

Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи бакалавра
на тему: «Проект мікроконтролерної системи
управління світлодіодним дисплеєм для задач
декорування дитячих виробів»

Виконав студент 4-го курсу, групи КІ-3
спеціальності 123 «Комп'ютерна інженерія»

_____ В.С. Шпет

Керівник к.т.н., доцент, доцент кафедри КІ та КБ

_____ І.В. Пулеко

Рецензент старший викладач кафедри КІ та КБ

_____ В.О. Філіпов

Житомир – 2021

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
(назва прописними)

КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА КІБЕРБЕЗПЕКИ
(назва прописними)

СПЕЦІАЛЬНІСТЬ 123 «КОМП'ЮТЕРНА ІНТЕЖЕНЕРІЯ»
(назва прописними)

ЗАТВЕРДЖУЮ

Зав. кафедри _____

(назва кафедри)

« 12 » квітня 2021 р.

ЗАВДАННЯ

на випускню кваліфікаційну роботу

Студента Шпета Вадима Сергійовича

Тема роботи: Проект мікроконтролерної системи управління світлодіодним дисплеєм для задач декорування дитячих виробів

Затверджена Наказом університету від «12» квітня 2021 р. № 133/с

Термін здачі студентом закінченої роботи _____

Вихідні дані роботи (зазначається предмет і об'єкт дослідження) _____

Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
<i>1</i>	<i>Пулеко І.В.</i>	12.04.21	16.06.21
<i>2</i>	<i>Пулеко І.В.</i>	12.04.21	16.06.21
<i>3</i>	<i>Пулеко І.В.</i>	12.04.21	16.06.21

Керівник _____
(підпис)

Календарний план

№ з/п	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	<i>Видача завдання</i>	12.04.21	виконано
2	<i>Розробка плану роботи</i>	15.04.21 – 20.04.21	виконано
3	<i>Розробка розгорнутого плану</i>	21.04.21 – 29.04.21	виконано
4	<i>Подача чернетки керівнику для перевірки</i>	20.05.21	виконано
5	<i>Доопрацювання роботи та перевірка керівником</i>	21.05.21 – 3.06.21	виконано
6	<i>Затвердження графічних матеріалів у керівника</i>	8.06.21	виконано
7	<i>Проведення попереднього захисту</i>	14.06.21	виконано
8	<i>Отримання відгуку керівника</i>	15.06.21	виконано
9	<i>Допуск завідувача кафедри</i>	15.06.21	виконано
10	<i>Подання роботи на рецензування</i>	16.06.21	виконано

Студент _____
(підпис)

Керівник _____
(підпис)

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра направлена на розробку проєкту мікроконтролерної системи управління світлодіодним дисплеєм для задач декорування дитячих виробів на базі методів створення IoT-систем для рішень в сфері декорування дитячих виробів.

Проведено обґрунтування актуальності теми, аналіз існуючих технологій підсвіток та декорування виробів, а також досліджено характеристики різних плат та світлодіодів.

Метою роботи є розробка мікроконтролерної системи управління світлодіодним дисплеєм для задач декорування дитячих виробів на прикладі декорування рюкзака адресною світлодіодною матрицею, що підключена до плати з wi-fi модулем, та керується за допомогою смартфона.

Об'єктом дослідження є процес побудови мікроконтролерної системи управління світлодіодним дисплеєм для задач декорування дитячих виробів. Предметом дослідження є технологія розробки мікропроцесорних систем IoT на прикладі декорування рюкзака адресною світлодіодною матрицею, що підключена до плати з wi-fi модулем, та керується за допомогою смартфона.

Пояснювальна записка до випускної бакалаврської роботи містить 72 сторінки, 38 рисунків, 3 додатка, 2 таблиці та 15 літературних джерел.

КЛЮЧОВІ СЛОВА: IoT, ІНТЕРНЕТ РЕЧЕЙ, АРДУІНО, АДРЕСНА СВІТЛОДІОДНА МАТРИЦЯ, БЕЗПРОВІДНІ ТЕХНОЛОГІЇ, LED ДЕКОРУВАННЯ, ПРОГРАММУВАННЯ.

					КІ.КР.Б – 123 – 21 – ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата	Проект мікроконтролерної системи управління світлодіодним дисплеєм для задач декорування дитячих виробів Пояснювальна записка	Літ.	Арк.	Аркушів
Розроб.		Шпет В.С.					4	78
Керівник		Пулеко І.В.				Житомирська політехніка, група КІ-3		
Реценз		Філіпов В.О.						
Н. контр.								
Зав. каф.		Сфіменко А.А.						

ABSTRACT

The final qualifying work of the bachelor is aimed at developing a project of a microcontroller control system of the LED display for the tasks of decorating children's products based on methods of creating IoT systems for solutions in the field of decorating children's products.

The relevance of the topic, analysis of existing technologies of lighting and decoration of products, as well as the characteristics of different boards and LEDs.

The aim of the work is to develop a microcontroller control system for the LED display for decorating children's products on the example of decorating a backpack with an address LED matrix connected to a board with a wi-fi module and controlled by a smartphone.

The object of the study is the process of building a microcontroller control system with an LED display for decorating children's products. The subject of the research is the technology of development of IoT microprocessor systems on the example of decorating a backpack with an address LED matrix connected to a board with a wi-fi module and controlled by a smartphone.

The explanatory note to the final bachelor's thesis contains 72 pages, 38 figures, 3 appendices, 2 tables and 15 references.

KEY WORDS: IOT, INTERNET OF THINGS, ARDUINO, ADDRESSED LED MATRIX, WIRELESS TECHNOLOGIES, LED DECORATION, PROGRAMMING.

		<i>Шпет В.С.</i>			КІ.КР.Б. – 123 – 21 – ПЗ	<i>Арк.</i>
		<i>Пулеко І.В.</i>				
<i>Змн.</i>	<i>Арк.</i>	<i>№ докум.</i>	<i>Підпис</i>	<i>Дата</i>		5

ЗМІСТ

РЕФЕРАТ	4
ABSTRACT	5
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	8
ВСТУП	9
РОЗДІЛ 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ОБҐРУНТУВАННЯ АКТУАЛЬНОСТІ ТЕМИ РОБОТИ.....	11
1.1 Обґрунтування актуальності теми роботи.....	11
1.2 Аналіз існуючих мікроконтролерних систем для задач декорування дитячих виробів	13
1.3 Аналіз типової архітектури IoT	16
1.4 Аналіз технічних характеристик мікроконтролерів	21
1.5 Постановка завдання на розробку	22
РОЗДІЛ 2 РОЗРОБКА СИСТЕМИ УПРАВЛІННЯ СВІТЛОДІОДНИМ ДИСПЛЕЄМ ДЛЯ ДЕКОРУВАННЯ ДИТЯЧИХ ВИРОБІВ	24
2.1 Розробка структурної схеми системи управління світлодіодним дисплеєм для декорування дитячих виробів	24
2.2 Розробка функціональної схеми системи управління світлодіодним дисплеєм для декорування дитячих виробів	26
2.3 Розробка програмного забезпечення системи управління.....	28
РОЗДІЛ 3 ПЕРЕВІРКА ПРАЦЕЗДАТНОСТІ ТА ЕФЕКТИВНОСТІ СИСТЕМИ УПРАВЛІННЯ СВІТЛОДІОДНИМ ДИСПЛЕЄМ ДЛЯ ДЕКОРУВАННЯ ДИТЯЧИХ ВИРОБІВ.....	35
3.1 Розробка моделі пристрою	35
3.2 Перевірка працездатності та ефективності по моделі пристрою управління	40
3.3 Рекомендації по застосуванню розробленого пристрою управління	42
ВИСНОВКИ.....	44

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				6
Змн.	Арк.	№ докум.	Підпис	Дата		

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ	45
ДОДАТКИ.....	47

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				7
Змн.	Арк.	№ докум.	Підпис	Дата		

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

IoT – Інтернет речей (Internet of Things)

RFID – Radio-Frequency IDentification

IP – Інтернет протокол

ПК – персональний комп'ютер

В – вольт

IDE – інтегроване середовище розробки (Integrated Development Environment)

GIF – Graphics Interchange Format (Формат для обміну зображеннями)

USB – Universal Serial Bus

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				8
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

Актуальність теми. У нинішню епоху цифрових технологій все більше інтегрується в наше життя Інтернет речей (IoT). Системи розумних будинків, метеостанції, охорона здоров'я, промисловість, розумні машини, навіть одяг, та безліч інших сфер. В даній роботі буде розглянуте застосування IoT на елементах одягу.

Бренди змагаються один з одним, хто зробить більш унікальну річ, ту, яка буде виділяти людину серед інших. Те, що змусить споживачів звернути увагу саме на їхню фірму і технологія IoT надає величезне місце для креативу таким компаніям. Зробити одяг, який буде змінювати свій колір, чи малюнок, який буде програвати різні цікаві ефекти і ще купу цікавих речей, за допомогою смартфона, який є у кожного. Якщо набридне візерунок, просто зміни його зі свого смартфона – за цим майбутнє.

Тому, **тема роботи**, що направлена на розробку IoT рішення для декорування дитячих виробів на прикладі рюкзака з дисплеєм є **актуальною**.

Мета і завдання роботи. Метою роботи є розробка мікроконтролерної системи управління світлодіодним дисплеєм для задач декорування дитячих виробів на прикладі декорування рюкзака адресною світлодіодною матрицею, що підключена до плати з wi-fi модулем, та керується за допомогою смартфона.

Досягнення поставленої мети передбачає розв'язання таких **завдань**:

- проведення аналізу існуючих способів застосуванні IoT у різних сферах діяльності людини;
- аналізу існуючих методів та можливостей реалізації одягу зі світлодіодами;
- розробки моделі пристрою на основі рюкзака з дисплеєм;
- розроблення програмного коду;
- перевірки та тестування працездатності розробленої технології.

Об'єктом дослідження є процес побудови мікроконтролерної системи управління світлодіодним дисплеєм для задач декорування дитячих виробів.

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		9

Предметом дослідження є технологія розробки мікропроцесорних систем IoT на прикладі декорування рюкзака адресною світлодіодною матрицею, що підключена до плати з wi-fi модулем, та керується за допомогою смартфона.

Це дозволить підкреслити унікальність кожної людини та можливість спростити життя батькам, за допомогою речей, що будуть розважати дитину та відволікати в довгих подорожах.

Методи дослідження. В кваліфікаційній роботі було використано такі методи: побудови IoT-систем, візуальне програмування та математичне моделювання, мобільне-програмування.

Практичне значення роботи полягає у такому:

- розроблений рюкзак, який дає можливість виділитись людині, а також допомагає розважити дітей та дорослих за допомогою різних ефектів, анімацій та ігор;

- розроблені пропозиції використання даної технології в промисловості, зокрема в сфері одягу.

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ОБҐРУНТУВАННЯ АКТУАЛЬНОСТІ ТЕМИ РОБОТИ

1.1 Обґрунтування актуальності теми роботи

Бажання виділятися завжди було і буде притаманне людині. Кожен робить це по своєму. Хтось фарбує волосся, хтось шукає одяг з унікальними принтами, а хтось намагається власноруч створити щось унікальне. На даний момент набирають популярність речі із світлодіодною матрицею. Такі матриці дають волю фантазії. На матриці можна відображати будь яке зображення, створювати власний малюнок, запускати відео та навіть грати в ігри. Такі матриці частіше за все присутні на рюкзаках або куртках, бо вони займають чимало місця і блок живлення потрібно десь ховати.

Відправною точкою створення та поширення речей зі світлодіодами можна вважати фільм «Назад у майбутнє 2» [1]. Саме для цього фільму у 1989 році компанією «Nike» були розроблені перші кросівки, де використовувалась підсвітка (рис. 1.0).



Рисунок 1.0 – Кадр з фільму «Назад в майбутнє 2»

Це були звичайні кросівки, всередину яких помістили світлодіод, та елемент живлення. Запасу роботи вистачало на 8 годин, після чого потрібно було

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

замінити батарею. Пізніше, для подовження роботи та придання взуттю цікавішого виду, контакти замикались і світлодіод вмикався лише тоді, коли людина робила крок, а увесь інший час він був неактивний. Це вивело довготривалість такого взуття на новий рівень, та дало можливість масово розповсюджувати його. Особливо полюбилось воно дітям. Батьки почали купувати для своєї малечі, щоб стимулювати їх більше ходити самостійно. Для дітей куди цікавіше робити кроки, коли вони бачать, що таким чином вони вмикають якійсь вогні на своєму взутті.

Але неймовірну популярність речам з підсвіткою приніс інший фільм, який вийшов у 2010 році – «Крок вперед 3», де танцюристи використовували одяг зі світлодіодами під час свого виступу (рис. 1.1).



Рисунок 1.1 – Кадр з фільму «Крок вперед 3»

Після цього фільму з'явилося безліч прихильників одягу, який світиться [2]. На усіляких фестивалях, де збиралась молодь, можна було все частіше помічати людей у саморобних костюмах, в яких використовувалась дана технологія. На даний момент жоден фестиваль Comic-Con не обходиться без героїв в костюмах зі світлодіодами. Такі костюми додають ефектності шоу. Це гарно продемонстрував український танцювальний гурт «Lightbalance», який в 2013 році на шоу «Україна має талант» вразив усіх своїм виступом (рис. 1.2). Їхні костюми були заздалегідь запрограмовані вмикатися, вимикатися та змінювати кольори в певний час, а учасникам залишалось лише рухатись в правильному ритмі.

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		12



Рисунок 1.2 – Кадр з відео https://youtu.be/RB1uoQ_9gWY

За роки технологія просунулась вперед і вже можна робити всі налаштування в режимі реального часу через додаток в смартфоні. Це відкриває великі можливості, як для проведення ефектних виступів так і для звичайного споживача, що може виділитись серед натовпу. Особливо, це цікаво дітям. Під час довгих подорожей малечі буде в радість відволіктись і побавитись зі своїм чемоданом чи рюкзаком. Дитина може малювати в смартфоні і цей малюнок буде відображатись на дисплеї, або навіть можна запустити такі ігри як «змійка» чи «тетріс». Це позбавить батьків від необхідності постійно займати чимось дитину та інших мандрівників від дискомфорту подорожувати з примхливими дітьми. Саме розроблення рюкзаку з дисплеєм і буде розглянуто в цій роботі.

1.2 Аналіз існуючих мікроконтролерних систем для задач декорування дитячих виробів

Дисплей. Так як дисплей розробляється для рюкзаку, то його розмір обмежений. Є можливість або використати вже готову матрицю (рис 1.3), або окремо придбати світлодіодну стрічку (рис 1.4) [4].

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		13

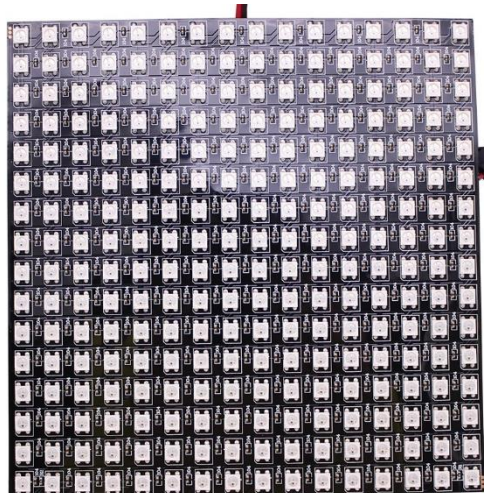


Рисунок 1.3 – Світлодіодна матриця



Рисунок 1.4 – Світлодіодна стрічка

Таблиця 1 – Переваги та недоліки різних дисплеїв

	Готова матриця	Світлодіодна стрічка
Ціна	+	-
Готовність	+	-
Масштабованість	-	+
Працездатність	-	+

Було виділено 4 параметри, за якими порівнювались дисплеї: ціна, готовність, масштабованість, працездатність. Готова матриця на двадцять п'ять відсотків дешевша. Також матрицю можна використовувати відразу з коробки, в тей

час як світлодіодну стрічку спочатку потрібно буде спаяти й привести до матричного вигляду. На цьому плюси матриці завершуються. Стрічка вигідніша тим, що її розмір можна обрати самому, просто спаявши потрібну кількість стрічок відповідної довжини і не потрібно буде підбирати рюкзак під певний розмір. Ще одна перевага стрічки проявляється в тому, що в разі несправності частини дисплея, можна просто замінити цю ділянку на іншу стрічку, а не замінювати увесь дисплей.

На основі перелічених вище даних, зроблено висновок, що світлодіодна стрічка краще підійде для даної задачі. Так як паяння не велика проблема, та ціна не сильно різниться, то в пріоритет береться працездатність, та масштабованість.

При виборі стрічки необхідно звернути увагу, щоб це була саме адресна світлодіодна стрічка. Дана стрічка дає можливість керувати кожним світлодіодом окремо.

Плата. На вибір є безліч плат, які можливо використовувати в даній роботі. Наприклад такі як: iQLED RF 1.4, WeMos D1 mini, T1000s [10,11]. Можливо навіть використовувати міні-комп'ютер Raspberry Pi 3. Але деякі з них відразу відпадають по різних причинах. Raspberry Pi 3 – задорогий для даної задачі і не використається увесь його потенціал. T1000s та iQLED RF 1.4 – не мають wi-fi модуля, тому керуванням дисплеєм з смартфона стає важче. Зрештою було обрано плату WeMos D1 mini (рис 1.5). Вона базується на мікроконтролері ESP8266, що містить в собі wi-fi модуль та вирізняється відсутністю флеш-пам'яті. Але внутрішній накопичувач в цьому проєкті і не потрібний, адже використовуватись буде зовнішній – смартфон.



