телефони, побутові прилади та периферійні пристрої для комп’ютерних систем.
2. Окреслено основні компоненти МК та його можливості.
3. Сформовано задачі дослідження.

РОЗДІЛ 2.
МЕТОДИ ТА ЗАСОБИ ВИРІШЕННЯ ЗАДАЧІ З ПРОЕКТУВАННЯ СИСТЕМИ КЕРУВАННЯ ЗВУКОВІДТВОРЮЮЧИМИ ПРИСТРОЯМИ

2.1. Огляд існуючих систем дистанційного керування

Дистанційне керування (ДК) – це передача керуючого впливу або сигналу чи імпульсу від керуючої системи чи конкретної особи – оператора до об’єкта керування, що знаходиться на певній відстані або якщо при керуванні певною системою чи технічним пристроєм немає можливості передавати сигнал напряму, якщо об’єкт управління змінює своє розташування у просторі.

Зазвичай ДК складається із передавача – пульту дистанційного керування та приймача який надалі може віддавати керуючі сигнали всій конкретній системі , та виконавчих механізмів, та інших складових систему керування якою відбувається [4].

Системи ДК мають можливість використовувати різні канали зв’язку.

Канал зв’язку для виконання механічного керування – це канал зв’язку, який використовується у випадку коли об’єкти віддалені один від одного на невелику відстань або якщо треба забезпечити миттєву , не спотворену різними видами впливу реакцію систему над якою здійснюється процес керування.

Радіоканал – канал зв’язку при передачі сигналів по якому використовується принцип дистанційного керування при якому керуючий вплив та зворотній зв’язок керуємої системи відбувається через радіоканал за допомогою радіохвиль, тобто принципом вільного поширення у просторі електромагнітних хвиль під час якого передаючий пристрій та приймач можуть не знаходитись в зоні прямого контакту.

Ультразвуковий канал – це канал зв’язку що функціонує за принципом передавання звукових хвиль певного діапазону в системах принципу роботи передавач – приймач, та використовується для керування мобільними та стаціонарними об’єктами, але на суттєво обмеженій відстані від передавачем керуючого

ультразвукового імпульсу та приймачем системи над якою здійснюється керування [13].

Канал випромінювання світла інфрачервоного спектру – канал зв'язку робота якого заснована на принципі фізичного явища випромінювання електромагнітного випромінювання, що займає спектральну область між червоним кінцем видимого людським оком світла та коротким електромагнітним, мікрохвильовим та радіовипромінюванням.

Великі світові технологічні компанії такі як Google, Samsung, Apple, Intel, Qualcomm займаються розробкою власних продуктів заснованих на використанні технології інтернету речей, створення сучасних та якісних електроприладів та спрощення виробництва та використання для кінцевого користувача систем керування даними електроприладами.

Згідно з дослідженням міжнародної консалтингової компанії International Data Corporation розподіл ринку інформаційних технологій збільшить рівень коштів що циркулюють на ринку інтернету речей с 1.9 трильйонів доларів США у 2014 році і за прогнозами International Data Corporation досягне 7 трильйонів доларів США у 2021 році (рис. 2.1).

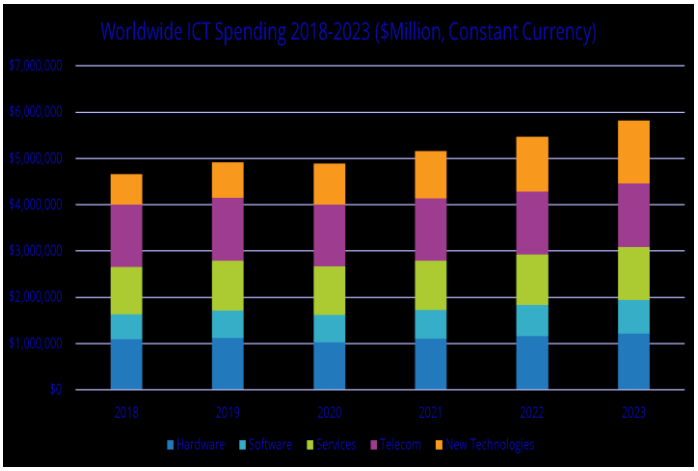

Рис. 2.1 - Загально-світовий оборотний об'єм ринку IoT з сегментами розподілу технологій до 2023 року

А за звітом аналітичного відділу компанії Gartner у 2020-му році кількість пристроїв, що використовуються у сфері технології інтернету речей та підключених до глобальної комунікаційної мережі інтернет налічує 10 мільярдів пристроїв (рис. 2.2).

Саме дякуючи великій розповсюдженості та доступності сьогодні кожна охоча людина має вільний доступ до використання технологій інтернету речей та зручного дистанційного керування побутовою електроапаратурою у повсякденному житті, тому ми маємо вільний доступ до «інтелектуальних» автомобілів, помешкань, побутової техніки та предметів щоденного використання.

Зацікавленість ринку мобільних додатків для смартфонів є еквівалентною кількості пристроїв що підключені до технології інтернету речей, тому як смартфони виступають надійною та зручною системою керування електроприладами для кінцевого користувача, адже вони виступають зв'язуючою ланкою між кінцевим користувачем та пристроєм так як через них і виконується керування речами [7].

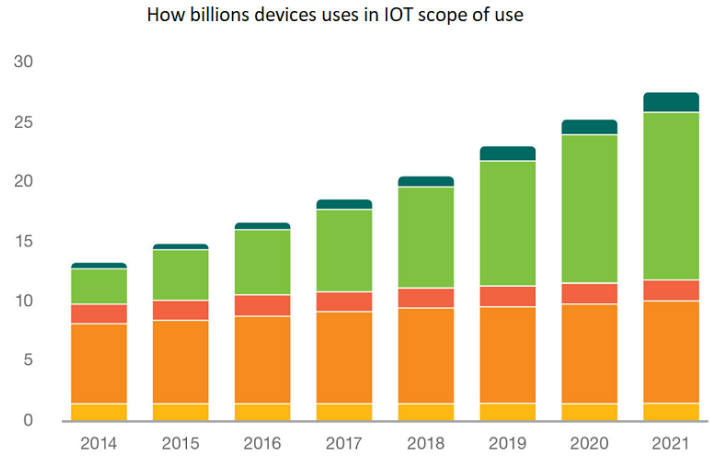

Рис. 2.2 - Графік загальної кількості пристроїв, що використовуються IoT

Розвиток IoT та галузі розробки мобільних застосунків мають на меті створення доступу великому сегменту користувачів до широкої кількості можливостей та переваг над стандартними системами керування різного роду пристроїв.

Саме тому галузь IoT стала однією з пріоритетних напрямків розробки мобільних додатків за останні роки.

Користувацькі мобільні застосунки в загальному випадку можна розділити на дві великі за призначенням групи [11]:

− додатки для накопичування та аналізу даних;

− додатки для керування.

Основне завдання першої групи додатків це зняття показань з пристроїв та їх зберігання в пам’яті додатку до цієї групи можна віднести такі додатки як:

− додатки для фітнес-трекєрів;

− ваг;

− вимірювачів атмосферної вологості;

− камер та пристроїв що застосовуються у системах безпеки;

− пожежних сигналізаціях;

− різних датчиків та іншої вимірювальної техніки;

Друга група додатків спеціалізується на слідкуванні за пристроями, зніманню повної інформації з нього а також зміна його робочого стану тобто виконання процесу керування пристроєм.

До цієї групи належать додатки за допомогою яких здійснюється повне керування пристроями [5]:

− додатки для керування кавоварками;

− електричними чайниками;

− телевізорами;

− кондиціонерами, обігрівачами, вентиляторами;

− пристрої системи керування «розумного будинку»;

Додатки як і самі пристрої інтернету речей можна віднести до будь-якої категорії – «фітнес та здоров’я», «медицина», «побутова техніка», «розумний будинок».

Все залежить від того якого типу пристрій над яким здійснюється керування. У будь-якому випадку, якщо цей пристрій є у продажу , то додаток до нього можна скачати із каталогу додатків «PlayMarket» для мобільної операційної системи Android.

Функціональні можливості кожного додатку залежать від того якого типу є виконавчий пристрій яким виконується керування.

Розпосюдженим є явище коли виробники електроприладів самі поставляють необхідні додатки для керування електроприладами їх виробничої лінійки при цьому додатки для керування приладами є сумісними та коректно працюють з усією виробничою лінійкою електроприладів даного виробника. В цьому випадку користувач лише виконує процес синхронізації додатків з власними побутовими пристроями.

2.2. Засоби програмування сучасних мікроконтролерів

Сучасні програмовані логічні контролери (ПЛК) програмуються відповідно до стандарту МЕК 61131, в якому описані стандартизовані мови програмування, що представляють інтерес для практичного використання. У даному розділі наводиться їх детальний аналіз [10].

1. IL (Instruction List - список команд)-текстова мова низького рівня, синтаксично схожий на асемблер. Програма мовою IL має приблизно такий вигляд:

CAL Fp (MODE: = Sinus, BASE: = TRUE,
PERIOD: = t # 2s, AMPLITUDE: = 100)
LD Fp.Out
ST Out_Gen_1.

На його основі можна створювати швидкодіючі програмні одиниці, тому він застосовується при створенні компактних компонентів, ретельного опрацювання.

2. ST (Structured Text - структурований текст) - текстовий мова високого рівня, синтаксично - адаптований мову Паскаль. Програма мовою ST має наступний вигляд:

Fp (MODE: = Triangle, BASE: = TRUE,
PERIOD: = t # 2s, AMPLITUDE: = 100);
Out_Gen_1: = INT_TO_REAL (Fp.OUT);
Filter1.In: = Out_Gen_1 - Filter1.Out;

Filter1 (Tm: = 100, RESET: = FALSE);

Variable1: = SQRT (Filter1.Out).

На його основі можна створювати гнучкі процедури обробки даних. Мова ST є основним для програмування кроків мови SFC.

3. LD (Ladder Diagram - релейні діаграми) - графічна мова, що є стандартизованим варіантом класу мов релейно-контактних схем. Логічні вирази цією мовою описуються у вигляді реле, які широко застосовувалися в області автоматизації в 60-х роках. Стандартом МЕК 61131-3 передбачено використання не тільки базових елементів програмування (\ "контакт \" і \ "котушка \"), але і будь-яких інших функціональних блоків.

4. FBD (Functional Block Diagram - діаграми функціональних блоків) - графічна мова, за своєю суттю схожий на LD, але замість реле використовує функціональні блоки. FBD-схеми дуже чітко відображають взаємозв'язок входів і виходів діаграми, використовуючи технологію інкапсуляції алгоритмів обробки даних. Все програмування зводиться до \ "склеюванню \" готових компонентів.

5. SFC (Sequential Function Chart - послідовні функціональні схеми) - графічна мова, що дозволяє описати алгоритм у вигляді набору пов'язаних пар \ "крок-перехід \". Крок представляє собою набір операцій над змінними, а перехід - набір виразів, що визначає передачу правління наступного кроку. SFC має можливість розпаралелювання алгоритму, але не має коштів для опису кроків і переходів, які можуть бути виражені лише засобами інших мов стандарту.

Основною перевагою SFC є висока виразність графічного представлення алгоритму.

Для створення програм на мовах стандарту МЕК 61131-3 використовуються інструментальні пакети. Зазвичай вони включають редактори (для кожної мови) і деякі додаткові функціональні розширення. Асортимент таких пакетів досить широкий (CoDeSys, Concept, IsaGRAF, OpenPCS, Multiprog, Virgo2000, KONGRAF), але ми

розглянемо тільки два з них - CoDeSys (компанії 3S Software) і Concept (Schneider Electrics).

Пакет CoDeSys є одним з найбільш відомих універсальних інструментів МЕК-програмування для ПЛК і промислових комп'ютерів. Його використовують більше ста відомих компаній-виробників апаратних засобів індустрії автоматизації.

Крім п'яти стандартних мов, в CoDeSys включений редактор CFC-діаграм, заснований на мові FBD, але більш зручний і наочний за рахунок вільного розміщення блоків. CoDeSys може генерувати машинний код для більшості поширених процесорів (Motorola, Intel (в т.ч. 80x86 і Pentium), Siemens, Hitachi та ін.) Всі компоненти CoDeSys докладно документовані і мають вбудовану систему допомоги.

До основних особливостей пакету можна віднести наступні [12].

1. Швидке впровадження - адаптація для будь-якої стандартної процесорної платформи займає не більше двох днів.

2. Ефективні засоби введення - функції автоматичного оголошення та форматування, асистент введення і інші максимально спрощують роботу. Команди мають можливість керування мишею і введення з клавіатури.

3. Висока продуктивність - вбудований компілятор безпосередньо генерує швидкий машинний код (на відміну від звичайних трансляторів), що забезпечує максимальну продуктивність прикладних проектів при збереженні високої швидкості компіляції.

4. Низькі системні вимоги програми - CoDeSys сумісний з будь-якою операційною системою сімейства Microsoft ® Windows ®.

5. Компактність проекту - на відміну від більшості конкуруючих пакетів весь проект знаходиться в одному файлі, що зручно при його перенесення.

6. Розширена реалізація всіх п'яти МЕК-мов. Редактори для всіх мов програмування зосереджені в одному додатку, що дуже зручно. Вибір мови написання модуля здійснюється при його створенні і після цього не може бути змінений надалі. CoDeSys включає багатий набір засобів налагодження та супроводження (явне

оголошення змінних, автоматичне формування списку параметрів підпрограм, моніторинг / запис / фіксація значень змінних, покрокове виконання, он-лайн корекція коду, трасування і емуляція).

