МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

Кафедра електроніки і комп'ютерної техніки

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи бакалавра на тему:

«**Пристрій обміну та зберігання даних з використанням хмарних сховищ»**

Завідувач кафедри: Опанасюк А.С.

Керівник кваліфікаційної
роботи бакалавра: Бережна О.В.

Виконав студент
гр. ЕС-71: Погуляй О.Р.

Суми

2021 р

Сумський державний університет

Факультет ЕлІТ
Кафедра електроніки і комп'ютерної техніки

Напрямок підготовки 171 ”Електроніка”

ЗАТВЕРДЖУЮ

Зав. кафедри ЕКТ Опанасюк А.С..
"___"_____ 2021 р.

ЗАВДАННЯ

на кваліфікаційну роботу бакалавра

студенту **Погуляй Олександр Ростиславович**

1. Тема проекту: **«Пристрій обміну та зберігання даних з використанням хмарних сховищ»**

затверджена наказом по університету " 05 " квітня 2021 р. № 0153-VI

2. Термін здачі студентом закінченої роботи 01.06.21_____

3. Вихідні дані до роботи _____

_____

_____

_____

4. Зміст розрахунково-пояснювальної записки:

– огляд літератури і постановка задачі проектування;

– розроблення, обґрунтування алгоритму функціонування та структурної схеми;

– розроблення функціональної схеми пристрою;

– розробка принципової схеми та вибір елементної бази;

– розроблення програмного забезпечення пристрою хмарного сховища.

5. Перелік графічного матеріалу:

– креслення схеми алгоритму;

– креслення схеми електричної структурної;

– креслення схеми електричної функціональної;

– креслення схеми електричної принципової.

Дата видачі завдання _____

Завдання прийняв до виконання _____Погуляй О.Р.

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів дипломного проекту	Термін виконання етапів роботи	Примітки
1	Огляд літератури і постановка завдання на проектування	27.03.21 – 05.04.21	
2	Синтез структурної схеми і алгоритма роботи пристрою	05.04.21 – 11.04.21	
3	Розробка функціональної схеми пристрою	12.04.21 – 20.04.21	
4	Розробка принципових схем блоків пристрою	21.04.21 – 28.04.21	
5	Розробка програмного забезпечення пристрою	29.04.21 – 14.05.21	
6	Оформлення пояснювальної записки і креслень	15.05.21 – 25.05.21	
7	Представлення роботи керівнику і отримання відгуку	24.05.21	
8	Представлення роботи на кафедру для отримання рецензії	31.05.21	

Студент _____

Керівник _____

"___"_____2021 р.

РЕФЕРАТ

Кваліфікаційна робота містить 65 сторінок, 37 рисунків, 16 таблиць, 20 джерел літератури.

Пояснювальна записка складається з п’яти розділів: огляд літератури і постановка задачі проектування, розроблення, обгрунтування алгоритму функціонування та структурної схеми, розробка функціональної та принципової схеми пристрою, розроблення функціональної схеми, розробка принципової схеми та вибір елементної бази, розроблення програмного забезпечення пристрою хмарного сховища

Графічна частина роботи містить алгоритм, структурну, функціональну і принципову схеми.

У першому розділі проведений огляд літературних джерел по вибраному напрямку проектування.

Другий розділ містить розробку та обґрунтування алгоритму функціонування і структурної схеми пристрою.

Третій розділ розкриває розробку функціональної схеми пристрою.

Четвертий розділ містить розробку та розрахунок принципової схеми пристрою. Також виконаний відбір елементної бази.

У п’ятому розділі пояснення до програми взаємодії з одноплатним комп’ютером.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ПК – персональний комп’ютер;
ІТ – інтернет-технології;
IP – Internet Protocol;
RAID – Redundant Array of Independent Disks;
ПЗ – програмне забезпечення;
TCO – Transmission Control Protocol;
ARM – Advanced RISC Machine;
ПЗП – Постійний запам'ятовуючий пристрій;
ОЗП – Оперативний запам'ятовуючий пристрій;
HDD – Hard Disk Driver;
NAS – Network Attached Storage;
RAID – Redundant Array of Independent Disks;
DAS – Direct-Attached Storage;
S.M.A.R.T. – Self-Monitoring, Analysis and Reporting Technology;
SSD – Solid-State Drive.

ВСТУП

З кожним роком розвиток хмарних технологій все більше набирає обертів, разом з цим виникає багато онлайн сервісів, які частіше використовують в різних сферах нашого життя. Наприклад, на сайті Figma люди можуть створювати сайти і додатки або за допомогою сервісу Adobe редагувати відео в онлайн режимі, хоча декілька років тому, це можливо було робити лише з використанням потужності власного ПК. Зараз технології Nvidia дозволяють купувати підписку, яка дає змогу використовувати віддалено ресурси їх серверів, для графічних розрахунків, обробки 3D графіки, більш швидкий рендерінг відео, тощо.

Кожного дня люди не замислюються над тим, що користуються хмарними сховищами. Людям більше не обов'язково носити з собою фізичні носії, щоб перенести інформацію в інше місце або хвилюватися над тим, що може щось статися з інформацією на HDD.

В бізнес сфері всі ці технології набули ще більш широкого розвитку. Наприклад в IT-компаніях, людям доволі комфортно використовувати одночасно один і той же файл, яки зберігається в хмарному сервісі Google disk, OneDrive або MegaCloud. Але разом с розвиток технології виникала інша проблема – ціна виділеного місця на сервері залишиається дорогою. У тому же Google disk безкоштовно лише 15Gb, а 2Tb коштують 2762 грн на рік, що доволі не дешево. Тому деякі фірми почали використовувати власні хмарні сховища NAS [1].

NAS можна описати як сервер, на який можна записати файл будь-якого формату і, відповідно, скачати з будь-якого місця на планеті де є доступ до мережі інтернет. Потративши суму на яку ми могли би купити річну підписку на гугл диск, на багато вигідніше купити HDD, одноплатний комп'ютер та підключити це до маршрутизатора. Звісно на ринку IT-технологій є готові рішення, але вони будуть коштувати набагато дорожче, хоча все одно будуть вигідним рішенням.

1 ОГЛЯД ЛІТЕРАТУРИ І ПОСТАНОВКА ЗАДАЧІ ПРОЕКТУВАННЯ

1.1 Огляд літератури

Хмарні сховища. Хмарне сховище – модель обчислень в хмарі, яка передбачає зберігання даних на сторонніх серверах в інтернеті за допомогою постачальника (власника) хмарних обчислювальних ресурсів, який надає сховище даних як сервіс і забезпечує користувачеві повний доступ на попередньо домовлений час. Хмарне сховище надається по договору в необхідному обсязі, оплачується методом підписки на конкретний термін і позбавляє від необхідності купувати власну інфраструктуру для зберігання даних. Це забезпечує гнучкість, глобальну масштабованість і надійність.

Постачальників таких послуг, на даний момент, існує багато тому й конкуренція між ними велика. Користувачів приваблюють додатковими послугами, які входять у відповідні пакети. Розглянемо найпопулярніші з них та порівняємо їх між собою. Але потрібно зазначити, що нас насамперед цікавить об’єм місця сховища даних, які нам будуть запропоновані.

OneDrive. Пропонує користувачам простий спосіб для зберігання, синхронізації та обміну усіма видами файлів з іншими людьми і пристроями в мережі, через веб-інтерфейс або інстальовану програму. Також можлива синхронізація ваших системних і візуальних налаштувань, тем, налаштувань програм. Залишається навіть історія та збережені паролі з веб-браузерів.

Хмарний сервіс від Microsoft добре інтегрований в усі програмні продукти Microsoft, тому для кожного тарифного плану були певні додаткові служби та програми. Наведемо тарифні плани Microsoft в табл. 1.1 [2].

Таблиця 1.1– Тарифні плани Microsoft OneDrive

Тарифний план	Об'єм памяті, Гб	Ціна за місяць, грн	Ціна за рік, грн	Ціна 1 Гб, грн
OneDrive Standalone	100	59	708	7,08
Microsoft 365	1000	189	1899	18,99

Google диск. Його функції включають зберігання файлів в Інтернеті, загальний доступ до них і спільне редагування. До складу Google Диску входять Google Документи, Таблиці та Презентації – набір офісних додатків для спільної роботи над текстовими документами, електронними таблицями, презентації, кресленнями, веб-формами та іншими файлами. Google диск має самий великий об’єм з усіх перерахованих. Максимально в нього можна завантажити до 30 Тбайт. Редагувати файли можна навіть без мережі – дані зберігаються автоматично. Рівні доступу дозволять керувати обміном папок і файлів і встановити заборону на копіювання важливих документів. Наведемо тарифні плани Google в табл. 1.2 [3].

Таблиця 1.2 – Тарифні плани Google диск

Тарифний план	Об'єм памяті, Гб	Ціна за місяць, грн	Ціна за рік, грн	Ціна 1 Гб
«G1»	15	0	0	0
«G2»	100	54,60	548,45	5,48
«G3»	200	82,03	822,82	4,11
«G4»	2000	274,09	2743,36	1,37

Mega Cloud. Хмарне сховище, яке має мінімум додаткових служб та має найбільший безкоштовний тарифний план в 50 Гб, але має обмеження на завантаження файлів на добу. Наведемо тарифні плани Mega Cloud в табл. 1.3 [4].

Таблиця 1.3 – Тарифні плани Mega Cloud

Тарифний план	Об'єм памяті, gb	Ціна за місяць, грн	Ціна за рік, грн	Середня ціна 1 gb
Pro Standart	50	0	0	0
Pro Lite	400	167,39	1676,95	4,19
Pro I	2000	335,12	3354,24	1,68
Pro II	8000	670,58	6708,82	0,83
Pro III	16000	1006,04	10063,40	0,63

NAS. NAS це пристрій зберігання, підключений до мережі, яке дозволяє зберігати та видавати дані з центрального розташування для авторизованих

користувачів мережі і різних клієнтів, приклад такої системи показана на рис. 1.1. Пристрої NAS є гнучкими і масштабованими. NAS – це як приватна хмара в офісі. Це швидше, дешевше і надає всі переваги загальнодоступної хмари на місці, надаючи вам повний контроль.

Тома NAS відображаються для користувача як том, підключений до мережі. Файли, які обслуговуються, зазвичай, містяться на одному або декількох дисках зберігання, часто у вигляді логічних надлишкових контейнерів зберігання або як RAID масив.

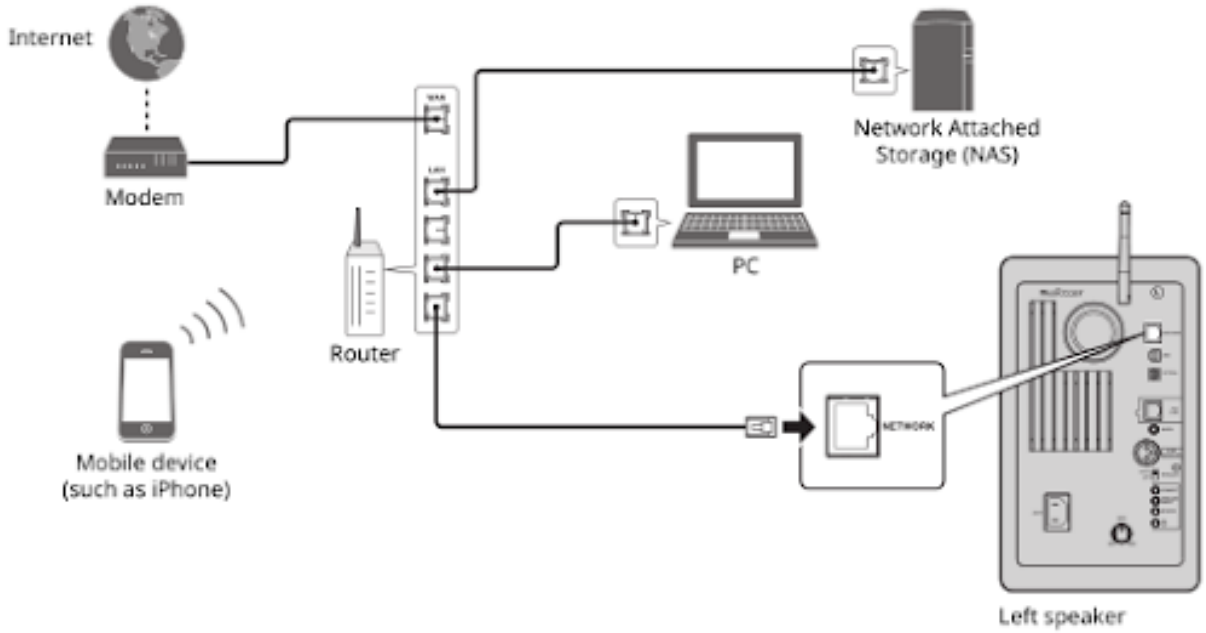

Рисунок 1.1 – NAS в системі

Сам пристрій являє собою мережевий вузол, дуже схожий на комп'ютер і інші пристрої TCP/IP, які підтримують свої власні IP-адреса і можуть ефективно взаємодіяти з іншими мережевими пристроями.

Хоча NAS зазвичай не призначений для використання в якості сервера загального призначення, постачальники NAS і незалежні постачальники все частіше пропонують програмне забезпечення для забезпечення серверної функціональності на NAS.

Пристрої NAS забезпечують легкий доступ до даних для декількох користувачів в різних місцях, що корисно при спільній роботі над проектами або при обміні інформацією. NAS забезпечує хороший контроль доступу і безпеку для підтримки спільної роботи, а також дозволяє тому, хто не є IT-спеціалістом,

адмініструвати та керувати доступом до даних. Він також забезпечує хорошу фундаментальну безпеку даних за рахунок використання надлишкових структур даних – часто RAID.

NAS часто є наступним кроком для домашнього офісу або малого бізнесу, який використовує DAS. Перехід на NAS обумовлений бажанням обмінюватися файлами локально і віддалено, наявністю файлів цілодобово, надмірністю даних, можливістю заміни та оновлення жорстких дисків в системі, а також доступністю інших служб, таких як автоматичне резервне копіювання.

Короткий виклад переваг NAS:

– відносно недорогий;
– цілодобова і віддалена доступність даних;
– гарна розширюваність;
– архітектура резервованого сховища;
– гнучкість конфігурації.

Слабкі сторони NAS пов'язані з масштабом і продуктивністю. Оскільки доступ до нього потрібно більшій кількості користувачів, сервер може не впоратися, а також вимагати додаткових ресурсів. Інший недолік пов'язаний з природою самого Ethernet. За своєю природою Ethernet передає дані з одного місця в інше через пакети, розділяючи джерело на кілька сегментів і відправляючи їх, структура Ethernet фрейму представлена на рис. 1.2.

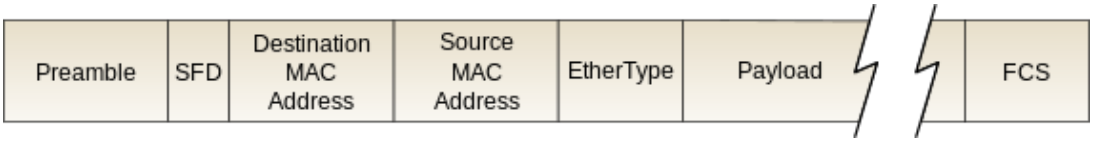

Рисунок 1.2 – Ethernet фрейм

Будь-який з цих пакетів може бути затриманий або відправлений не по порядку і може бути недоступний користувачеві доти, поки всі пакети не будуть доставлені і не повернуті в порядок.

Будь-яка затримка (повільні або повторні підключення) зазвичай не помічаються користувачами для невеликих файлів, але може бути серйозною проблемою у вимогливих середовищах, де файли дуже великі і затримка більш декількох мілісекунд може порушити такі етапи виробництва, як рендерінг.

NAS WD My Cloud Duo. My Cloud Duo, на рис. 1.3, являє собою простий у використанні дводисковий персональний накопичувач, який підключається безпосередньо до домашнього маршрутизатора з Wi-Fi. Проста у використанні персональна хмарна система для централізованого зберігання цифрових матеріалів. На відміну від мережевого пристрою зберігання даних, це просте рішення для централізованого зберігання резервних файлів, завдяки додатку My Cloud Home можна копіювати на накопичувач і відкривати з нього файли, а також надавати до них спільний доступ з будь-якого місця, де є підключення до Інтернету.

Рисунок 1.3 – My Cloud Duo

Кожен пристрій My Cloud Home™ Duo містить у собі два жорсткі диски, що працюють в режимі дзеркального запису (режим RAID 1), тому всі файли, які надходять у NAS автоматично копіюються на два диска, що забезпечує додаткову впевненість в їх збереженні. Звичайно, в налаштуваннях можна вимкнути цю функцію та збільшити об’єм внутрішнього простору вдвічі. Основні параметри пристрою наведемо в табл. 1.4 [4].