Рисунок 1.5 – Плата WeMos D1 mini

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				15
Змн.	Арк.	№ докум.	Підпис	Дата		

Живлення. Для живлення плати необхідно 5V, для цього підійде будь-який powerbank. Також павербанк зручно поміщати в рюкзак і пізніше підзаряджати.

1.3 Аналіз типової архітектури IoT

IoT, або ж Інтернет речей (з англ. Internet of Things) – це концепція обчислювальної мережі «речей», що оснащені технологіями для взаємодії із середовищем, або один з одним, в якій мінімізується участь людини [1].

Найпершим пристроєм IoT вважається автомат з газованою водою, який був підключений до інтернету у 1982 році в університеті Карнегі-Мелон. Сильний поштовх для розвитку технологія Інтернету речей дала організація Bluetooth SIG у 1998 році. Через те, що технологія Bluetooth була набагато економніша, з боку споживання енергії, автономність пристроїв стала сильно краща.

На ранніх етапах розвитку інтернету (2000-і роки) вважалося, що максимальне число користувачів завжди буде обмежене чисельністю людей. Але з розвитком технологій таких технологій як RFID (Radio Frequency Identification), NFC (Near Field Communication), WSN (Wireless Sensor Networks) та M2M (Machine-to-Machine) користувачами інтернету стали не лише люди, але й усі пристрої, так звані «речі», число яких постійно збільшується. Саме 2008-2009 роки аналітики компанії Cisco назвали «Справжнім народженням інтернету речей», бо саме в цей період кількість «речей» перевищила кількість населення землі. Але прийнято вважати, що досі відбувається перехід від теперішнього інтернету – «Інтернету людей» до майбутнього «Інтернету речей» [3].

Для того, щоб пристрої могли взаємодіяти між собою, вони повинні містити ідентифікаційні та сенсорні датчики. Зв'язуються ж вони за допомогою інтернет протоколу (IP).

Архітектура Інтернет речей складається з чотирьох рівнів (рис 1.6), з цього випливає формула:

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		16

IoT = додатки + керування + мережі + датчики

Нижче буде розглянуто кожен з цих рівнів детально.

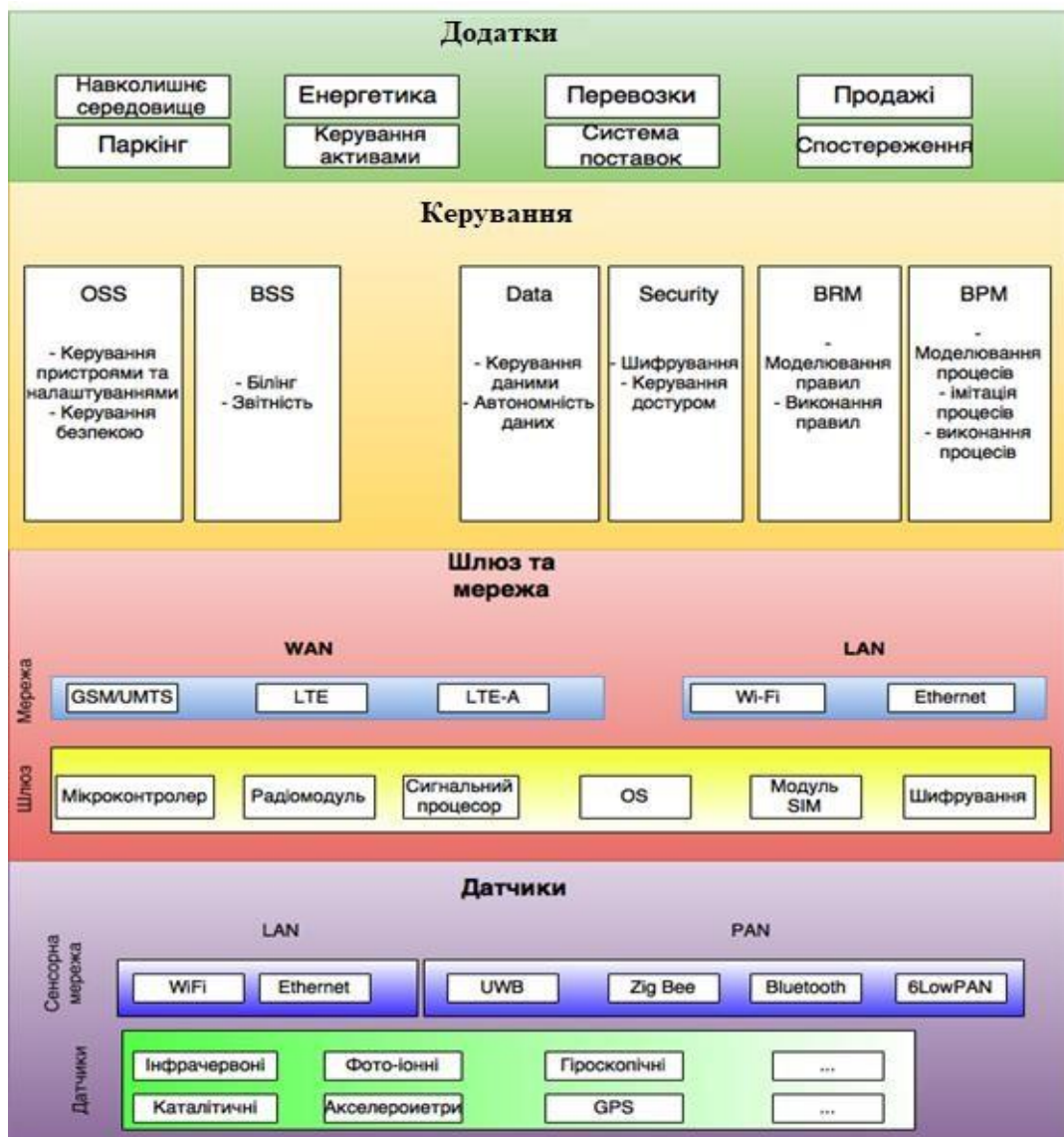


Рисунок 1.6 – Архітектура IoT

Датчики. Найперший рівень – датчики, або ж іншим словом – сенсори. Він є найнижчим рівнем та складається зі smart-об’єктів, які поєднанні з датчиками. Сенсори є провідником між фізичним та цифровим світом. Вони збирають та обробляють інформацію в режимі реального часу. На даний момент існує велика кількість різних сенсорів, які відслідковують температуру, геолокацію, швидкість руху, навіть радіацію. А також завдяки своєму маленькому розмірі, їх можна інтегрувати майже у будь-який пристрій. Зараз кожен смартфон обладнаний десятком сенсорів, які допомагають користувачам отримувати приємний досвід. Більше того, навіть в наручний годинник вбудовують датчики, які допомагають

не лише орієнтуватися у просторі, але й слідкувати за своїм здоров'ям в реальному часі.

Більшості сенсорам потрібне з'єднання зі шлюзом, так званим агрегатором сенсорів. Це з'єднання може бути реалізовано за допомогою LAN (локальної мережі), такої як WIFI, або Ethernet, також за допомоги PAN (персональної мережі), наприклад Bluetooth, та для маленьких дистанцій може використовуватись ультраширокополосний бездротовий зв'язок (UWB, Ultra-Wide Band). Якщо ж сенсорам не потрібне з'єднання з агрегатором, то зв'язок може передаватись через глобальні мережі WAN, це такі сенсори як LTE, GSM [5].

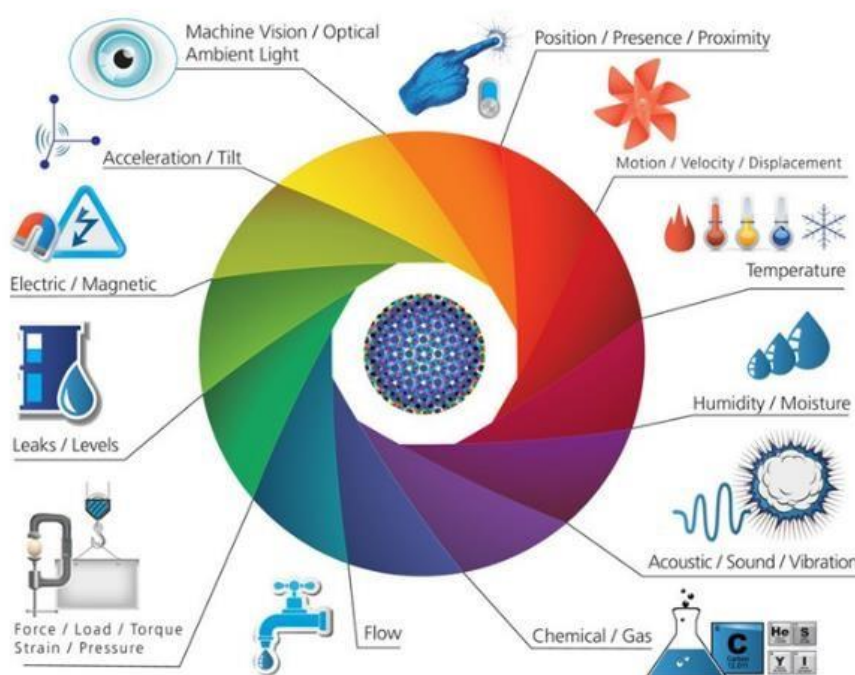


Рисунок 1.7 – Схематичне зображення сил та явищ, для обробки яких може використовуватись IoT

Шлюз та мережа. Другий рівень – шлюз та мережа. На цей рівень лягає відповідальність за увесь обсяг даних, що створюється на першому рівні. Він має через дротову, або бездротову мережу передавати дані. Для транспортування даних з датчиків до інтернет-простору необхідно дві речі: маршрутизатор-шлюз та опорні інтернет протоколи, що забезпечують ефективність обміну даними. Маршрутизатор дуже важливий в аспектах безпеки, управління, та напрямом

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				18
Змн.	Арк.	№ докум.	Підпис	Дата		

даних. Також маршрутизатор відіграє велику роль у створенні VPN (віртуальних локальних мереж), VLAN (віртуальних локальних мереж) та програмно-визначених глобальних мереж. Ці мережі можуть містити тисячі вузлів, обробляти які буде маршрутизатор, який служить розширенням для хмари (edge device).

Цей рівень складається з конвергентної мережної інфраструктури, що дозволяє через певні шлюзи різним користувачам використовувати ресурси в єдиній мережі незалежно і без небезпеки конфіденційності та продуктивності.

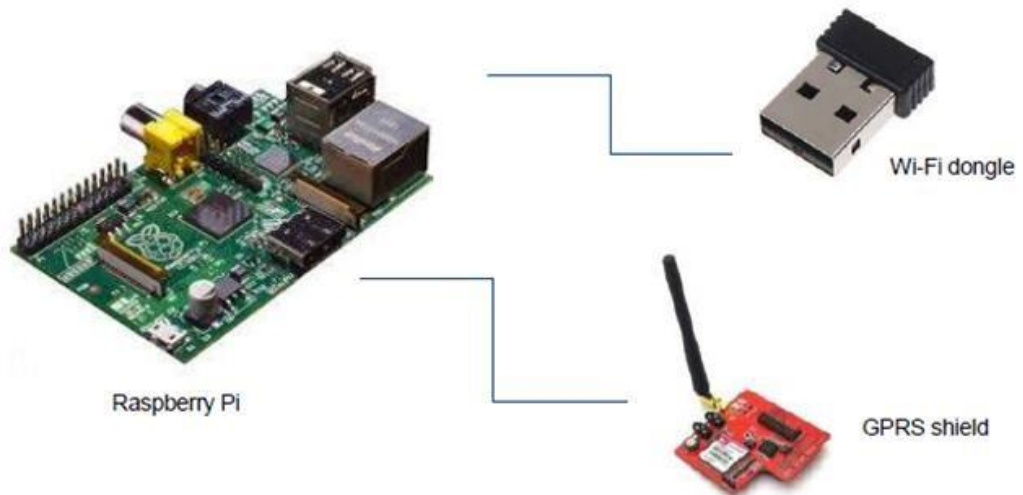


Рисунок 1.8 – Приклади пристроїв для передачі даних в мережі LAN

Керування. Третій рівень – керування, або ж сервісний рівень. Саме цей рівень має автоматизовувати усі процеси, такі як: аналітична обробка інформації, зберігання даних, управління бізнес-правилами (Business Rule Management), управління бізнес-процесами (Business Process Management), забезпечення інформаційної безпеки та ін. Також знати, що немає необхідності відправляти усі дані у хмару. Пересилання усього обходиться значно дорожче, ніж якщо обробити ці дані на кордоні мережі за допомогою граничних обчислень (Edge computing), або включити граничний маршрутизатор в зону, яку обслуговує хмарний сервіс і таким чином використати туманні обчислення (Fog computing).

Додатки. Четвертий, останній рівень – додатки [3]. На даному рівні існують різноманітні види додатків для різних сфер діяльності (бізнес, медицина, спорт, торгівля та ін.). Додатки поділяються на два види: горизонтальні – ті, що

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				19
Змн.	Арк.	№ докум.	Підпис	Дата		

використовуються зазвичай, та вертикальні, які використовуються в специфічних напрямках.

Мережа мереж (ще одна назва IoT) – це мережа різних пристроїв, що підключені до інтернету та які реалізують різні взаємодії, такі як: «річ-річ» (Thing-Thing), «річ-користувач» (Thing-User), «річ-веб-Об’єкт» (Thing-web-Object).

Через складнощі з електроживленням, знижується популярність мереж стандарту 802.11. Для малих відстаней все більше використовуються стандарти, що використовують mesh-архітектуру – вони спеціально розроблені, щоб зменшити енергоспоживання. Це такі стандарти як: Bluetooth Low Energy, 6LoWPAN, ZigBee IP. А для великих відстаней розробляються окремі радіотехнології з меншим енергоспоживанням. Найбільш перспективними в цьому плані є технології EC-GSM (Extended Coverage GSM) та NB-CIoT (Narrowband Cellular IoT). Вони передбачають використання мобільного зв’язку для передачі даних. В мобільних мережах потрібно виділити частотні смуги нижче використовуваних, а операторам треба додати відповідні трансівери на станціях і оновити своє програмне забезпечення.

Головне в цьому це ідентифікація кожного елементу. Беручи до уваги необхідну розрядність, більше за все підходить унікальна Ipv6 адреса, яка надається кожному пристрою в сучасній мережі. Ідентифікатор потрібен не тільки для маршрутизації, але і для зіставлення параметрів пристроїв, таких як: mac-адреса, RFID, qr-код, EID. Смарт об’єкти, які мають унікальний ідентифікатор, можуть не лише передавати дані з сенсорів, але й передавати команди для зміни стану інших пристроїв, що до них підключені.

Основоположні характеристики інтернету речей:

- *Взаємозв’язанність.* Усі пристрої взаємодіють через локальну, чи глобальну інфраструктуру інформаційного обміну;
- *Сервіси, що орієнтовані на пристрої.* IoT узгоджує фізичні об’єкти реального світу з їх поданням у мережі та об’єднує фізичні пристрої з урахуванням усіх правил і обмежень;

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		20

- *Гетерогенність*. Різні мережі та апаратні платформи не заважають пристроям комунікувати між собою;
- *Динамічність*. Кількість пристроїв може змінюватись як і їхній стан;
- *Масштабованість*. Кількість пристроїв постійно зростає, тому потрібно вирішувати питання інтерпретації даних з подальшим їх застосуванням.

1.4 Аналіз технічних характеристик мікроконтролерів

Плата WeMos D1 mini не має великої кількості входів та виходів, але має повноцінний функціонал бездротового зв'язку. Також цю плату можна прошивати використовуючи Arduino IDE, але перед цим встановивши необхідні бібліотеки [13].