Крім середовища програмування, до складу комплексу CoDeSys входять: SP RTE (емуляція ПЛК на ПК), Soft Motion (набір засобів управління рухом - до багатовимірної інтерполяції сучасних систем ЧПУ), ARTI (забезпечує символьний доступ до змінних в ПЛК) і ENI Server (дозволяє працювати з одним проектом кільком користувачам).

Також CoDeSys володіє досить широкими графічними можливостями, які за функціональністю не поступаються найпростішим.

Даний пакет є фірмовим, тобто він призначений для програмування контролерів тільки одного виробника - компанії Schneider Electric. Крім стандартних, Concept підтримує мову LL984 (Ladder Logic - сходова логіка). Він використовується для програмування ПЛК старої версії (Modsoft).

За допомогою засобів конфігурації Concept дуже просто вибирати, розміщувати і переміщати об’єкти (блоки, кроки чи переходи) в графічній формі.

Функції імпорту / експорту дозволяють перетворювати програми на мовах ST і IL в програми на FBD, SFC, IL або ST і навпаки. Великим плюсом пакета є наявність великого вибору бібліотек блоків, що істотно спрощує процес програмування і розширює можливості пакета. При цьому існує можливість доповнити бібліотеки своїми власними блоками, які виконують необхідні функції, за допомогою утиліти Concept DFB (Функціональні блоки користувача).

Також ця утиліта призначена для програмування кроків мови SFC. Засобами Concept можливо документування проекту, що передбачає роздруківку всіх даних проекту або їх частина за вибором програміста.

При цьому користувачеві немає необхідності турбуватися про оформлення документації - вона буде виконана у відповідності з нормами.

Ще однією перевагою пакета Concept є така функціональна можливість, як забезпечення безпеки роботи з додатком. Існує сім рівнів доступу до

використання програмних засобів Concept, що забезпечують різні права різним користувачам. Є можливість резервування до 128 користувачів і їх паролів. Вбудований симулятор дозволяє проводити налагодження програм, не використовуючи ПЛК [5].

Недоліком пакету можна вважати неможливість одночасного запуску декількох додатків з комплексу, що, безсумнівно, є утрудняють обставиною, особливо при розробці складних діаграм SFC.

Обидва розглянутих пакета володіють широкими можливостями і є лідерами на ринку програмного забезпечення для ПЛК.

Головною перевагою пакета CoDeSys є його безкоштовне поширення, що дозволяє використовувати його для навчання МЕК-програмуванню. У той же час він є і зручним засобом реального програмування, яке можна здійснювати після покупки ліцензії у компанії-виробника.

Пакет Concept, навпаки, є спочатку ліцензійним, що робить його непридатним для навчання. Але він підтримує найпопулярніші моделі контролера одного з лідерів «залізного» ринку - компанії Schneider Electric.

2.3. Підходи до проектування системи та алгоритм її створення

По суті, МК збирає вхідні дані, обробляє цю інформацію та виводить певну дію на основі зібраної інформації. Мікроконтролери зазвичай працюють на нижчих швидкостях, в діапазоні від 1 МГц до 200 МГц, і повинні бути спроектовані так, щоб споживати менше енергії, оскільки вони вбудовані в інші пристрої, які можуть споживати більше енергії в інших областях [18].

Отже, МК можна розглядати як невеликий комп’ютер, і через основні компоненти всередині нього; центральний процесор (ЦП), оперативне запам’ятовуючий пристрій (ОЗУ), флеш-пам’ять, інтерфейс послідовної шини, порти вводу/виводу (порти вводу/виводу) і, у багатьох випадках, програмне пристрій, що електрично стирається, тільки для читання Пам’ять (ЕСППЗУ). На рис. 1 показана велика схема основних

частин, а також інших частин мікроконтролера. Давайте заглибимося в кожен із цих компонентів і подивимося, як вони працюють усередині МК (рис. 2.3).

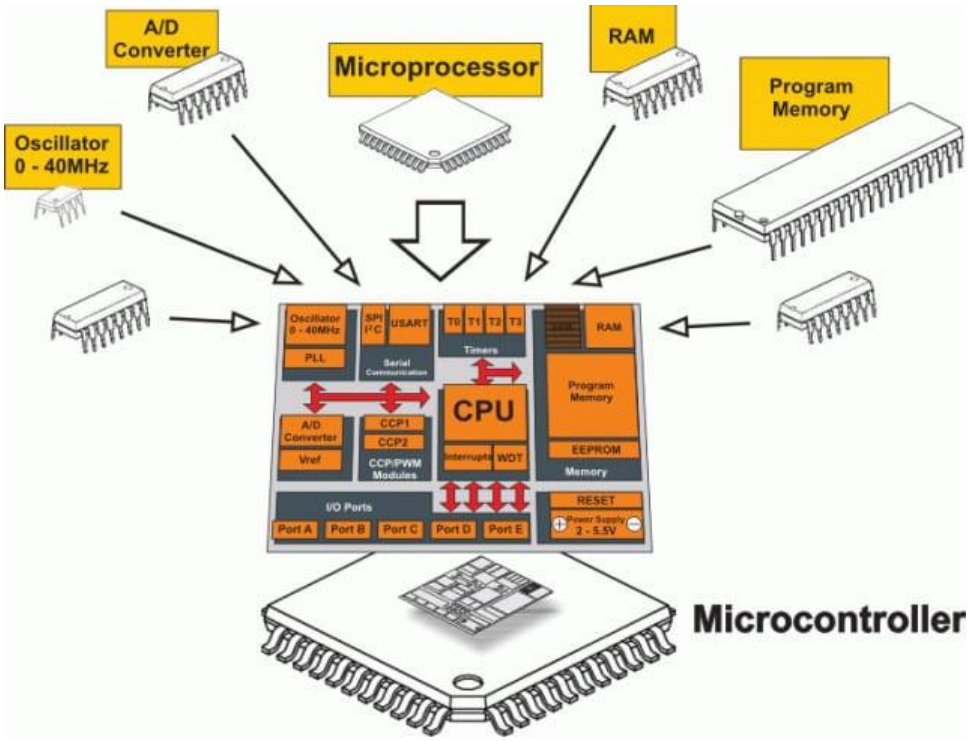

Рис. 2.3 - Складові МК

Проектування ЦП мікроконтролера.

Центральний процесор, іноді званий процесором або мікропроцесором, керує всіма потоками інструкцій/даних, які він отримує. Ви можете думати про нього як про мозок системи, що обробляє всі дані, що надходять і виконує необхідні інструкції. Двома його основними компонентами є арифметико-логічний пристрій (ALU), який виконує арифметичні та логічні операції, та керуючий пристрій (CU), який обробляє виконання всіх інструкцій процесора. На рис. 2.4 показаний звичайний "машинний цикл", через який проходить ЦП.

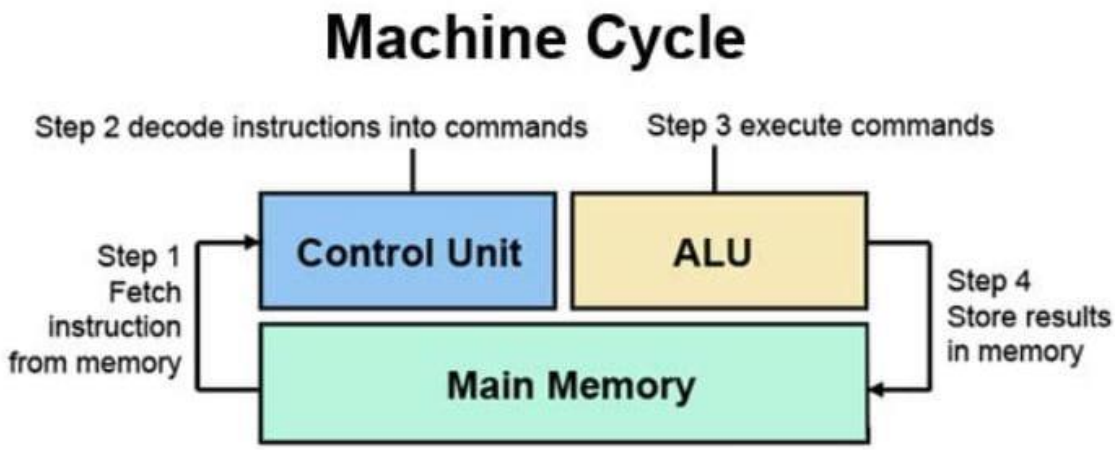

Рис. 2.4 - Типовий машинний цикл, який виконує ЦП

Оперативна пам’ять - це компонент, який тимчасово зберігає дані та до якого можна швидко отримати доступ. Він забезпечує швидкий доступ для читання та запису до пристрою зберігання. Це відрізняється від більшості інших спогадів, оскільки для отримання даних потрібно більше часу, оскільки дані не завжди доступні. Ви можете бачити це як оперативну пам’ять, що має доступ до поверхні даних - легко доступну - але все, що занурюється глибше, вимагатиме іншого типу пам’яті. Оперативна пам’ять покращує загальну продуктивність системи, оскільки дозволяє мікроконтролерам одночасно обробляти більше інформації. Оскільки оперативна пам’ять є тимчасовими даними, її вміст завжди стирається при вимиканні мікроконтролера.

Флеш-пам’ять - це тип енергонезалежної пам’яті, яка, на відміну від оперативної пам’яті, зберігає свої дані протягом тривалого часу, навіть якщо мікроконтролер вимкнено. Це зберігає збережену програму, яку ви, можливо, завантажили у мікроконтролер. Флеш-пам’ять записується в блок або сектор за раз, тому, якщо вам потрібно просто перезаписати один байт, флеш-пам’яті потрібно перезаписати весь блок, в якому знаходиться байт, що може швидше зношуватися [34].

EEPROM схожий на флеш-пам’ять, будучи незалежною пам’яттю і зберігаючи свої дані навіть після вимкнення. Різниця в тому, що флеш-пам’ять перезаписує «блок» байтів, EEPROM може перезаписувати будь-який конкретний байт у будь-який

час. Це продовжує термін служби EEPROM у порівнянні з флеш-пам’яттю, але також означає, що вона дорожча.

Інтерфейс послідовної шини - це послідовний зв’язок у мікроконтролері, що відправляє дані по одному біту за раз. З платами мікроконтролерів він з’єднує ІВ із сигнальними доріжками на друкованій платі (PCB). Для ІС вони використовують послідовну шину передачі даних, щоб зменшити кількість контактів у корпусі, що робить їх економічнішими. Прикладами послідовних шин ІС є SPI або I2C.

Порти вводу-виводу - це те, що мікроконтролер використовує для підключення до реальних програм. Вхідні дані отримують зміни в реальному світі, від вимірювання температури до виявлення руху, натискання кнопок та багато іншого. Потім вступ вступає в ЦП і вирішує, що робити з цією інформацією. Коли настав час виконати певну команду на основі певного значення на вході, він відправляє сигнал на вихідні порти, де він може змінюватись від простого вимикання світлодіода до запуску двигуна для певної частини та багато іншого.

На рис. 2.5 показані деякі поширені вхідні та вихідні компоненти.

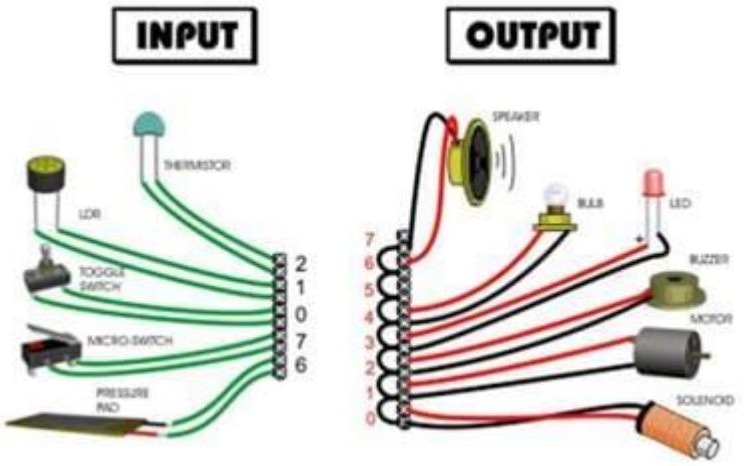

Рис. 2.5 - Загальні компоненти введення та виведення, що застосовуються для МК

В сучасних пристроях логіка функціонування зав’язана на використанні керуючих пристроїв – мікроконтролерах, які виконують роль інтерфейсу між кінцевим користувачем ЗВП та його керуючим пристроєм, зазвичай мобільним телефоном, який

в сучасному світі є надійним помічником кожної людини та має в собі всі необхідні користувачеві функції для взаємодії з навколишнім світом за допомогою програмних додатків функціонал яких і дає змогу виконувати дистанційне керування ЗВП та здійснює цей процес по радіоканалу, який використовують вбудовані в мобільний телефон чіпи через які виконується з'єднання між мобільним телефоном та іншими пристроями за допомогою бездротових дистанційних технологій передачі інформації Wi-fi або Bluetooth (рис. 2.6).

Керування МК може виконуватись у різних режимах роботи [19]:

− імпульсному режимі роботи;

− дистанційно – релейному режимі роботи;

Імпульсний режим роботи – це коли управління виходами та вбудованим у МК реле здійснюється імпульсним сигналом. Тобто сигнал від пристрою передавача поступає на приймач у мікроконтролері у вигляді короткого імпульсу. Режим роботи може бути одно або двонаправленим. При двунаправленому режимі стан вихідних релейних контактів можна контролювати за допомогою вбудованого світлового сигнального індикатора [12].