Таблиця 1.4– Основні параметри NAS WD My Cloud Duo

NAS WD My Cloud Duo	
HDD	2 x 3,5 SATA
Raid, рівень	JBOD, Raid 1
Мережеві інтерфейси	1 × Gigabit Ethernet RJ-45
Додаткові інтерфейси	2 × USB Host 3.0
Габарити, мм	179 × 160 × 102
Ціна пристрою, грн	9425
Ціна HDD на 2 tb	2408
Ціна за 1 gb, грн	5,92

NAS Synology DS220+. NAS Synology DS220 + призначений для роботи з двома дисками і відноситься до серій домашніх мережевих накопичувачів з індексом Plus, зображений на рис. 1.4. І хоча саме цей файловий сервер представляє саму молоду модель цієї серії, але NAS з індексом Plus відрізняються високою продуктивністю та цікавими можливостями [5].

Рисунок 1.4 – NAS Synology DS220+

В основі цього NAS лежить однокристальна платформа Intel Gemini Lake Refresh з двоядерний процесором Celeron J4025. Діапазон робочих тактових частот для даного процесору становить від 2,0 до 2,9 ГГц. Це один з наймолодших процесорів у серії, але він має вбудовану графічну систему Intel UHD 600 з можливістю транскодування відеосигналу з роздільною здатністю 4K (4096 × 2160, 60 Гц). Основні параметри пристрою наведемо в табл. 1.5 [6].

Таблиця 1.5 – Основні параметри NAS Synology DS220+

NAS Synology DS220+	
HDD	2 x 3,5 або 2 x 2,5 SATA
Raid, рівень	JBOD, Raid 0, Raid 1, Basic
Мережеві інтерфейси	2 × Gigabit Ethernet RJ-45
Додаткові інтерфейси	2 × USB Host 3.0
Габарити, мм	165 × 108 × 233
Ціна пристрою, грн	11 932
Ціна HDD на 2 tb	2408
Ціна за 1 gb, грн	7,17

NAS оснащений двома гігабітними мережевими інтерфейсами, представленими парою широко поширених контролерів RTL8111HS виробництва Realtek Semiconductor Corp. Пристрій дозволяє об'єднувати мережеві підключення завдяки функції Link Aggregation з налаштуванням типу підключення.

1.2 Постановка завдання на проектування

Метою роботи є розробка пристрою обміну та зберігання даних з використанням локальних хмарних сховищ, який повинен забезпечувати базові потреби користувача, а саме:

- завантаження файлів через мережу;
- доступ до завантажених файлів через мережу;
- редагування файлів в локальної хмарі;
- захищений доступ до даних.

Для досягнення цієї мети необхідно виконати наступне:

1. Визначити основні функції та завдання, які повинен виконувати пристрій обміну та зберігання даних з використанням хмарних сховищ.

2. Розробити алгоритм функціонування пристрою.

3. Розробити схему електричну структурну пристрою обміну та зберігання даних.

4. Розробити схему електричну функціональну пристрою.

5. Розробити схему електричну принципову пристрою обміну та зберігання даних з використанням хмарних сховищ.

2 РОЗРОБЛЕННЯ, ОБГРУНТУВАННЯ АЛГОРИТМУ ФУНКЦІОНУВАННЯ ТА СТРУКТУРНОЇ СХЕМИ

2.1 Розроблення алгоритму функціонування пристрою

Для наочного уявлення роботи мережевого сховища даних, нам потрібно зрозуміти принцип роботи пристрою. Для цього, перечислимо всі функції, які має виконувати пристрій:

- прийом файлів поширених форматів;
- зберігання файлів;
- завантаження файлів;
- редагування файлів різних форматів;
- видалення файлів;
- захист даних, які зберігаються;
- контроль рівня завантаженості сховища;
- контроль фізичного стану сховища.

На підставі перелічених функцій складемо блок-схему алгоритму роботи системи мережевого сховища даних. Схема алгоритму наведена на рисунку 2.1.

Першим пунктом є необхідність під’єднати пристрій до мережі. Так як система не має в своїй конструкції ніяких індикаторів, по яким можна провести налаштування або безпосередньо скористатися мережевим сховищем, потрібно під’єднатися до web-інтерфейсу. Він автоматично розгортається при вмиканні системи.

Потрібно відмітити, що підключення для самого мережевого сховища повинно бути високошвидкісним, бажано щоб маршрутизатор мав у своїй конструкції гігабітний порт [8]. Можливе підключення і до менш швидкісних портів але при такій конфігурації зменшується пропускна здатність, маршрутизатор стає вузьким місцем в ланцюзі обміну файлів. Наведемо приклад, середня пропускна спроможність жорсткого диску – 150-200 МБ/с, а швидкість звичайного, дешевого маршрутизатора – 12,5 МБ/с, при використанні маршрутизатора з гігабітним портом, швидкість зростає аж до 125 МБ/с.

Другим пунктом є запит на авторизацію. Як писалося вище, при звертанні до мережевого сховища користувач потрапляє на web-інтерфейс, приклад якого зображений на рис. 2.2.

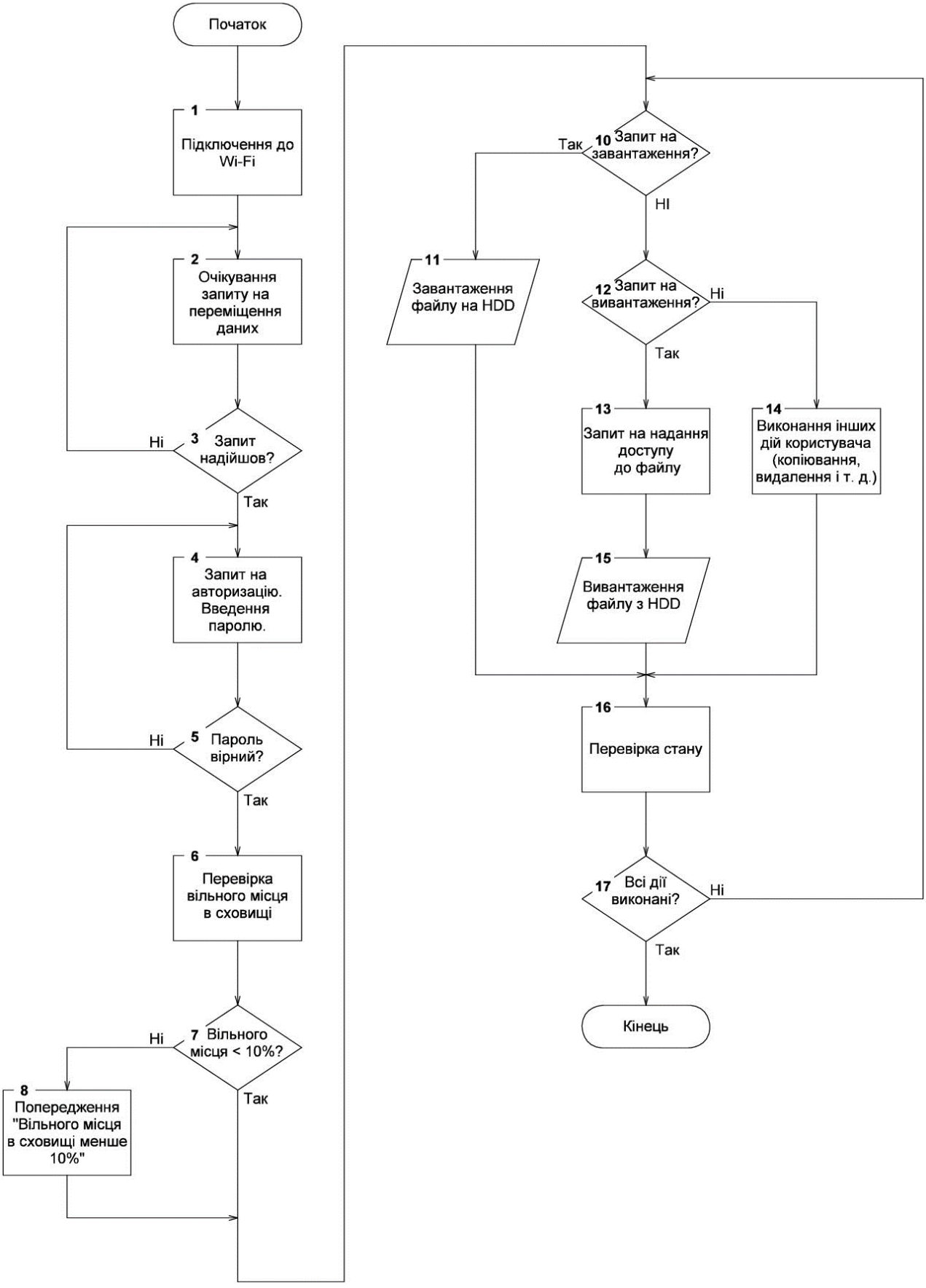

Рисунок 2.1 – Блок схема мережевого сховища

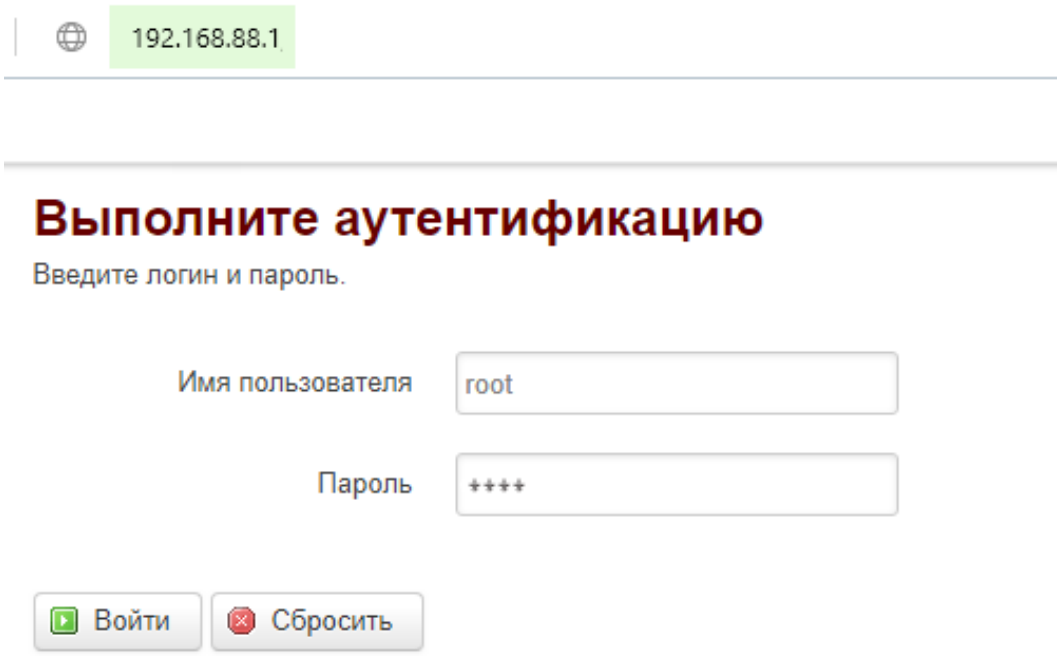

Рисунок 2.2 – Приклад автентифікації користувача за допомогою web-інтерфейсу

Користувач повинен ввести свій логін та пароль для доступу до сховища. Розділення на користувачів дозволяє в майбутньому, за потреби, організувати кожному свій вільний простір, а пароль захищає персональні дані від сторонніх користувачів.

Третім пунктом є діагностика дискового простору. Майже 100% дискових накопичувачів останнього десятиріччя, функціонує з використанням технології S.M.A.R.T. На українській мові абревіатура звучить як механізм самоконтролю, аналізу та звітності. Коли на дисковий накопичувач надходить живлення, він починає свою штатну роботу, паралельно відслідковуючи власний стан за допомогою спеціальних параметрів і атрибутів. Ці параметри відслідковує окрема зона накопичувача, доступ до якої, за звичайних умов – неможливий.

Четвертим пунктом є перевірка вільного місця в сховищі. Після вірного введення логіну та паролю, користувач безпосередньо потрапляє у графічний інтерфейс, де бачить свій простір. За технологіє S.M.A.R.T. програмне забезпечення отримує інформацію з жорсткого диску та адаптує його для зручного розуміння людиною. Якщо ж вільного місця в мережевому сховищі залишається менше 10%, то користувач побачить відповідне повідомлення.

П’ятим пунктом є запит користувача для надання доступу взаємодії з файлами. Звичайним користувачам, які далекі від роботи в linux-терміналі, простіше користуватися графічним інтерфейсом, тому на нашому сервері буде встановлено програмне забезпечення з графічним інтерфейсом яке зображене на рис. 2.3. Запит для взаємодії з файлом здійснюється кліком миші по символу «…», якщо це комп’ютер та довгим натисканням, якщо це смартфон. Після чого обирається дія, яка повинна бути виконана над файлом, доступно це лише авторизованим користувачам. Доступні дії користувача:

– копіювання;
– видалення;
– завантаження;
– перейменування.

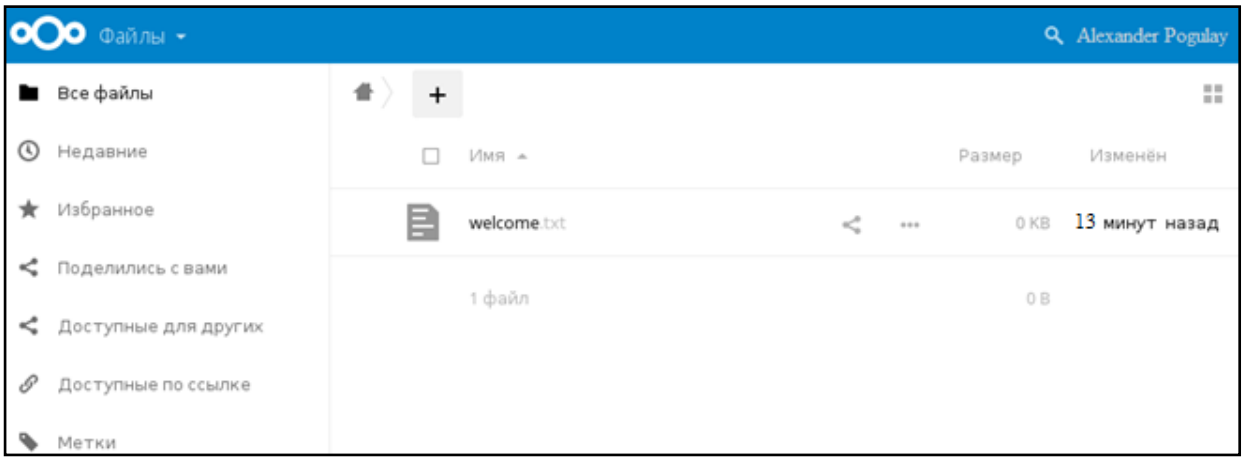

Рисунок 2.3 – Програмне забезпечення для взаємодії з файлами

Шостим кроком є переміщення файлів. При запиті на переміщення файлу, програмне забезпечення розраховує приблизний час, за який буде здійснено переміщення файлу, показуючи цю інформацію користувачеві. Час передачі залежить від швидкості маршрутизатора, розміру файлу, кількості файлів, швидкості мережі, та швидкості приймача. Після чого, на IP-адресу користувача, який ініціалізував переміщення файлів буде надіслана відповідна інформація.

Сьомим кроком, є закінченням роботи системи. Після завершення прийому-передачі даних, коли користувач закриває web-інтерфейс відбувається його розлогінювання з системи.

Варто відмітити, що в системі крім службових аккаунтів також є аккаунт адміністратора. Адміністративний обліковий запис має більше прав, чим

звичайний аккаунт, він може відслідковувати дії користувачів, надавати їм більші права та напряму впливати на розмір їх сховища.

2.2 Розробка структурної схеми пристрою

Згідно з технічного завдання, необхідно розробити систему, яка реалізує функцію мережевого сховища інформації. Система повинна бути швидкою, зручною і в той же час безпечною.

Мережеве сховище повинно забезпечувати виконання наступних функцій:

- прийом файлів поширених форматів;
- зберігання файлів;
- завантаження файлів;
- редагування файлів різних форматів;
- видалення файлів;
- захист даних, які зберігаються;
- контроль рівня завантаженості сховища;
- контроль фізичного стану сховища;
- можливість підключення до системи через технологію Ethernet;
- можливість підключення до системи через технологію wi-fi.

Важливо відмітити, що маршрутизатор не входить в розроблювану систему, але його наявність необхідна для функціонування системи, структурна схема системи наведена на рис. 2.4. Рішення, не включати маршрутизатор в систему, було виваженим, воно дозволяє користувачеві самому обрати пристрій, який задовільняє його потреби. Також, не самою останньою причиною відмови від маршрутизатора в готовій системі мережевого сховища – завантаженість wi-fi каналів, у великих містах це дуже виражена проблема. Ну і звісно, відсутність маршрутизатора позитивно вплине на ціну фінального пристрою.