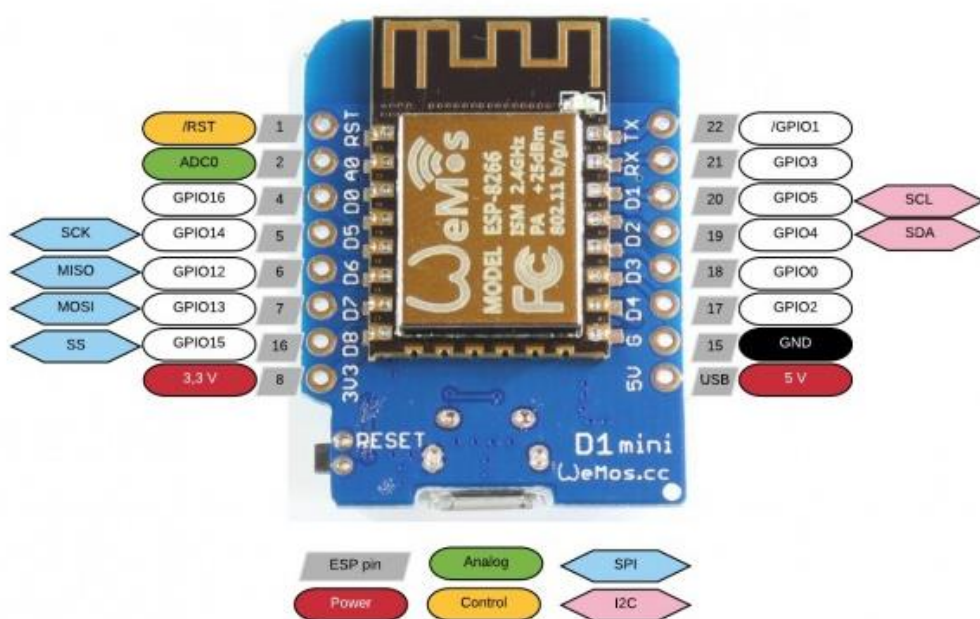


Рисунок 1.9 – Плата WeMos D1 mini з розписаними контактами

Таблиця 1.2 – Технічні характеристики плати

WeMos D1 mini	
Напруга	5 В
Напруга логічних сигналів	3,3 В
Дискретні входи-виходи	11
Аналоговий вхід	1 (максимальна вхідна напруга 3,3В)
Тактова частота процесора	80 МГц/160 МГц
USB-UART конвертер	CH340
Флеш пам'ять	4Мб
Довжина	34,2 мм
Ширина	25,6 мм
Вага	10 г

1.5 Постановка завдання на розробку

Одним з ключових чинників при прогнозуванні майбутньої проблеми є розробка моделі рюкзака з дисплеєм. Модель рюкзака з дисплеєм – це те, що необхідно отримати, зібравши до купи плату, світлодіодну стрічку, живлення, резистор, та прошивку.

В даному проєкті буде побудована модель рюкзака з дисплеєм на основі адресної світлодіодної матриці та плати з wifi модулем. Побудова моделі буде включати такі етапи:

- Розробка структурної схеми системи управління світлодіодним дисплеєм для декорування дитячих виробів;
- Розробка функціональної схеми системи управління світлодіодним дисплеєм;
- Розробка програмного забезпечення системи управління;
- Підготування середовища розробки до завантаження прошивки в плату;

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				22
Змн.	Арк.	№ докум.	Підпис	Дата		

- Розробка моделі пристрою;
- Завантаження коду в плату;
- Перевірка працездатності та ефективності по моделі пристрою управління;

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				23
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 2 РОЗРОБКА СИСТЕМИ УПРАВЛІННЯ СВІТЛОДІОДНИМ ДИСПЛЕЄМ ДЛЯ ДЕКОРУВАННЯ ДИТЯЧИХ ВИРОБІВ

2.1 Розробка структурної схеми системи управління світлодіодним дисплеєм для декорування дитячих виробів

Рюкзак з дисплеєм – це взаємодія таких елементів як: дисплей, контролер, батарея та пристрій керування. Для зручності, як пристрій керування, буде використовуватись телефон. Завдяки спеціальному додатку, можливо надсилати різні команди-сигнали, через wi-fi на контролер, який буде їх обробляти та передавати на дисплей. В якості батареї буде виступати powerbank, який буде живити плату та світлодіоди. Структурну схему пристрою подано на рис.2.0.



Рисунок 2.0 – Структурна схема системи управління рюкзака з дисплеєм

Смартфон. Пристрій за допомогою якого відбувається взаємозв'язок користувача та контролера. Обов'язково потрібен телефон з wi-fi та достатньою кількістю вільного місця, щоб завантажити додаток. За допомогою розробленого додатку для телефону, з'являється можливість легко та без перешкод обирати картинку, чи малюнок, який буде відображатись на дисплеї.

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

Інтернет. Завдяки інтернету сигнал з телефону буде доходити до контролера. Саме тому потрібно, щоб обидва пристрої (телефон та плата) були здатні до передачі та прийняття даних через wifi.

Живлення. Так як плата та світлодіодна стрічка на 5В, то потрібне живлення, яке буде мати напругу в 5В. Для цього підійде powerbank. Також, як варіант можна приєднати 3 батарейки по 1.5В, цього має вистачити.

Контролер. Завдання контролеру прийняти дані, що йому відправить смартфон. Після того як він отримає команду, йому потрібно буде обробити її та відправити правильний сигнал до дисплею. В цьому допоможе прошивка, що буде написана на Arduino.

Дисплей. Дисплей отримує сигнал з контролера, та завдяки тому, що адресна світлодіодна стрічка дозволяє задавати колір кожному світлодіоду окремо, наш дисплей відмальовує рисунок, чи відео в потрібному вигляді.

Алгоритм дій користувача (рис. 2.1) буде виглядати наступним чином:

1. В додатку виконати пошук мережі;
2. В разі невдалого підключення повернутися до пункту 1 і спробувати знову;
3. Якщо підключення вдале, то обрати ефект для відображення на дисплеї;
4. Споглядати обраний ефект;
5. Якщо потрібно змінити ефект, то повернутись до пункту 3;
6. Якщо ж змінювати не потрібно, то кінець дій.

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				25
Змн.	Арк.	№ докум.	Підпис	Дата		



Рисунок 2.1 – Блок-схема алгоритму дій користувача

2.2 Розробка функціональної схеми системи управління світлодіодним дисплеєм для декорування дитячих виробів

На основі запропонованої структурної схеми системи проведемо розробку функціональної схеми.

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				26
Змн.	Арк.	№ докум.	Підпис	Дата		

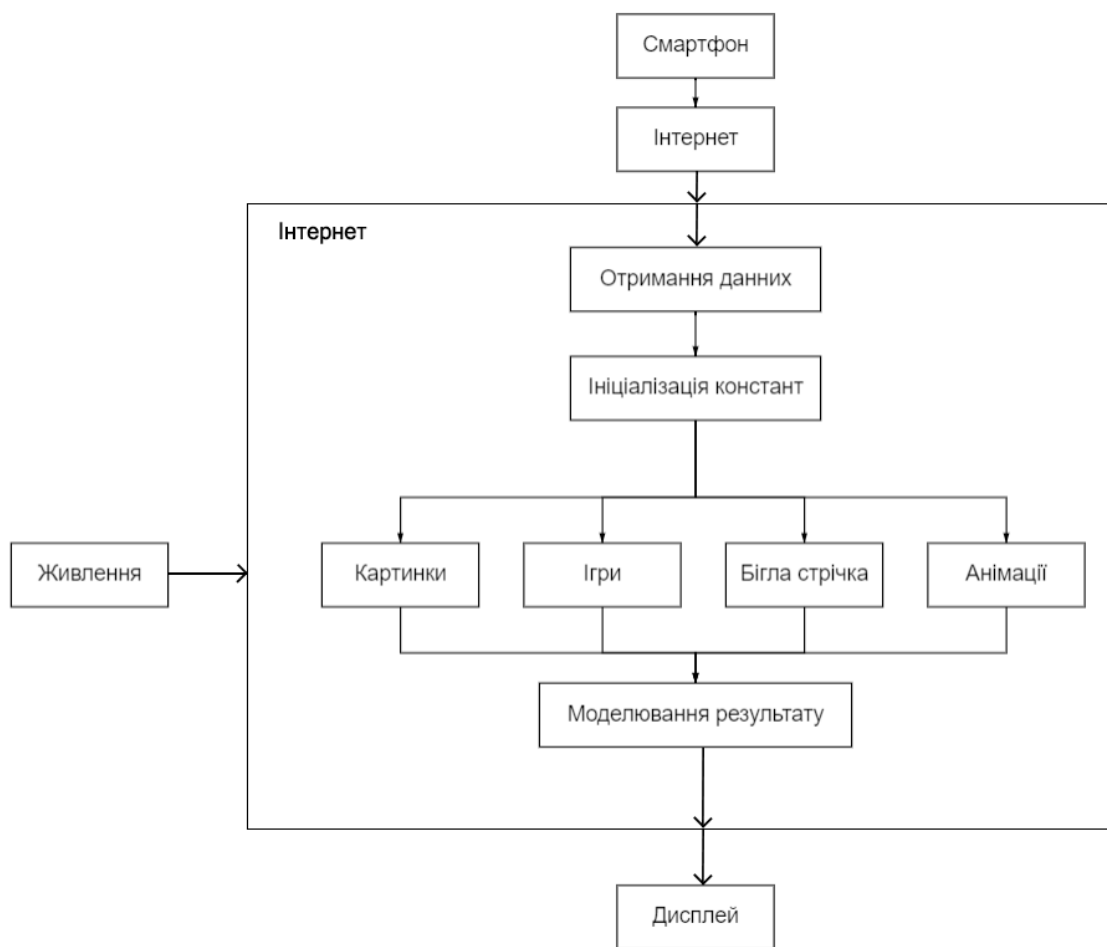


Рисунок 2.2 – Функціональна схема системи управління рюкзака з дисплеєм

На даній схемі розширений блок «контролер» структурної схеми, що дозволяє більше поглибитись в те, що відбувається всередині плати.

Спершу контролер повинен отримати данні, які йому відправив телефон. Далі на основі цих даних контролер ініціалізує потрібні константи. Наприклад, якщо надійшла команда з телефону, що потрібно відобразити певну анімацію, то ініціалізуються такі константи як: висота, ширина, та назва ефекту. Після ініціалізації констант, відкривається відповідний файл зі скриптом, в даному випадку «animation.ino». В ньому обирається потрібна анімація, на основі її назви, яку було отримано раніше, та оголошуються змінні такі як: колір, швидкість анімації. Якби ж була обрана звичайна картинка, а не анімація, то така змінна як швидкість анімації вже не потрібна. Далі моделюється результат та передається на дисплей.

2.3 Розробка програмного забезпечення системи управління

Розробляти програмне забезпечення буде в середовищі Arduino IDE. Arduino IDE – це інтегроване середовище розробки, яке створене спеціально для розроблення програм на Arduino-сумісні та інші плати. Можливо використовувати мову C та C++, а також спеціально бібліотеку Wiring, яка надає багато спільних процедур введення та виведення. Далі буде розглянуто деякі частини створеної програми [14].

Для початку потрібно підключити усі необхідні бібліотеки для повноцінної роботи проекту.

```
12 // Подключение используемых библиотек
13 #include <ESP8266WiFi.h>
14 #include <WiFiUdp.h>
15 #include <TimeLib.h>
16 #include <EEPROM.h>
17 #include <SoftwareSerial.h>
18 #include "FastLED.h"
19 #include "DFRobotDFPlayerMini.h" /
20 #include "timerMinim.h"
21 #include "GyverButton.h"
22 #include "GyverFilters.h"
23
24 #include "bitmap1.h"
25 #include "bitmap2.h"
26 #include "bitmap3.h"
27 #include "bitmap4.h"
28 #include "bitmap5.h"
29
```

Рисунок 2.3 – Бібліотеки проекту

Більшість з цих бібліотек стандартні, без яких неможлива компіляція та робота в цілому, але є й кастомні, наприклад серія «bitmap*.h». Бібліотеки «bitmap1.h», «bitmap2.h» та інші потрібні для відображення анімацій. В цих файлах вказується масив матриць. В кожній матриці відмічений потрібний світлодіод, та колір, яким він має бути на кожному фреймі. (Рис 2.4). Колір задається у форматі RGB565 – це спеціальний формат для матриці 16x16. В даному форматі колір закодований таким чином: 5 бітів (червоного) + 6 бітів (зеленого) + 5 бітів (синього). Таким чином, червона та синя компонента має 32 можливих значення, а зелена 64. Для простішого переводу із звичайної шістнадцяткової системи в

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		28

потрібно, можна використати онлайн калькулятор, наприклад:

<https://trolsoft.ru/ru/articles/rgb565-color-picker>.

```
58 const uint16_t mario3[] PROGMEM = {
59 0x0000, 0x0000, 0x0000, 0x0000, 0x8000, 0xF800, 0xF800, 0xF800, 0xF806, 0x8000, 0x8000, 0x8000, 0x1040, 0x0000, 0x0000,
60 0x0000, 0x0000, 0x0000, 0x9200, 0xF84A, 0xF827, 0xF860, 0xFB80, 0xFB80, 0xF828, 0xFB40, 0xF806, 0xF806, 0xAB40, 0x0000, 0x0000,
61 0x0000, 0x0000, 0x0000, 0x1040, 0xAB40, 0xAB40, 0xBC20, 0xFF20, 0xFF00, 0x8B80, 0xF68E, 0xAC80, 0xAC80, 0x0000, 0x0000, 0x0000,
62 0x0000, 0x0000, 0x1040, 0xAB40, 0xF68E, 0xAB40, 0xF70B, 0xFF40, 0xFF40, 0x93A0, 0xF70B, 0xFF40, 0xFF40, 0xC520, 0x0000, 0x0000,
63 0x0000, 0x0000, 0x1040, 0xAB40, 0xF70B, 0xBC00, 0xBC40, 0xFF40, 0xFF40, 0xEEA0, 0xAC80, 0xF70B, 0xFF20, 0xFF40, 0x93A0, 0x0000,
64 0x0000, 0x0000, 0x0000, 0x8000, 0xBC00, 0xF70B, 0xFF20, 0xFF40, 0xFF40, 0xAC80, 0x8B60, 0x93A0, 0x93A0, 0xAC80, 0x1040, 0x0000,
65 0x0000, 0x0000, 0x0000, 0x0000, 0x1040, 0xFF00, 0xFF20, 0xFF20, 0xFF20, 0xFF00, 0xF70B, 0xF70B, 0x93A0, 0x0000, 0x0000,
66 0x0000, 0x9A60, 0xF8E0, 0xF8E0, 0xF8E0, 0xF827, 0x72FD, 0xF848, 0xFB20, 0xF826, 0x825D, 0x8B60, 0x0000, 0x0000, 0x0000, 0x0000,
67 0xFF40, 0xFF20, 0xFB80, 0xF804, 0xF800, 0xF827, 0x429F, 0x5AFE, 0xF848, 0xF826, 0xF08D, 0x72FD, 0xF848, 0xFF20, 0xFF40, 0xFF40,
68 0xFF40, 0xFF40, 0xF70B, 0x8000, 0xF06C, 0xF06C, 0x627E, 0xEF6F, 0x5AFE, 0x429F, 0x63DE, 0xEF6F, 0xF927, 0xF927, 0xFF00, 0xFF20,
69 0xFF20, 0xF70B, 0x93A0, 0x7A96, 0x5AFE, 0x3B1F, 0x12BF, 0x3B1F, 0x02BF, 0x02BF, 0x0ABF, 0x429F, 0x7A96, 0x1040, 0x9A60, 0x1040,
70 0x0000, 0x0000, 0x6A99, 0x1ABF, 0x02BF, 0x02BF, 0x02BF, 0x02BF, 0x02BF, 0x02BF, 0x02BF, 0x02BF, 0x429F, 0x9A62, 0x9A60, 0x1040,
71 0x0000, 0x1040, 0x429F, 0x0ABF, 0x02BF, 0x02BF, 0x02BF, 0x02BF, 0x02BF, 0x02BF, 0x02BF, 0x02BF, 0x429F, 0x9A62, 0x9A60, 0x1040,
72 0x1040, 0x9240, 0x7A96, 0x32BE, 0x12BF, 0x2A9F, 0x7A96, 0x7A96, 0x7A96, 0x7A96, 0x2A9F, 0x02BF, 0x429F, 0x9A62, 0x9A40, 0x1040,
73 0x1040, 0x9A60, 0x9A61, 0x9200, 0x7A96, 0x0820, 0x0000, 0x0000, 0x0000, 0x0000, 0x0820, 0x7A96, 0x1040, 0x1040, 0x1040, 0x0000,
74 0x0000, 0x1040, 0x9A60, 0x9A60, 0x9A40, 0x1040, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
75 };
76
77 const uint16_t* const mario_array[4] PROGMEM = {
78 mario0, mario1, mario2, mario3};
```

Рисунок 2.4 – Матриця з відміченими світлодіодами

Наступним кроком потрібно оголосити константи, які будуть використовуватись в проєкті (рис 2.4).

```
#define DEFAULT_AP_NAME "MatrixAP"
#define DEFAULT_AP_PASS "12341234"
#define localPort 2390
#define LED_PIN 2

#define D_TEXT_SPEED 100
#define D_TEXT_SPEED_MIN 10
#define D_TEXT_SPEED_MAX 255

#define D_EFFECT_SPEED 80
#define D_EFFECT_SPEED_MIN 0
#define D_EFFECT_SPEED_MAX 255

#define D_GAME_SPEED 250
#define D_GAME_SPEED_MIN 25
#define D_GAME_SPEED_MAX 375

byte IP_STA[] = {192, 168, 0, 106};
```

Рисунок 2.5 – Оголошення частини констант

Командою `#define` ми задаємо ті значення, які не будуть змінюватись – константи. Наприклад: назва мережі, пароль мережі, мінімальна та максимальна швидкість ефектів, значення за змовчуванням, та інші. Зручніше задати відразу константи, та мати їх в одному місці – відразу після додавання бібліотек.

Завдяки бібліотеці «FastLED.h» налаштування світлодіодної стрічки проходить доволі швидко та без проблем.