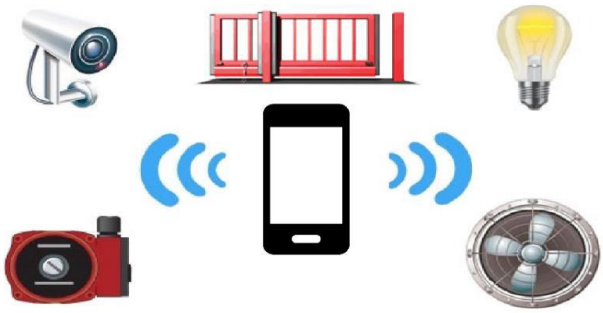

Рис. 2.6 – Схема дистанційної взаємодії мобільного пристрою з різними виконавчими кінцевими пристроями

Дистанційно – релейний режим роботи – це коли керування релейними контактами здійснюється неперервним сигналом, тобто передача керуючого сигналу з пристрою керування відбувається постійно у часі під час чого реле замкнуто на протязі всього часу поки контакт знаходиться в стані замкнення відповідного каналу пристрою з якого відбувається передача сигналу керування, в процесі роботи відбувається передача сигналу кожні 80 секунд. Цей сигнал обновлює стан входів для синхронізації з виходами приймаючого сигнал пристрою вбудованого у мікроконтролер. Цей режим має більш високий захист від зовнішніх впливів або втрати живлення [20].

Контроль за станом виходів відбувається в реальному часі. Управління відбувається в ручному або автоматично запрограмованому режимі від зовнішніх сигналопередаючих пристроїв керування.

Протоколи дистанційної взаємодії – це певний набір промислових специфікацій бездротових персональних мереж реалізованих, як інтерфейс завдяки якому пристрої які підтримують технологію цю технологію можуть взаємодіяти між собою. Вони забезпечують обмін інформацією різні пристрої над якими можна здійснювати процес дистанційного керування , наприклад керування зі спеціального пульта або мобільного телефона засобом мобільного додатку.

До таких пристроїв можна віднести [16]:

− персональні комп’ютери;

− мобільні телефони;

− принтери та різні офісні переферійні пристрої;

− акустичні системи;

− навушники та гарнітури;

Розглянемо основні технології дистанційного обміну даними які на сьогодні доступні для використання та не потребують спеціального дозволу є безкоштовними та доступними для розробки власних продуктів (рис. 2.7).

Технологія зв'язку	Wi-Fi	Bluetooth	ZigBee	Thread
Стандарт зв'язку	IEEE 802.11	IEEE 802.15.4	IEEE 802.15.4	IEEE 802.15.4
Швидкість передачі даних	300+ Мбит/с	до 3 Мбит/с	250 Кбит/с	250 Кбит/с
Енергоспоживання	Высокое	Низкое	Низкое	Низкое
Частотний діапазон	2,4 ГГц	2,4 ГГц	2,4 ГГц	2,4 ГГц
Підтримка IP-технологій	+	-	-	+
Топологія	«звезда»	«звезда»	«mesh»	«mesh»

Рис. 2.7 - Характеристики різних бездротових технологій

Wi-Fi - це технологія бездротової локальної мережі з пристроями на основі стандартів IEEE 802.11. Логотип Wi-Fi є торговою маркою Wi-Fi Alliance. Під абревіатурою Wi-Fi (від англійського словосполучення Wireless Fidelity[2], яке можна дослівно перекласти як «бездротова точність»), в даний час розвивається ціла родина стандартів передачі цифрових потоків даних по радіоканалах. Основними діапазонами Wi-Fi вважаються 2.4 ГГц (2412 МГц-2472 МГц) та 5 ГГц (5160-5825 МГц).

Сигнал Wi-Fi може передаватися на кілометри навіть за низької потужності передачі, але для прийому Wi-Fi-сигналу зі звичайного Wi-Fi-маршрутизатора на далекій відстані потрібна антена з високим коефіцієнтом посилення (наприклад, параболічна антена або Wi-Fi-гармата).

Будь-яке обладнання, що відповідає стандарту IEEE 802.11, може бути протестоване у Wi-Fi Alliance та отримати відповідний сертифікат та право нанесення логотипу Wi-Fi.

Переваги Wi-Fi.

Дозволяє розгорнути мережу без прокладки кабелю, що може зменшити вартість розгортання та розширення мережі. Місця, де не можна прокласти кабель, наприклад, поза приміщеннями та в будинках, що мають історичну цінність, можуть обслуговуватися бездротовими мережами.

Дозволяє мати доступ до мережі мобільних пристроїв.

Пристрої Wi-Fi поширені над ринком. Гарантується сумісність обладнання за рахунок обов'язкової сертифікації обладнання з логотипом Wi-Fi. Мобільність. Ви більше не прив'язані до одного місця і можете користуватися інтернетом в зручній для вас обстановці. В межах зони Wi-Fi в інтернеті можуть виходити кілька користувачів з різних пристроїв [30].

Випромінювання від пристроїв Wi-Fi в момент передачі даних на порядок (в 10 разів) менше, ніж у мобільного телефону (рис.2.8).

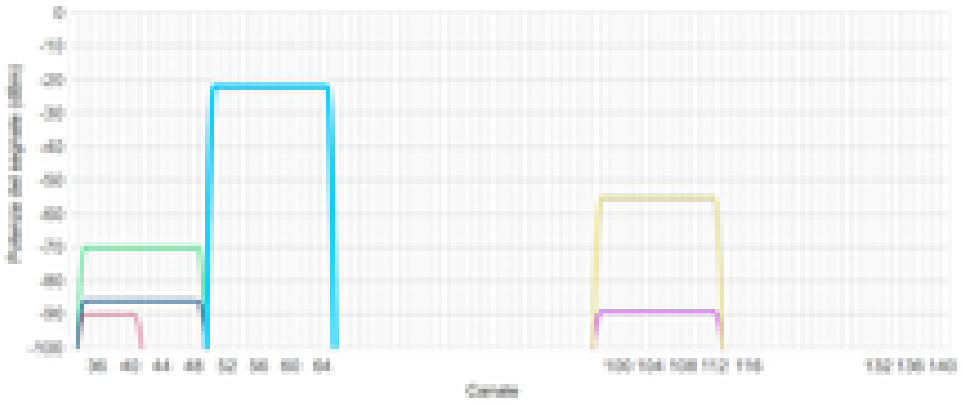

Рисунок 2.8 - Схема швидкості Wi-Fi

Bluetooth – це стандарт бездротової технології малого радіусу дії, який використовується для обміну даними між стаціонарними та мобільними пристроями на коротких відстанях з використанням радіохвиль УВЧ у діапазонах ISM, від 2,402 ГГц до 2,48 ГГц, а також для побудови персональних мереж (PAN).

Він в основному використовується як альтернатива дротовим з'єднанням, для обміну файлами між сусідніми портативними пристроями і для підключення мобільних телефонів і музичних плеєрів з бездротовими навушниками. У найбільш широко використовуваному режимі потужність передачі обмежена 2,5 мВт, що забезпечує дуже малу дальність до 10 метрів.

Bluetooth керується Bluetooth Special Interest Group (SIG), до якої входять понад 35 000 компаній-членів у галузі телекомунікацій, обчислювальної техніки, мереж та

побутової електроніки. IEEE стандартизував Bluetooth як IEEE 802.15.1, але більше не підтримує цей стандарт. Bluetooth SIG спостерігає за розробкою специфікації, керує програмою кваліфікації та захищає товарні знаки.

Виробник повинен відповідати стандартам Bluetooth SIG, щоб продавати його як пристрій Bluetooth [21].

До технології застосовується мережа патентів, що ліцензуються для окремих відповідних пристроїв. З 2009 року інтегральна схема Bluetoothчіпи поставляється приблизно 920 мільйонів одиниць на рік. До 2017 року щорічно відвантажувалося 3,6 мільярда пристроїв Bluetooth, і очікується, що поставки продовжуватимуть зростати приблизно на 12% на рік.

Bluetooth працює на частотах від 2,402 до 2,480 ГГц або від 2,400 до 2,4835 ГГц, включаючи захисні смуги шириною 2 МГц у нижній частині та 3,5 МГц у верхній частині.

Це глобально неліцензований (але не нерегульований) промисловий, науковий та медичний (ISM) діапазон радіочастот близької дії 2,4 ГГц.

Bluetooth використовує радіотехнологію, звану стрибкоподібною перебудовою частоти. Bluetooth ділить дані на пакети і передає кожен пакет по одному з 79 призначених каналів Bluetooth.

Кожен канал має смугу пропускання 1 МГц.

Зазвичай він виконує 1600 стрибків частоти в секунду з адаптивною стрибкоподібною перебудовою частоти (AFH) включений. Bluetooth Low Energy використовує інтервал 2 МГц, що відповідає 40 каналам.

Спочатку модуляція з частотною маніпуляцією за Гауссом (GFSK) була єдиною доступною схемою модуляції. З моменту появи Bluetooth 2.0+EDR між сумісними пристроями також можуть використовуватися модуляція π/4 – DQPSK (диференціальна квадратурна фазова маніпуляція) та модуляція 8-DPSK. Говорять, що пристрої, що працюють з GFSK, працюють у режимі базової швидкості (BR), де можлива миттєва швидкість передачі даних 1 Мбіт/с.

Термін Enhanced Data Rate (EDR) використовується для опису схем π/4-DPSK (EDR2) та 8-DPSK (EDR3), кожна з яких дає 2 та 3 Мбіт/с відповідно. Комбінація цих режимів (BR та EDR) у радіотехнології Bluetooth класифікується як радіо BR/EDR.

У 2019 році Apple опублікувала розширення під назвою HDR, яке підтримує швидкість передачі даних 4 (HDR4) та 8 (HDR8) Мбіт/с з використанням модуляції π/4-DQPSK на каналах 4 МГц із прямою корекцією помилок (FEC)

Bluetooth - це пакетний протокол із архітектурою «головний/відомий». Один головний може зв'язуватися з сімома послідовниками в пікосеті.

Всі пристрої в даній пікосеті використовують годинник, наданий майстром, як основа для обміну пакетами.

Головний годинник цокає з періодом 312,5 мкс, два такти складають слот 625 мкс, а два слоти складають пару слотів 1250 мкс. У найпростішому випадку пакетів з одним слотом основна передача здійснюється в парних слотах, а прийом – непарних [23].

Повторювач, навпаки, отримує у парних слотах і передає у непарних слотах.

Пакети можуть мати довжину 1, 3 або 5 слотів, але завжди передача основного пакета починається в парних слотах, а передача веденого - в непарних.

Щоб використовувати бездротову технологію Bluetooth, пристрій повинен мати можливість інтерпретувати певні профілі Bluetooth, які є визначенням можливих програм і визначають загальну поведінку, яку використовують пристрої Bluetooth для зв'язку з іншими пристроями Bluetooth. Ці профілі включають настройки для параметризації та керування зв'язком із самого початку.

Дотримання профілів заощаджує час на повторну передачу параметрів до того, як двонаправлений зв'язок стане ефективним. Існує безліч профілів Bluetooth, що описують безліч різних типів програм або варіантів використання пристроїв (рис. 2.9).

Рис. 2.9 - Типова Bluetooth-гарнітура для мобільного телефону

Щоб розширити сумісність пристроїв Bluetooth, пристрої, що відповідають стандарту, використовують інтерфейс, що називається HCI (інтерфейс хост-контролера), між хост-пристроєм (наприклад, ноутбуком, телефоном) та пристроєм Bluetooth (наприклад, бездротовою гарнітурою Bluetooth).

Протоколи високого рівня, такі як SDP (протокол, який використовується для пошуку інших пристроїв Bluetooth у межах діапазону зв'язку, також відповідає за визначення функції пристроїв у межах діапазону),RFCOMM (протокол, який використовується для емуляції з'єднань послідовного порту) та TCS (протокол керування телефоном).

Взаємодіяти з контролером основної лінії частот через L2CAP (протокол управління та адаптації логічного зв'язку). Протокол L2CAP відповідає за сегментацію та повторне складання пакетів.

Апаратне забезпечення, що становить пристрій Bluetooth, логічно складається із двох частин; які можуть бути фізично розділені. Радіопристрій, що відповідає за модуляцію та передачу сигналу; та цифровий контролер.

Цифровий контролер, ймовірно, є ЦП, однією з функцій якого є запуск контролера зв'язку та інтерфейси з хост-пристроєм.

Проте деякі функції можуть бути делеговані апаратного забезпечення.

Контролер каналу відповідає за обробку основної смуги частот та управління протоколами ARQ та FEC фізичного рівня.

Крім того, він обробляє функції передачі (як асинхронні, так і синхронні), кодування звуку (наприклад, SBC (кодек)) та шифрування даних. ЦП пристрою відповідає за виконання інструкцій, пов'язаних з Bluetooth хост-пристрої, щоб спростити його роботу.

Для цього ЦП запускає програмне забезпечення з назвою Link Manager, яке має функцію зв'язку з іншими пристроями через протокол LMP [8].

Bluetooth-пристрій - це бездротовий пристрій малого радіусу дії. Пристрої Bluetooth виготовляються на мікросхемах інтегральної схеми RF CMOS.