Система є доволі складною, вона повинна керувати потоками даних, розуміти основні формати даних з якими працює, мати зрозумілий користувачеві інтерфейс ще й забезпечувати безпеку файлів. Для цього завдання мікроконтролер уже не підходить, його потужності не вистачить, доцільно застосувати одноплатний комп’ютер на ARM архітектурі.

Жорсткий диск має в собі SATA-інтерфейс для передачі інформації але для одноплатних комп’ютерів наявність повноцінного SATA порта – розкіш, з огляду на їх фізичні розміри. Потрібно застосувати перехідник із SATA в USB.

Одноплатні комп’ютері, в своїй конструкції не мають пам’яті, яку б міг використовувати користувач але є порт для встановлення SD-карти.

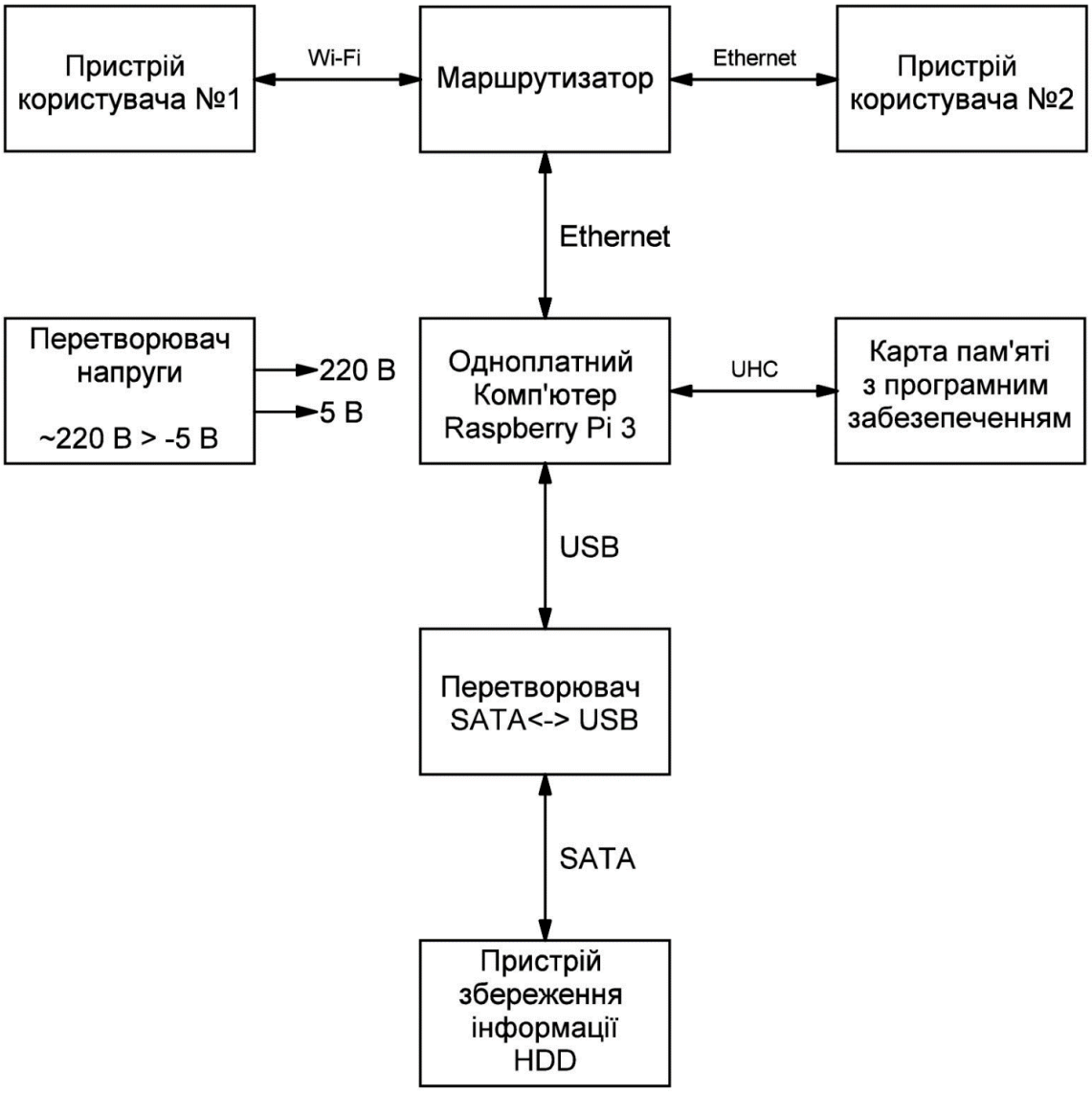

Рисунок 2.4 – Структурна схема мережевого сховища даних

Оптимальним вибором для одноплатного комп’ютера для побудови мережевого сховище є SD-карта [9] на 16 Гб 10-го класу. Такого об’єму пам’яті вистачить для програмного забезпечення, а 10 клас накопичувача забезпечить необхідний рівень швидкодії.

3 РОЗРОБЛЕННЯ ФУНКЦІОНАЛЬНОЇ СХЕМИ ПРИСТРОЮ

Для того, щоб людина змогла скористатися системою зберігання даних, вона повинна мати будь-який пристрій під керуванням операційної системи Android або IOS, персональний комп'ютер під керуванням операційної системи Windows або Linux. На структурній схемі, ці варіанти розглянуті і записані як «Пристрій користувача №1» та «Пристрій користувача №2». Якщо розібратися, то ці пункти не входять в розроблювану систему, але без них функціонування системи просто втрачає будь-який сенс.

Після того, як користувач зробив запит в браузері свого пристрою, чи то з смартфона, чи то з персонального комп’ютера, запит потрапляє на маршрутизатор, рис. 3.1 через wi-fi антену або ж ethernet кабель. На структурній схемі, цей пристрій розглянутий і записаний як «Маршрутизатор».

Рисунок 3.1 – Маршрутизатор

Маршрутизатор приймає запит з пристрою користувача та намагається знайти відповідну ір-адресу в своїй таблиці маршрутизації, якщо ж процес відбувається вперше, то потрібна адреса не буде знайдена. В такому випадку, спрацьовує автоматичний алгоритм дій. Маршрутизатор посилає на всі пристрої, які підключені до нього широкомовна запит. Пристрої, які отримали такий запит, відправляють всю необхідну маршрутизатору інформацію, для вдалої побудови таблиці маршрутизації. Після чого, запит користувача буде направлений на

одноплатний комп’ютер, так як, щоб користувачу отримати доступ до свої даних, йому потрібно пройти процес підтвердження особи.

Одноплатний комп’ютер з’єднаний з маршрутизатором через ethernet кабель для максимальної швидкості передачі даних та зменшення циклів перезапиту на відправку файлів. На структурній схемі, цей пристрій розглянутий і записаний як «Одноплатний комп’ютер raspberry pi 3 b+». Одноплатний комп’ютер, який зображений на рис. 3.2, у нашій системі відіграє ключову роль, саме він формує пакети для відправки їх користувачеві, здійснює захист персональних даних, розгортає на собі графічний інтерфейс та взаємодіє з жорстким диском.

Рисунок 3.2 – Одноплатний комп’ютер

Як і звичайний комп’ютер, він має в своєму розпорядженні центральний процесор, відеоядро, оперативну пам’ять, flash-пам’ять для команд, та радіомодуль. Пам’яті, для операційної системи та інших системних файлів на одноплатному комп’ютері немає, але є відповідний протокол UHS за яким комп’ютер має змогу працювати з картами пам’яті формату SD. На структурній схемі, цей пристрій розглянутий і записаний як «Карта пам’яті з програмним забезпеченням». Живлення до такого одноплатного комп’ютера підводиться через інтерфейс microusb. Вимоги до живлення доволі прості, це 3А струму та 5В постійного струму. Такий великий струм обумовлений великою кількість периферії, яку можна під’єднати до такого одноплатного комп’ютера. У нашому випадку, до одноплатного комп’ютера буде під’єднаний лише жорсткий диск і живиться він буде від нього через інтерфейс USB, тому джерело живлення повинно мати запас по струму та невеликі пульсації на виході. На структурній схемі, цей пристрій розглянутий і записаний як «Перетворювач напруги».

Після вдалої авторизації в системі та сформованій команді на взаємодію з файлами на жорсткому дискі, операційна система надсилає запит на жорсткий диск через перехідник USB-SATA. В своїй конструкції, одноплатний комп'ютер не має SATA роз’єму, а всі сучасні жорсткі диски мають в свої конструкції лише SATA інтерфейс. Тому для узгодження їх роботи застосовується конвертер USB-SATA, який формує запит, зрозумілий для жорсткого диску, після чого відправляє на одноплатний комп’ютер отриману від HDD інформацію. На структурній схемі, цей пристрій розглянутий і записаний як «Перетворювач SATA<->USB».

[13] Вся інформація користувача зберігається на жорстокому диску, який з’єднаний з одноплатним комп’ютером. Саме на нього жорсткий диск відправляє всі дані користувача та свої діагностичні дані. На структурній схемі, цей пристрій розглянутий і записаний як «Пристрій збереження інформації».

На основі описаного вище побудуємо функціональну схему та зобразимо її на рис. 3.3.

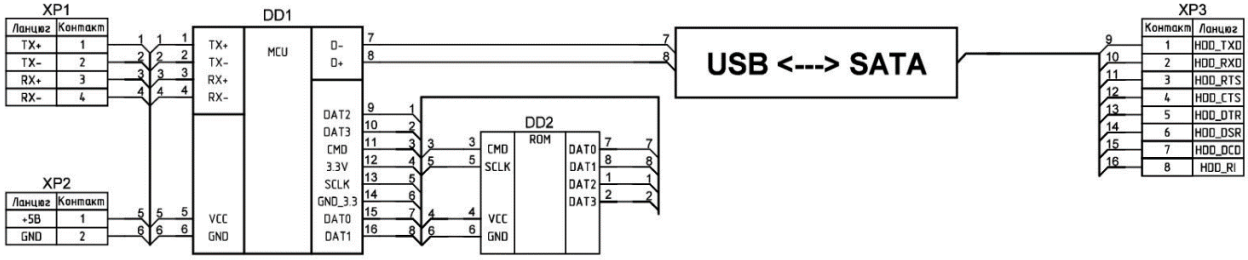

Рисунок 3.3 – Функціональна схема пристрою для обміну та зберігання даних з використанням хмарних сховищ

4 РОЗРОБКА ПРИНЦИПОВОЇ СХЕМИ ТА ВИБІР ЕЛЕМЕНТНОЇ БАЗИ

4.1 Вибір обчислювального модуля

Згідно структурної схеми, потрібен одноплатний комп’ютер, який забезпечить необхідну потужність та швидкодію, також він повинен мати хоча б один порт USB та один порт під 8P8C (його ще помилково називають RJ45). Розглянемо 2 одноплатних комп’ютера.

Asus Tinker Board. Tinker Board це мініатюрний комп'ютер на одній платі, що володіє гарною продуктивністю в своєму класі, а також безліччю можливостей по створенню систем на його основі. Tinker Board призначений для розробників, ентузіастів та інших користувачів, які хочуть втілити свої ідеї в реальність.

Сам пристрій являє собою друковану плату розмірами 85x55 мм з розпаяними на ній контролерами і портами, зображений на рис. 4.1. На лицьовій стороні плати розпаяна більшість найважливіших компонентів [10].

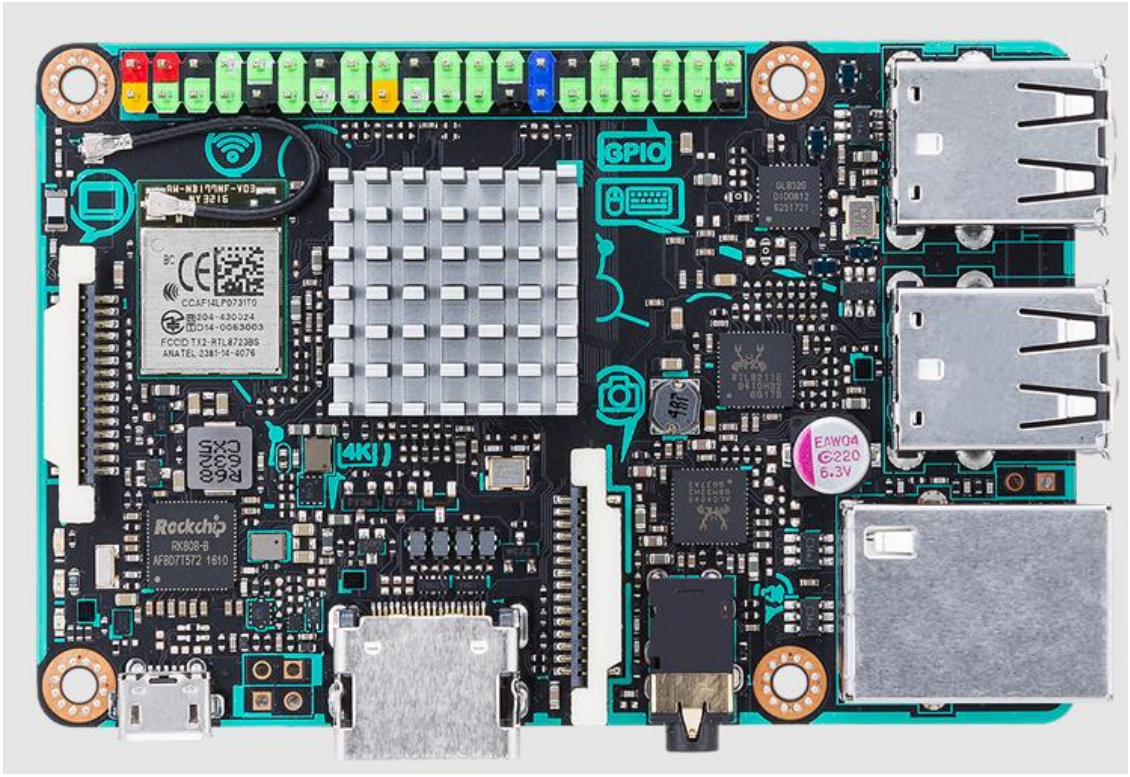

Рисунок 4.1 – ASUS Tinker Board

Схема основних інтерфейсів зображена на рис. 4.2. Компонентна база перелічена в табл. 4.1.

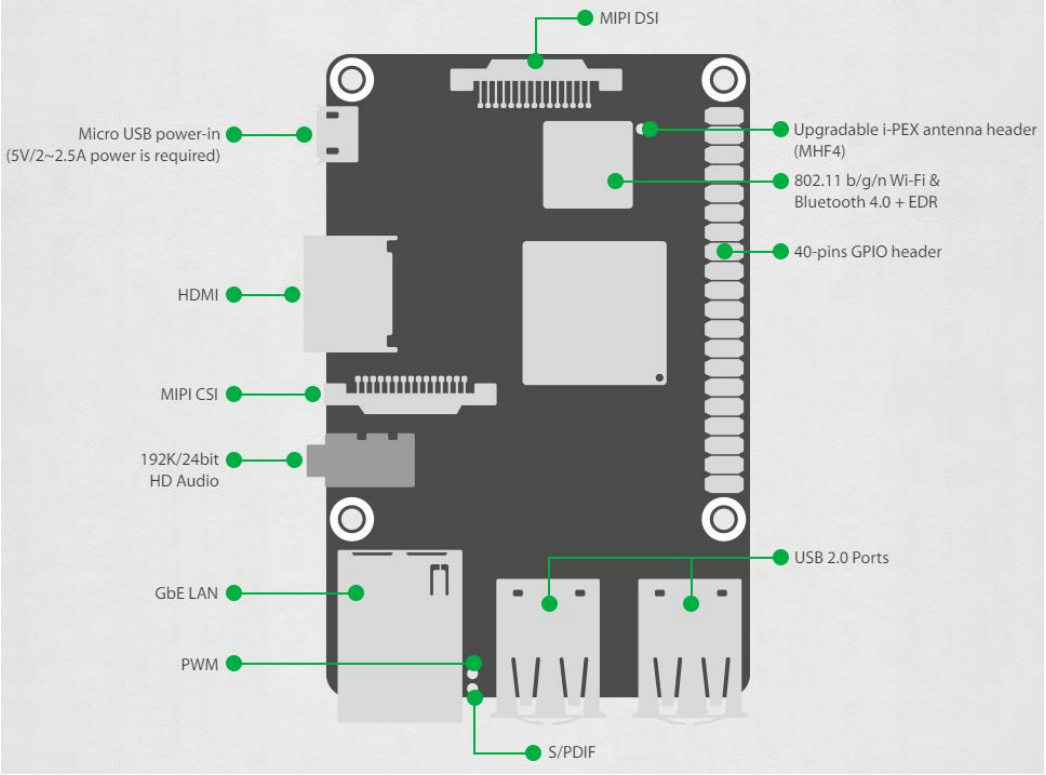

Рисунок 4.2 – Основні інтерфейси ASUS Tinker Board

Таблиця 4.1 – Компонентна база ASUS Tinker Board

Назва компоненту	Характеристика
SoC	RK3288
Центральний процесор	4-ядерний 1.8 ГГц ARM Cortex-A17
Графічний процесор	4-ядерний 600 МГц Mali-T764
Оперативна пам'ять	2 ГБ 2-канальна LPDDR3
Слот для microSD (SDIO3.0)	+
Швидкість мережевого модуля	1 Гбіт
Безпровідні технології	Bluetooth 4.0 + EDR, Wi-Fi 802.11 b/g/n
HDMI	HDMI 1.4 4K/30 fps
USB	4 x USB2.0

Враховуючи всі характеристики Asus Tinker Board, ціна на який коливається від 2500 грн до 3100 грн, можна дійти висновку, що плата має чудові

характеристики, особливо чотирьох-ядерний процесор з частотою 1.8 ГГц. Тобто, гідний претендент на роль мозку проектованої системи.