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		29

```
FastLED.addLeds<WS2812, LED_PIN, COLOR_ORDER>(leds, NUM_LEDS).setCorrection( TypicalLEDStrip );
FastLED.setBrightness(globalBrightness);
if (CURRENT_LIMIT > 0) FastLED.setMaxPowerInVoltsAndMilliamps(5, CURRENT_LIMIT);
FastLED.clear();
FastLED.show();
```

Рисунок 2.6 – Використання бібліотеки FastLED.h для налаштування стрічки

Найбільш важливо, успішно створити точку доступу wifi, щоб можливо було під'єднатись з телефону. Завдяки додатковій бібліотеці «WiFiUdp.h» це виконується дуже швидко. За цю реалізацію відповідає код на рисунку 2.7.

```
void startSoftAP() {
    WiFi.softAPdisconnect(true);
    ap_connected = false;

    ap_connected = WiFi.softAP(apName, apPass);

    for (int j = 0; j < 10; j++ ) {
        WiFi.enableAP(false);
        WiFi.softAPdisconnect(true);
        ESP.wdtFeed();
        delay(1000);

        ap_connected = WiFi.softAP(apName, apPass);
    }

    if (!ap_connected)
        Serial.println(F("Не удалось создать WiFi точку доступа."));
}
```

Рисунок 2.7 – Створення точки доступу

Розглянемо частину коду, що відповідає за відображення gif-зображень. При виборі анімації, її індекс передається до умовного оператора «switch», який обробляє і запускає функцію «loadImage» з потрібними параметрами під дану анімацію.

```
void loadImage(const uint16_t (*frame)) {
    int8_t offset_x = (WIDTH - FRAME_WIDTH) / 2;
    int8_t offset_y = (HEIGHT - FRAME_HEIGHT) / 2;
    for (byte i = 0; i < FRAME_WIDTH; i++) {
        if (offset_x + i < 0 || offset_x + i > WIDTH - 1) continue;
        for (byte j = 0; j < FRAME_HEIGHT; j++) {
            if (offset_y + j < 0 || offset_y + j > HEIGHT - 1) continue;
            drawPixelXY(offset_x + i, offset_y + j, gammaCorrection(expandColor((pgm_read_word(&(frame[(HEIGHT - j - 1) * WIDTH + i]))))));
        }
    }
}
```

Рисунок 2.8 – Код функції «loadImage»

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		30

```

switch (n) {
case 1:
    // EFFECT_ANIMATION_1
    loadImage(mario_array[frameNum]);
    if (++frameNum >= sizeof(mario_array) / sizeof(mario_array[0])) frameNum = 0;
    break;
case 2:
    // EFFECT_ANIMATION_2
    loadImage(flame_array[frameNum]);
    if (++frameNum >= sizeof(flame_array) / sizeof(flame_array[0])) frameNum = 0;
    break;
case 3:
    // EFFECT_ANIMATION_3
    loadImage(ghost_array[frameNum]);
    if (++frameNum >= sizeof(ghost_array) / sizeof(ghost_array[0])) frameNum = 0;
    break;
case 4:
    // EFFECT_ANIMATION_4
    loadImage(ironMan_array[frameNum]);
    if (++frameNum >= sizeof(ironMan_array) / sizeof(ironMan_array[0])) frameNum = 0;
    break;
}

```

Рисунок 2.9 – Код оператору Switch

Як видно на рисунку 2.8, ми спершу задаємо змінні `offset_x` та `offset_y`, які відповідають за розміщення анімації посередині дисплею. Координати вираховуються доволі просто: береться ширина та висота дисплею і ділиться на 2, таким чином отримується точний центр дисплею. Далі, в циклі, завдяки бібліотеці «FastLED.h», можливо використати функцію «drawPixelXY», за допомоги якої передасться сигнал на світлодіодну матрицю. Функція приймає три параметри: координата X, координата Y та колір світлодіоду. Колір світлодіоду записаний у масив матриць, який був продемонстрований на рисунку 2.4.

Не менш цікавою розробкою в даному пристрої є гра «Змійка». Класична гра, яку можна зробити під будь-який пристрій і даний рюкзак не став винятком. Суть гри полягає в тому, щоб їсти яблука, та рости, не торкаючись при цьому свого тіла та стін. На подальших скріншотах буде розглянуто, як реалізована ця гра в даному проєкті.

```

int8_t vectorX, vectorY;
int8_t headX, headY, buttX, buttY;
int8_t appleX, appleY;
boolean apple_flag, missDelete = false;
int snakeLength;

```

Рисунок 2.10 – Оголошення змінних в «змійці»

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		31

На рисунку вище оголошуються необхідні змінні для гри. VectorX, vectorY – це напрямок, у якому повинна рухатись змійка. Наприклад якщо vectorX рівняється 1, то змійка буде рухатись вправо, якщо дорівнюватиме -1, то в ліво, якщо ж 0, то по осі X змійка не має рухатись. Важливо врахувати, що якщо змійка вже рухається в осі X, то vectorY має дорівнювати 0 і навпаки.

Змінні headX та headY – це координати голови змійки, а buttX, buttY – тіла. Різні змінні для голови і тіла потрібні для того, щоб була змога відмічати, коли голова «з’їдає» яблуко, та куди потрібно додавати додатковий «вирісший» блок. Координати тіла, це саме координати кінця змійки. Туди ж ми додаємо додаткові блоки та прибираємо старі, при русі змійки. Змінні appleX та appleY – це координати яблука. При зіткненні з головою яблуко має пропасти, а також потрібна перевірка, щоб яблуко не з’явилося на місці самої змійки.

Також є дві булеві змінні, вони містять лише значення «true», або «false» та потрібні для того, щоб відмічати, чи існує яблуко на карті (apple_flag) та чи треба відмальовувати додатковий хвіст змійці, якщо вона росте, або не додавати хвіст, якщо не росте (missDelete).

Остання змінна – довжина змійки – snakeLength. Потрібна для руху змійки на полі та для зберігання результату гравця.

```
while (!apple_flag) {
    appleX = random(0, WIDTH);
    appleY = random(0, HEIGHT);

    if ((long)getPixColorXY(appleX, appleY) == 0) {
        apple_flag = true;
        drawPixelXY(appleX, appleY, GLOBAL_COLOR_2);
        FastLED.show();
    }
}
```

Рисунок 2.11 – Створення яблука для «змійки»

Процес створення яблука полягає таким чином: перевірка чи існує вже яблуко на полі, якщо немає, то генеруються випадкові координати, куди розміс-

титься яблуко. Після генерації проводиться перевірка, чи не співпадають координати яблука з координатами змійки, якщо не співпадають, то переключється `apple_flag` у положення «true» та відмальовується яблуко на дисплеї.

```
if ((long)(getPixColorXY(headX, headY) != 0 && (long)getPixColorXY(headX, headY) != GLOBAL_COLOR_2)) {
    fail = true;
}
```

Рисунок 2.12 – Перевірка, чи не врізалась змійка у щось, окрім яблука

```
if (fail) {
    fail = false;

    for (byte bright = 0; bright < 15; bright++) {
        FastLED.setBrightness(bright);
        for (int i = 0; i < NUM_LEDS; i++) {
            leds[i] = CRGB::Red;
        }
        FastLED.show();
        delay(10);
    }
    delay(100);
    FastLED.clear();
    FastLED.show();
    FastLED.setBrightness(globalBrightness);
    displayScore(snakeLength - START_LENGTH);
    delay(1000);
    FastLED.clear();
    FastLED.show();
    newGameSnake();
}
```

Рисунок 2.13 – Обробка програшу в «змійці»

На рисунках 2.12, та 2.13 перевіряється та обробляється випадок, якщо гравець програє. Якщо ж змійка врізається в своє тіло, або в стіну, то переключється прапорець «fail» та запускається код умовного оператора «if». В середині обробки програшу спочатку повертається прапорець в положення за змовчуванням, а після відбувається плавне затемнення екрану в циклі «for». Записується результат гри, та починається нова гра.

```
if (!fail && (long)getPixColorXY(headX, headY) == (long)GLOBAL_COLOR_2) {
    apple_flag = false;
    snakeLength++;
    buttVector[snakeLength] = 4;
}
```

Рисунок 2.14 – Поїдання яблука в «змійці»

Якщо ж змійка врізалась саме в яблуко, то потрібно переключити прапорець наявності яблука в негативне положення, та збільшити інкрементувати

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		33

розмір змійки. Також треба вказати, що в даний момент не потрібно буде зати-
рати хвіст, адже він збільшився.

```
switch (buttVector[0]) {
    case 0: buttX += 1;
        break;
    case 1: buttX -= 1;
        break;
    case 2: buttY += 1;
        break;
    case 3: buttY -= 1;
        break;
    case 4: missDelete = true;
        break;
}

for (byte i = 0; i < snakeLength; i++) {
    buttVector[i] = buttVector[i + 1];
}

if (!missDelete) {
    drawPixelXY(buttX, buttY, 0x000000);
}
else missDelete = false;

drawPixelXY(headX, headY, GLOBAL_COLOR_1);
FastLED.show();
}
```

Рисунок 2.15 – Рух змій

Якщо змія з’їла яблука, то обирається «case 4», який вказує, що не потрібно замальовувати старий хвіст. А тому виконується лише цикл зі зміщенням масиву векторів вліво, та відображення змій в новому положені зі вже збільшеним ті-
лом.

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				34
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3 ПЕРЕВІРКА ПРАЦЕЗДАТНОСТІ ТА ЕФЕКТИВНОСТІ СИСТЕМИ УПРАВЛІННЯ СВІТЛОДІОДНИМ ДИСПЛЕЄМ ДЛЯ ДЕКОРУВАННЯ ДИТЯЧИХ ВИРОБІВ

3.1 Розробка моделі пристрою

Після розроблення коду саме час створювати модель пристрою. Спочатку потрібно підготувати світлодіодну матрицю. Для цього була використана пластина розміром 27.5x28 см – як основа матриці, до якої буде кріпитися стрічка. Адресна світлодіодна стрічка ділиться на шістнадцять частин, по 16 світлодіодів у відрізку. Важливо зберігати порядок накладення відрізків, щоб не збити напрямлення стрічки. На більшості світлодіодних стрічках є позначки у вигляді стрілок, тож можна орієнтуватися по ним. Краї пластини потрібно ізолювати, наприклад клейкою стрічкою, для того, щоб випадково не замкнути контакти, адже в тих місцях будуть з'єднуватись стрічки одна до одної.



Рисунок 3.0 – Приклеєні стрічки

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				35
Змн.	Арк.	№ докум.	Підпис	Дата		

Наступним кроком потрібно спаяти контакти. Паяти треба послідовно (зигзагом), а не паралельно. Паралельне з'єднання важче в реалізації, та можлива поява перешкод на матриці, через довжину дроту, що відповідає за логіку.



Рисунок 3.1 – Паяльна станція

Дана стрічка, як і плата, дуже чутлива до температур і може пошкодитись від перегріву. Через це використовувати звичайний паяльник небезпечно, тож буде використана паяльна станція, яка дозволяє регулювати температуру жала. Таким чином буде збережена плата та матриця від небезпечних температур.

Після з'єднання усіх відрізків, плата зі стрічкам стає адресною світлодіодною матрицею 16x16.

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		36

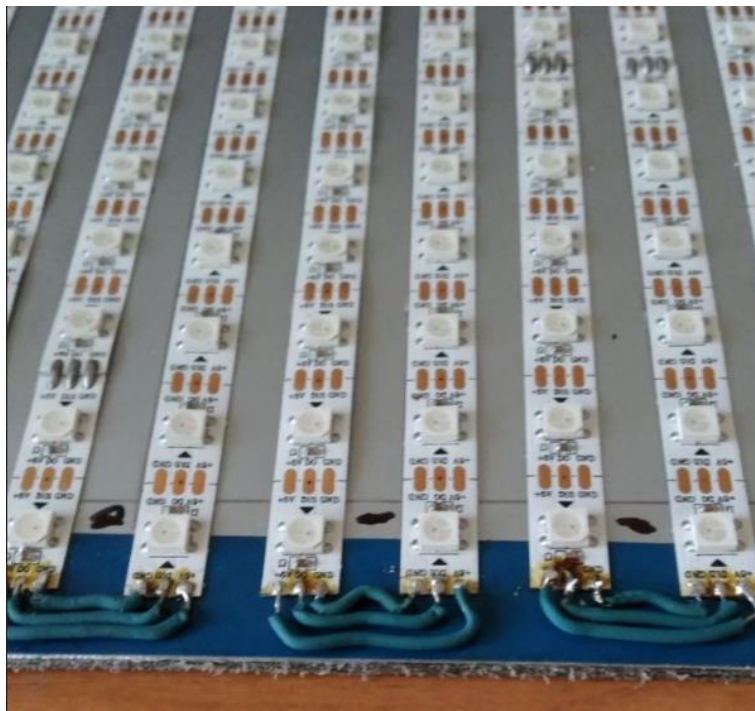


Рисунок 3.2 – З'єднання за принципом «зигзаг»

Коли матриця створена, саме час переходити до плати. До плати потрібно під'єднати дріт від світлодіодної стрічки, там кабель USB – для живлення плати та стрічки. Важливо відмітити, що ESP8266 не можна жити від 5В, через це на матриці можуть з'явитись різні артефакти, тому до схеми потрібно додати діод. Підійде майже будь-який кремнієвий (1N4001-1N4007), в даному проєкті був використаний Д223Б. Діод знизить напругу на платі і перешкоди не повинні виникати. Також, при роботі зі світлодіодною стрічкою, рекомендується використовувати резистор. Можна взяти резистор, в якому регулюється опір і самому підлаштувати потрібний.



Рисунок 3.3 – Схема плати

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				37
Змн.	Арк.	№ докум.	Підпис	Дата		

Оголяємо та лудимо дроти. Після цього, за рисунком 3.3, збираємо схему. В кінцевому результаті має вийти щось схоже на рисунок 3.4.



Рисунок 3.4 – Зібрана плата

Коли зібрані та підготовлені усі компоненти, саме час завантажувати прошивку, яка була розроблена в пункті 2.3, в плату. Перед цим потрібно встановити усі необхідні компоненти для плати esp8266, та обрати потрібні налаштування.

Спершу встановити пакет ESP8266 Community, через менеджер плат у середовищі Arduino IDE.

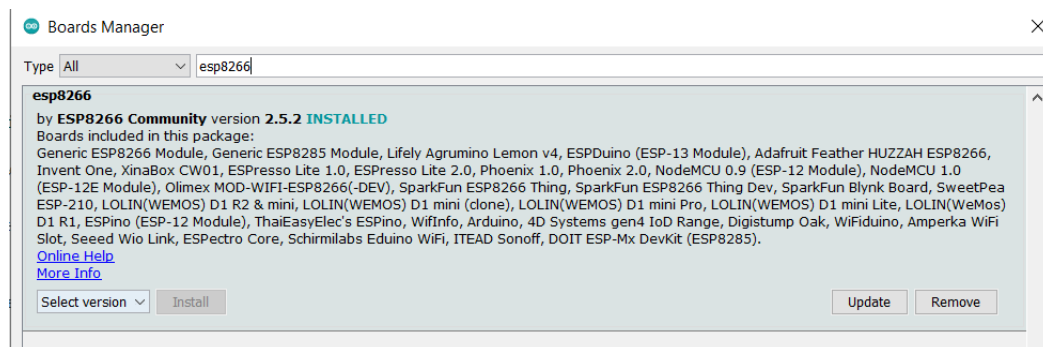


Рисунок 3.5 – Встановлення необхідних пакетів

Після цього в «Інструментах» обрати потрібні налаштування, як на рисунок 3.6.

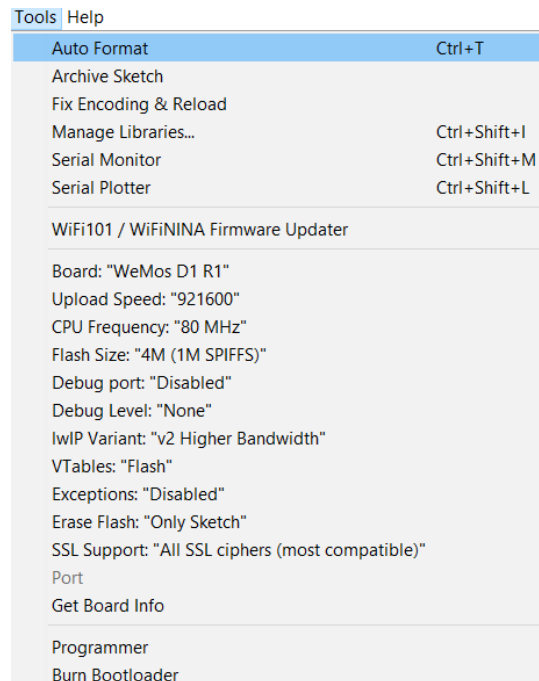


Рисунок 3.6 – Налаштування плати

Опція «Port» стане доступною після під'єднання пристрою до ПК. Порт потрібно обрати будь який, відмінний від «Com1».

Коли усі підготовлення завершені, залишається лиш завантажити прошивку до плати. Бажано надати платі, окрім живлення від USB, ще й альтернативне, в даному випадку – powerbank.