Bluetooth визначається як багаторівнева архітектура протоколу, що складається з основних протоколів, протоколів заміни кабелю, протоколів керування телефоном та прийнятих протоколів.

Обов'язковими протоколами для всіх стеків Bluetooth є LMP, L2CAP та SDP.

Крім того, пристрої, що взаємодіють із Bluetooth, майже повсюдно можуть використовувати ці протоколи: HCI та RFCOMM.

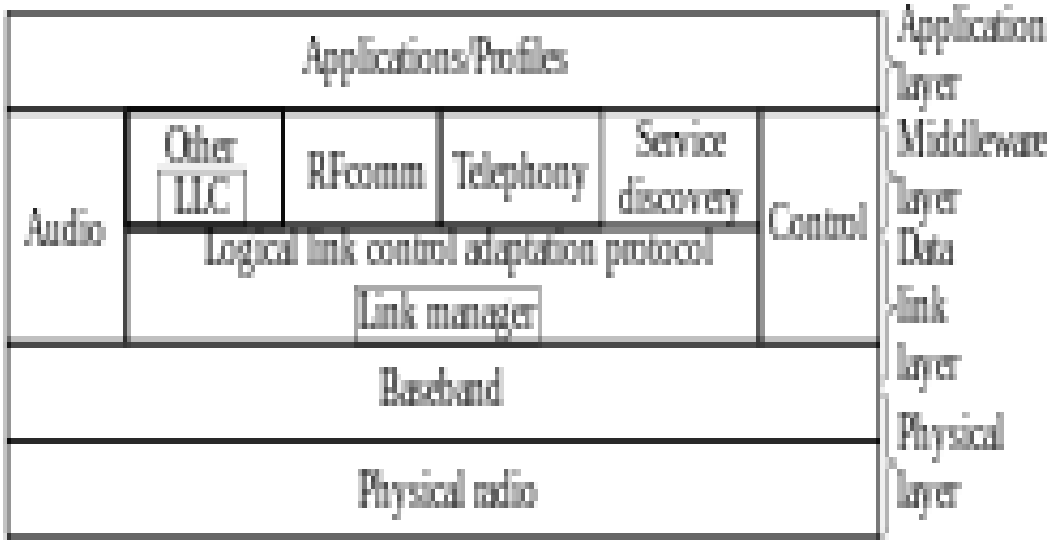

Рис. 2.10 - Стек протоколів Bluetooth

Технологія Bluetooth у своїй роботі використовує певний стек протоколів (правил роботи), який можна розділити на дві групи (рис. 2.10):

− протоколи універсального призначення;

− протоколи для використання у вбудованих системах;

Протоколи універсального призначення створенні для забезпечення функціональності та гнучкості використання на платформах різних пристроїв що підтримують використання цієї технології.

В даному проекті планується реалізувати систему, що складається з 4 основних компонентів, що взаємодіють через об’єкт управління, роль якого виконуватиме мікроконтролер.

1. ESP8266 - мікроконтролер китайського виробника Espressif Systems з інтерфейсом Wi-Fi. Крім Wi-Fi, мікроконтролер відрізняється відсутністю флеш-пам’яті в SoC, програми користувача виконуються із зовнішньої флеш-пам’яті з інтерфейсом SPI [15].

Мікроконтролер привернув увагу у 2014 році у зв’язку з виходом перших товарів на його основі за надзвичайно низькою ціною.

Весною 2016 року почалося виробництво ESP8285, що поєднує ESP8266 та флеш-пам’ять на 1 МБайт. Восени 2015 року Espressif представила розвиток лінійки - мікросхему ESP32 та модулі на її основі.

– Мікроконтролер 80 MHz

– 32-bit процесор Tensilica (англ.) рос. Xtensa L106

– Можливий негарантований розгін до 160 МГц.

– IEEE 802.11 b/g/n Wi-Fi.

– Підтримується WEP та WPA/WPA2.

– 14 портів вводу-виводу (з них можна використовувати 11), SPI, I²S, UART, 10-bit АЦП. I²C можливий лише через bit-banging.

– Живлення 2,2 ... 3,6 В.

– Споживання - до 215 мА в режимі передачі, 100 мА - в режимі прийому, 70 мА - в режимі очікування.

– Підтримуються три режими зниженого споживання, все без збереження з’єднання з точкою доступу: Modem sleep (15 мА), Light sleep (0.4 мА), Deep sleep (15 мкА).

– Мікроконтролер не має на кристалі користувальницької енергонезалежної пам’яті. Виконання програми ведеться із зовнішнього SPI ПЗУ шляхом динамічного підвантаження необхідних ділянок програми в кеш інструкцій. Підвантаження відбувається апаратно, прозоро для програміста.

– Підтримується до 16 МБ зовнішньої пам’яті програм. Можливий інтерфейс Standard, Dual або Quad SPI.

Електричні параметри, цоколівки, схеми включення можна знайти в документах 0A-ESP8266EX__Datasheet і 0B-ESP8266__System_Description з Espressif SDK.

Джерело програми ESP8266 задається станом портів GPIO0, GPIO2 і GPIO15 в момент закінчення сигналу Reset (тобто подачі живлення). Найбільш цікаві два режими: виконання коду з UART (GPIO0 = 0, GPIO2 = 1 та GPIO15 = 0) та із зовнішньої ПЗУ (GPIO0 = 1, GPIO2 = 1 та GPIO15 = 0). Режим виконання коду UART використовується для перепрошивки підключеної флеш-пам’яті, а другий режим - штатний робітник.

Засоби розробки.

Програмні засоби розробки (програмний комплект розробника, SDK) складаються з:

Компілятори. Компілятор Xtensa LX106 входить до пакету компіляторів GNU Compiler Collection. Компілятор має відкриті вихідні тексти. У різних SDK можуть міститися різні збірки цього компілятора, що трохи відрізняються підтримуваними опціями.

Бібліотек до роботи з периферією контролера, стеків протоколів WiFi, TCP/IP.

Засобів завантаження файлу, що виконується, в пам’ять програм мікроконтролера.

Опціональний IDE.

Espressif вільно розповсюджує свій комплект розробника. У цей комплект входить компілятор GCC, бібліотеки Espressif та завантажувальна утиліта XTCOM. Бібліотеки

постачаються у вигляді скомпілованих бібліотек, без вихідних текстів. Espressif підтримує дві версії SDK: одна на основі RTOS, інша на основі зворотного виклику (callback).

Крім офіційної SDK існує низка проектів альтернативних SDK [8]. Ці SDK використовують бібліотеки Espressif або пропонують власний еквівалент бібліотек Espressif, отриманий методами реверсинжинірингу.

"esp-open-sdk". Удосконалена версія SDK від Expressif. Містить GCC компілятор та деякі бібліотеки Expressif. Тільки Лінукс [15].

"Unofficial Development Kit" Михайла Григор'єва. У комплект входить Windows-інсталятор, компілятор GCC власної збірки з інтеграцією з графічною IDE Eclipse, актуальні комплекти бібліотек та документації Espressif, деякі утиліти. Є російськомовний форум.

"Arduino IDE for ESP8266" - доповнення до IDE Arduino, що дозволяє програмувати ESP8266 так само легко, як будь-які інші модулі Ардуїно. При цьому є мережна функціональність ESP8266. Компілятор GCC, завантажувач прошивки ESPTool.

Детальний російськомовний опис процесу встановлення та доступного API тут, приклад роботи тут.

"GNU toolchain for esp8266". Має можливість інтеграції у Visual Studio.

"ESP8266 GCC Toolchain" Макса Філіппова.

"Sming" - проект додавання Arduino сумісних бібліотек поверх стандартних бібліотек Espressif, але без середовища Ардуіно.

Мережева інфраструктура.

Типове застосування ESP8266 як апаратної основи Internet of Things найчастіше передбачає встановлення у будинках чи офісах. При цьому мережне підключення здійснюється до домашньої/офісної локальної мережі з виходом в інтернет через роутер. Користувач пристрою може контролювати його за допомогою планшета або комп'ютера через свою локальну мережу або віддалено через Інтернет.

В свою чергу, ESP32 – серія недорогих мікросхем з малим енергоспоживанням компанії Espressif Systems. Є системою на кристалі з інтегрованим контролерами радіозв'язку Wi-Fi, Bluetooth і Thread. У серіях ESP32 та ESP32-S використовуються процесорні ядра з архітектурою компанії Tensilica, а в серіях ESP32-C та ESP32-H – ядра з відкритою архітектурою RISC-V. У мікросхему інтегрований радіочастотний тракт: симетруючий трансформатор, вбудовані антенні комутатори, радіочастотні компоненти, малошумний підсилювач, підсилювач потужності, фільтри та модулі керування живленням. ESP32 створено та розроблено компанією, розташованою в Шанхаї, а виробляється компанією TSMC за техпроцесом 40 нм та 28 нм. Серія є наступником мікросхем ESP8266.

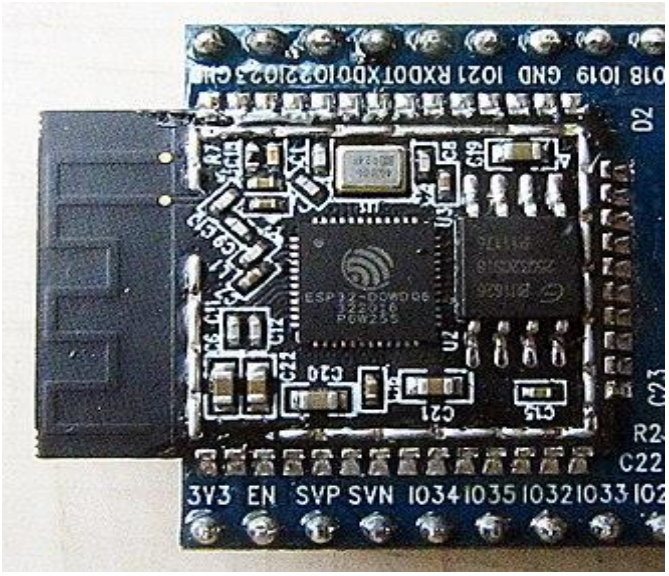

Рис. 2.11- Зовнішній вигляд мікросхеми ESP32

Особливості ESP32 [15]:

Серії ESP32 і ESP32-S включають:

– Мікроконтролер та управління.

– Tensilica Xtensa LX6 двоядерний (або одноядерний) 32-розрядний процесор, з тактовою частотою 160 або 240 МГц та продуктивністю до 600 DMIPS (Dhrystone MIPS).

– Співпроцесор із ультранизким енергоспоживанням.

– Пам’ять: 520 КБ пам’яті SRAM.

– Бездротовий зв’язок:

– Wi-Fi: 802.11 b/g/N.

– Bluetooth: v4.2 BR/EDR and BLE.

– Периферійні інтерфейси:

– 12-розрядний АЦП до 18 каналів.

– 2 × 8 біт ЦАП.

– 10 × портів для підключення ємнісних датчиків (що вимірюють ємність GPIO).

– Датчик температури відсутній. Інформація про нього видалена зі специфікації V2.2.

– 4 × SPI майстер-інтерфейсу (провідні пристрої).

– 2 × I²S майстер-інтерфейсу.

– 2 × I²C майстер-інтерфейсу.

– 3 × UART інтерфейсу.

– SD/SDIO/CE-ATA/MMC/ eMMC хост-контролер.

– SDIO/SPI слейв-контролери (відомі пристрої).

– Ethernet MAC interface з виділеним DMA та IEEE 1588 Precision Time Protocol support.

– CAN bus 2.0.

– ІЧ дистанційне керування (передавач/приймач, до 8 каналів).

– Можливість підключення двигунів та світлодіодів через ШІМ-вихід.

– Датчик холлу.

– Аналоговий підсилювач низького енергоспоживання.

– Безпека:

– Підтримуються всі функції безпеки стандарту IEEE 802.11, зокрема WFA, WPA/WPA2 та WAPI.

– Безпечне завантаження.

– Шифрування флеш-диску.

– 1024-бітний ключ до 768 біт для клієнтів.

– Криптографічне апаратне прискорення: AES, SHA-2, RSA, криптографії на основі еліптичних кривих (ECC), апаратний генератор випадкових чисел при включеному WiFi або Bluetooth, інакше використовується генератор псевдовипадкових чисел.

Управління живленням:

– Лінійний регулятор з низьким рівнем падіння напруги.

– Індивідуальне харчування для RTC.

– споживання 5-2,5 мкА у режимі «глибокий сон».

– Пробудження з переривання від GPIO, таймера, вимірювання АЦП, переривання ємнісного сенсорного датчика.

– Робоча напруга від 2,2 до 3,6 В.

Від -40 °C до + 125 °C робоча температура.

Максимальна швидкість передачі даних 150 Мбіт/с при 11n HT40, 72 Мбіт/с при 11n HT20, 54 Мбіт/с @ 11g, та 11 Мбіт/с при 11b.

Максимальна потужність передачі 19,5 дБм @ 11b, 16,5 дБм @ 11 г, 15,5 дБм @ 11n.

Мінімальна чутливість приймача-98 дБм.

Стійка пропускна здатність UDP 135 Мбіт/с.

ESP32 випускається в планарному корпусі (QFN) з 48 контактами по периметру та одним великим тепловідведенням по центру, що одночасно виконує функцію сигнальної землі.

ESP32-D0WDQ6 містить два малої потужності Xtensa® 32-біт LX6 мікропроцесорів. Внутрішня пам’ять включає:

448 КБ ПЗУ для завантаження та основних функцій.