Raspberry Pi 3 B+. Обчислювальний модуль Raspberry Pi 3 – це ряд механічно сумісних систем DDR2-SODIMM на модулях, що містять процесор, пам’ять та підтримку схем живлення. Ці модулі дозволяють дизайнеру використовувати апаратно-програмний стек Raspberry Pi у своїх власних системах та форм-факторах. Сам пристрій являє собою друковану плату розмірами 85x56 мм з розпаяними на ній контролерами і портами, зображений на рис. 4.3. На лицьовій стороні плати розпаяно більшість найважливіших компонентів.

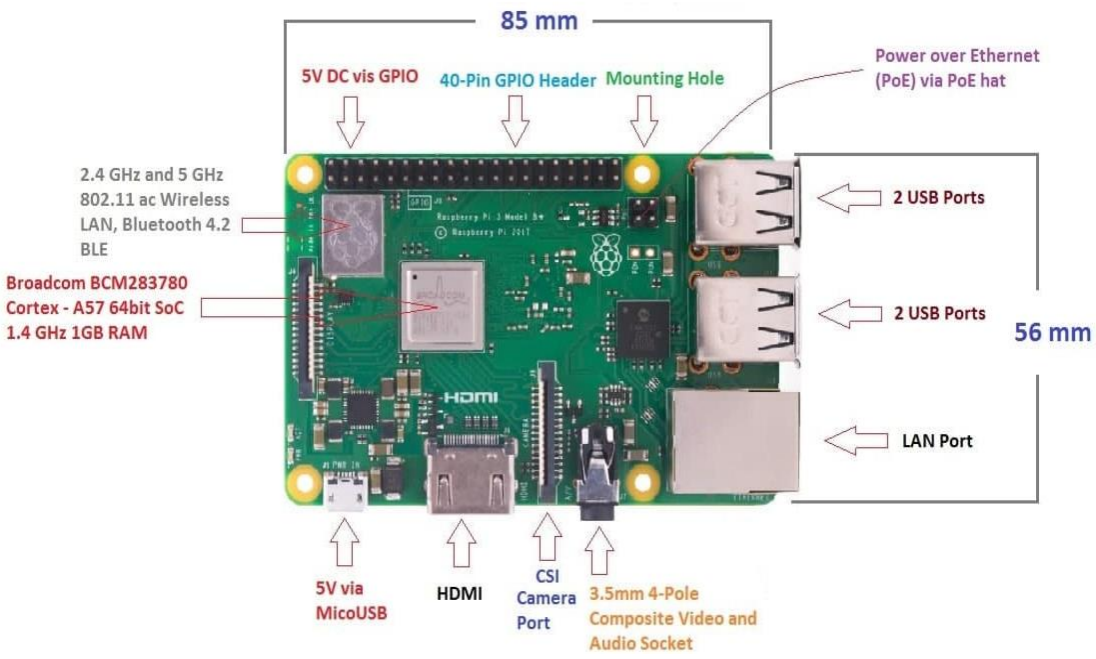

Рисунок 4.3 – Основні інтерфейси Raspberry Pi 3 B+

На Raspberry Pi 3 встановлений 64-х бітний процесор Broadcom BCM2837 на архітектурі ARM Cortex-A53 з тактовою частотою 1,2 ГГц і модулем оперативної пам'яті на 1 ГБ. Процесор і пам'ять розміщені за технологією «package-on-package» безпосередньо на процесорі. BCM2837 включає в себе також двох-ядерний графічний співпроцесор Video Core IV® Multimedia, який забезпечує Open GL ES 2.0, апаратне прискорення Open VG і 1080p30 H.264 декодування.

На Raspberry Pi 3 встановлена мережева карта LAN9514i, яка містить інтегрований концентратор USB 2.0, чотири інтегрованих USB 2.0 PHY,

контролер Ethernet 10/100, контролер TAP та контролер EEPROM. Структурна схема LAN9514i представлена на рис. 4.4.

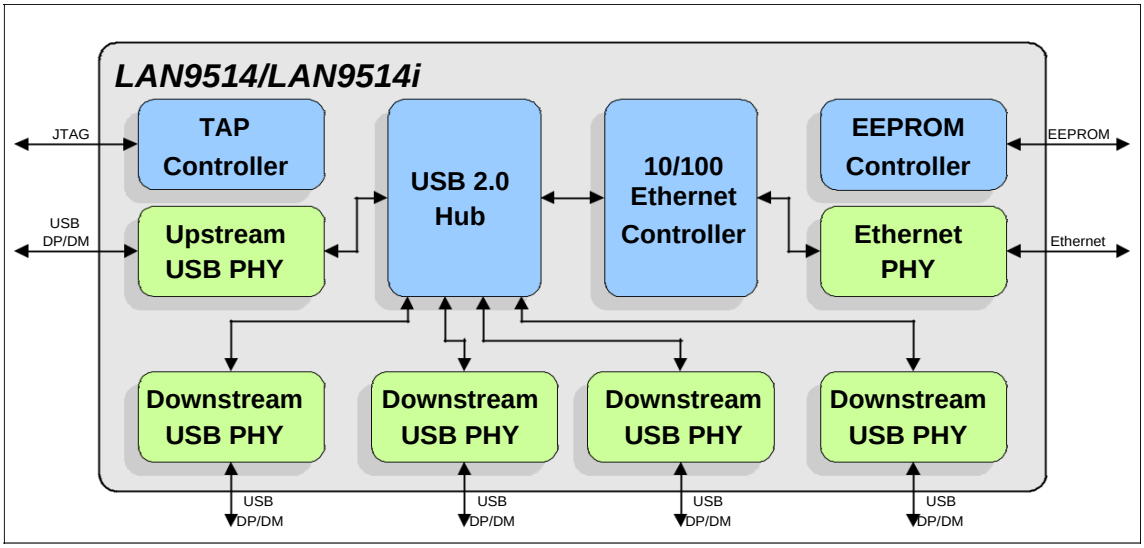

Рисунок 4.4 – Структурна схема LAN9514i

LAN9514i містить контролер EEPROM для підключення до зовнішнього EEPROM. Це дозволяє автоматично завантажувати статичні дані конфігурації під час відсутності живлення, скидання контактів або скидання програмного забезпечення. EEPROM можна налаштувати для завантаження дескрипторів USB, конфігурації USB-пристрою та MAC-адреси.

Зведемо в табл. 4.2 основну компонентну базу Raspberry Pi 3 B+ та оберемо одноплатний комп'ютер для проекту.

Таблиця 4.2 – Компонентна база Raspberry Pi 3 B+

Назва компоненту	Характеристика
Центральний процесор	4-ядерний 1.2 ГГц ARM Cortex-A53
Графічний співпроцесор	2-ядерний 400 Мгц Video Core IV
Оперативна пам'ять	1 ГБ LPDDR2
Слот для microSD (SDIO3.0)	+
Швидкість мережевого модуля	300 Мбіт/сек
Безпровідні технології	Bluetooth 4.2 LE, Wi-Fi 802.11 b/g/n/ac
HDMI	HDMI 1.4 4K/30 fps
USB	4 x USB2.0

Контролер Ethernet 10/100 забезпечує вбудований Ethernet MAC і PHY, які повністю відповідають стандартам IEEE 802.3 10BASE-T та 802.3u 100BASE-TX. Контролер Ethernet 10/100 також підтримує численні функції пробудження живленням, включаючи “Magic Packet”, “Wake on LAN” та “Link Status Change”. Ці події пробудження можна запрограмувати на ініціювання віддаленого пробудження через USB. 10/100 Ethernet PHY інтегрує фізичний рівень IEEE 802.3 для додатків Ethernet. Блок PHY включає підтрюимку автоматичного узгодження, повної або напівдуплексної конфігурації, корекції автополярності та Auto-MDIX. Для використання інтегрованого PHY необхідні мінімальні зовнішні компоненти.

Враховуючи всі характеристики Raspberry Pi 3 B+, ціна на який коливається від 1300 грн до 2100 грн, можна дійти висновку, що плата більше підходить для нашого проекту, так як має достатньо швидкий мережевий модуль, достатню кількість оперативної пам’яті, має в наявності USB порт і основний аргумент в користю Raspberry Pi 3 B – її ціна.

4.2 Особливості обчислювального модуля Raspberry Pi

Raspberry Pi – серія невеликих одноплатних комп'ютерів, розроблених в Сполученому Королівстві Raspberry Pi Foundation. Не включає периферійні пристрої (наприклад, клавіатури і миші) або корпусу. Всі моделі оснащені системою Broadcom на мікросхемі (SoC) з вбудованим ARM-сумісним центральним процесором (CPU) і вбудований графічний процесор (GPU).

Raspberry Pi 3 Model B була випущена в лютому 2016 року зі 64-бітовим чотирьохядерним процесором 1,2 ГГц, вбудованим 802.11nWi-Fi, Bluetooth і з можливістю завантаження через USB. У 2018 був випущений Raspberry Pi 3 Model B + з більш швидким процесором 1,4 ГГц і в три рази швидшим гігабітним Ethernet (пропускна здатність обмежена 300 Мбіт/с через внутрішнє з'єднання USB 2.0) або 2,4/5 ГГц двохдіапазонний 802.11ac Wi-Fi (100 Мбіт/с).

Апаратне забезпечення Raspberry Pi еволюціонувало через кілька версій, в яких представлені різні типи центрального процесора, обсяг пам'яті ємність, мережева підтримка та підтримка периферійних пристроїв. Частота процесора коливається від 700 МГц до 1,4 ГГц для Pi 3 Model B + або 1,5 ГГц для Pi 4. Вбудована пам'ять варіюється від 256 МБ до 1 ГБ оперативної пам'яті (ОЗП).

Secure Digital (SD) в форм-факторі MicroSDHC (SDHC на ранніх моделях) використовуються для зберігання операційної системи і пам'яті програм. Плати мають від одного до п'яти портів USB.

Всі SoC, що використовуються в Raspberry Pis розроблені Broadcom у співпраці з Raspberry Pi Foundation. Raspberry Pi 3 Model B використовує SoC Broadcom BCM2837 з 64-бітовим чотирьохядерним процесором 1,2 ГГц ARM Cortex-A53 процесор з 512 Кбайт загальної кеш-пам'яті L2. Моделі A + і B + мають тактову частоту 1,4 ГГц.

Raspberry Pi 3 з чотирьохядерним процесором ARM Cortex-A53 описується як має в десять разів більшу продуктивність, ніж Raspberry Pi 1. Тести показали, що Raspberry Pi 3 приблизно на 80% швидше, ніж Raspberry Pi 2 в розпаралелених завданнях.

4.3 Версії архітектури ARM процесорів сімейства Cortex

Архітектура визначає набір команд сімейства процесора Cortex, алгоритми управління пам'яттю, кешем і т.д. Одними з найпоширеніших таких архітектур є 32 біт архітектура ARMv7 і 32/64 розрядна архітектура ARMv8-A - (ядро Cortex-A53, Cortex-A57, Cortex-A72). Cortex-A це дуже швидкі по частотам процесора (1 ГГц і більше), але архітектура у них така ж як і у низькочастотних Cortex-M.

Версії архітектури ARM (до версії ARMv8) і процесори, розроблені за цими архітектурам, представлені на рис. 4.5.

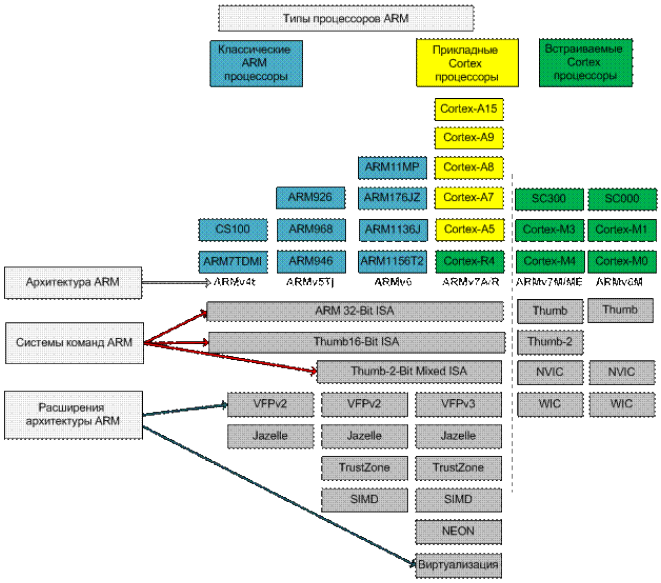

Рисунок 4.5 – Версії архітектури ARM (до версії ARMv8) і процесори, розроблені за цими архітектурам

Процесори з архітектурою ARMv4, ARMv5 і ARMv6 до сих пір використовуються в не дуже складних пристроях, так як дозволяють обійтися меншими витратами за ліцензії.

Процесори Cortex-Mi (архітектура ARMv7M) Cortex-M0, Cortex-M1, Cortex-M3, Cortex-M4 – прийшли на зміну 8- і 16-розрядних мікроконтролерів для вбудованих систем. Низький рівень споживання енергії, мінімальне тепловиділення і маленькі габарити поряд з достатньою продуктивністю дозволяють їм стати на чолі «розумної» побутової техніки, а також інтелектуальних контролерів в областях автомобільної електроніки і ігрових консолей. Чим більше номер і в назві мікроконтролера Cortex-Mi, тим більші можливості і продуктивність мікроконтролера.

Приклади поєднання в одному пристрої процесорів з архітектурою Cortex наведені на рисунках 4.6.

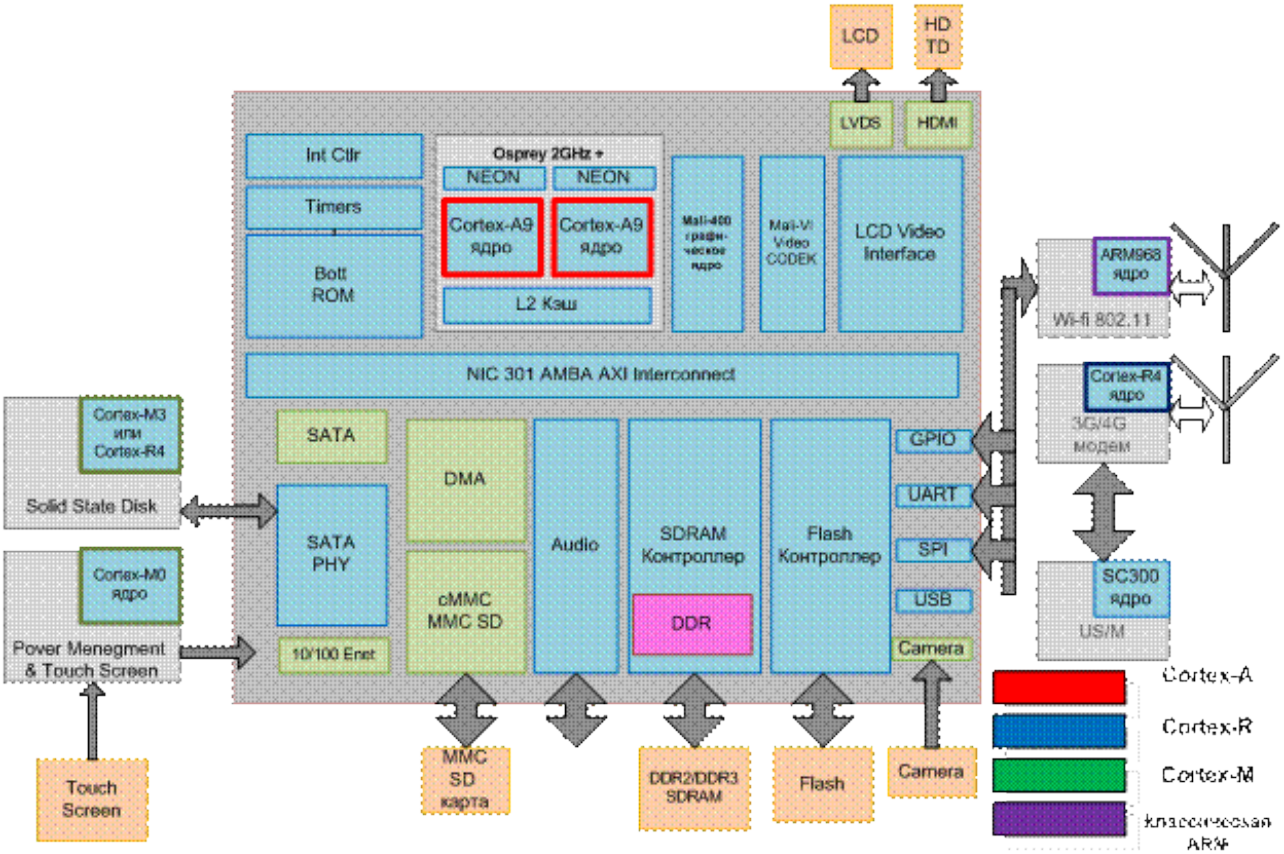

Рисунок 4.6 – Система на кристалі типового планшета

4.4 Особливості архітектури ARMv8

Процесори ARM довели свою перевагу у вбудованих системах, мобільних телефонах, планшетах, але їх використання для більш серйозних додатків головним чином було обмежено можливостями 32-розрядної адресації. Як проміжний рішення в процесорах Cortex-A15, при збереженні загальної 32-розрядної архітектури, для адресації до оперативної пам'яті до 1 Тбайт було запропоновано розширення 40-bit Large Physical Address Extensions. Однак, це рішення не забезпечувало майбутній розвиток процесорів фірми ARM як в уже завойованих областях, так і в нових областях застосування типу серверів і т.д.