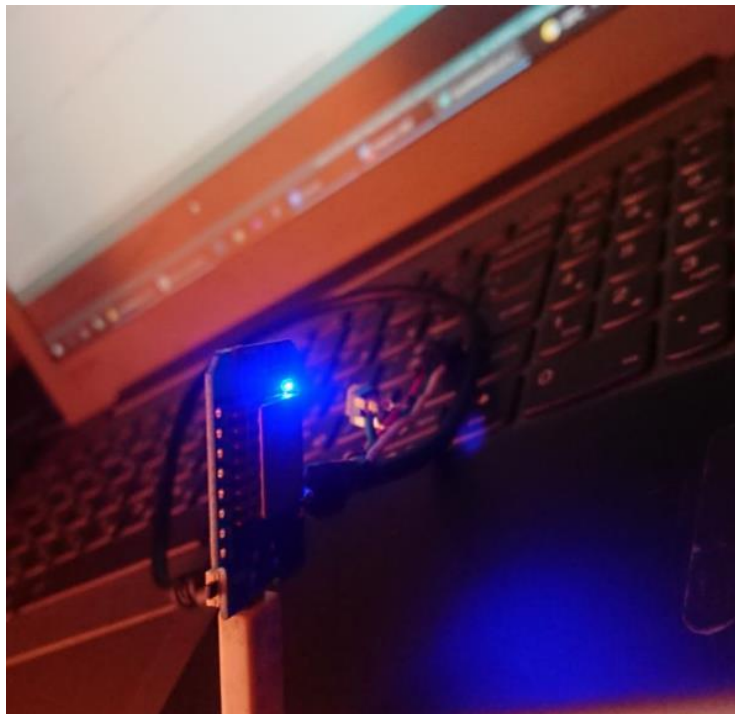


Рисунок 3.7 – Завантаження прошивки до плати

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				39
Змн.	Арк.	№ докум.	Підпис	Дата		

3.2 Перевірка працездатності та ефективності по моделі пристрою управління

Після того, як усе зібране та прошите, потрібно протестувати на працездатність. Треба з'єднати адресну світлодіодну матрицю, павербанк, та плату. Далі за допомогою телефону потрібно під'єднатись до точки доступу wifi «Matrix», використавши пароль який був створений в пункті 2.3. Наступним кроком, зайти в додаток та ввести ір-адресу, порт і під'єднатись. Усі дані були створені в 2 розділі, при розробці програмного коду. Якщо з'єднання успішне, то на матриця має відтворювати різні ефекти (рис. 3.8-3.10).



Рисунок 3.8 – Відтворення ефекту №1

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				40
Змн.	Арк.	№ докум.	Підпис	Дата		

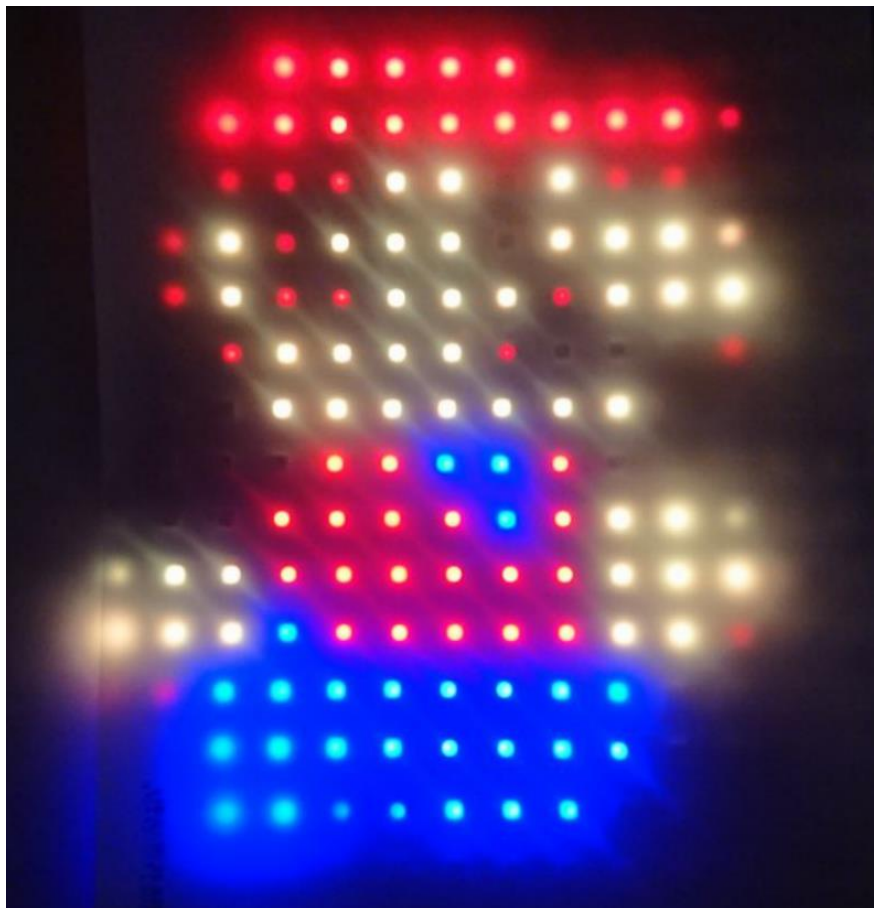


Рисунок 3.9 – Відтворення ефекту №2



Рисунок 3.10 – Відтворення ефекту №3

Для кращого розсіювання світла, був використаний білий пластик, який наклеївся поверх дисплею.

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		41

3.3 Рекомендації по застосуванню розробленого пристрою управління

Дану технологію можна використовувати не лише на елементах одягу, а також і торгівельній сфері чи сфері громадського харчування. На вітринах багатьох магазинів та ресторанів часто красується стрічка з назвою закладу, або з текстом «відчинено». При технології, що надає можливість керувати стрічкою з телефону, власники закладів можуть підлаштовувати дисплей під знижки, страви дня та ін. І ця маніпуляція займе всього пару кліків, з телефону який завжди під рукою.

Також, можливе, застосування технології в сфері декору. На даний момент є певні спроби інтегрувати цю технологію в декор приміщень. Наприклад, стіни, які зроблені у вигляді піксельного екрану, але поки це занадто дороге.



Рисунок 3.11 – Застосування технології для декорування стін

Куди більш цікаво буде, коли так звані «пікселі» стануть меншими, та око не буде спроможне бачити такий грубий перехід від одного пікселя до іншого. Тоді стане можливим змінювати колір кімнати, меблів в декілька кліків на телефоні, та обирати кожен раз різну палітру на різні дні тижня.

Компанія «Nike» вже має модель кросівок, в яких через додаток можна змінювати колір логотипу. Саме по такому принципу, можна буде декорувати усе, що завгодно.

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		42



Рисунок 3.12 – Кросівки зі змінним кольором логотипу

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				43
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

У випускній кваліфікаційній роботі бакалавра було вирішено актуальне завдання по розробці IoT рішення задачі декорування дитячих виробів на основі рюкзака з дисплеєм.

Обґрунтування актуальності теми дало змогу оцінити внесок LED-одягу у сферу моди та промисловості.

Після проведення аналізу технічних характеристик мікроконтролерів, була обрана плата та підібрані інші деталі, що відповідали вимогам плати Wemos D1 mini, на базі ESP8266.

Ознайомившись з середовищем розробки Arduino IDE, було виділено етапи розробки: створення робочої області Arduino, написання основних функцій, написання різних ефектів, підготовка даних, налаштування та підготовка проєкту до завантаження в плату.

Готова модель рюкзака з дисплеєм включає такі модулі: світлодіодна матриця, блок живлення, плата, рюкзак, телефон з додатком.

Дана модель розроблена за допомогою принципів IoT. Включає в себе пристрій керування – плата, де обробляються усі сигнали з додатку, додаток, за допомогою якого відбувається надсилання сигналів до плати, мережа – wifi, через яку відбувається з'єднання телефону та плати.

Розглянуто доцільність використання запропонованої моделі у промисловій сфері та сфері громадського харчування. Основними перевагами є легке управління та зрозумілий інтерфейс. Використання даної технології дає змогу виділити, або підкреслити те, що потрібно людині. Стане в нагоді власникам закладів, та просто звичайним людям, що хочуть показати свою індивідуальність.

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		44

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Росляков А. В., Ваняшин С. В., Гребешков А. Ю. Интернет вещей: учеб. пособ. / А. В. Росляков, С. В. Ваняшин, А. Ю. Гребешков. – Самара. 2015. – 136 с.
2. Грингард С. Интернет вещей. Будущее уже здесь / С. Грингард. – Альпина Паблишер – 2017. – 188 с.
3. Грінгард С. Інтернет речей. / С. Грінгард. – К. «Клуб сімейного дозвілля» – 2018. – 176 с.
4. Интернет вещей. Исследования и область применения: монография / Е.П. Зараменских, И.Е. Артемьев. - М.: НИЦ ИНФРА-М, 2015. - 200 с.
5. Приемышев А. В., Крутов В. Н., Треяль В. А. Технологии создания интеллектуальных устройств, подключенных к интернет / А. В. Приемышев, В. Н. Крутов, В. А. Треяль. – Лань, 2017. – 100 с.
6. Доменна система імен [Електронний ресурс]. – 2018. – Режим доступу до ресурсу:
https://uk.wikipedia.org/wiki/%D0%94%D0%BE%D0%BC%D0%B5%D0%BD%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%96%D0%BC%D0%B5%D0%BD
7. Интернет вещей и требования к сети [Електронний ресурс]. – 2017. – Режим доступу до ресурсу: <https://www.orange-business.com/ru/blogs/get-ready/upravlenie-it/Internet-veshchey-i-trebovaniya-k-seti>
8. Интернет вещей и требования к сети [Електронний ресурс]. – 2017. – Режим доступу до ресурсу: <https://www.orange-business.com/ru/blogs/get-ready/upravlenie-it/Internet-veshchey-i-trebovaniya-k-seti>
9. Wemos D1 mini – характеристики [Електронний ресурс] – 2021 – Режим доступу до ресурсу: http://geekmatic.in.ua/nodemcu_esp8266_wemos
10. Светодиодный костюм [Електронний ресурс] – 2020 – Режим доступу до ресурсу: <https://habr.com/ru/post/490460/>

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		45

11. Контроллер iqLED RF 1.2 и iqLED RF 1.4 [Электронный ресурс] – 2019 – Режим доступа до ресурсу: <https://iqled.org/forum/viewtopic.php?t=104>
12. Контроллер умных светодиодов. Достойный ответ китайским разработчикам [Электронный ресурс] – 2017 – Режим доступа до ресурсу: <https://mysku.ru/blog/ebay/50015.html>
13. Among Us Backpack [Электронный ресурс] – 2021 – Режим доступа до ресурсу: <https://create.arduino.cc/projecthub/tglizzy/among-us-backpack-75af0f>
14. How to make a portable LED backpack sign! [Электронный ресурс] – 2018 – Режим доступа до ресурсу: <https://djanes.xyz/how-to-make-a-portable-led-backpack-sign/>
15. Калькулятор цвета RGB565 для LCD [Электронный ресурс] – 2017 – Режим доступа до ресурсу: <https://trolsoft.ru/ru/articles/rgb565-color-picker>

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				46
Змн.	Арк.	№ докум.	Підпис	Дата		



ДОДАТКИ

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				47
Змн.	Арк.	№ докум.	Підпис	Дата		

```

// Подключение используемых библиотек
#include <ESP8266WiFi.h>
#include <WiFiUdp.h>
#include <TimeLib.h>
#include <EEPROM.h>
#include <SoftwareSerial.h>
#include "FastLED.h"
#include "DFRobotDFPlayerMini.h"
#include "timerMinim.h"
#include "GyverButton.h"
#include "GyverFilters.h"

#include "bitmap1.h"
#include "bitmap2.h"
#include "bitmap3.h"
#include "bitmap4.h"
#include "bitmap5.h"

// ***** МАТРИЦА *****

#define BRIGHTNESS 32
#define CURRENT_LIMIT 1800

#define WIDTH 16           // ширина матрицы
#define HEIGHT 16          // высота матрицы
#define SEGMENTS 1         // диодов в одном "пикселе" (для создания матрицы из кус-
ков ленты)
#define DEVICE_TYPE 1      // Использование матрицы: 0 - свернута в трубу для лампы;
1 - плоская матрица в рамке

#define COLOR_ORDER GRB    // порядок цветов на ленте. Если цвет отображается некор-
ректно - меняйте. Начать можно с RGB

#define MATRIX_TYPE 0      // тип матрицы: 0 - зигзаг, 1 - параллельная
#define CONNECTION_ANGLE 0 // угол подключения: 0 - левый нижний, 1 - левый верхний,
2 - правый верхний, 3 - правый нижний
#define STRIP_DIRECTION 0  // направление ленты из угла: 0 - вправо, 1 - вверх, 2 -
влево, 3 - вниз

// ***** ПОДКЛЮЧЕНИЕ К СЕТИ *****

#define NETWORK_SSID ""    // Имя WiFi сети - пропишите здесь или за-
дайте из программы на смартфоне
#define NETWORK_PASS ""   // Пароль для подключения к WiFi сети -
пропишите здесь или задайте из программы на смартфоне
#define DEFAULT_AP_NAME "MatrixAP" // Имя точки доступа по умолчанию
#define DEFAULT_AP_PASS "12341234" // Пароль точки доступа по умолчанию
#define localPort 2390    // Порт матрицы для подключения из програ-
ммы на смартфоне
byte IP_STA[] = {192, 168, 0, 106}; // Статический адрес матрицы в локальной
сети WiFi

#define DEFAULT_NTP_SERVER "ru.pool.ntp.org" // NTP сервер по умолчанию "time.nist.gov"

// ***** ПИНЫ ПОДКЛЮЧЕНИЯ *****

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		48

```

// пины подписаны согласно pinout платы, а не надписям на пинах!
// esp8266 - плату выбирал "Node MCU v3 (SP-12E Module)"
#define PHOTO_PIN 0 // пин фоторезистора
#define LED_PIN 2 // пин ленты
#define SRX D4 // D4 is TX of ESP8266, connect to RX of DFPlayer module
#define STX D3 // D3 is RX of ESP8266, connect to TX of DFPlayer
#define PIN_BTN D6 // кнопка подключена сюда (PIN --- КНОПКА --- GND)

// ***** ЭФФЕКТЫ И РЕЖИМЫ *****

#define EFFECT_LIST F("Дыхание,Цвета,Снегопад,Шарик,Радуга,Радуга пикс,Огонь,The Matrix,Шарики,Часы,Звездопад,Конфетти,Радуга диагональная,Цветной шум,Облака,Лава,Плазма,Радужные переливы,Полосатые переливы,Зебра,Шумящий лес,Морской прибой,Лампа,Рассвет,Анимация 1,Анимация 2,Анимация 3,Анимация 4,Анимация 5")
#define ALARM_LIST F("Снегопад,Шарик,Радуга,Огонь,The Matrix,Шарики,Звездопад,Конфетти,Радуга диагональная,Цветной шум,Облака,Лава,Плазма,Радужные переливы,Полосатые переливы,Зебра,Шумящий лес,Морской прибой,Рассвет,Анимация 1,Анимация 2,Анимация 3,Анимация 4,Анимация 5")
#define GAME_LIST F("Змейка,Тетрис,Лабиринт,Runner,Flappy Bird,Арканоид")

#define D_TEXT_SPEED 100 // скорость бегущего текста по умолчанию (мс)
#define D_TEXT_SPEED_MIN 10
#define D_TEXT_SPEED_MAX 255

#define D_EFFECT_SPEED 80 // скорость эффектов по умолчанию (мс)
#define D_EFFECT_SPEED_MIN 0
#define D_EFFECT_SPEED_MAX 255

#define D_GAME_SPEED 250 // скорость игр по умолчанию (мс)
#define D_GAME_SPEED_MIN 25
#define D_GAME_SPEED_MAX 375

#define D_GIF_SPEED 100 // скорость гифок (мс)
#define DEMO_GAME_SPEED 60 // скорость игр в демо режиме (мс)

boolean AUTOPLAY = 1; // 0 выкл / 1 вкл автоматическую смену режимов (откл. можно со смартфона)
#define AUTOPLAY_PERIOD 30 // время между авто сменой режимов (секунды)
#define IDLE_TIME 10 // время бездействия кнопок или Bluetooth (в минутах) после которого запускается автосмена режимов и демо в играх

#define GLOBAL_COLOR_1 CRGB::Green // основной цвет №1 для игр
#define GLOBAL_COLOR_2 CRGB::Orange // основной цвет №2 для игр

#define SCORE_SIZE 0 // размер символов счёта в игре. 0 - маленький для 8x8 (шрифт 3x5), 1 - большой (шрифт 5x7)
#define FONT_TYPE 1 // (0 / 1) два вида маленького шрифта в выводе игрового счёта
#define SMOOTH_CHANGE 0 // плавная смена режимов через чёрный

// ***** ОТКЛЮЧЕНИЕ КОМПОНЕНТОВ СИСТЕМЫ (для экономии памяти) *****

#define USE_NOISE_EFFECTS 1 // крутые полноэкранные эффекты (0 нет, 1 да) СИЛЬНО ЖРУТ ПАМЯТЬ!!!
#define OVERLAY_CLOCK 1 // часы на фоне всех эффектов и игр. Жрёт SRAM память!
#define USE_FONTS 1 // использовать буквы (бегущая строка) (0 нет, 1 да)