520 Кб (8 КБ RTC швидка пам’ять у комплекті) on-chip SRAM для даних та інструкцій.

8 КБ SRAM в RTC, який називається RTC швидкої пам’яті та зберігання даних використовується для; це для доступу до нього з боку Головного процесора під час завантаження RTC з режиму глибокого сну.

8 КБ SRAM в RTC, який називається повільною пам’яттю RTC і може бути доступний з процесором під час режиму глибокого сну.

1 кбіт eFuse, з яких 256 біт використовуються для системи (MAC-адреса та конфігурація чіпа) та інші 768 біт зарезервовані для клієнтських програм, включаючи шифрування флеш-пам’яті та ідентифікатор чіпа.

ESP32 підтримує до чотирьох банків 16-Мб зовнішньої flash QSPI і SRAM з апаратним шифруванням на основі AES із захистом користувальницьких програм і даних.

ESP32 може отримати доступ до зовнішньої flash QSPI та SRAM через швидкісні канали.

До 16 Мб зовнішньої флеш-пам’яті зіставлені з кодовим простором ЦП, що підтримує 8, 16 і 32-біт доступу. Підтримується виконання коду.

До 8 Мб зовнішньої flash/SRAM карти пам’яті на ЦП простору даних, підтримка 8, 16 та 3 2-біт доступу. Читання даних підтримується на флеш-пам’яті та SRAM. Запис даних підтримується SRAM.

ESP32-WROVER інтегрує 4-16 Мб зовнішньої SPI flash. 4-мб SPI flash може бути картка пам’яті на процесор простір, що підтримують 8, 16 та 32 біт доступу.

Підтримується виконання коду.

Крім 4-16 МБ SPI flash, ESP32-WROVER також інтегрує 4-8 Мб PSRAM для більшого простору пам’яті.

Мікропрограма ESP32 Wi-Fi/BT може підтримувати лише кварцовий генератор 40 МГц.

[RTC та управління низьким споживанням].

За допомогою сучасних технологій керування живленням ESP32 може перемикатися між різними режимами живлення (див. таблицю нижче).

[Потужність режими/Power modes].

Active mode / Активний режим: чіп радіо увімкнено. Чіп може отримувати, передавати чи слухати.

Modem-sleep mode / Режим сну модему: ЦП працює і годинник налаштовується. Базова смуга Wi-Fi/Bluetooth I радіо вимкнено.

Light-sleep mode / Режим сну: ЦП припинено. Пам’ять RTC та периферійні пристрої RTC, а також ULPСопроцесор працює. Всі події пробудження (MAC, хост, таймер RTC або зовнішні переривання) будуть прокидатися до chip.

Deep-sleep mode / Режим глибокого сну: Тільки пам’ять RTC та периферійні пристрої RTC увімкнені. Wi-Fi та Bluetooth дані з’єднання зберігаються у пам’яті RTC. Співпроцесор ULP може працювати.

Hibernation mode / Режим глибокого сну: внутрішній 8 МГц осцилятор і со-процесор ULP відключені. RTC відновлення пам’яті вимкнено. Тільки один таймер RTC на повільному годиннику і деякі GPIOs RTC активні. Таймер RTC або GPIOs RTC можуть розбудити чіп у режимі сплячки.

[Сон/Sleep Patterns]

Association sleep pattern / Шаблон Association sleep: режим живлення перемикається між активним режимом, модемом та LightsleepРежим під час цього сну CPU, Wi-Fi, Bluetooth та радіо прокидаються на заздалегідь визначеному Інтервали для збереження з’єднання Wi-Fi/BT живими.

ULP sensor-monitored pattern / ULP датчик-контрольований шаблон: Головний процесор перебуває у режимі глибокого сну. Комбінований процесор ULPВимір датчиків і Пробудження основної системи на основі даних, зібраних з датчиків.

Модульні SMT-плати.

Модулі SMT-плати на основі ESP32 містять ESP32 SoC та призначені для легкого інтегрування в інші плати.

Вимірювані інвертовані F-антенні конструкції використовуються для трасування PCB антени на модулях, перерахованих нижче.

Крім флеш-пам’яті, деякі модулі включають псевдостатичну оперативну пам’ять (pSRAM)

2. Контролер Arduino (рис.2.12).

Arduino - це електронна платформа з відкритим вихідним кодом, заснована на простому у використанні апаратному та програмному забезпеченні (рис.2.12).

Рис. 2. 12- Arduino IDE із прикладом простої програми

Класичні Arduino та Arduino-сумісні плати спроектовані для монтажу в стопки через штирьові роз'єми. Таким чином, базову мікропроцесорну плату доповнюють необхідною периферією і зовнішніми підключеннями [20].

Існують плати Uno, Pro, Leonardo, Mega 2560, Due та плати, наприклад Zero, з розширеним набором штирьових роз'ємів для них. Плати розширення стандартної довжини можуть встановлюватися і розширені процесорні плати (рис. 2.13).

Arduino приймає навколишнє середовище, отримуючи дані від безлічі датчиків, і впливає на навколишнє середовище, керуючи освітленням, двигунами та іншими виконавчими механізмами [14].

Основне призначення плати Arduino - взаємодія з сенсорами і пристроями, тому Arduino відмінно підходить для апаратних проектів, де потрібно просто реагувати на різні сигнали сенсорів і ручне введення. Може здатися, що в цьому немає нічого

особливого, проте на ділі Arduino - складна вивірена система, значно полегшує управління пристроями.

Рис. 2.13 - Класичний конструктив Arduino з платами розширення

Вона відмінно підходить саме для організації взаємодії інших пристроїв і виконавчих механізмів, де повновага операційна система просто не потрібно, так як мова йде просто про отримання сигналів з сенсорів і реагуванні на них (рис. 2.14).

Апаратне забезпечення типової плати Arduino засноване на мікроконтролері Microchip AVR із серії megaAVR, такому як ATmega328 [14].

Відхилення від цього можна знайти, наприклад, у платах Arduino Due (32-бітний процесор Atmel SAM3X8E Arm Cortex-M3), Yun, Tre, Gemma та Zero, де використовуються інші мікроконтролери Atmel.

Плати Arduino Yún і Tre, які мають потужніший мікропроцесор на додаток до мікроконтролера, також є особливими.

Рис. 2.14 - Arduino UNO R3 - версія SMD з інтерфейсом USB та мікроконтролером ATmega 328.

Також є варіанти з напругою живлення 3,3 і варіанти з іншою тактовою частотою. Ряд інших мікроконтролерів, таких як ESP8266, ESP32, STM32 або MSP430 також можна запрограмувати через Arduino IDE через розширення.

Arduino поставляється з власним інтегрованим середовищем розробки (IDE), заснованим на Wiring IDE . Ця програма Java доступна безкоштовно для поширених операційних систем Windows, Linux і macOS . Він заснований на Processing IDE, середовищі розробки, що спеціалізується на графіку, моделюванні та анімації. Arduino IDE поставляється з редактором коду і інтегрує gcc як компілятор. Крім того, бібліотека avr-gcc та інші бібліотеки Arduino («бібліотеки»), які значно спрощують програмування на C та C++.

У даній модифікації платформи Arduino Uno, на платформі розташовані 11 контактів. Але, на відміну від оригіналу, не всі з них можуть бути використані для цифрового введення і виведення. Всі вони працюють з напругою 3,3 В, і розраховані на струм до 40 мА. Також кожен контакт має вбудований, але відключений за замовчуванням резистор на 20 - 50 кОм.

ESP8266 - мікроконтролер китайського виробника Espressif з інтерфейсом Wi-Fi. Крім Wi-Fi мікроконтролер відрізняється важливою для даного проекту можливістю виконувати програми з зовнішньої flash-пам'яті з інтерфейсом SPI. Цей чіп вбудований в плату Wemos D1 і використовується для взаємодії з Wi-Fi з Web сервером.

Софтверна частина розробки СДК включає в себе:

1. Android software development kit (Android SDK) – набір засобів розробки програмного забезпечення який дозволяє створювати додатки для певної апаратної платформи , комп'ютерної системи , мобільних та вбудованих операційних систем та середовищ виконання програмних додатків. В даному випадку технологія розробки програмного забезпечення для мобільних платформ Android SDK – це універсальний засіб розробки мобільних додатків та допоміжних компонентів для мобільної операційної системи Android , даний пакет надає розробникам програмного забезпечення функціональні можливості завдяки яким можна запускати тестування , налагодження отриманих в процесі розробки артефактів в режимі сумісності з різними видами операційної системи Android та спостерігати результат роботи в режимі реального часу. Даний набір програмних засобів підтримує більшу частину мобільних пристроїв серед яких є:

− мобільні телефони;

− портативні планшетні комп'ютери;

− автомобілі з вбудованими бортовими комп'ютерами , що працюють під керуванням операційної системи Android;

− телевізори з підтримкою «інтелектуального функціоналу роботи»;

− наручні годиники;

− та інші малогабаритні технічні пристрої;

2. Flask - мікрофреймворк для вебдодатків, створений з використанням Python. Його основу складає інструментарій Werkzeug та рушій шаблонів Jinja2.

Поширюється відповідно до умов ліцензії BSD. Flask називається мікрофреймворком, оскільки він не вимагає спеціальних засобів чи бібліотек.

3. У ньому відсутній рівень абстракції для роботи з базою даних, перевірки форм або інші компоненти, які надають широковживані функції за допомогою сторонніх бібліотек.

Властивості мікрофреймворку Flask:

- Містить сервер для розробки та відлагоджувач.
- Вбудована підтримка юніт-тестів.
- Управління запитами RESTful.
- Використовує шаблони Jinja2.
- Має підтримку безпечних куків (сесії на стороні клієнта).
- 100% відповідність WSGI 1.0.
- Підтримка Unicode.
- Докладна документація.
- Сумісність з Google App Engine.
- Наявність розширень для забезпечення бажаної поведінки

4. Бібліотека Servo.h.

Ця бібліотека функцій для контролера Arduino надає набір функцій для управління сервоприводами. Стандартні сервоприводи дозволяють повертати привод на певний кут від 0 до 180 градусів. Деякі сервоприводи дозволяють здійснювати повні оберти заданої швидкості.

У загальному випадку сервопривід підключається трьома проводами: живлення, земля та сигнальний.

Зазвичай живлення – червоний провід і може бути підключений до виведення +5V на платі Arduino. Чорний дріт земля підключається до GND виводу Arduino, сигнальний, зазвичай жовтий, провід підключається до цифрового виводу котроллера Arduino.

Слід зазначити, що потужні сервоприводи можуть створювати велике навантаження, у цьому випадку він повинен бути окремо (не через вихід +5V Arduino). Теж правильне для випадку підключення відразу декількох сервоприводів. Переконайтеся, що привід та контролер підключено до спільної землі.

3. СУБД – NoSQL.

InterSystems Caché (/ k æ ʃ eɪ / kashay) - Комерційна система управління операційними базами даних від InterSystems , що використовується для розробки програмних додатків для управління охороною здоров'я, банківськими та фінансовими послугами, урядом та іншими секторами.

Програмне забезпечення замовника може використовувати базу даних з об'єктним та SQL-кодом.

Caché також дозволяє розробникам безпосередньо маніпулювати базовими структурами даних: ієрархічними масивами, відомими як M-технологія.

Всередині Caché зберігає дані у багатовимірних масивах, здатних зберігати ієрархічно структуровані дані. Це ті самі «глобальні» структури даних, які використовуються мовою програмування MUMPS, які вплинули на дизайн Caché, і аналогічні тим, що використовуються в системах MultiValue (також відомих як PICK).

Технологія Caché Server Pages (CSP) дозволяє створювати веб-застосунки на основі тегів, які генерують динамічні веб-сторінки, зазвичай використовуючи дані з бази даних Caché. Caché також включає InterSystems Zen, реалізацію AJAX, яка дозволяє розробляти багатофункціональні веб-програми на основі компонентів.

Inter Systems Caché надає найважливішим додаткам наших клієнтів можливість зберігати, використовувати та аналізувати транзакційні та історичні дані одночасно у будь-яких необхідних формах.

Високошвидкісний SQL працює послідовно та безперешкодно у всіх моделях даних [9].

У міру збільшення вимог до пропускної спроможності та обсягу даних Caché підтримує незмінно високу продуктивність. У Caché дані можна моделювати та зберігати у вигляді таблиць, об'єктів або багатовимірних масивів (ієрархій). Різні моделі можуть безперешкодно отримувати доступ до даних без необхідності зіставлення між моделями, що знижує продуктивність. Вбудована підтримка об'єктів динамічних даних (таких як XML та JSON) забезпечує простоту взаємодії та швидку розробку веб-додатків [21].

SQL – це мова спілкування для запитів до даних Caché у всіх моделях даних.

Завдяки своїй ефективній архітектурі Caché забезпечує більш високу продуктивність SQL ніж інші технології баз даних.

Він підтримує традиційні індекси, а також індекси з бітовою карткою та бітовими зрізами, які можна використовувати з транзакційними даними у реальному часі.

Дзеркала бази даних Caché не вимагають великих інвестицій у обладнання, підтримку, ліцензії на операційну систему чи сховище. Крім того, дзеркала баз даних Caché легко налаштовуються та обслуговуються, тому витрати на адміністрування зведені до мінімуму.