Архітектура ARMv8 була представлена в кінці 2011 року і стала восьмою за рахунком версією архітектури ARM, звідси і її назва – ARMv8 (рис.4. 7).

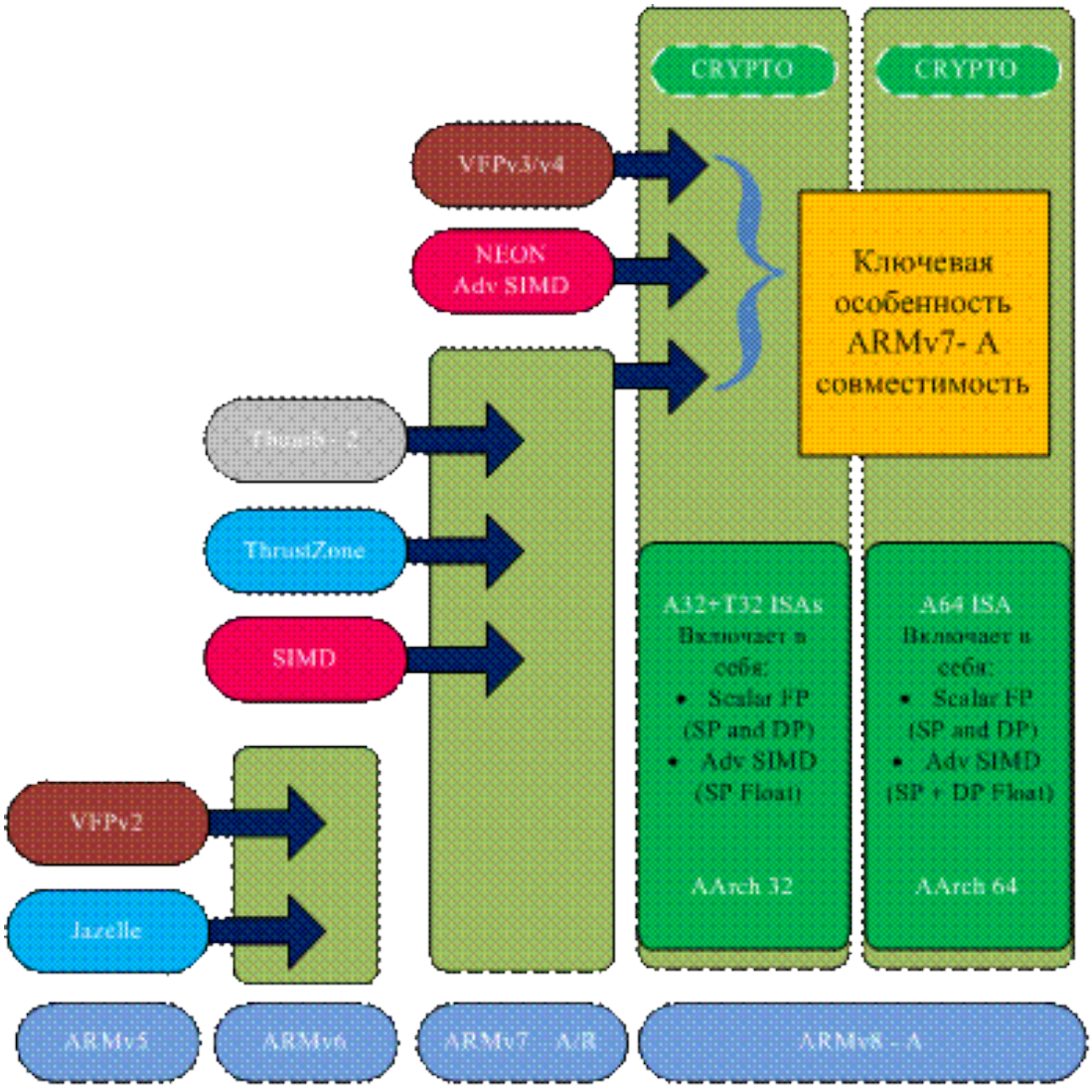

Рисунок 4.7 – Архітектура ARMv8 в ряду інших архітектур ARM

В архітектурі ARMv8-A реалізований один з найважливіших принципів при впровадженні нових технологій – абсолютно повна зворотна програмна сумісність з версією архітектури ARMv7-A (режим AArch32).

Режим AArch64 є принципово новим рішенням з новим набором команд A64. Згодом, коли не потрібно буде підтримувати успадковані додатки, планується відмовитися від режиму AArch32.

В архітектурі ARMv8 в порівнянні з архітектурою ARMv7 більш повно відбивається ідеологія архітектури RISC, створеної не для вбудованих систем, а для вирішення класичних задач рахунки і обробки даних. В архітектурі ARMv7 і в її попередників спочатку були закладені властивості, яка започаткована ще в вимоги з боку вбудованих систем і диктують необхідність спеціальним чином використовувати регістри, щоб встигати в реальному часі обробляти переривання. Ці особливості не знаходять своє відображення в режимі AArch64, тому в архітектурі ARMv8 були поєднані обидві системи команд. Набір команд режиму AArch64 адаптований до роботи в одному з двох режимів в залежності від операційної системи. Якщо встановлена операційна система Linux або інші поширені клони UNIX, то вибирається режим LP64, де цілі числа мають довжину 32 розрядв і довгі цілі – 64 розряда, а якщо операційна система Windows, то режим LLP64, де цілі і довгі цілі мають довжину 32 розряда, а дуже довгі цілі (long long integers) – 64 розряда.

У режимі AArch64 використовується новий набір команд A64 з 32-розрядними командами і 5-розрядними покажчиками регістрів для доступу до 32/64/128-розрядних регістрів. При виконанні більшості команд доступний виділений нульовий регістр. Покажчик стека і лічильник команд не входять в групу регістрів.

Нові команди підтримують 32- і 64-розрядні дані. У наборі A64 відсутні кілька команд завантаження або зберігання довільної довжини. Підтримка архітектури SIMD і операцій з плаваючою крапкою схожі з підтримкою попередньою версією архітектури – ARMv7A.

В архітектурі з'явився режим Embedded Trace Mode 4 (ETM4), який розширює адреси до 64 розрядів. Він дозволяє виявити «проміжні етапи» виконання команд, що ефективніше, ніж надання інформації про адреси.

Апаратна підтримка криптографії, здатна прискорити процеси, що використовують шифрування, в десятки разів.

Cortex-A53 і Cortex-A57 були першими процесорами, в яких реалізована архітектура ARMv8-A.

В архітектурі ARMv8 реалізовані 2 режиму Aarch32 (32-розрядний ARMv7 і 64-розрядний режим AArch64). До основних особливостей 64-розрядної режиму AArch64 відносяться:

- Новий набір команд A64.
- 31 64-розрядний регістр загального призначення.
- Окремі регістри SP і PC.
- Команди 32-розрядні і багато збігаються з командами архітектури ARMv7.
- Більшість команд працюють як з 32, так і з 64-розрядними аргументами.
- Адреси 64-розрядні.
- Реалізовано розширення NEON.
- З 16 до 32 збільшено кількість 128-розрядних регістрів, доступних через співпроцесори NEON, VFPv4, кріптокоманди AES, SHA.

4.5 Архітектура процесорного ядра Cortex

Процесорні ядра Cortex мають багато в чому загальну архітектуру і систему команд з урахуванням відмінностей між ними – це і є Гарвардська архітектура з триступінчатим конвеєром.

Процесорний ядро є повністю 32/64-розрядне і має конвеєрну RISC-архітектуру, розраховану на виконання більшості команд за один такт, і низьке енергоспоживання, що особливо важливо у вбудованих системах управління. Шини адреси і даних 32/64-розрядні, як і всі регістри загального призначення.

Одним з переваг процесорних ядер Cortex є інтеграція на кристалі центрального процесора найважливіших периферійних пристроїв, необхідних у кожному мікроконтролері для керування в реальному часі.

Кінцеві користувачі можуть самі розробляти або купувати готові операційні системи реального часу і монітори, що використовують системні периферійні пристрої. При цьому забезпечується незалежність цих програмних продуктів від обсягів пам'яті і наборів периферійних пристроїв в конкретних типах мікроконтролерів.

Як і будь-який RISC-процесор, ядра Cortex мають систему команд, оптимізовану для використання мов високого рівня і відповідних компіляторів. Система команд процесорів настільки потужна, що, найчастіше, реалізувати потрібну функцію на Асемблері навіть простіше, ніж на С.Спрощена блок-схема процесора Cortex-M4 зображена на рис. 4.8.

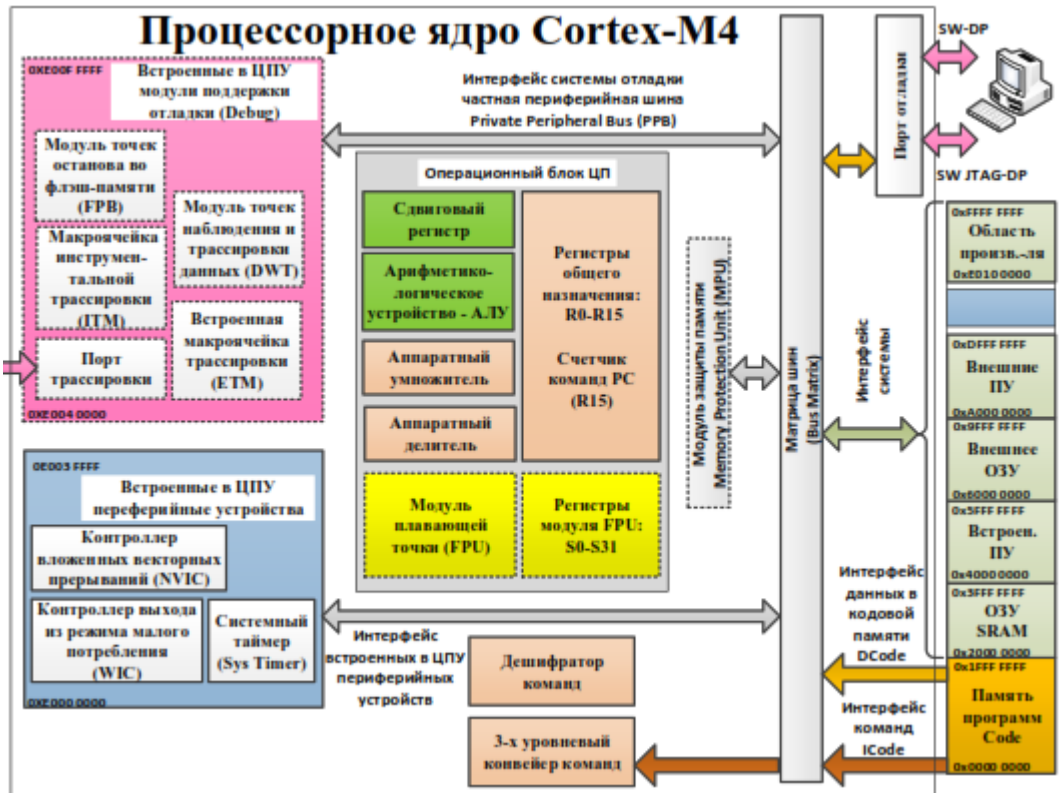

Рисунок 4.8 – Структура процесорного ядра Cortex-M4

Передбачається, що до складу операційного блоку включений опціональний модуль підтримки обчислень в форматі з плаваючою точкою FPU

Операційний блок ЦПУ містить в собі:

1) Арифметико-логічний пристрій (АЛП) для виконання арифметичних, логічних і інших операцій над 32-розрядними операндами з додатковим кільцевим зсувними регістром, що дозволяє виконувати так звані «попутні» операції зсуву при виконанні більшості звичайних операцій.

2) Апаратний помножувач 32-розрядних чисел з можливістю отримання 64-бітного результату.

3) Апаратний дільник 32-розрядних чисел.

4) Файл регістрів загального призначення (РОН) r0-r15, кожен з яких може бути джерелом або приймачем операції.

5) Співпроцесор для обробки чисел в форматі з плаваючою точкою FPU.

6) Файл регістрів співпроцесора S0-S31, кожен з яких може бути джерелом або приймачем операції з плаваючою точкою.

Один з регістрів загального призначення відіграє особливу роль (R15), будучи лічильником команд PC, вміст якого задає адресу чергової команди в кодової пам'яті, яка підлягає виконанню.

4.6 Організація пам'яті

Адресний простір пам'яті (рис.4.9) лінійне розміром 4Г байт. Використовується стандартний розподіл пам'яті з зумовленими виділеними адресними просторами для пам'яті програм, ОЗП, пристроїв зовнішньої пам'яті або периферійних пристроїв, а також вбудованою периферією. Також є спеціальна область пам'яті, яка містить адреси, зарезервовані виробником.

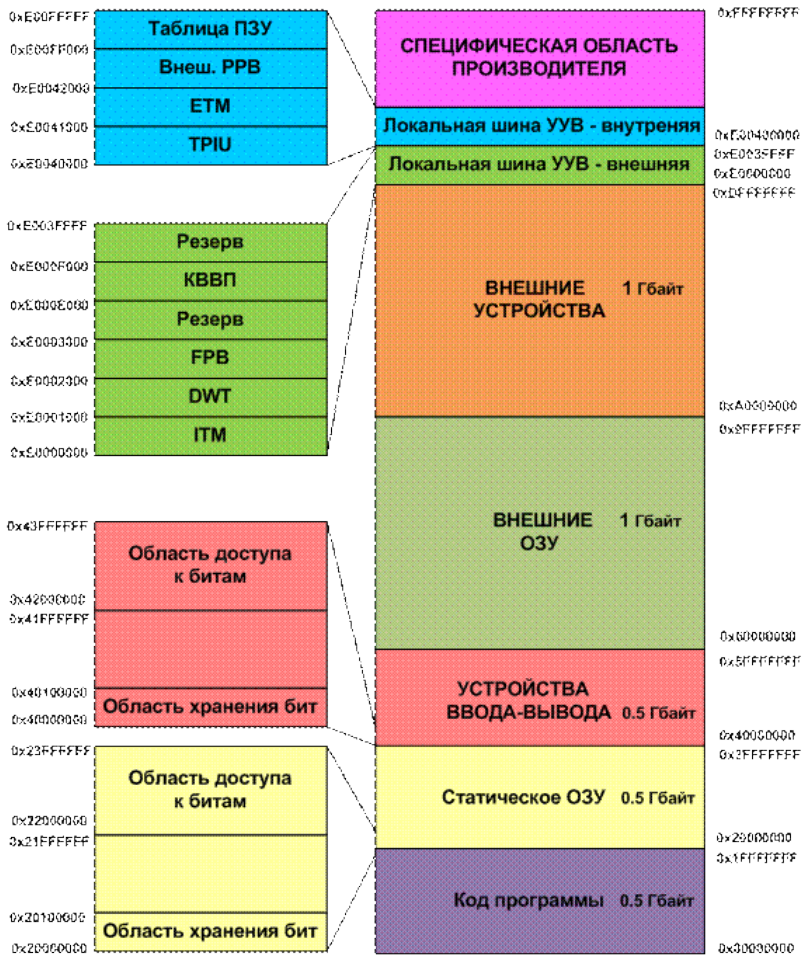

Рисунок 4.9 – Структура адресного простору пам'яті Cortex-M4

4.7 Особливості Гарвардської архітектури процесорів Cortex

У процесорах з Гарвардської архітектурою всі три шини (шина даних, шина адреси і шина управління) умовно об'єднуються в одну і називаються шинами інтерфейсу процесора з кодовою пам'яттю, пам'яттю даних, периферійними пристроями. Для кожної з таких шин розробники процесорів визначають розрядність даних, адрес, порядок обміну інформацією, тобто протоколи обміну даними. Шина управління застосовується окремо для кодової пам'яті, пам'яті даних і периферійних пристроїв.