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		49

```

// включение / отключение игр
#define USE_SNAKE 1          // игра змейка (0 нет, 1 да)
#define USE_TETRIS 1         // игра тетрис (0 нет, 1 да)
#define USE_MAZE 1           // игра лабиринт (0 нет, 1 да)
#define USE_RUNNER 1         // игра бегалка-прыгалка (0 нет, 1 да)
#define USE_FLAPPY 1         // игра flappy bird
#define USE_ARKAN 1          // игра арканойд

#define DEBUG 0
#define NUM_LEDS WIDTH * HEIGHT * SEGMENTS

#if (USE_FONTS == 1)
#include "fonts.h"
#endif

CRGB leds[NUM_LEDS];

// Ориентация отображения часов
byte CLOCK_ORIENT = 0;      // 0 горизонтальные, 1 вертикальные

// Макрос центрирования отображения часов на матрице
#define CLOCK_X_H (byte((WIDTH - (4*3 + 3*1)) / 2.0 + 0.51)) // 4 цифры * (шрифт 3
пикс шириной) 3 + пробела между цифрами) / 2 - в центр
#define CLOCK_Y_H (byte((HEIGHT - (1*5)) / 2.0 + 0.51))      // Одна строка цифр 5
пикс высотой / 2 - в центр
#define CLOCK_X_V (byte((WIDTH - (2*3 + 1)) / 2.0 + 0.51))   // 2 цифры * (шрифт 3
пикс шириной) 1 + пробел между цифрами) / 2 - в центр
#define CLOCK_Y_V (byte((HEIGHT - (2*5 + 1)) / 2.0 + 0.51))   // Две строки цифр 5 пикс
высотой + 1 пробел между строками / 2 - в центр

// Макрос центрирования отображения календаря на матрице
#define CAL_X (byte((WIDTH - (4*3 + 1)) / 2.0))              // 4 цифры * (шрифт 3
пикс шириной) 1 + пробел между цифрами) / 2 - в центр
#define CAL_Y (byte((HEIGHT - (2*5 + 1)) / 2.0))            // Две строки цифр 5 пикс
высотой + 1 пробел между строками / 2 - в центр

// Позиции отображения часов и календаря
byte CALENDAR_X = CAL_X;
byte CALENDAR_Y = CAL_Y;
byte CLOCK_X = CLOCK_X_H;      // Для вертикальных часов CLOCK_X_V и CLOCK_Y_V
byte CLOCK_Y = CLOCK_Y_H;

// Режим цвета часов
byte COLOR_MODE = 0;
//
// 0 - монохромные цвета, определенные для режимов
NORMAL_CLOCK_COLOR, CONTRAST_COLOR_1, CONTRAST_COLOR_2, CONTRAST_COLOR_3 в clock.ino
//
// 1 - радужная смена (каждая цифра)
//
// 2 - радужная смена (часы, точки, минуты)
//
// 3 - заданные цвета (часы, точки, минуты) - HOUR_COLOR,
DOT_COLOR, MIN_COLOR в clock.ino

// Порядок следования эффектов и игр в демо-режиме (см. customModes() в custom.ino) ID
с 0 до MODES_AMOUNT-1
// Если требуется изменить порядок следования эффектов в демо-режиме - (а также доба-
вить/удалить режимы) - редактируйте

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		50

```

// последовательность тут. Наличие - тут и в customModes() в custom.ino
#define DEMO_TEXT_0          0
#define DEMO_TEXT_1          1
#define DEMO_TEXT_2          2
#define DEMO_NOISE_MADNESS   3
#define DEMO_NOISE_CLOUD     4
#define DEMO_NOISE_LAVA       5
#define DEMO_NOISE_PLASMA     6
#define DEMO_NOISE_RAINBOW    7
#define DEMO_NOISE_RAINBOW_STRIP 8
#define DEMO_NOISE_ZEBRA      9
#define DEMO_NOISE_FOREST     10
#define DEMO_NOISE_OCEAN      11
#define DEMO_SNOW              12
#define DEMO_SPARKLES          13
#define DEMO_MATRIX            14
#define DEMO_STARFALL          15
#define DEMO_BALL              16
#define DEMO_BALLS             17
#define DEMO_RAINBOW           18
#define DEMO_RAINBOW_DIAG      19
#define DEMO_FIRE              20
#define DEMO_SNAKE             21
#define DEMO_TETRIS            22
#define DEMO_MAZE              23
#define DEMO_RUNNER            24
#define DEMO_FLAPPY            25
#define DEMO_ARKANOID          26
#define DEMO_CLOCK             27
#define DEMO_FILL_COLOR        28 // Заливка матрицы одним цветом
#define DEMO_DAWN_ALARM        29 // Режим эффекта будильника "Рассвет"
// -----
#define DEMO_ANIMATION_1       30
#define DEMO_ANIMATION_2       31
#define DEMO_ANIMATION_3       32
#define DEMO_ANIMATION_4       33
#define DEMO_ANIMATION_5       34

// не забудьте указать количество режимов для корректного переключения с последнего на
// первый
// количество кастомных режимов (которые переключаются сами или кнопкой)
#define MODES_AMOUNT 35

// -----
#define DEMO_DAWN_ALARM_SPIRAL 253 // Специальный режим, вызывается из DEMO_DAWN_ALARM
// для ламп на круговой матрице - огонек по спирали
#define DEMO_DAWN_ALARM_SQUARE 254 // Специальный режим, вызывается из DEMO_DAWN_ALARM
// для плоских матриц - огонек по спирали на плоскости матрицы
// -----

// Сквозная нумерация (ID) эффектов в группе эффектов
#define EFFECT_BREATH          0
#define EFFECT_COLOR           1
#define EFFECT_SNOW            2
#define EFFECT_BALL            3
#define EFFECT_RAINBOW         4
#define EFFECT_RAINBOW_PIX     5

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		51

```

#define EFFECT_FIRE                6
#define EFFECT_MATRIX              7
#define EFFECT_BALLS               8
#define EFFECT_CLOCK               9
#define EFFECT_STARFALL            10
#define EFFECT_SPARKLES            11
#define EFFECT_RAINBOW_DIAG        12
#define EFFECT_NOISE_MADNESS       13
#define EFFECT_NOISE_CLOUD         14
#define EFFECT_NOISE_LAVA          15
#define EFFECT_NOISE_PLASMA        16
#define EFFECT_NOISE_RAINBOW       17
#define EFFECT_NOISE_RAINBOW_STRIP 18
#define EFFECT_NOISE_ZEBRA         19
#define EFFECT_NOISE_FOREST        20
#define EFFECT_NOISE_OCEAN         21
#define EFFECT_FILL_COLOR          22
#define EFFECT_DAWN_ALARM          23
#define EFFECT_ANIMATION_1         24
#define EFFECT_ANIMATION_2         25
#define EFFECT_ANIMATION_3         26
#define EFFECT_ANIMATION_4         27
#define EFFECT_ANIMATION_5         28

#define MAX_EFFECT                  29          // количество эффектов, определенных в
прошивке

// Сквозная нумерация (ID) игр в группе игр
#define GAME_SNAKE                  0
#define GAME_TETRIS                 1
#define GAME_MAZE                   2
#define GAME_RUNNER                 3
#define GAME_FLAPPY                 4
#define GAME_ARKANOID               5

#define MAX_GAME                    6          // количество игр, определенных в проши-
вке

// ID типа эффектов (тип группы - текст, игры имеют один ID типа на все подтипы)
#define MC_TEXT                     0
#define MC_CLOCK                    1
#define MC_GAME                     2
#define MC_NOISE_MADNESS            3
#define MC_NOISE_CLOUD              4
#define MC_NOISE_LAVA               5
#define MC_NOISE_PLASMA             6
#define MC_NOISE_RAINBOW            7
#define MC_NOISE_RAINBOW_STRIP      8
#define MC_NOISE_ZEBRA              9
#define MC_NOISE_FOREST             10
#define MC_NOISE_OCEAN              11
#define MC_SNOW                     12
#define MC_SPARKLES                 13
#define MC_MATRIX                   14
#define MC_STARFALL                 15
#define MC_BALL                     16
#define MC_BALLS                    17

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		52

```

#define MC_RAINBOW                18
#define MC_RAINBOW_DIAG           19
#define MC_FIRE                   20
#define MC_DAWN_ALARM             21
#define MC_FILL_COLOR             22
#define MC_IMAGE                  23

// Типы эффектов (см. выше), в которых могут отображаться часы в наложении
#if (OVERLAY_CLOCK == 1)
byte overlayList[] = {
    MC_NOISE_MADNESS,
    MC_NOISE_CLOUD,
    MC_NOISE_LAVA,
    MC_NOISE_PLASMA,
    MC_NOISE_RAINBOW,
    MC_NOISE_RAINBOW_STRIP,
    MC_NOISE_ZEBRA,
    MC_NOISE_FOREST,
    MC_NOISE_OCEAN,
    MC_SNOW,
    MC_SPARKLES,
    MC_MATRIX,
    MC_STARFALL,
    MC_BALL,
    MC_BALLS,
    MC_RAINBOW,
    MC_RAINBOW_DIAG,
    MC_FIRE,
    MC_DAWN_ALARM,
    MC_FILL_COLOR,
};
#endif

// ***** Глобальные переменные *****

// Соответствие индексов списка эффектов будильника ALARM_LIST индексам списка существ-
// вующих эффектов EFFECT_LIST
const byte ALARM_LIST_IDX[] PROGMEM = {EFFECT_SNOW, EFFECT_BALL, EFFECT_RAINBOW,
EFFECT_FIRE, EFFECT_MATRIX, EFFECT_BALLS,
                                     EFFECT_STARFALL, EFFECT_SPARKLES,
EFFECT_RAINBOW_DIAG, EFFECT_NOISE_MADNESS, EFFECT_NOISE_CLOUD,
                                     EFFECT_NOISE_LAVA, EFFECT_NOISE_PLASMA,
EFFECT_NOISE_RAINBOW, EFFECT_NOISE_RAINBOW_STRIP,
                                     EFFECT_NOISE_ZEBRA, EFFECT_NOISE_FOREST,
EFFECT_NOISE_OCEAN, EFFECT_DAWN_ALARM,
                                     EFFECT_ANIMATION_1, EFFECT_ANIMATION_2,
EFFECT_ANIMATION_3, EFFECT_ANIMATION_4, EFFECT_ANIMATION_5
                                     };

// ---- Подключение к сети

WiFiUDP udp;

bool useSoftAP = false;           // использовать режим точки доступа
bool wifi_connected = false;      // true - подключение к wifi сети выполнена
bool ap_connected = false;        // true - работаем в режиме точки доступа;

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		53

```

bool wifi_print_ip = false;          // В качестве бегущей строки отображается текущий IP
WiFi соединения
String wifi_current_ip = "";

// Буфер для загрузки имен/пароля из EEPROM текущих имени/пароля подключения к сети /
точке доступа
// Внимание: к длине +1 байт на \0 - терминальный символ.
char apName[17] = "";                // Имя сети в режиме точки доступа
char apPass[9] = "";                // Пароль подключения к точке доступа
char ssid[25] = "";                 // SSID (имя) вашего роутера (конфигурируется
подключением через точку доступа и сохранением в EEPROM)
char pass[17] = "";                 // пароль роутера

// ---- Синхронизация часов с сервером NTP

IPAddress timeServerIP;
#define NTP_PACKET_SIZE 48          // NTP время в первых 48 байтах сообщения
uint16_t SYNC_TIME_PERIOD = 60;    // Период синхронизации в минутах
byte packetBuffer[NTP_PACKET_SIZE]; // буффер для хранения входящих и исходящих
пакетов

int8_t timeZoneOffset = 7;          // смещение часового пояса от UTC
long ntp_t = 0;                     // Время, прошедшее с запроса данных с NTP-
сервера (таймаут)
byte ntp_cnt = 0;                   // Счетчик попыток получить данные от сервера
bool init_time = false;             // Флаг false - время не инициализировано;
true - время инициализировано
bool refresh_time = true;           // Флаг true - пришло время выполнить синхро-
низацию часов с сервером NTP
bool useNtp = true;                 // Использовать синхронизацию времени с NTP-
сервером
char ntpServerName[31] = "";        // Используемый сервер NTP

// ---- Часы и будильник

bool isAlarming = false;            // Сработал будильник "рассвет"
bool isPlayAlarmSound = false;      // Сработал настоящий будильник - играет звук бу-
дильника
bool isAlarmStopped = false;        // Сработавший будильник "рассвет" остановлен поль-
зователем

byte alarmWeekDay = 0;              // Битовая маска дней недели будильника
byte alarmDuration = 1;             // Проигрывать звук будильнике N минут после сра-
батывания (по окончанию рассвета)

byte alarmHour[7] = {0, 0, 0, 0, 0, 0, 0}; // Часы времени срабатывания будильника
byte alarmMinute[7] = {0, 0, 0, 0, 0, 0, 0}; // Минуты времени срабатывания будильника

int8_t dawnHour = 0;               // Часы времени начала рассвета
int8_t dawnMinute = 0;             // Минуты времени начала рассвета
byte dawnWeekDay = 0;              // День недели времени начала рассвета (0 - выклю-
чено, 1..7 - пн..вс)
byte dawnDuration = 0;              // Продолжительность "рассвета" по настройкам
byte realDawnDuration = 0;         // Продолжительность "рассвета" по вычисленому вре-
мени срабатывания будильника
bool useAlarmSound = false;        // Использовать звуки в будильнике

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		54

```

int8_t alarmSound = 0;          // Звук будильника - номер файла в папке SD:/01 [-1
не использовать; 0 - случайный; 1..N] номер файла
int8_t dawnSound = 0;          // Звук рассвета - номер файла в папке SD:/02 [-1
не использовать; 0 - случайный; 1..N] номер файла
byte maxAlarmVolume = 30;      // Максимальная громкость будильника (1..30)
byte alarmEffect = EFFECT_DAWN_ALARM; // Какой эффект используется для будильника
"рассвет". Могут быть обычные эффекты - их яркость просто будет постепенно увеличи-
ваться

boolean showDateInClock = true; // Показывать дату при отображении часов
byte showDateDuration = 5;      // на 5 секунд
byte showDateInterval = 20;    // через каждые 20 секунд
byte showDateState = false;    // false - отображаются часы; true - отображается
дата
long showDateStateLastChange = 0; // Время, когда отображение часов сменилось на отоб-
ражение календаря и наоборот

int8_t hrs = 0, mins = 0, secs = 0, aday = 1, amnth = 1;
int16_t ayear = 1970;
boolean dotFlag;               // Флаг отрисовки разделяющих точек в часах

// ---- Спец.режимы (при включении спец-режимов отключается автосмена режимов и демо)

int8_t specialModeId = -1;     // Номер текущего спец.режима
bool specialMode = false;     // Спец.режим, включенный вручную со смартфона или с
кнопок быстрого включения режима
bool specialClock = false;     // Спец.режим использует overlay часов
byte specialBrightness = 1;    // Яркость в спец.режиме
bool isNightClock = false;     // Включен режим ночных часов с минимальной яркостью
bool isTurnedOff = false;     // Включен черный экран (т.е всё выключено)

// ---- Авторегулировка яркости

GFilterRA brightness_filter;  // фильтр для плавного изменения яркости
bool useAutoBrightness = false; // Использовать автоматическую регулировку яркости
byte autoBrightnessMin = 1;    // Минимальный уровень яркости при автоматической
регуливке

// ---- Режимы, включаемые в заданное время

bool AM1_running = false;     // Режим 1 по времени - работает
byte AM1_hour = 0;            // Режим 1 по времени - часы
byte AM1_minute = 0;          // Режим 1 по времени - минуты
int8_t AM1_effect_id = -5;    // Режим 1 по времени - ID эффекта или -5 - выключе-
нено; -4 - выключить матрицу (черный экран); -3 - ночные часы, -2 - камин с часами, -1
- бегущая строка, 0 - случайныйб 1 и далее - эффект ALARM_LIST
bool AM2_running = false;     // Режим 2 по времени - работает
byte AM2_hour = 0;            // Режим 2 по времени - часы
byte AM2_minute = 0;          // Режим 2 по времени - минуты
int8_t AM2_effect_id = -5;    // Режим 2 по времени - ID эффекта или -5 - выключе-
нено; -4 - выключить матрицу (черный экран); -3 - ночные часы, -2 - камин с часами, -1
- бегущая строка, 0 - случайныйб 1 и далее - эффект ALARM_LIST

// ---- Таймеры

uint32_t idleTime = ((long)IDLE_TIME * 60 * 1000); // минуты -> миллисек
uint32_t autoplayTime = ((long)AUTOPLAY_PERIOD * 1000); // секунды -> миллисек