Рішення NoSQL відрізняються як проектуванням з урахуванням масштабування. Іншими характерними рисами NoSQL-рішень є :

– Застосування різних типів сховищ.
– Можливість розробки бази даних без завдання схеми.
– Лінійна масштабованість (додавання процесорів збільшує продуктивність).
– Інноваційність: «не лише SQL» відкриває багато можливостей для зберігання та обробки даних.

2.4. Висновки до розділу

Підсумовуючи другий розділ, можемо зробити такі висновки:

1. Розглянуто існуючі СДК.
2. Зясовано засоби програмування сучасних МК.
3. Визначено підходи до проектування СДК ЗВП та алгоритм її створення.

РОЗДІЛ 3.
ПРАКТИЧНА РЕАЛІЗАЦІЯ СИСТЕМИ КЕРУВАННЯ ЗВУКОВІДТВОРЮЮЧИМИ ПРИСТРОЯМИ НА БАЗІ МІКРОКОНТРОЛЕРА ESP-8266/ESP32

3.1. Обгрунтування вибору функційної парадигми програмування

У сучасній індустрії парадигма програмування дуже часто визначається набором інструментів програміста, а саме, мовою програмування і використовуваними бібліотеками, тому парадигм за останні роки виникло дуже багато, їх розподіл зображено на рис. 3.1.

Рис. 3.1 - Розподіл парадигм програмування

Парадигма програмування визначає те, в яких термінах програміст описує логіку програми. Наприклад, в імперативному програмуванні програма описується як послідовність дій, а в ФП представляється у вигляді вираження і безлічі визначень функцій.

Функційне програмування - це парадигма програмування, в якій програми створюються шляхом застосування та складання функцій. Це парадигма декларативного програмування , де визначення функцій є дерева виразів , які відображають значення інші значення, а чи не послідовність імперативних операторів , які оновлюють поточний стан програми.

У ФП функції розглядаються як громадяни першого класу, що означає, що вони можуть бути прив’язані до імен (включаючи локальні ідентифікатори), передаватися як аргументи та повертатися з інших функцій, як і будь-який інший тип даних. Це дозволяє писати програми у декларативному та складовому стилі, де невеликі функції об’єднуються модульним чином.

ФП іноді розглядається як синонім суто функціонального програмування, підмножини функціонального програмування, яке розглядає всі функції як детерміновані математичні функції чи чисті функції. Коли чиста функція викликається з деякими заданими аргументами, вона завжди буде повертати той самий результат, і на неї не можуть вплинути будь-які стани, що змінюються, або інші побічні ефекти.

Це контрастує з нечистими процедурами, поширеними в імперативному програмуванні, які можуть мати побічні ефекти (такі як зміна стану програми або отримання даних від користувача). Прихильники суто ФП стверджують, що, обмежуючи побічні ефекти, програми можуть мати менше помилок, їх легше налагоджувати та тестувати, і вони більше підходять для формальної перевірки.

Ряд концепцій і парадигм специфічний для функціонального програмування і взагалі далекий від імперативного програмування (включаючи об’єктно-орієнтоване програмування). Однак мови програмування часто обслуговують кілька парадигм програмування, тому програмісти, які використовують «в основному імперативні» мови, могли використовувати деякі з цих концепцій.

Функції вищого порядку - це функції, які можуть або приймати інші функції як аргументи, або повертати їх як результати. У обчисленні прикладом функції вищого порядку є диференціальний оператор {\Displaystyle д/дх}д/дх, який повертає похідну функції{\Displaystyle е}ф.

Функції вищого порядку тісно пов’язані з функціями першого класу у тому сенсі, що і функції вищого порядку, і функції першого класу дозволяють використовувати функції як аргументи та результати інших функцій.

Функції вищого порядку допускають часткове застосування чи каррирование - метод, у якому функція застосовується до її аргументів по одному, у своїй кожен додаток повертає нову функцію, яка приймає наступний аргумент. Це дозволяє програмісту коротко виразити, наприклад, функцію наступника як оператор додавання, частково застосований до натурального числа один.

Чисті функції (або вирази) не мають побічних ефектів (пам’ять або введення-виведення). Це означає, що чисті функції мають кілька корисних властивостей, багато з яких можна використовувати для оптимізації коду.

Якщо результат чистого виразу не використовується, його можна видалити, не торкаючись інших виразів.

Якщо чиста функція викликається з аргументами, які не викликають побічних ефектів, результат є постійним по відношенню до цього списку аргументів (іноді це називається прозорістю або ідемпотентністю), тобто повторний виклик чистої функції з тими ж аргументами повертає той же результат. (Це може включити оптимізацію кешування, таку як запам’ятовування.)

Якщо між двома чистими виразами немає залежності даних, їх порядок може бути зворотним, або вони можуть виконуватися паралельно і вони не можуть заважати один одному (іншими словами, обчислення будь-якого чистого виразу потокобезпечно).

Якщо вся мова не допускає побічних ефектів, можна використовувати будь-яку стратегію оцінки; це дає компілятор свободу переупорядковувати або комбінувати оцінку виразів у програмі (наприклад, за допомогою обезліснення).

Хоча більшість компіляторів для імперативних мов програмування виявляють чисті функції та виконують усунення загальних виразів для викликів чистих функцій, вони не завжди можуть це зробити для попередньо скомпільованих бібліотек, які зазвичай не надають цю інформацію, що запобігає оптимізації, що включає ці зовнішні функції. Деякі компілятори, такі як gcc , додають додаткові ключові слова для програміста, щоб помітити зовнішні функції як чисті, щоб включити таку оптимізацію. Фортран 95 також дозволяє позначати функції як чисті. [57] У C++11 додано constexpr ключове слово з аналогічною семантикою.

Ітерація (зациклювання) у функціональних мовах зазвичай виконується за допомогою рекурсії. Рекурсивні функції викликають самі себе, дозволяючи повторювати операцію до тих пір, поки не буде досягнуто базового випадку. Загалом, рекурсія вимагає підтримки стека, який споживає простір у лінійній пропорції до глибини рекурсії. Це може зробити використання рекурсії надмірно дорогим замість імперативних циклів. Однак особлива форма рекурсії, відома як хвостова рекурсія, може бути розпізнана та оптимізована компілятором у той же код, який використовується для реалізації ітерації в імперативних мовах.

Оптимізація хвостової рекурсії може бути реалізована шляхом перетворення програми в стиль передачі продовження під час компіляції, серед інших підходів.

Функціональні мови програмування зазвичай менш ефективні використання ЦП і пам’яті, ніж імперативні мови, такі як C і Pascal . Це пов’язано з тим, що деякі структури даних, що змінюються, такі як масиви, мають дуже просту реалізацію з використанням існуючого обладнання. До плоских масивів можна отримати дуже ефективний доступ за допомогою процесорів із глибокою конвеєризацією, ефективною попередньою вибіркою через кеші (без складної погоні за вказівниками).) або обробляється за допомогою SIMD-інструкцій. Також непросто створити їх так само ефективні незмінні аналоги загального призначення.

Для чисто функціональних мов найгірше уповільнення є логарифмічним за кількістю використовуваних осередків пам’яті, тому що пам’ять, що змінюється, може

бути представлена чисто функціональною структурою даних з логарифмічним часом доступу (такий як збалансоване дерево).

Однак такі уповільнення не є універсальними.

Для програм, які виконують інтенсивні числові обчислення, функціональні мови, такі як OCaml і Clean лише трохи повільніше, ніж C, згідно The Computer Language Benchmarks Game.

Для програм, що обробляють великі матриці та багатовимірні бази даних, функціональні мови масивів (такі як J і K) були розроблені з оптимізацією швидкості.

3.2. Розробка структурної схеми та архітектури прототипу

При дослідженні сучасних програмних додатків для дистанційного керування ЗВП можна дійти висновку, що програмний модуль повинен бути реалізований у вигляді мобільного додатку для смартфону під керуванням мобільної операційної системи Android з задіянням інтерфейсу дистанційної передачі файлів по каналу радіозв'язку Bluetooth та включати в себе наступний функціонал:

– набір функцій та методів для реалізації головного вікна програми та користувацького інтерфейсу та точки запуску додатку – MainActivity.java , та активності з реалізацією інтерфейсу керування activity_main.xml;

– файл реалізації інтерфейсу дистанційної взаємодії по радіоканалу Bluetooth мобільного додатку з самим керованим пристроем AdapterList.java

– файли реалізації користувацького меню пошуку інших пристроїв device_item.xml та device_view_list.xml;

– файл основних налаштувань додатку та установок взаємодії з мобільною операційною системою Android.

Структуру компонентів системи дистанційного керування ЗВП, а також її функціональну схему відображено на рис. 3.2-3.3.

Апаратний модуль має таке функціональне забезпечення:

1. Ввімкнення та вимкнення пристрою;

2. Контроль рівню температури;

3. SMS повідомлення про закінчення роботи.

Програмний модуль має таке функціональне забезпечення:

1. Підключення до апаратного модулю;

2. Дистанційне керування пристроєм.

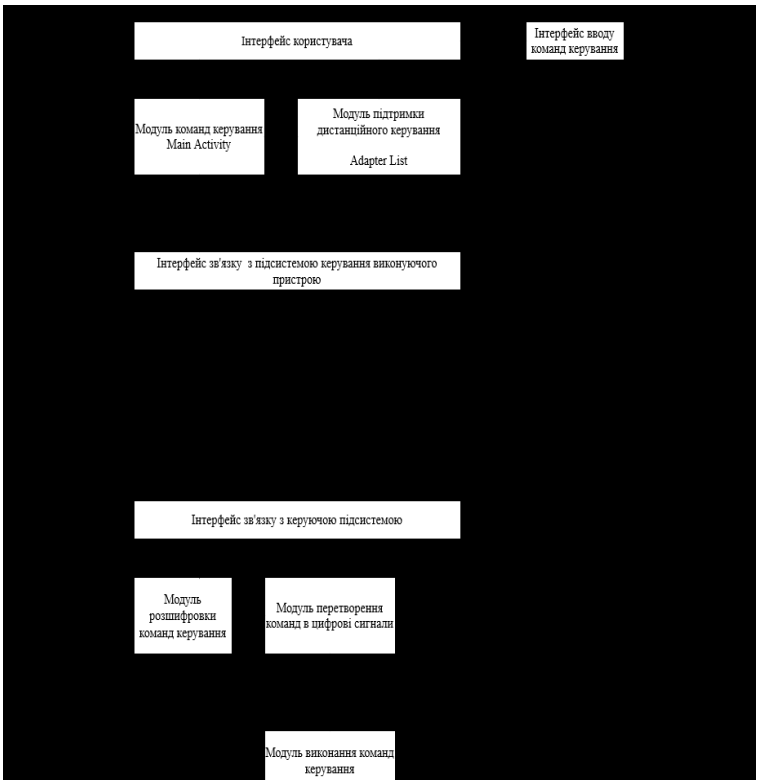

Рис. 3.2 - Структурна схема системи дистанційного керування ЗВП

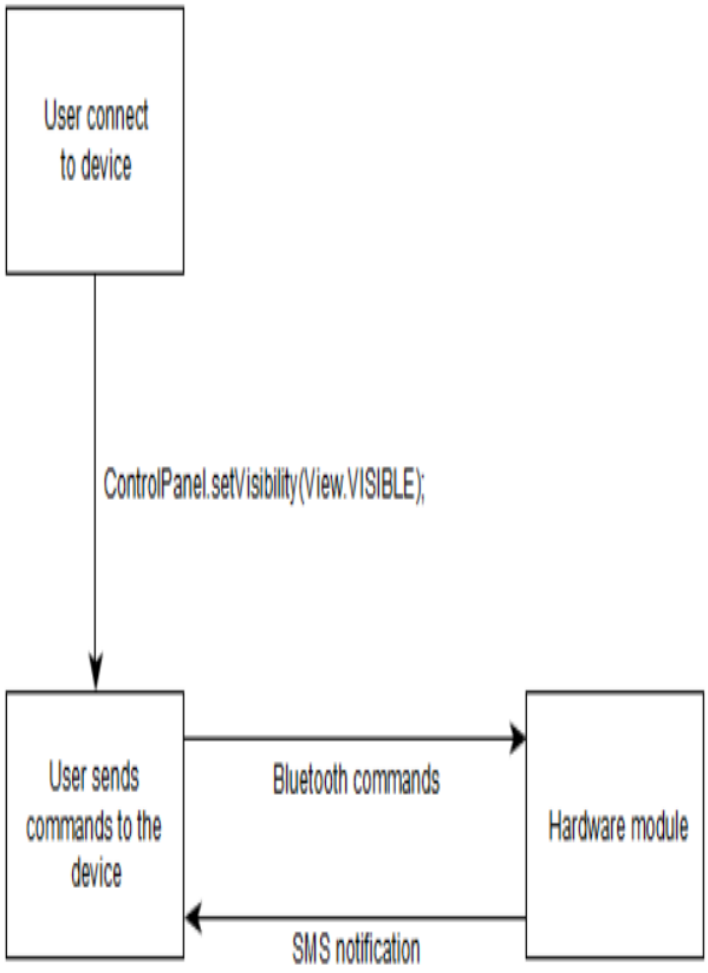

Рис. 3.3 - Функціональна схема СДК ЗВП

Зв’язок між модулями реалізовано за допомогою технології Bluetooth та стандарту GSM.

Мобільний додаток працює на пристроях під управлінням системи Android. Мінімальна версія системи – 4.1. Остання версія, на якій проводилося тестування додатку – Android 11. Відповідно до інформації від розробника Android компанії Google, це покриває приблизно 99,8% активних пристроїв під управлінням даної операційної системи, або іншими словами, пристрої випущені, починаючи з 2012-2013 років.