Отже, відмінна риса Гарвардської архітектури – можливість паралельного виконання кількох дій відразу:

– зчитування коду чергової команди з кодовою пам'яті;
– читання значень операндів з пам'яті даних або збереження в ній результату попередньої операції.

Паралельно можуть виконуватися також операції отримання чергової команди з кодовою пам'яті і читання/запису в пристрої введення/виводу. Якщо пам'ять даних є дво-портовою (допускає одночасну паралельну запис в неї даних за однією адресою і зчитування даних за іншою адресою), то можливий ще більший паралелізм – зчитування чергового операнда для поточної операції і одночасне збереження результату попередньої операції.

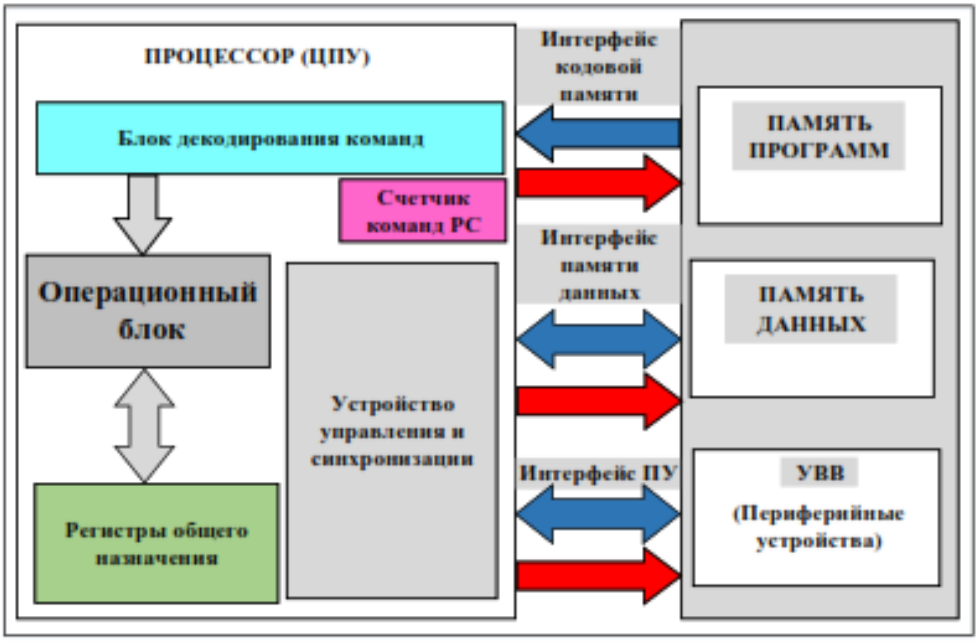

Рисунок 4.10 – Гарвардська архітектура процесорів

З одного боку, використання цієї архітектури – істотне підвищення швидкодії системи, а з іншого – значне ускладнення апаратної реалізації процесора. Велике число шин означає також велике число пінів процесора, якщо елементи пам'яті – зовнішні, а також наявність для кожної з них свого власного пристрою управління і синхронізації. Саме ускладнення апаратури затримало розробку процесорів з Гарвардською архітектурою на десятиліття. Вона стала можливою тільки при різкому підвищенні рівня інтеграції транзисторів на кристалі і здешевлення процесорних БІС.

Процесори з Гарвардської архітектурою мають окремі оптимізовані інтерфейси з кодовою пам'яттю, пам'яттю даних і периферійними пристроями, що дозволяє поєднати кілька типових операцій по часу і різко підняти продуктивність процесора. Гарвардська архітектура передбачає (опціонально) наявність додаткових модулів управління пам'яттю, які мають апаратні засоби блокування несанкціонованого доступу до заданих програмістом областей пам'яті по запису для підвищення надійності системи в процесі експлуатації.

Практично всі сучасні процесори, що мають Гарвардською архітектуру, обробляють команди на так званому «конвеєрі команд», приклад зображений на рис. 4.3. Завдання обробки команди розбивається на кілька етапів, на кожному з яких над командою виконуються певні дії. Ці дії залежать від місця розташування команди на конвеєрі. Кожне місце (каскад конвеєра) має свій власний обробник команди. У найпростішому випадку команду необхідно, як мінімум:

1) вибрати з пам'яті по інтерфейсу «Процесор» – «Кодова пам'ять». Оброблювач цього каскаду витягує чергову команду з пам'яті і розміщує її на конвеєрі – завантажує конвеєр новою командою;

2) декодувати код операції, що вже знаходиться на конвеєрі команди – визначити, що ж повинен робити операційний блок процесора, яка мікропрограма обробки даних повинна бути запущена;

3) виконати. Обробником цього каскаду конвеєра є операційний блок

Всі етапи конвеєра виконуються за один і той же час, так званий такт конвеєра. Вище ми привели список завдань, характерний саме для 3-х рівневого конвеєра процесорних ядер ARM-Cortex-M3 / M4 / M4F. Далі на кожному черговому такті одночасно будуть виконуватися відразу три команди. З боку

спостерігача на виході конвеєра створюється повне враження про те, що кожна команда виконується лише за один такт.

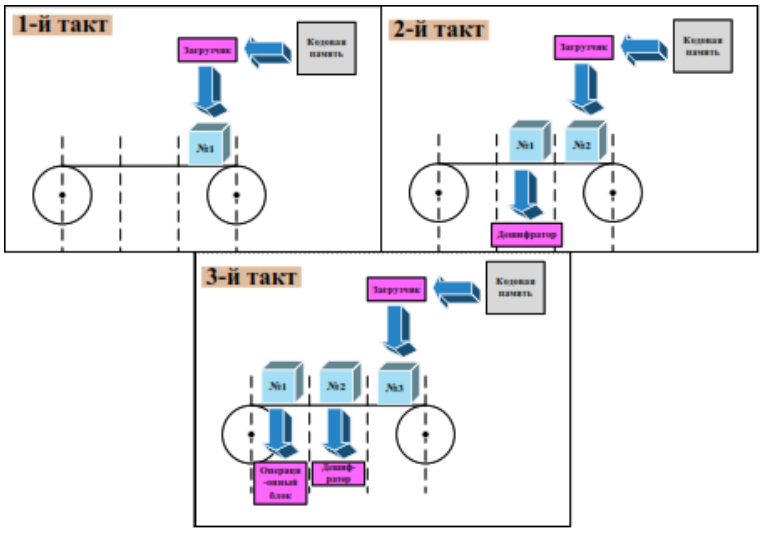

Рисунок 4.11 – 3-х рівневий конвеєр Гарвардської архітектури

Насправді кожна команда знаходиться на конвеєрі три такту, але на різних етапах виконання: завантаження, дешифрування, власне виконання.

Конвеєр виконує попередню вибірку команд з пам'яті і послідовну їх обробку, істотно підвищуючи продуктивність процесора.

Будь-який сучасний мікроконтролер має процесор, вбудовану пам'ять програм і дані певного розміру, обмежений набір вбудованих периферійних пристроїв. У більшості практичних застосувань список цих ресурсів мікроконтролера є підставою для його вибору. Це так звана «закрита», нерозширювана архітектура мікроконтролера. Однак, в ряді програм виникає необхідність при збереженні мікроконтролера або розширити його пам'ять програм або даних, або підключити до нього якісь інші периферійні пристрої. Такі мікроконтролери є розширюваними і повинні містити шини для підключення зовнішнього обладнання. Вони повинні мати так звану відкриту архітектуру.

Модифікована Гарвардська архітектура передбачає що внутрішня архітектура мікроконтролера є справжньою багатошинною Гарвардською, а для сполучення з зовнішніми БІС пам'яті і зовнішніми периферійними пристроями використовуються загальні шини адреси і даних і загальна шина управління, що містить сигнали читання і запису, а також сигнали вибірки потрібної області пам'яті.

4.8 Flash-накопичувач для збереження операційної системи

Згідно структурної схеми, flash-накопичувач, який буде зберігати в собі операційну систему та службову інформацію для коректної роботи системи. До flash-накопичувача висувається лише 3 умови:

– microSD формат;

– об’єм 16 Гбайт;

– 10 клас flash-накопичувача.

Розглянемо зведену табл. 4.3 в якій наведемо декілька flash-накопичувачів.

Таблиця 4.3 – Вибір flash-накопичувача

Назва	Об’єм, Гб	Клас	Ціна, грн
Kingston Canvas Select	16	10	90
Apacer microSDHC	16	10	116
SanDisk Ultra microSDHC	16	10	130
Transcend Premium 400x	16	10	160
Kingston Industrial Temperature	16	10	449

З табл. 4.3 нас задовольняє Kingston Canvas Select, він підходить нам по головним параметрам та має найбільш привабливу ціну. Опишемо його детальніше.

Карти пам'яті Canvas Select Plus microSD рис. 4.12, компанії Kingston сумісні з пристроями Android і володіють продуктивністю класу A1.

Рисунок 4.12 – Карта пам'яті Canvas Select Plus microSD

Наведемо всі важливі характеристики flash-накопичувача в табл. 4.4.

Таблиця 4.4 – Характеристики Kingston Canvas Select

Об’єм, Гб	16
Швидкість зчитування, Мб/с	100
Розміри (В х Д х Т), мм	11 х 15 х 1
Напруга, В	3,3

4.9 Блок живлення

З технічної документації нам відомо, що одноплатному комп’ютеру Raspberry Pi 3 B+ для нормальної роботи потрібна напруга в 5 вольт та струм 2.5 ампери. [18] Жорсткий диск буде отримувати живлення через розпаяний на платі USB-порт, тому для стабільної роботи всієї системи, потрібно підібрати блок живлення за наступними характеристиками:

– напруга 5 В;

– струм 3 А.

Розглянемо зведену табл. 4.5 в якій наведемо декілька блоків живлення.

Таблиця 4.5 – Вибір блоку живлення

Назва	Струм, А	Напруга, В	Ціна, грн
Samsung EP-TA800	3	5	279
Florence FL-1050	3	5	264
Hoco C69A	4,5	5	402
Grand-X CH-850	4,5	5	373

З табл. 4.5 нас задовільняє Samsung EP-TA800, він підходить нам по головним параметрам та від перевіреного виробника, що у виборі блоків живлення відіграє не останню роль [12].

4.10 Конвертор SATA в USB

Конвертор SATA в USB. Для зв’язку між одноплатним комп’ютером та жорстким диском необхідно узгодити їх роботу так як в Raspberry Pi 3 B+ немає реалізованого SATA порту. Для цього необхідно застосувати спеціальний конвертор. Спроектуємо його на спеціалізованій мікросхемі FT232BM.

Мікросхема FT232BM, структурна схема якої зображена на рис. 4.13, є представителем другого покоління перетворень. Вона була розроблена на базі FT8U232AM і реалізувала передачу даних з протоколу USB в UART і навпаки. Функціональні можливості були розширені завдяки новому режиму Bit Bang, коли вихідний інтерфейс мікросхем дозволяє реалізувати вісім незалежних лінійних вводів та виводів.

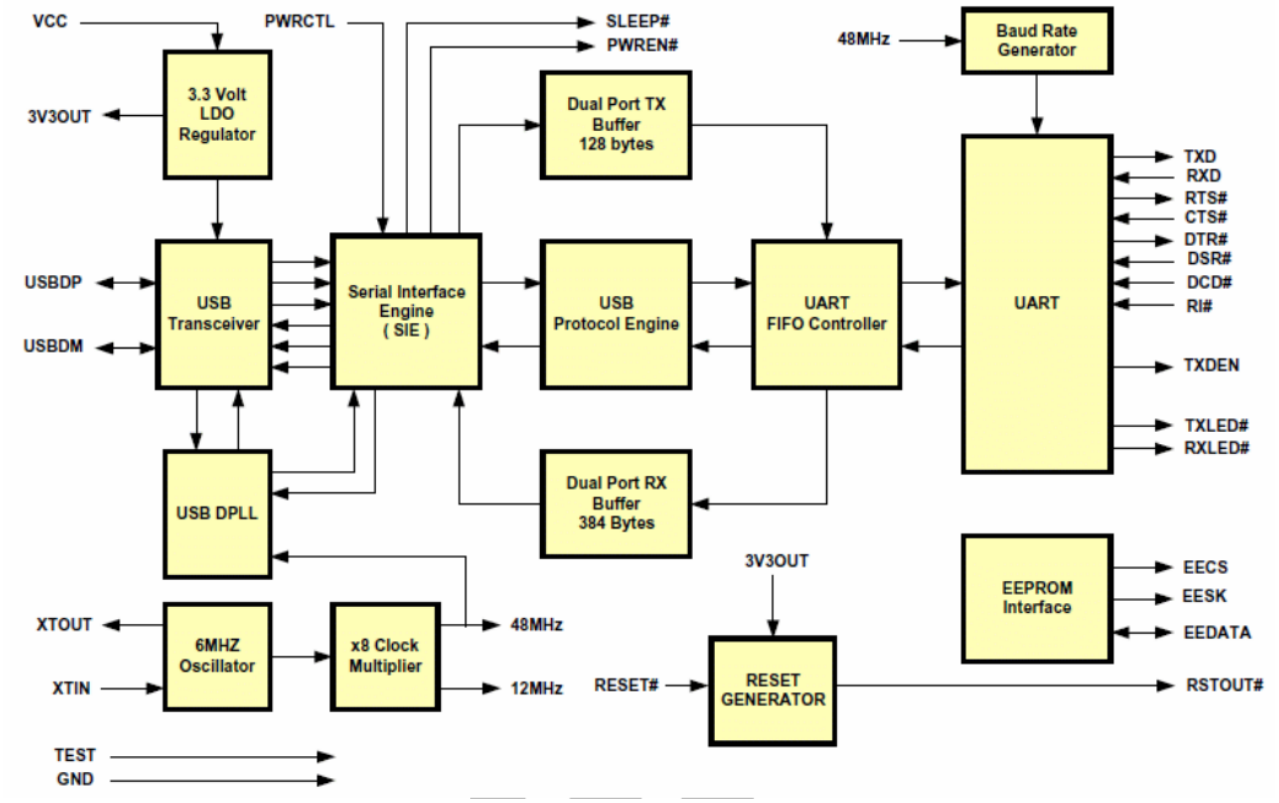

Рисунок 4.13 – структурна схема FT232BM

Мікросхема FT232BM забезпечує передачу даних із швидкістю до 2000 кБіт/с. Вбудований конвертер рівнів тепер дозволяє обробляти сигнали UART на логічних рівнях 5В CMOS. Відповідний контакт VCC-IO дозволяє пристрою підключити до пристрою 3.3 вольтову логіку, таким чином, відкидаючи необхідність у зовнішньому конверторі логічних рівнів.

У нових ревізіях мікросхем, значно покращена система керування живленням для високовольтних пристроїв. У мікросхемах попереднього покоління, USBEN контакт залишався активним при підключенні до пристрою USB. Для контролю живлення, сигнал необхідно було підключати через зовнішню комунікацію з контактами SLEEP # і RESET #. Тепер такий затвор реалізований в середині мікросхеми, сигнал PWREN # може бути використаний

для прямого управління транзистором або p-канальним MOSFET-том у додатках, де потрібна комутація живлення зовнішніх схем.

Завдяки EEPROM можливостям, стало можливо повільно відключати лінії UART-інтерфейсу при вимкненні живлення. У такому режимі будь-яка залишкова напруга на зовнішній схемі перенаправляється на землю при вимкненні живлення, забезпечивши, таким чином, надійній перезапуск зовнішніх схем, контактом PWREN #, при відновленні подачі живлення.

Час затримки буферу приймача в мікросхемі FT232BM може бути запрограмована в діапазоні від 1 до 255 мс з кроком 1 мс. Завдяки цьому, пристрій може бути оптимізовано для протоколів, що вимагають малого часу відклику для малого обсягу переданих даних.