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		55

```

uint32_t autoplayTimer;

timerMinim effectTimer(D_EFFECT_SPEED);           // Таймер скорости эффекта
(шага выполнения эффекта)
timerMinim gameTimer(DEMO_GAME_SPEED);           // Таймер скорости игры (шага
выполнения игры)
timerMinim scrollTimer(D_TEXT_SPEED);             // Таймер прокрутки текста эф-
фекта бегущей строки
timerMinim changeTimer(70);                      // Таймер шага плавной смены
режима - Fade
timerMinim halfsecTimer(500);                   // Полусекундный таймер точек
часов
timerMinim idleTimer(idleTime);                 // Таймер бездействия ручного
управления для автоперехода а демо-режим
timerMinim alarmSoundTimer(4294967295);         // Таймер выключения звука бу-
дильника
timerMinim fadeSoundTimer(4294967295);          // Таймер плавного включения /
выключения звука
timerMinim autoBrightnessTimer(500);            // Таймер отслеживания показа-
ний датчика света при включенной авторегулировки яркости матрицы
timerMinim saveSettingsTimer(30000);            // Таймер отложенного сохране-
ния настроек
timerMinim ntpSyncTimer(1000 * 60 * SYNC_TIME_PERIOD); // Таймер синхронизации времени
с NTP-сервером
timerMinim dawnTimer(4294967295);              // Таймер шага рассвета для бу-
дильника "рассвет"

// ---- MP3 плеер для проигрывания звуков будильника

SoftwareSerial mp3Serial(SRX, STX);
DFRobotDFPlayerMini dfPlayer;
bool isDfPlayerOk = false;
int16_t alarmSoundsCount = 0;                  // Кол-во файлов звуков в папке '01' на SD-карте
int16_t dawnSoundsCount = 0;                  // Кол-во файлов звуков в папке '02' на SD-карте
byte soundFolder = 0;
byte soundFile = 0;
int8_t fadeSoundDirection = 1;                // направление изменения громкости звука: 1 - уве-
личение; -1 - уменьшение
byte fadeSoundStepCounter = 0;                // счетчик шагов изменения громкости, которое
осталось сделать

// ---- Физическая кнопка управления режимами

GButton butt(PIN_BTN);                        // Физическая кнопка
#define HOLD_TIMEOUT 2000                      // Время удержания кнопки перед выполнением дей-
ствия ( + debounce time) суммарно - около 3 сек
bool isButtonHold = false;                    // Кнопка нажата и удерживается
long hold_start_time = 0;                     // Время обнаружения состояния "Конпка нажата и
удерживается"

// ---- Прочие переменные

int globalBrightness = BRIGHTNESS;           // Текущая яркость
byte breathBrightness;                       // Яркость эффекта "Дыхание"
uint32_t globalColor = 0xffffffff;           // Цвет рисования при запуске белый

int scrollSpeed = D_TEXT_SPEED;               // скорость прокрутки текста бегущей строки

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				56
Змн.	Арк.	№ докум.	Підпис	Дата		

```

int gameSpeed = DEMO_GAME_SPEED;    // скорость в играх
int effectSpeed = D_EFFECT_SPEED;    // скорость изменения эффекта

boolean BTcontrol = false;           // флаг: true - ручное управление эффектами и
игры; false - в режиме Autoplay
boolean loadingFlag = true;          // флаг: выполняется инициализация параметров режима
boolean runningFlag;                 // флаг: текущий режим ручном режиме - бегущая
строка
boolean drawingFlag;                  // флаг: текущий режим в ручном режиме - рисование
или картинка на матрице
boolean effectsFlag;                  // флаг: текущий режим в ручном режиме - эффекты
boolean gamemodeFlag = false;         // флаг: текущий режим - игра
boolean controlFlag = false;          // флаг: управление игрой перехвачено с кнопок или
смартфона
boolean gamePaused;                  // флаг: игра приостановлена
boolean gameDemo = true;              // флаг: игра в демо-режиме
boolean idleState = true;             // флаг холостого режима работы
boolean fullTextFlag = false;         // флаг: текст бегущей строки показан полностью
(строка убежала)
boolean eepromModified = false;       // флаг: данные, сохраняемые в EEPROM были изменены
и требуют сохранения

byte game;                           // индекс текущей игры в ручном режиме
byte effect;                          // индекс текущего эффекта в ручном режиме

byte modeCode;                       // тип текущего эффекта: 0 бегущая строка, 1 часы, 2
игры, 3 нойс маднесс и далее, 21 гифка или картинка,
int8_t thisMode = 0;                  // текущий режим

byte frameNum;                       // Номер фрейма проигрывания анимации
byte buttons = 4;                     // Управление играми: нажата кнопка: 0 - верх, 1 -
право, 2 - низ, 3 - лево, 4 - не нажата

#if (SMOOTH_CHANGE == 1)
byte fadeMode = 4;                    // фаза плавного смена режима
boolean modeDir;                      // направление: затухание / проявление
#endif

// Текст для демо-режимов "бегущая строка"
String TEXT_1 = "";                  // Строка заданного цвета
String TEXT_2 = "";                  // Строка с плавной сменой цвета
String TEXT_3 = "";                  // Строка с разноцветными буквами
String runningText = "";              // Текущий текст бегущей строки, задаваемый пользо-
вателем со смартфона

// Сервер не может инициировать отправку сообщения клиенту - только в ответ на запрос
клиента
// Следующие две переменные хранят сообщения, формируемые по инициативе сервера и
отправляются в ответ на ближайший запрос от клиента,
// например в ответ на периодический ping - в команде sendAcknowledge();
String cmd95 = "";                   // Строка, формируемая sendPageParams(95) для отпра-
вки по инициативе сервера
String cmd96 = "";                   // Строка, формируемая sendPageParams(96) для отпра-
вки по инициативе сервера

void setup() {

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		57

```

// Watcdog Timer - 8 секунд
ESP.wdtEnable(WDTO_8S);

// Инициализация последовательного порта
Serial.begin(115200);
delay(10);

// Инициализация EEPROM и загрузка сохраненных параметров
EEPROM.begin(512);

// Проверяем, что при старте зажата кнопка
if(!digitalRead(PIN_BTN)) {

    Serial.println(F("\nКнопка зажата. Ждем 5 секунд.\n"));

    int cnt = 0;
    while( cnt++ < 5 ) {

        if (digitalRead(PIN_BTN)) {
            Serial.println(F("\nКнопку отпустили.\n"));
            break;
        }

        Serial.print(cnt);
        Serial.print(".");

        ESP.wdtFeed();
        delay(1000);
    }

    // Если держали 5 и более секунд, загружаем дефолтные настройки
    if ( cnt == 6 ) {
        Serial.println(F("\nСброс настроек\n"));

        // Устанавливаем параметры Wi-Fi по-умолчанию
        strcpy(ssid, NETWORK_SSID);
        strcpy(pass, NETWORK_PASS);
        setSsid(ssid);
        setPass(pass);

        // Сбрасываем в дефолт
        saveDefaults();
    }
}

loadSettings();

// Пинаем генератор случайных чисел
randomSeed(analogRead(1));

// Первый этап инициализации плеера - подключение и основные настройки
InitializeDfPlayer1();

// WiFi всегда включен
WiFi.setSleepMode(WIFI_NONE_SLEEP);

// Выполнить подключение к сети / создание точки доступа

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				58
Змн.	Арк.	№ докум.	Підпис	Дата		

```

connectToNetwork();

// UDP-клиент на указанном порту
udp.begin(localPort);

// Таймер бездействия
idleTimer.setInterval(idleTime == 0 ? 4294967295 : idleTime);

// Настройки ленты
FastLED.addLeds<WS2812, LED_PIN, COLOR_ORDER>(leds, NUM_LEDS).setCorrection(
TypicalLEDStrip );
FastLED.setBrightness(globalBrightness);
if (CURRENT_LIMIT > 0) FastLED.setMaxPowerInVoltsAndMilliamps(5, CURRENT_LIMIT);
FastLED.clear();
FastLED.show();

// Второй этап инициализации плеера - проверка наличия файлов звуков на SD карте
InitializeDfPlayer2();
if (!isDfPlayerOk) {
    Serial.println(F("MP3 плеер недоступен."));
}

// Проверить соответствие позиции вывода часов и календаря размерам матрицы
checkClockOrigin();

/*
    butt.setDebounce(50);          // настройка антидребезга (по умолчанию 80 мс)
    butt.setTimeout(300);          // настройка таймаута на удержание (по умолчанию 500
мс)
    butt.setIncrStep(2);           // настройка инкремента, может быть отрицательным (по
умолчанию 1)
    butt.setIncrTimeout(500);      // настройка интервала инкремента (по умолчанию 800
мс)
*/

// Текст для демо-режимов "бегущая строка"
TEXT_1 = String(F("Матрица"));    // Строка заданного цвета
TEXT_2 = String(F("на адресных")); // Строка с плавной сменой цвета
TEXT_3 = String(F("светодиодах")); // Строка с разноцветными буквами

// Установка коэффициента фильтрации (0.0... 1.0). Чем меньше, тем плавнее фильтр
brightness_filter.setCoef(0.1);

// Если был задан спец.режим во время предыдущего сеанса работы матрицы - включить
его
// Номер спец-режима запоминается при его включении и сбрасывается при включении
обычного режима или игры
// Это позволяет в случае внезапной перезагрузки матрицы (например по wdt), когда был
включен спец-режим (например ночные часы или выкл. матрицы)
// снова включить его, а не отображать случайный обычный после включения матрицы
int8_t spc_mode = getCurrentSpecMode();
if (spc_mode >= 0 && spc_mode <= 7)
    setSpecialMode(spc_mode);
else {
    while (1) {
        // Если режим отмечен флагом "использовать" - используем его, иначе берем следую-
щий (и проверяем его)

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		59

```

        if (getUsageForMode(thisMode)) break;
        thisMode++;
        if (thisMode >= MODES_AMOUNT) {
            thisMode = 0;
            break;
        }
    }
    setTimersForMode(thisMode);
}

void loop() {
    bluetoothRoutine();
    ESP.wdtFeed();    // пнуть собаку
    yield();
}

// -----

void startWiFi() {

    WiFi.disconnect(true);
    wifi_connected = false;
    WiFi.mode(WIFI_AP_STA);

    // Пытаемся соединиться с роутером в сети
    if (strlen(ssid) > 0) {
        Serial.print(F("\nПодключение к "));
        Serial.print(ssid);

        if (IP_STA[0] + IP_STA[1] + IP_STA[2] + IP_STA[3] > 0) {
            WiFi.config(IPAddress(IP_STA[0], IP_STA[1], IP_STA[2], IP_STA[3]), //
192.168.0.106
                        IPAddress(IP_STA[0], IP_STA[1], IP_STA[2], 1),          //
192.168.0.1
                        IPAddress(255, 255, 255, 0));
        }
        WiFi.begin(ssid, pass);

        // Проверка соединения (таймаут 10 секунд)
        for (int j = 0; j < 10; j++ ) {
            wifi_connected = WiFi.status() == WL_CONNECTED;
            if (wifi_connected) {
                // Подключение установлено
                Serial.println();
                Serial.print(F("WiFi подключен. IP адрес: "));
                Serial.println(WiFi.localIP());
                break;
            }
            ESP.wdtFeed();
            delay(1000);
            Serial.print(".");
        }
        Serial.println();

        if (!wifi_connected)
            Serial.println(F("Не удалось подключиться к сети WiFi.));
    }

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				60
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }
}

void startSoftAP() {
    WiFi.softAPdisconnect(true);
    ap_connected = false;

    Serial.print(F("Создание точки доступа "));
    Serial.print(apName);

    ap_connected = WiFi.softAP(apName, apPass);

    for (int j = 0; j < 10; j++ ) {
        if (ap_connected) {
            Serial.println();
            Serial.print(F("Точка доступа создана. Сеть: '"));
            Serial.print(apName);
            // Если пароль совпадает с паролем по умолчанию - печатать для информации,
            // если был изменен пользователем - не печатать
            if (strcmp(apPass, "12341234") == 0) {
                Serial.print(F("' . Пароль: '"));
                Serial.print(apPass);
            }
            Serial.println(F("' ."));
            Serial.print(F("IP адрес: "));
            Serial.println(WiFi.softAPIP());
            break;
        }

        WiFi.enableAP(false);
        WiFi.softAPdisconnect(true);
        ESP.wdtFeed();
        delay(1000);

        Serial.print(".");
        ap_connected = WiFi.softAP(apName, apPass);
    }
    Serial.println();

    if (!ap_connected)
        Serial.println(F("Не удалось создать WiFi точку доступа."));
}

bool startSmartConfig() {
    Serial.print(F("\nЗапуск режима SmartConfig "));

    // Устанавливаем режим клиента
    WiFi.mode(WIFI_STA);
    WiFi.hostname((String("ShpetMatrix") + WiFi.macAddress()).c_str());

    // Отключаем Wi-Fi и забываем настройки
    WiFi.disconnect(true);

    WiFi.beginSmartConfig();

    int smartCount = 0;
    bool smartDone = false;

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		61

```

while (smartCount++ < 60) { // Пробуем получить настройки 60 раз

    if (WiFi.smartConfigDone()) {

        Serial.print(F("\nАвтонастройка успешно завершена \n"));
        Serial.printf("\tSSID = %s\n" , WiFi.SSID().c_str());
        Serial.printf("\tPSK = %s\n\n" , WiFi.psk().c_str());

        // Сохраняем SSID и пароль, присланный из приложения
        setSsid(WiFi.SSID());
        setPass(WiFi.psk());

        // Сбрасываем настройки IP для активации DHCP
        saveStaticIP(0, 0, 0, 0);

        return true;
    }

    ESP.wdtFeed();
    delay(500);
    Serial.print(".");
}

WiFi.stopSmartConfig();
Serial.print(F("\nАвтонастройка не удалась\n"));
ESP.wdtFeed();
return false;
}

void printNtpServerName() {
    Serial.print(F("NTP-сервер "));
    Serial.print(ntpServerName);
    Serial.print(F(" -> "));
    Serial.println(timeServerIP);
}

void connectToNetwork() {
    // Подключиться к WiFi сети
    startWiFi();

    if (!wifi_connected) {

        // Пробуем получить настройки через ESP8255 SmartConfig
        if (startSmartConfig()) {
            // Удалось, поднимаем Wi-Fi
            startWiFi();
        } else {
            // Если режим точки доступа не используется и к WiFi сети подключиться не удалось
            - создать точку доступа
            WiFi.mode(WIFI_AP);
            startSoftAP();
        }
    }
}

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				62
Змн.	Арк.	№ докум.	Підпис	Дата		

```

if (useSoftAP && !ap_connected) startSoftAP();

// Сообщить UDP порт, на который ожидаются подключения
if (wifi_connected || ap_connected) {
    Serial.print(F("UDP-сервер на порту "));
    Serial.println(localPort);
}
}

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				63
Змн.	Арк.	№ докум.	Підпис	Дата		

```

const uint16_t mario0[] PROGMEM = {
0x0000, 0x0000, 0x0000, 0x0000, 0x8000, 0xF800, 0xF800, 0xF800, 0xF800,
0xF820, 0x8000, 0x8000, 0x8000, 0x1040, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x9220, 0xF06C, 0xF827, 0xF860, 0xFB40, 0xFB40,
0xF827, 0xFB40, 0xF820, 0xF820, 0xAB40, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x1040, 0xAB40, 0xAB40, 0xBC20, 0xFF20, 0xFF20,
0x8B80, 0xF6ED, 0xAC80, 0xAC80, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x1040, 0xAB40, 0xF6ED, 0xAB40, 0xF6ED, 0xFF40, 0xFF40,
0x93A0, 0xF6ED, 0xFF40, 0xFF40, 0xC520, 0x0000, 0x0000,
0x0000, 0x0000, 0x1040, 0xAB40, 0xF6ED, 0xBC20, 0xBC40, 0xFF40, 0xFF40,
0xF6ED, 0xAC80, 0xF70C, 0xFF20, 0xFF40, 0x93A0, 0x0000,
0x0000, 0x0000, 0x0000, 0x8000, 0xBC20, 0xF70C, 0xFF20, 0xFF40, 0xFF40,
0xAC80, 0x8B60, 0x93A0, 0x93A0, 0xAC80, 0x1040, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x1040, 0xFF20, 0xFF20, 0xFF20, 0xFF20,
0xFF20, 0xFF20, 0xF70C, 0x93A0, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x9220, 0xF800, 0xFA80, 0xF800, 0x5B9E, 0xF800,
0xF860, 0x8B60, 0x8B60, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0xAB00, 0xF800, 0xF800, 0xF801, 0xF800, 0x329F, 0x5B9E,
0xF800, 0xF06C, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0xAB00, 0xF800, 0xF800, 0xF827, 0x725D, 0x331F, 0xEF6F,
0x627E, 0x627E, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0xAB40, 0xF827, 0xF801, 0xF801, 0xF16C, 0x535E, 0x433F,
0x0ABF, 0x429F, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x1040, 0x725D, 0xF06C, 0xF940, 0xFEE0, 0xF70C, 0x331F,
0x02BF, 0x627E, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x1040, 0x7A5D, 0xF16C, 0xFF20, 0xF70C, 0x331F,
0x329F, 0x0820, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x725D, 0x6A7E, 0x6BDE, 0x6BDE, 0x531E,
0x627E, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x9220, 0x9A60, 0x9A40, 0x8A00, 0x8A00,
0x8A00, 0x1040, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x9200, 0x9A60, 0x9A60, 0x9A60, 0x8A00,
0x8A00, 0x8A00, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000
};

```

```

const uint16_t mario1[] PROGMEM = {
0x0000, 0x0000, 0x0000, 0x0000, 0x8000, 0xF800, 0xF800, 0xF800, 0xF800,
0xF826, 0x8000, 0x8000, 0x8000, 0x1040, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x9200, 0xF06B, 0xF827, 0xF860, 0xF960, 0xF940,
0xF868, 0xF802, 0xF826, 0xF826, 0xAB40, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x1040, 0xAB40, 0xAB40, 0xBC20, 0xFF20, 0xFF20,
0x8B80, 0xEEA0, 0xAC80, 0xAC80, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x1040, 0xAB40, 0xEEA0, 0xAB40, 0xEEA0, 0xFF40, 0xFF40,
0x93A0, 0xEEA0, 0xFF40, 0xFF40, 0xC520, 0x0000, 0x0000,