Під час проектування системи складено діаграму послідовностей (рис. 3.4). На діаграмі послідовностей відображено послідовність дій , що відбуваються під час використання системи.

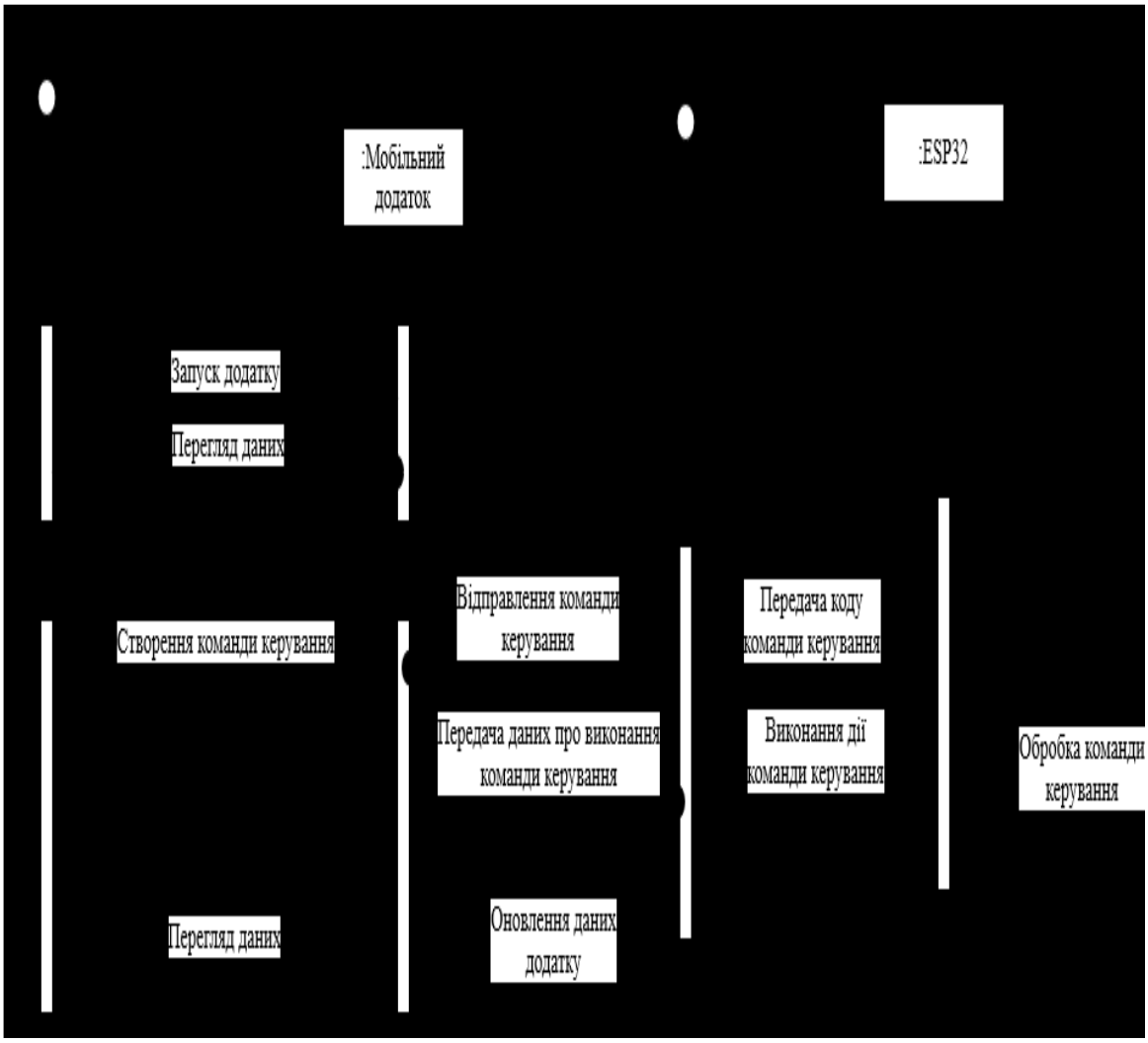

Рис. 3.4 - Діаграма послідовностей для демонстрації процесу роботи системи дистанційного керування ЗВП на базі ESP-8266/ESP32

3.3. Впровадження програми, її тестування та валідація

Під час проектування системи складено діаграму послідовностей (рис. 3.5.). На діаграмі послідовностей відображено послідовність дій, що відбуваються під час використання системи.

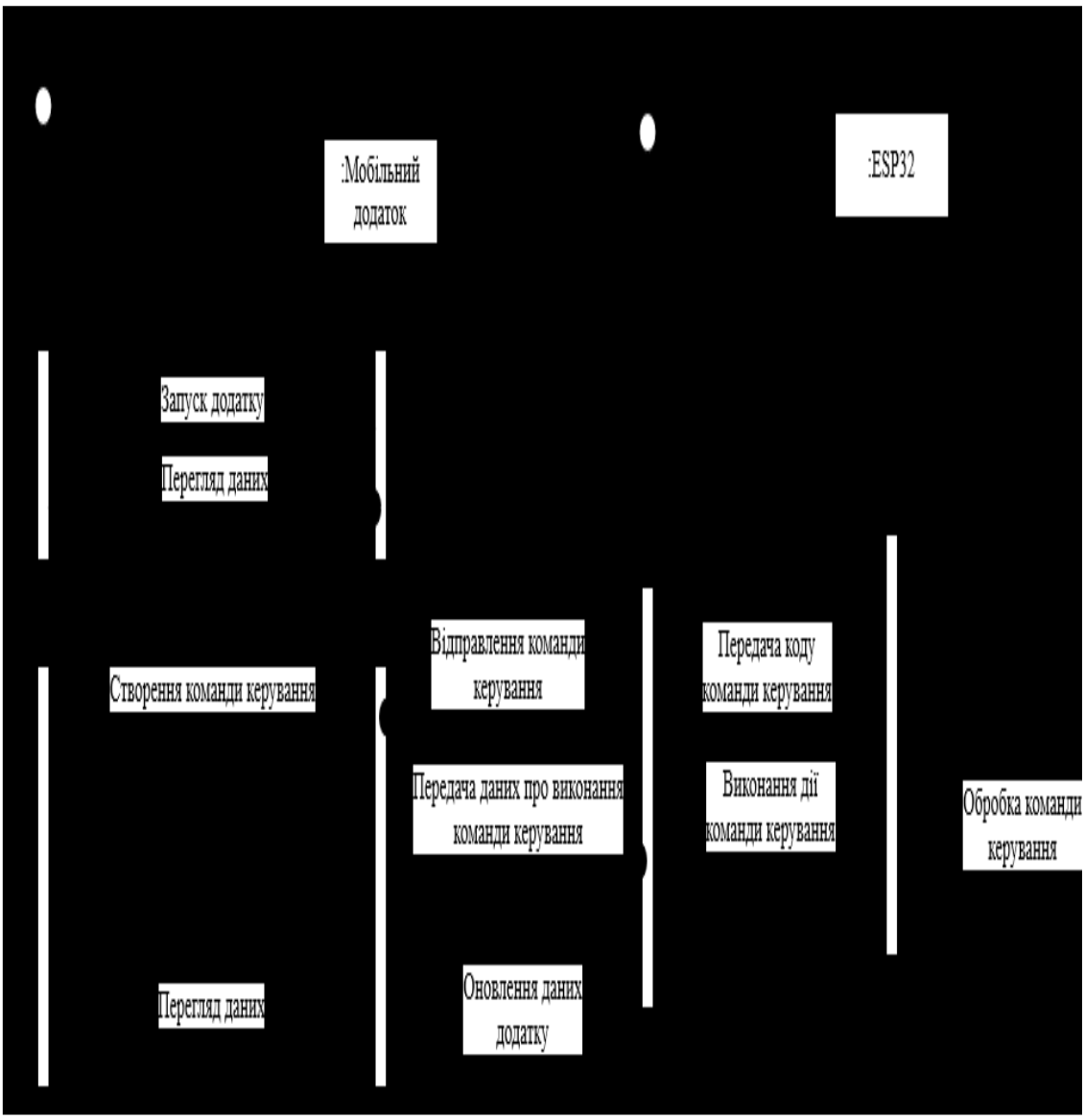

Рис. 3.5 - Діаграма послідовностей для демонстрації процесу роботи системи дистанційного керування ЗВП на базі ESP-8266/ESP32

На рис. 3.6. зображено загальну схему реалізації алгоритму роботи додатку, яка містить всі етапи процесу дистанційного керування ЗВП при виконанні процесу керування.

Рис. 3.6 - Загальна схема реалізації алгоритму дистанційного керування ЗВП

Діаграма класів розробленого додатку СДК ЗВП, яка наочно демонструє всі необхідні елементи для створення мобільного додатку, зображена на рис. 3.7.

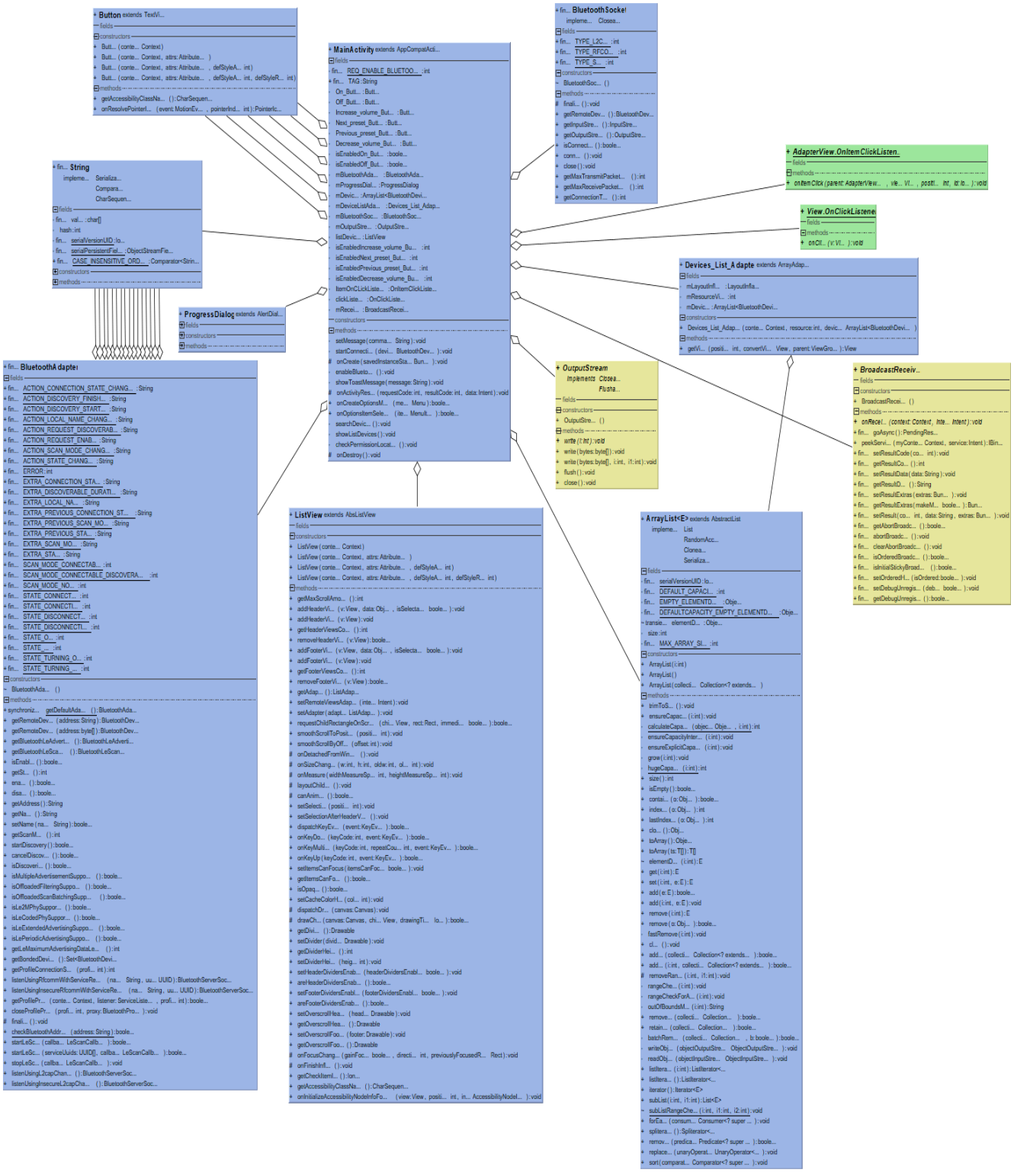

Рис. 3.7 - Діаграма класів мобільного додатку СДК ЗВ

Представимо програмний код керування функціями ЗВП в розробленій СДК на базі МК ESP-8266/ESP32.

```
private void setMessage (String command)
{
byte[] buffer = command.getBytes();
if (mOutputStream != null)
{
try
{
mOutputStream.write(buffer);
mOutputStream.flush();
}
catch (IOException e)
{
showToastMessage("Command sending error!");
e.printStackTrace();
}
}
}
private View.OnClickListener clickListener = new View.OnClickListener() {
@Override
public void onClick(View view)
{
String command = null;
if (view.equals(On_Button))
{
isEnabledOn_Button = !isEnabledOn_Button;
command = "77";
Log.d(TAG, "onClick: isEnabledOn_Button = " + isEnabledOn_Button);
}
```

```
if (view.equals(Off_Button))
{
isEnabledOff_Button = !isEnabledOff_Button;
command = "77";
Log.d(TAG, "onClick: isEnabledOff_Button = " + isEnabledOff_Button);
}
if (view.equals(Increase_volume_Button))
{
isEnabledIncrease_volume_Button = isEnabledIncrease_volume_Button + 1;
command = "22";
Log.d(TAG, "onClick: isEnabled_Increase_volume_Button = " +
isEnabledIncrease_volume_Button);
}
if (view.equals(Next_preset_Button))
{
isEnabledNext_preset_Button = isEnabledNext_preset_Button + 1;
command = "44";
Log.d(TAG, "onClick: isEnabledNext_preset_Button = " +
isEnabledNext_preset_Button);
}
if (view.equals(Previous_preset_Button)) {
isEnabledPrevious_preset_Button = isEnabledPrevious_preset_Button + 1;
command = "55";
Log.d(TAG, "onClick: isEnabledPrevious_preset_Button = " +
isEnabledPrevious_preset_Button);
}
if (view.equals(Decrease_volume_Button))
{
isEnabledDecrease_volume_Button = isEnabledDecrease_volume_Button + 1;
```

```
                command = "33";
        Log.d(TAG, "onClick: isEnabledDecrease_volume_Button = " +
                isEnabledDecrease_volume_Button);
                }
                setMessage(command);
                }
                };
```

3.4. Висновки за розділом 3

Підсумовуючи третій розділ, можемо зробити такі висновки:

1. Обгрунтовано вибір функційної парадигми програмування.
2. Розроблено структурну схему та архітектуру прототипу.
2. Проведено впровадження програми.