Мікросхема випускається лише в форматі LQFP, що дещо ускладнює розробку та виготовлення печатної плати але в той же час, готова схема буде займати мінімум корисного простору, мікросхема зображена на рис. 4.14.

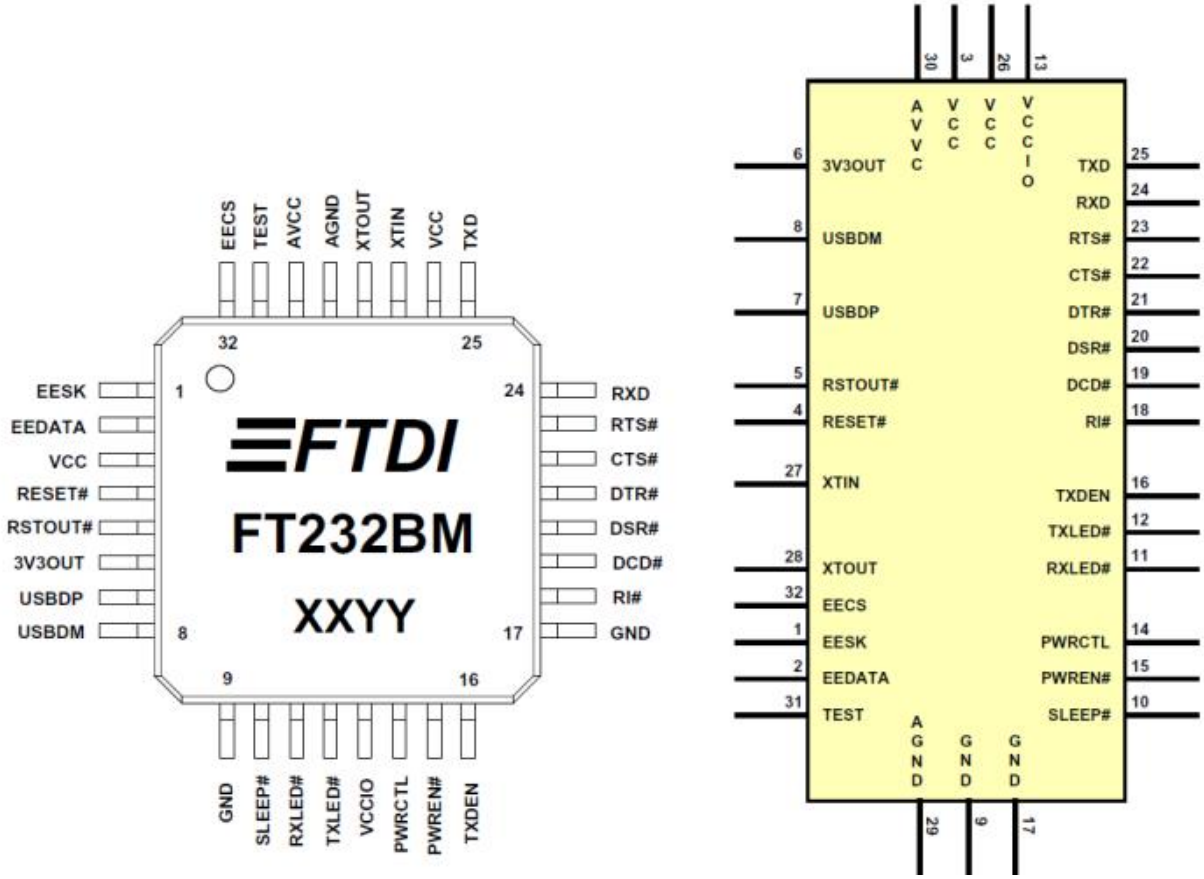

Рисунок 4.14 – Розпіновка корпусу та схематичне відображення FT232BM

За рис. 4.14 складемо табл. 4.6, де приведемо всі доступні порти мікросхеми FT232BM та зробимо короткий їх опис.

Таблиця 4.6 – Перелік та призначення портів FT232BM

№	Назва	Тип	Опис
1	EESK	Out	Сигнал лічильника на EEPROM. Додавання 10кОм резистора на EESK призведе до того, що FT232BM використовуватиме USB-ідентифікатор продукту 6004
2	EEDATA	I/O	EEPROM – введення та вивід даних безпосередньо підключається до входу даних EEPROM і до виходу даних EEPROM через резистор 2,2кОм.
3,26	VCC	pwr	Від +4,35В до +5,25В
4	RESET#	In	Активний пін скидання низьким сигналом.
5	RSTOUT	Out	Вихід внутрішнього генератора скидання. Залишається високим імпедансом протягом ~5 мс після VCC > 3,5 В
6	3V3OUT	Out	3,3 вольта на виході від вбудованого регулятора LDO. Його основна мета - забезпечити внутрішнє живлення 3,3 В до комірки USB-трансивера та виводу RSTOUT #.
7	USDP	I/O	USB Data Signal Plus
8	USDM	I/O	USB Data Signal Minus
9,17	GND	pwr	Пристрій заземлення пінів
10	SLEEP#	Out	Режим призупинення USB від низького рівня.
11	RXLED	O.C	Вхід для світлодіода – індикатор при отриманні даних через USB
12	TXLED	O.C	Вхід для світлодіода – індикатор при передачі даних через USB
13	VCCIO	pwr	+3,0 В до +5,25 В VCC до контактів інтерфейсу UART 10 ... 12, 14 ... 16 і 18 ... 25.
14	PWRCT	In	Шина з живленням
15	PWREN#	Out	Низький рівень після налаштування пристрою через USB, потім високий під час призупинення роботи USB.
16	TXDEN	Out	Увімкнення передачі даних для RS485.
18	RI#	In	Вхід управління кільцевим індикатором.
19	DCD#	In	Вхід управління виявлення носіїв даних.
20	DSR#	In	Набір даних Data Set Ready / сигнал рукостискання.
21	DTR#	Out	Data Terminal Ready Control / сигнал рукостискання.
22	CTS#	In	Clear To Send Control / сигнал рукостискання.
23	RTS#	Out	Request to Send Control/ сигнал рукостискання.
24	RXD	In	Отримання асинхронних даних.
25	TXD	Out	Передача асинхронних даних.
27	XTIN	In	Вхід для кристалічного генератора 6 МГц.
28	XOUT	Out	Вихід для кристалічного генератора 6 МГц. XTOUT перестає коливатися під час призупинення роботи USB.
29	AGND	pwr	Пристрій аналогового заземлення для внутрішнього тактового множника x8
30	AVCC	pwr	Пристрій аналогового джерела живлення для внутрішнього тактового множника x8
31	TEST	In	Переводить пристрій у режим перевірки мікросхеми. Повинен бути прив’язаний до GND для нормальної роботи.
32	EECS	I/O	EEPROM - вибір мікросхеми. Для роботи 48 МГц підтягніть EECS до GND за допомогою резистора 10 кОМ.

Тепер розглянемо докладніше кожну функцію в FT232BM.

+ 3,3 В LDO регулятор. 3,3 В LDO-регулятор генерує опорну напругу 3,3 вольта для керування вихідними буферами комірки трансивера USB. Він вимагає зовнішнього роз'єднуючого конденсатора, який повинен бути приєднаний до вихідного піна регулятора «3V3OUT». Він також забезпечує живлення 3,3 В на контакті «RSTOUT #». Основна функція цього блоку полягає у живленні USB-трансивера та сброс генератора, а не у зовнішній логіці. Однак зовнішні схеми, що вимагають номінальної напруги 3,3 В при струмі, що не перевищує 5 мА, за необхідності можуть також отримувати свій струм через вивід «3V3OUT».

USB трансивер. Стільник трансивера USB забезпечує повношвидкісний фізичний інтерфейс USB 1.1/USB 2.0 до кабелю USB. Вихідні драйвери забезпечують сигналізацію контролю рівня зниження рівня + 3,3 В, тоді як диференціальний вхідний приймач і два односторонні вхідні приймачі забезпечують дані USB в режимах Single-Ended-0 (SE0) та USB-виявлення. Ця функція також включає внутрішні термінальні резистори серії USB на лініях передачі даних USB та підтягуючий резистор 1,5 кОм на USBDP.

USB DPLL. Осередок USB DPLL блокується на вхідні дані USB NRZI та генерує відновлені сигнали лічильники та даних для блоку послідовного інтерфейсу (SIE).

Внутрішній осцилятор 12 МГц. Внутрішня комірка осцилятора 12 МГц генерує контрольний тактовий сигнал 12 МГц. Це забезпечує вхід до функції x4 Clock Multiplier. Осцилятор 12 МГц також використовується як еталонний лічильник для блоків контролерів SIE, USB Protocol Protocol і UART FIFO.

Дільник та множник частоти. Тактовий множник або дільник приймає 12 МГц на вхідну функцію внутрішнього генератора і генерує опорні тактові сигнали 48 МГц, 24 МГц, 12 МГц і 6 МГц. Для генерування тактової частоти 48 МГц, потрібно задіяти блоки USB DPLL та генератором швидкості передачі.

Механізм послідовного інтерфейсу (SIE). Блок механізму послідовного інтерфейсу (SIE) виконує паралельне послідовне та послідовне перетворення даних USB. Відповідно до специфікації USB 2.0, він виконує набивання/ розбивання бітів та генерацію CRC5 / CRC16. Він також перевіряє CRC на потоці даних USB.

Механізм протоколу USB. Механізм протоколу USB керує потоком даних від кінцевої точки керування USB пристрою. Він обробляє запити протоколу USB низького рівня, що генеруються контролером USB-хосту, та команди для

управління функціональними параметрами UART відповідно до розділу 9 специфікації USB 2.0.

Буфер FIFO RX (128 байт). Дані, надіслані з контролера хоста USB на UART через кінцеву точку USBdata OUT, зберігаються в буфері FIFO RX (отримання). Дані видаляються з буфера в регістр передачі UART під контролем контролера UART FIFO. (Rx щодо інтерфейсу USB).

Буфер FIFO TX (256 байт). Дані з реєстру прийому UART зберігаються в буфері TX. Контролер хоста USB видаляє дані з буфера FIFO TX, надсилаючи запит USB на дані з кінцевої точки даних пристрою. (Tx щодо інтерфейсу USB). Контролер UART FIFO обробляє передачу даних між буферами FIFO RX і TX та регістрами передачі та прийому UART.

Контролер UART з програмованою інверсією сигналу та високим приводом. Разом з контролером UART FIFO контролер UART обробляє передачу даних між буферами FIFO RX та FIFO TX та регістрами передачі та прийому UART. Він виконує асинхронні 7 або 8 біт паралельно послідовному та послідовному паралельному перетворенню даних на інтерфейсі RS232 (або RS422 або RS485). Сигнали управління, підтримувані режимом UART, включають RTS, CTS, DSR, DTR, DCD та RI. Також підтримуються варіанти рукостискання RTS / CTS, DSR / DTR та XON / XOFF. Встановлення зв'язку здійснюється апаратно, щоб забезпечити швидкий час відгуку. Інтерфейс UART також підтримує налаштування та умови виявлення RS232 BREAK. Крім того, сигнали UART можуть бути індивідуально інвертовані та мати конфігуровану високу потужність приводу. Обидві ці функції можна налаштувати в EEPROM.

Рис. 4.15 ілюструє FT232R у типовій конфігурації з автономним живленням USB. Пристрій USB із власним живленням отримує живлення від власного джерела живлення VCC і не забирає струм з шини USB. Основні правила для пристроїв з автономним живленням USB такі:

- пристрій з автономним живленням не повинен отримувати струм через шину USB, коли контролер USB-хоста або концентратора відключений;
- пристрій з автономним живленням може використовувати стільки струму, скільки йому потрібно під час нормальної роботи та призупиняти USB;
- пристрій з автономним живленням можна використовуватися з будь-яким USB-хостом, USB-концентратором, що працює від шини, або концентратором USB.

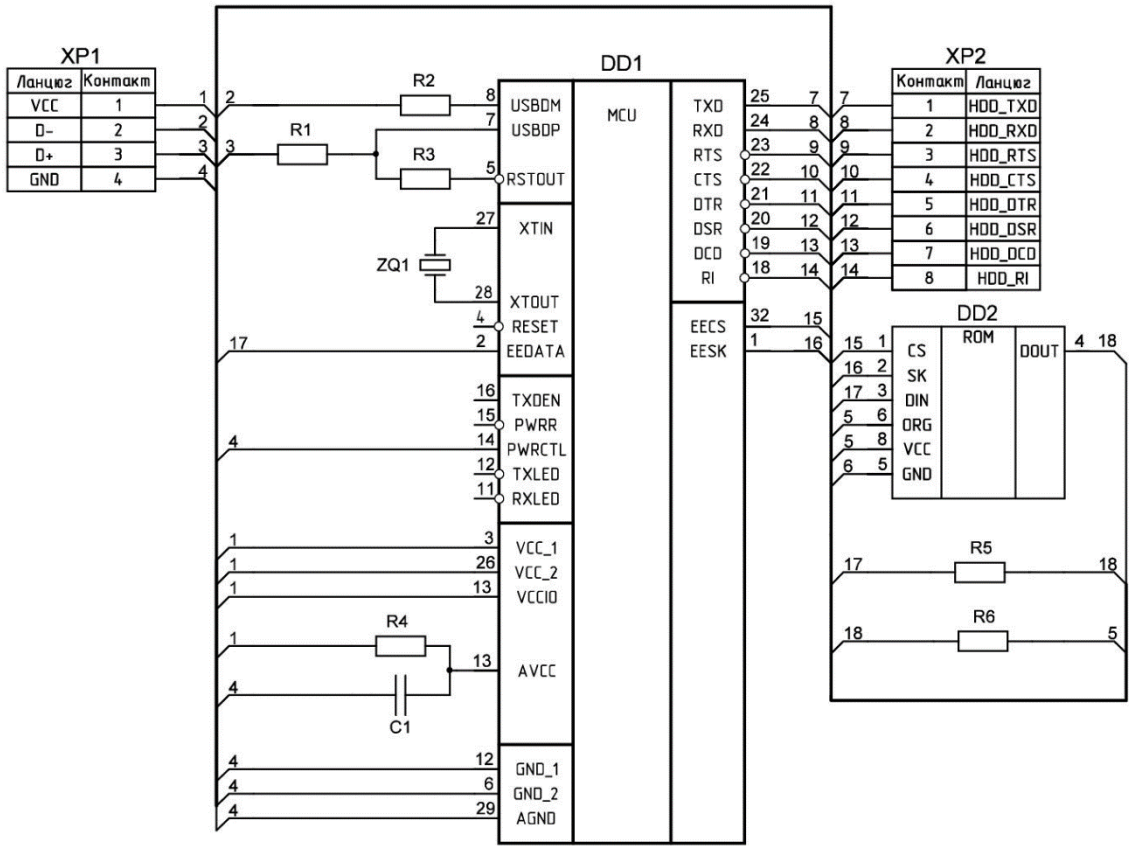

Рисунок 4.15 – Принципова схема конвертора

Дескриптор живлення у внутрішньому EEPROM FT232R повинен бути запрограмований на значення нуля (з автономним живленням). Для того, щоб відповідати першій вимозі вище, потужність USB-шини (контакт 1) використовується для управління контактом RESET # пристрою FT232R. Коли живлення або концентратор USB підключається, на USBDP отримаємо напргуго до + 3,3 В (генерується за допомогою резисторної мережі 4700Ом та 10кОм), таким чином виконується ідентифікація пристрою як повношвидкісного пристрою для USB-хосту або концентратора. Коли USB-хаб чи концентратор вимкнені, RESET # буде низьким, а FT232R утримується у скиданні. Оскільки RESET # низький, внутрішній резистор 1,5 кОм не підтягується до жодного джерела живлення (концентратор або хост відключений), тому струм не тече по USBDP через підтягуючий резистор 1,5 кОм. Якщо цього не зробити, деякі контролери хостів або концентраторів USB можуть нестабільно живити систему. Рис. 4.15 ілюструє конструкцію з автономним живленням, яка має джерело від + 4 В до + 5,25 В.

4.11 Пристрій збереження інформації HDD

Для збереження інформації користувача потрібно обрати жорсткий диск. Класичний HDD складається:

- плата керування;
- шпиндельний двигун;
- блок магнітних головок;
- магнітний диск.

Плата керування, яка зображена на рис. 4.16, являє собою інтегральну схему, яка керує роботою HDD і обробляє сигнали, отримані з магнітних головок, перетворюючи їх в зрозумілі комп'ютеру сигнали за технологією SATA. Плата контролера має свій процесор, пристрої ПЗП і ОЗП, а також мікросхему управління шпиндельним двигуном диска [11].

Рисунок 4.16 – Плата керування жорстким диском

Шпиндельний двигун, зображений на рис. 4.8, який отримує команди мікроконтролера на платі керування, приводить в рух магнітний диск, обертаючи його зі швидкістю 5400-10000 обертів в хвилину. Від нього напряму залежить швидкодія та рівень шуму жорсткого диску.