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				64
Змн.	Арк.	№ докум.	Підпис	Дата		

```

0x0000, 0x0000, 0x1040, 0xAB40, 0xEEA0, 0xBC20, 0xBC40, 0xFF40, 0xFF40,
0xEEA0, 0xAC80, 0xFF20, 0xFF20, 0xFF40, 0x93A0, 0x0000,
0x0000, 0x0000, 0x0000, 0x8000, 0xBC20, 0xFF20, 0xFF20, 0xFF40, 0xFF40,
0xAC80, 0x8B60, 0x93A0, 0x93A0, 0xAC80, 0x1040, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x1040, 0xFF20, 0xFF20, 0xFF20, 0xFF20,
0xFF20, 0xFF20, 0xFF20, 0x93A0, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x8B60, 0xF826, 0xF826, 0x627E,
0x529F, 0xF06D, 0x8B60, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x1040, 0x8B60, 0x8B60, 0xF868, 0xF800, 0xF800, 0xF849,
0x627E, 0xF849, 0xFF20, 0xFF40, 0xB4E0, 0x1040, 0x0000,
0x0000, 0x0000, 0xB4E0, 0xFF40, 0xFF20, 0xF868, 0xF801, 0xF800, 0xF801,
0xF868, 0xF940, 0xFF20, 0xFF40, 0xFF20, 0x93A0, 0x0000,
0x0000, 0x93A0, 0xFF20, 0xFF20, 0xFF20, 0x627E, 0xF06C, 0xF868, 0xF868,
0xF868, 0xF868, 0xFF20, 0xFF20, 0xAC80, 0x0000, 0x0000,
0x0000, 0x1040, 0xAB40, 0xAB40, 0x627E, 0x0ABF, 0x3B1F, 0x3B1F, 0x3B1F,
0x3B1F, 0x3B1F, 0x627E, 0x1040, 0x0000, 0x0000, 0x0000,
0x0000, 0x1040, 0x9A60, 0x9240, 0x529F, 0x02BF, 0x02BF, 0x02BF, 0x02BF,
0x02BF, 0x02BF, 0x3B1F, 0x0820, 0x0000, 0x0000, 0x0000,
0x0000, 0x1040, 0x9A60, 0x9240, 0x529F, 0x2A9F, 0x7298, 0x7298, 0x529F,
0x529F, 0x627E, 0x1040, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x1040, 0x9200, 0x1040, 0x1040, 0x0820, 0x0000, 0x0000, 0x9A60,
0x9A60, 0x9200, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x1040, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x9A60,
0x9A60, 0x9A60, 0x9240, 0x1040, 0x0000, 0x0000, 0x0000
};

```

```

const uint16_t mario2[] PROGMEM = {
0x0000, 0x0000, 0x0000, 0x0000, 0x8000, 0xF800, 0xF800, 0xF800, 0xF800,
0xF820, 0x8000, 0x8000, 0x8000, 0x1040, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x9220, 0xF06C, 0xF827, 0xF860, 0xFB40, 0xFB40,
0xF827, 0xFB40, 0xF820, 0xF820, 0xAB40, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x1040, 0xAB40, 0xAB40, 0xBC20, 0xFF20, 0xFF20,
0x8B80, 0xF6ED, 0xAC80, 0xAC80, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x1040, 0xAB40, 0xF6ED, 0xAB40, 0xF6ED, 0xFF40, 0xFF40,
0x93A0, 0xF6ED, 0xFF40, 0xFF40, 0xC520, 0x0000, 0x0000,
0x0000, 0x0000, 0x1040, 0xAB40, 0xF6ED, 0xBC20, 0xBC40, 0xFF40, 0xFF40,
0xF6ED, 0xAC80, 0xF70C, 0xFF20, 0xFF40, 0x93A0, 0x0000,
0x0000, 0x0000, 0x0000, 0x8000, 0xBC20, 0xF70C, 0xFF20, 0xFF40, 0xFF40,
0xAC80, 0x8B60, 0x93A0, 0x93A0, 0xAC80, 0x1040, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x1040, 0xFF20, 0xFF20, 0xFF20, 0xFF20,
0xFF20, 0xFF20, 0xF70C, 0x93A0, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x9220, 0xF800, 0xFA80, 0xF800, 0x5B9E, 0xF800,
0xF860, 0x8B60, 0x8B60, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0xAB00, 0xF800, 0xF800, 0xF801, 0xF800, 0x329F, 0x5B9E,
0xF800, 0xF06C, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		65

```

0x0000, 0x0000, 0xAB00, 0xF800, 0xF800, 0xF827, 0x725D, 0x331F, 0xEF6F,
0x627E, 0x627E, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0xAB40, 0xF827, 0xF801, 0xF801, 0xF16C, 0x535E, 0x433F,
0x0ABF, 0x429F, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x1040, 0x725D, 0xF06C, 0xF940, 0xFEE0, 0xF70C, 0x331F,
0x02BF, 0x627E, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x1040, 0x7A5D, 0xF16C, 0xFF20, 0xF70C, 0x331F,
0x329F, 0x0820, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x725D, 0x6A7E, 0x6BDE, 0x6BDE, 0x531E,
0x627E, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x9220, 0x9A60, 0x9A40, 0x8A00, 0x8A00,
0x8A00, 0x1040, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x9200, 0x9A60, 0x9A60, 0x9A60, 0x8A00,
0x8A00, 0x8A00, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
};

```

```

const uint16_t mario3[] PROGMEM = {
0x0000, 0x0000, 0x0000, 0x0000, 0x8000, 0xF800, 0xF800, 0xF800, 0xF800,
0xF806, 0x8000, 0x8000, 0x8000, 0x1040, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x9200, 0xF84A, 0xF827, 0xF860, 0xFB80, 0xFB80,
0xF828, 0xFB40, 0xF806, 0xF806, 0xAB40, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x1040, 0xABA0, 0xAB40, 0xBC20, 0xFF20, 0xFF00,
0x8B80, 0xF68E, 0xAC80, 0xAC80, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x1040, 0xAB40, 0xF68E, 0xABA0, 0xF70B, 0xFF40, 0xFF40,
0x93A0, 0xF70B, 0xFF40, 0xFF40, 0xC520, 0x0000, 0x0000,
0x0000, 0x0000, 0x1040, 0xAB40, 0xF70B, 0xBC00, 0xBC40, 0xFF40, 0xFF40,
0xEEA0, 0xAC80, 0xF70B, 0xFF20, 0xFF40, 0x93A0, 0x0000,
0x0000, 0x0000, 0x0000, 0x8000, 0xBC00, 0xF70B, 0xFF20, 0xFF40, 0xFF40,
0xAC80, 0x8B60, 0x93A0, 0x93A0, 0xAC80, 0x1040, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x1040, 0xFF00, 0xFF20, 0xFF20, 0xFF20,
0xFF00, 0xF70B, 0xF70B, 0x93A0, 0x0000, 0x0000, 0x0000,
0x0000, 0x9A60, 0xF8E0, 0xF8E0, 0xF8E0, 0xF827, 0x72FD, 0xF848, 0xFB20,
0xF826, 0x825D, 0x8B60, 0x0000, 0x0000, 0x0000, 0x0000,
0xFF40, 0xFF20, 0xFB80, 0xF804, 0xF800, 0xF827, 0x429F, 0x5AFE, 0xF848,
0xF826, 0xF08D, 0x72FD, 0xF848, 0xFF20, 0xFF40, 0xFF40,
0xFF40, 0xFF40, 0xF70B, 0x8000, 0xF06C, 0xF06C, 0x627E, 0xEF6F, 0x5AFE,
0x429F, 0x63DE, 0xEF6F, 0xF927, 0xF927, 0xFF00, 0xFF20,
0xFF20, 0xF70B, 0x93A0, 0x7A96, 0x5AFE, 0x3B1F, 0x12BF, 0x3B1F, 0x02BF,
0x02BF, 0x0ABF, 0x429F, 0x7A96, 0x1040, 0x9A60, 0x1040,
0x0000, 0x0000, 0x6A99, 0x1ABF, 0x02BF, 0x02BF, 0x02BF, 0x02BF, 0x02BF,
0x02BF, 0x02BF, 0x02BF, 0x429F, 0x9A62, 0x9A60, 0x1040,
0x0000, 0x1040, 0x429F, 0x0ABF, 0x02BF, 0x02BF, 0x02BF, 0x02BF, 0x02BF,
0x02BF, 0x02BF, 0x02BF, 0x429F, 0x9A62, 0x9A60, 0x1040,
0x1040, 0x9240, 0x7A96, 0x32BE, 0x12BF, 0x2A9F, 0x7A96, 0x7A96, 0x7A96,
0x7A96, 0x2A9F, 0x02BF, 0x429F, 0x9A62, 0x9A40, 0x1040,

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				66
Змн.	Арк.	№ докум.	Підпис	Дата		

```

0x1040, 0x9A60, 0x9A61, 0x9200, 0x7A96, 0x0820, 0x0000, 0x0000, 0x0000,
0x0000, 0x0820, 0x7A96, 0x1040, 0x1040, 0x1040, 0x0000,
0x0000, 0x1040, 0x9A60, 0x9A60, 0x9A40, 0x1040, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000
};

```

```

const uint16_t* const mario_array[4] PROGMEM = {
mario0, mario1, mario2, mario3};

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				67
Змн.	Арк.	№ докум.	Підпис	Дата		

```

// игра змейка!
#if (USE_SNAKE == 1)
// ***** НАСТРОЙКИ ЗМЕЙКИ *****
#define START_LENGTH 4 // начальная длина змейки
#define MAX_LENGTH 80 // максимальная длина змейки

int8_t vectorX, vectorY;
int8_t headX, headY, buttX, buttY;
int8_t appleX, appleY;
boolean apple_flag, missDelete = false;

int8_t buttVector[MAX_LENGTH];
int snakeLength;
boolean butt_flag, fail;

void snakeRoutine() {
    if (loadingFlag) {
        FastLED.clear();
        loadingFlag = false;
        newGameSnake();
        gamemodeFlag = true;
        modeCode = MC_GAME;
    }

    buttonsTickSnake();

    if (gameTimer.isReady()) {
        // БЛОК ГЕНЕРАЦИИ ЯБЛОКА
        while (!apple_flag) { // пока яблоко не создано
            appleX = random(0, WIDTH); // взять случайные координаты
            appleY = random(0, HEIGHT);

            // проверить, не совпадает ли координата с телом змеи
            if ((long)getPixColorXY(appleX, appleY) == 0) {
                apple_flag = true; // если не совпадает, считаем что яблоко со-
здано
                drawPixelXY(appleX, appleY, GLOBAL_COLOR_2); // и рисуем
                FastLED.show();
            }
        }

        // запоминаем, куда повернули голову
        // 0 - право, 1 - лево, 2 - вверх, 3 - вниз
        if (vectorX > 0) buttVector[snakeLength] = 0;

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				68
Змн.	Арк.	№ докум.	Підпис	Дата		

```

else if (vectorX < 0) buttVector[snakeLength] = 1;
if (vectorY > 0) buttVector[snakeLength] = 2;
else if (vectorY < 0) buttVector[snakeLength] = 3;

// смещение головы змеи
headX += vectorX;
headY += vectorY;

if (headX < 0 || headX > WIDTH - 1 || headY < 0 || headY > HEIGHT - 1) { // если
вышла за границы поля
    fail = true;
}

if (!fail) {
    // проверка на fail
    if ((long)(getPixColorXY(headX, headY) != 0 && (long)getPixColorXY(headX,
headY) != GLOBAL_COLOR_2)) { // если змея врезалась во что то, но не в яб-
локо
        fail = true;                // флаг на отработку
    }

    // БЛОК ОТРАБОТКИ ПОЕДАНИЯ ЯБЛОКА
    if (!fail && (long)getPixColorXY(headX, headY) ==
(long)GLOBAL_COLOR_2) { // если попали головой в яблоко
        apple_flag = false;        // флаг что яблока больше нет
        snakeLength++;             // увеличить длину змеи
        buttVector[snakeLength] = 4; // запоминаем, что надо будет не сти-
рать хвост
    }

    // вычисляем координату хвоста (чтобы стереть) по массиву вектора
    switch (buttVector[0]) {
        case 0: buttX += 1;
            break;
        case 1: buttX -= 1;
            break;
        case 2: buttY += 1;
            break;
        case 3: buttY -= 1;
            break;
        case 4: missDelete = true; // 4 значит не стирать!
            break;
    }

    // смещаем весь массив векторов хвоста ВЛЕВО

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				69
Змн.	Арк.	№ докум.	Підпис	Дата		

```

for (byte i = 0; i < snakeLength; i++) {
    buttVector[i] = buttVector[i + 1];
}

// если змея не в процессе роста, закрасить бывший хвост чёрным
if (!missDelete) {
    drawPixelXY(buttX, buttY, 0x000000);
}
else missDelete = false;

// рисуем голову змеи в новом положении
drawPixelXY(headX, headY, GLOBAL_COLOR_1);
FastLED.show();
}
if (gameDemo) snakeDemo();
}
if (fail) {
    fail = false;
    // ну в общем плавно моргнуть, типо змейке "больно"
    for (byte bright = 0; bright < 15; bright++) {
        FastLED.setBrightness(bright);
        for (int i = 0; i < NUM_LEDS; i++) {
            leds[i] = CRGB::Red;
        }
        FastLED.show();
        delay(10);
    }
    delay(100);
    FastLED.clear();
    FastLED.show();
    FastLED.setBrightness(globalBrightness);
    displayScore(snakeLength - START_LENGTH);
    delay(1000);
    FastLED.clear();
    FastLED.show();
    newGameSnake();                // миша, всё ху.я, давай по новой
}
}

void snakeDemo() {
    // смещение головы змеи
    int8_t nextX = headX + vectorX;
    int8_t nextY = headY + vectorY;

    // ищем яблоко

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				70
Змн.	Арк.	№ докум.	Підпис	Дата		

```

if (headX == appleX) {           // яблоко на одной линии по вертикали
    if (headY < appleY) buttons = 0;
    if (headY > appleY) buttons = 2;
}
if (headY == appleY) {           // яблоко на одной линии по горизонтали
    if (headX < appleX) buttons = 1;
    if (headX > appleX) buttons = 3;
}

if (getPixColorXY(nextX, nextY) == GLOBAL_COLOR_1) { // проверка на
столкновение с собой
    // поворачиваем налево
    if (vectorX > 0) buttons = 0;
    if (vectorX < 0) buttons = 2;
    if (vectorY > 0) buttons = 3;
    if (vectorY < 0) buttons = 1;
    return;
}

if (nextX < 0 || nextX > WIDTH - 1 || nextY < 0      // проверка на столкновение
со стеной
    || nextY > HEIGHT - 1) {

    // поворачиваем направо в обычном случае или налево в углу
    if (vectorX > 0) buttons = 2;
    if (vectorX > 0 && headY == 0) buttons = 0;

    if (vectorX < 0 && headY == HEIGHT - 1) buttons = 2;
    if (vectorX < 0) buttons = 0;

    if (vectorY > 0) buttons = 1;
    if (vectorY > 0 && headX == WIDTH - 1) buttons = 3;

    if (vectorY < 0 && headX == 0) buttons = 1;
    if (vectorY < 0) buttons = 3;
}
}

void buttonsTickSnake() {
    if (checkButtons()) {
        if (buttons == 3) { // кнопка нажата
            vectorX = -1;
            vectorY = 0;
            buttons = 4;
        }
    }
}

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				71
Змн.	Арк.	№ докум.	Підпис	Дата		

```

if (buttons == 1) { // кнопка нажата
    vectorX = 1;
    vectorY = 0;
    buttons = 4;
}
if (buttons == 0) { // кнопка нажата
    vectorY = 1;
    vectorX = 0;
    buttons = 4;
}
if (buttons == 2) { // кнопка нажата
    vectorY = -1;
    vectorX = 0;
    buttons = 4;
}
}
}

void newGameSnake() {
    FastLED.clear();
    // свежее зерно для генератора случайных чисел
    randomSeed(millis());

    // длина из настроек, начинаем в середине экрана, бла-бла-бла
    snakeLength = START_LENGTH;
    headX = WIDTH / 2;
    headY = HEIGHT / 2;
    buttY = headY;

    vectorX = 1; // начальный вектор движения задаётся вот здесь
    vectorY = 0;
    buttons = 4;

    // первоначальная отрисовка змейки и забивка массива векторов для хвоста
    for (byte i = 0; i < snakeLength; i++) {
        drawPixelXY(headX - i, headY, GLOBAL_COLOR_1);
        buttVector[i] = 0;
    }
    FastLED.show();
    buttX = headX - snakeLength; // координата хвоста как голова - длина
    missDelete = false;
    apple_flag = false;
}

#elif (USE_SNAKE == 0)

```

		Шпет В.С.			КІ.КР.Б. – 123 – 21 – ПЗ	Арк.
		Пулеко І.В.				72
Змн.	Арк.	№ докум.	Підпис	Дата		