ВИСНОВКИ

Підсумовуючи загальний зміст дослідження, можемо зробити такі висновки:

1. Визначено, що наразі важливе місце в автоматизаційних процесах належить системам дистанційного керування (СДК), які дозволяють за допомогою засобів штучного інтелекту здійснювати контроль за різноманітними пристроями. Такими засобами програмування є мікроконтролери, з яких складається велика кількість сучасної побутової та виробничо-промислової електронної техніки. Звичайний мікроконтролер представляє собою мікросхему для керування електронними пристроями будь-якої архітектури та складності виконання. За своєю архітектурою мікроконтролер - це однокристальний цифровий обчислювальний пристрій , що може мати на одному кристалі функціонал процесора – обчислювального модуля , різних допоміжних переферійних пристроїв, наприклад, таких як універсальні цифрові порти , вводу-виводу інформації, різні інтерфейси комунікації такі як UART, USB, Ethernet , різні запам’ятовуючі пристрої. Слід зауважити, що такого набору компонентів цілком достатньо для виконання задач простого – середнього рівня складності. Отже, завдяки використанню сучасних мікроконтролерів користувачеві доступні функції дистанційного керування різною виконавчою технікою через використання інтерфейсів керування таких як радіоканали, Wi-fi , Bluetooth та ZigBee.

2. З’ясовано, що мікроконтролер (МК) являє собою невеликий комп’ютер на одному чіпі інтегральної схеми (ІС) НВІС метал-оксид-напівпровідник (МОП). Мікроконтролер містить один або кілька процесорів (процесорних ядер), а також пам’ять та програмовані периферійні пристрої вводу/виводу. Програмна пам’ять у вигляді фероелектричної ОЗУ, флеш-пам’яті NOR або OTP ROM також часто включається до мікросхеми, а також невеликий обсяг ОЗУ. Мікроконтролери призначені для додатків, що вбудовуються, на відміну від мікропроцесорів, що використовуються в персональних комп’ютерах або інших додатках загального призначення, що складаються з різних дискретних мікросхем. У сучасній термінології мікроконтролер нагадує систему на кристалі (SoC), але менш складний. SoC може

підключати зовнішні мікросхеми мікроконтролера як компоненти материнської плати, але SoC зазвичай поєднує передові периферійні пристрої, такі як графічний процесор (GPU) і контролер інтерфейсу Wi-Fi, як внутрішні схеми блоку мікроконтролера. Мікроконтролери використовуються в продуктах і пристроях з автоматичним керуванням, таких як системи керування автомобільними двигунами, медичні пристрої, що імплантуються, пульти дистанційного керування, офісні машини, побутова техніка, електроінструменти, іграшки та інші вбудовувані системи.

3. Окреслено, що дистанційне керування (ДК) – це передача керуючого впливу або сигналу чи імпульсу від керуючої системи , чи конкретної особи – оператора до об’єкта керування, що знаходиться на певній відстані або якщо при керуванні певною системою чи технічним пристроєм немає можливості передавати сигнал напряму, якщо об’єкт управління змінює своє розташування у просторі. Зазвичай ДК складається із передавача – пульту дистанційного керування та приймача який надалі може віддавати керуючі сигнали всій конкретній системі , та виконавчих механізмів, та інших складових систему керування якою відбувається. Системи ДК мають можливість використовувати різні канали зв’язку. Канал зв’язку для виконання механічного керування – це канал зв’язку, який використовується у випадку коли об’єкти віддалені один від одного на невелику відстань або якщо треба забезпечити миттєву , не спотворену різними видами впливу реакцію систему над якою здійснюється процес керування.

4. Після аналізу наявних у доступі мобільних додатків для СДК ЗВП можна дійти висновку, що питання дистанційного керування пристроями є актуальним для безлічі користувачів, але є проблеми у знаходженні зручних та зрозумілих додатків для певних груп користувачів та досі існують конфлікти сумісності додатків керування з деякими конкретними моделями різноманітних пристроїв.

5. Проаналізовано, що ESP8266 – це мікроконтролер китайського виробника Espressif Systems з інтерфейсом Wi-Fi. Крім Wi-Fi, мікроконтролер відрізняється відсутністю флеш-пам’яті в SoC, програми користувача виконуються із зовнішньої флеш-пам’яті з інтерфейсом SPI.

6. При дослідженні сучасних програмних додатків для дистанційного керування ЗВП можна дійти висновку, що програмний модуль повинен бути реалізований у вигляді мобільного додатку для смартфону під керуванням мобільної операційної системи Android з задіянням інтерфейсу дистанційної передачі файлів по каналу радіозв'язку Bluetooth та включати в себе наступний функціонал:

– набір функцій та методів для реалізації головного вікна програми та користувацького інтерфейсу та точки запуску додатку – MainActivity.java , та активності з реалізацією інтерфейсу керування activity_main.xml;

– файл реалізації інтерфейсу дистанційної взаємодії по радіоканалу Bluetooth мобільного додатку з самим керованим пристроем AdapterList.java

– файли реалізації користувацького меню пошуку інших пристроїв device_item.xml та device_view_list.xml;

– файл основних налаштувань додатку та установок взаємодії з мобільною операційною системою Android.

7. Доведено, що апаратний модуль СДК на базі МК ESP8266 має таке функціональне забезпечення:

1. Ввімкнення та вимкнення пристрою;

2. Контроль рівню температури;

3. SMS повідомлення про закінчення роботи.

Програмний модуль має таке функціональне забезпечення:

1. Підключення до апаратного модулю;

2. Дистанційне керування пристроєм.

8. В результаті розроблено програмний модуль для функціонування на мобільній операційній системі Android, що буде реалізовувати своїм функціоналом дистанційне керування ЗВП на мікропроцесорній базі мікроконтролеру ESP32, який, в свою чергу, буде виконувати функції радіоприймача, а керування виконується через взаємодію мобільного пристрою (смартфону) та мікропроцесору шляхом реалізації бездротового інтерфейсу керування за використанням технології Bluetooth. Програмне

забезпечення реалізовано у вигляді мобільного додатку створеного на функційній мові програмування високого рівня Java та інструментальними засобами розробки Android studio, призначенням якого є керування даним ЗВП.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Александреску О. В. Сучасне проектування на C++. / О. В. Александреску – Вильямс, 2014. – 336 с.

2. Ардатський С. М., Бартунов О. С. Управління доступом в складних інформаційних системах. – Освітні портали. Випуск 1. - 2005. -187 с.

3. Басманов А. С., Широков Ю. Ф. Мікропроцесори і одно кристальні мікро ЕОМ: Номенклатура і функціональні можливості/Под ред. В. Г. Домрачева. М .: Вища школа, 1988. - 127 с.

4. Бодрова Г. О., Логвін В. І. Позиціонування та взаємодія в бездротових сенсорних мережах / Г. О. Бодрова, В. І. Логвін // Молодий вчений. – 2015. – № 6. – С. 129-132.

5. Вислоух С. П. Інформаційні технології в задачах технологічної підготовки приладо- та машинобудівного виробництва : монографія / С. П. Вислоух. – К. : НТУУ «КПІ», 2011. – 488 с.

6. Ворожко В. П., Корченко О. Г. Захист інформації з обмеженим доступом. Збірник нормативних документів. – К.: КУЦА, 1999. – 283 с

7. Вуд А. Мікропроцесори в питаннях і відповідях / Пер. з англ. М .: Вища школа, 1985. - 185 с.

8. Гайкович В. Ю., Єршов Д. В. Основи безпеки інформационих технологій. М.: МІФІ, 1995. – 285с.

9. Глушков В. М., Амосов М. М., Артеменко І. О. Енциклопедія кібернетики. Том 2. Київ, - 1974. – С. 33-54.

10. Голощапов О. Google Android. Программування для мобільних пристроїв , 2011. - 438 с.

11. Дейч А. М. Методи ідентифікації динамічних об’єктів. – М: Енергія, 1979 – 240 с.

12. Джошуа, Блох Java. Ефективне програмування / Блох Джошуа. - М.: ЛОРІ, 2014. - 292 с.

13. Інтернет-ресурс. АСУ. – Режим доступу: https://uk.wikipedia.org/wiki.

14. Інтернет-ресурс. Arduino. Вікіпедія. – Режим доступу: https://de.wikipedia.org/wiki/Arduino_(Plattform).

15. Інтернет-ресурс. ESP32. – Режим доступу: https://www.espressif.com/en/products/socs/esp32

16. Інтернет-ресурс. RISC. – Режим доступу: https://ru.wikipedia.org/wiki/ RISC.

17. Інтернет-ресурс. Функціональне програмування. – Режим доступу: https://en.wikipedia.org/wiki/Functional_programming.

18. Квашнін В. О. Методологія програмування мікроконтролерів Stm 32 F4 Discovery і практичного їх застосування для вирішення наукових та інженерних задач / В. О. Квашнін, А. В. Бабаш, В. В. Квашнін // Сучасна освіта – доступність, якість, визнання : збірник наукових. – Краматорськ : ДДМА, 2016. – 209 с.

19. Ліпунцов Ю. П. Управління процесами. М: Компанія АйТф, 2003. – С. 33-42.

20. Лотов О. В., Поспєлова І. І. Багатокритеріальні задачі прийняття рішень: навч.посіб. М.: МАКС Прес, 2008. – С. 77-89.

21. Офіційний сайт Arduino. Інтернет-ресурс.– Режим доступу: https://www.arduino.cc.

22. Поспєлов Г. С Штучний інтелект - основа нової системи автоматизації діяльності організації. // [Електронний ресурс] - Режим доступу: http://www.in-line.ru/solutions/business_appl.

23. Самохвалов Ю. Я., Темпіков В. О., Хорошко В. О. Організаційно-технічне забезпечення захисту інформації: Навчальний посібник / За ред. проф. Хорошка В.О. – К.: НАУ, 2002. – С. 207.

24. Соколов В. Ю. Інформаційні системи і технології: Навчальний посібник – Київ ДУІКТ. - 2010. – С. 33-49.

25. Сташин В. В. Проектування цифрових приладів на однокристальних мікроконтролерах / Сташин В.В. - М.:Енергоатомвидав, 2010. – 224 с.

26. Уокерлі Дж. Архітектура та програмування мікро ЕОМ: У 2-х кн./Пер. з англ. М .: Світ, 1984. - Кн. 2. - 359 с.

27. Хосе М. Ангуло. Мікропроцесори: Архітектура, програмування та проектування систем. Тбілісі: Ганатлеба, 1989. – С. 45-54.

28. Хоровіц П., Хілл У. Мистецтво схемотехніки. М .: Світ, 1983. - Т. 2. - 590 с.

29. Щєлкунов М. М. Мікропроцесорні засоби і системи / Щєлкунов М. М., Діанов А. Н. - М.: Радіо і зв'язок, 2009. – 360 с.

30. Alley P. Introductory Microcontroller Programming. Worcester, 2011. - 205 p.

31. Babash A. Proprietary data transfer protocol between personal computers and microcontrollers STM32F4DISCOVERY development method / Andrey Babash, Valeriy Kvashnin, Alexander Tarasov // Proceedings of the International Symposium on Embedded Systems and Trends in Teaching Engineering Nitra. – 2016. – P. 30–35.

32. Bird R. Thinking Functionally with Haskell. Cambridge, 2014. 344 p.

33. Graham H. Programming in Haskell. Nottingham, 2016. 318 p.

34. Kvashnin V. Generating PWM by using microcontroller Stm32F4disovery / V. Kvashnin, A. Babash // Electrotechnic and computer systems. – 2016. – № 22 (98). – P. 277–283. – URL: ui/static/active/en/resource/technical/document/user_manual/ DM00039084.pdf.

35. Watt D. Programming Language Concepts and Paradigms. Upper Saddle River, 1990. 322 p.

ДОДАТКИ