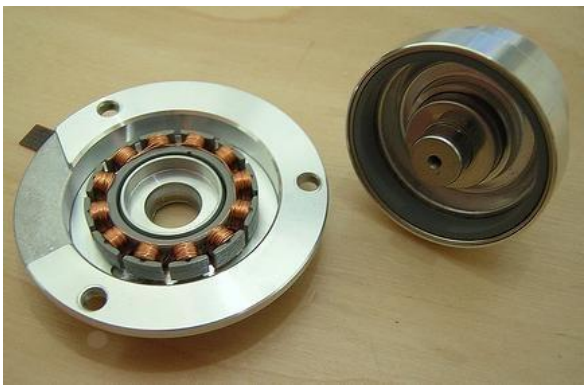

Рисунок 4.17 – Шпиндельний двигун HDD

Блок магнітних головок, зображений на рис. 4.18, складається з мікросхеми попереднього підсилювача-комутатора і коромисла, на кінці пластин якого розташовані магнітні головки, за допомогою них відбувається зчитування та запис інформації на диск.

Рисунок 4.18 – Блок магнітних головок

Магнітний диск, який зображений на рис. 4.19, головний елемент вінчестера, саме на ньому зберігається вся інформація користувача. Являє собою пластину круглої форми, зроблену з алюмінію або скла і покриту тонким шаром феромагнітного матеріалу, як правило, двоокисом хрому. Саме цей верхній магнітний шар і є основою для запису даних за допомогою магнітної головки.

Рисунок 4.19 – Магнітний дюиск HDD

Всі ці елементи напряму впливають на основні характеристики HDD – інтерфейс підключення, об’єм, форм-фактор, швидкість обертання шпинделя, об’єм буфера та час довільного доступу. З цього переліку нас цікавлять наступні параметри, за якими будемо підбирати HDD:

– форм-фактор, повинен бути 2,5;
– інтерфейс підключення SATA;
– об’єм 1 Тб.

Розглянемо зведену табл. 4.7, в якій наведемо декілька вінчестерів.

Таблиця 4.7 – Вибір жорсткого диску

Назва	Об’єм, Гб	Частота, об/хв	Буфер обміну, Мб	Ціна, грн
WD Blue 2.5"	1000	5400	128	1269
WD Black Performance 2.5"	1000	7200	64	2356
Seagate BarraCuda Pro 2.5	1000	7200	128	1785
Toshiba L200 2.5	1000	5400	128	1568
WD NasWare Red 2.5	1000	5400	16	2756

З табл. 4.7 нас задовільняє WD Blue 2.5", він підходить нам по головним параметрам та має найбільш привабливу ціну.

5 РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ПРИСТРОЮ ХМАРНОГО СХОВИЩА

5.1 Налаштування оболонки

NextCloudPi є операційною системою для Rasbperry Pi, призначена саме для NAS систем. Для неї використовували модель Raspberry Pi 3B+. Операційна система Raspbian поставляється з фірмовим інсталятором NOOBS (New Out Of Box Software) [14]. В даний час в NOOBS включені наступні операційні системи:

- Raspbian;
- LibreELEC;
- OSMC;
- Recalbox;
- Lakka;
- RISC OS;
- Screeny OSE;
- Windows 10 IoT Core;
- TLXOS.

Починаємо роботу з NOOBS v1.3.10 за вересень 2014 року. В NOOBS за замовчуванням офлайн встановлюється тільки Raspbian. Інші можуть бути встановлені за допомогою під'єднання до мережі Інтернет прямо в процесі встановлення, приклад вікна ОС Raspbian, наведений на рис. 5.1.

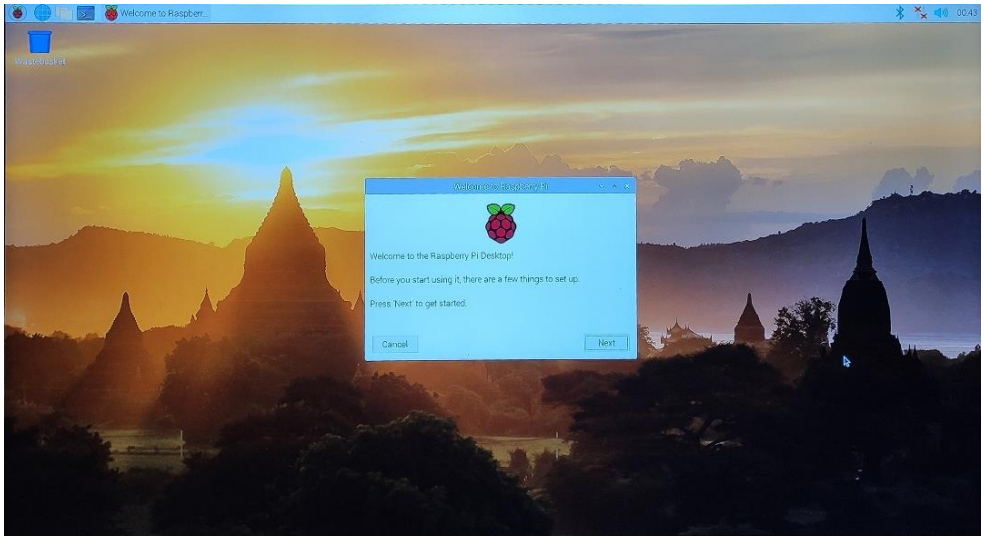

Рисунок 5.1 – Операційна система Raspbian

NOOBS доступний в двох форматах: змішаний (офлайн інсталятор та онлайн інстялятор) або тільки онлайн інсталятор. Повна версія включає ОС Raspbian, тому її можна встановити з SD-карти в автономному режимі, тоді як для використання NOOBS Lite або встановлення будь-якої іншої операційної системи потрібне підключення до мережі інтернет.

NextCloudPi – це встановлена і налаштована оболонка Nextcloud, що включає інтерфейс управління з усіма інструментами, необхідними для самостійного розміщення особистих даних в одному пакеті. Це офіційний проект спільноти з відкритим вихідним кодом, мета якого полегшити кожному контроль над власними даними.

На відміну від інших варіантів, дана ОС не вимагає великих знань в області програмування на Linux, має простий інтерфейс і установка займає на багато менше часу.

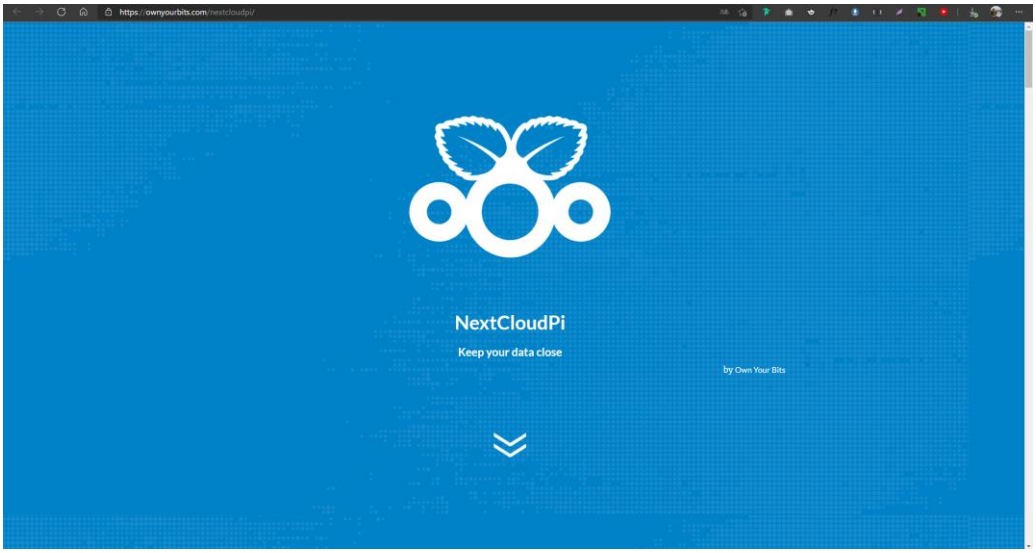

Рисунок 5.2 – Сайт NextCloudPi

Підготовка до встановлення операційної системи. Першим етапом є завантаження інсталятора операційної системи NextCloudPi. Він може бути завантажений у вигляді архіву .bz2 з офіційного сайту ownyourbits.com [15].

Поки йде завантаження, нам потрібно підготувати SD-картку, на яку буде і записана операційна система. Все, що знаходиться на ній буде видалено після форматування. Сформуємо чіткий порядок дій для підготування SD-карти:

– завантажити програмне забезпечення Rufus 3.14 [16];

– дотримуючись інструкцій виконати установки програмного забезпечення;
– вставити SD-карту в роз'єм ПК або ноутбука;
– у вікні програми, який зображений на рис. 5.3, обираємо образ з програмним забезпеченням NextCloudPi.img;
– натиснути кнопку «СТАРТ»

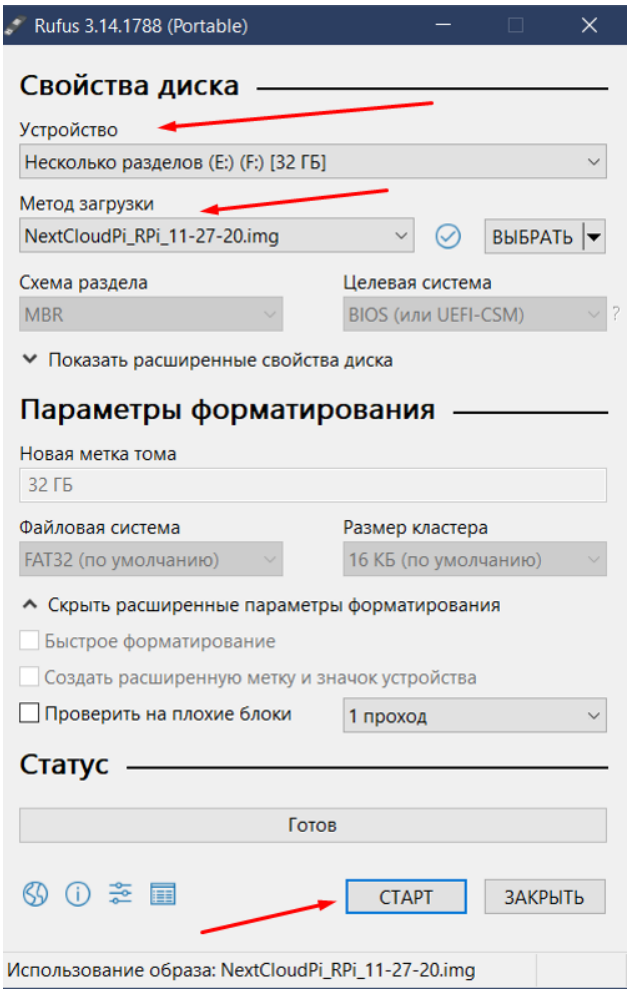

Рисунок 5.3 – Інтерфейс програми RUFUS

Після завантаження операційної системи на SD-карту за допомогою безкоштовної програми Rufus, збираємо компоненти в систему. Зібрана система наведена на рис. 5.4. Також наведемо алгоритм збірки необхідних компонентів:
– підключаємо до плати вивід зображення на монітор через HDMI кабель;
– підключаємо клавіатуру, можна використовувати бездротову;
– вставити флешку або HDD на якому будуть зберігатися файли.

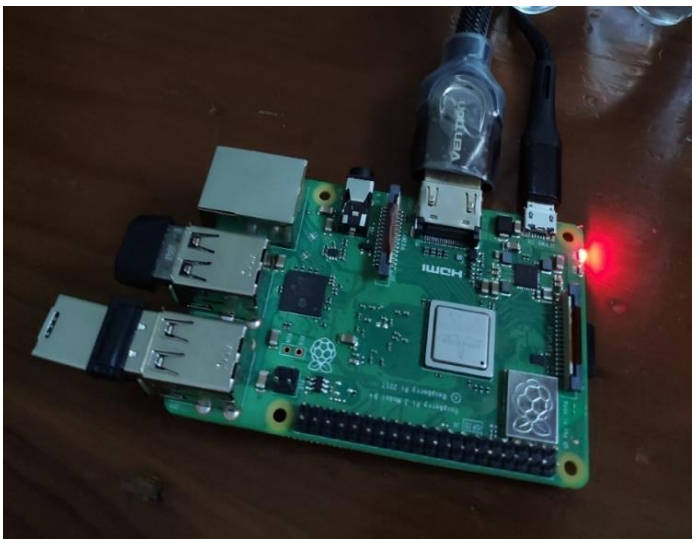

Рисунок 5.4 – Вигляд NAS на етапі налаштування

Перше завантаження може зайняти деякий час, процес завантаження зображений на рис. 5.5 [17].

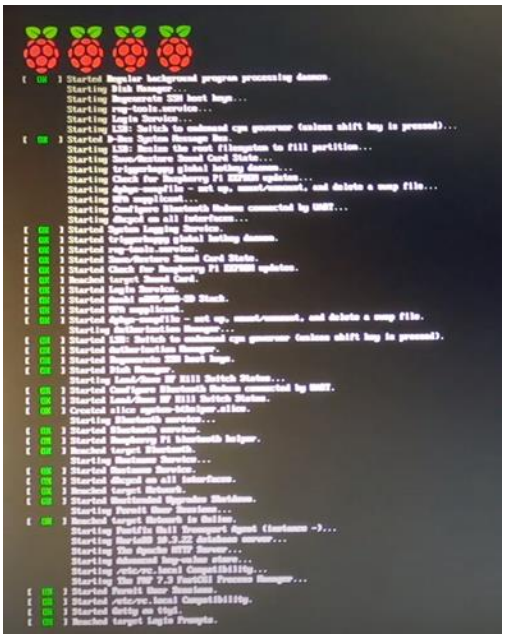

Рисунок 5.5 – Перше завантаження операційної системи

Під час першого завантаження користувачу буде доступний лише linux-термінал але навіть так, потрібно авторизуватися. За замовчуванням, при першому завантаженні ОС логін буде «pi», а пароль «raspberry». Звичайно, після всіх налаштування потрібно змінити логін та пароль але зараз ми просто надаємо права супер-користувача командою **sudo raspi-config.** Приклад виконання команди зображений на рис. 5.6.

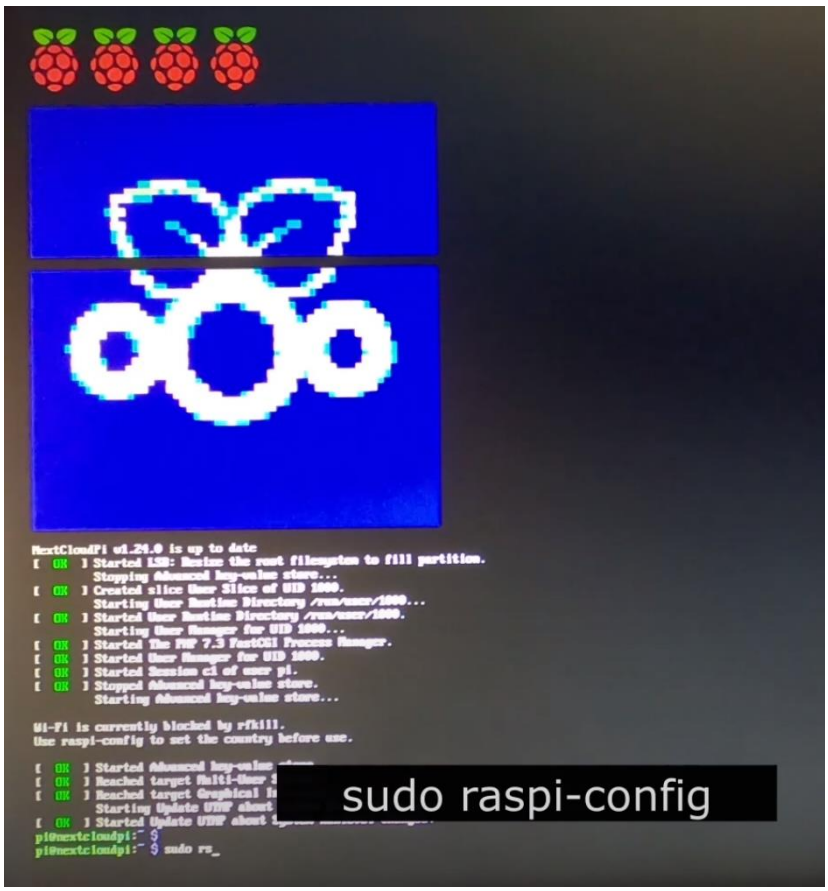

Рисунок 5.6 – Надання прав супер-користувача

Після цього ми потрапляємо у меню конфігурації системи, рис. 5.7.

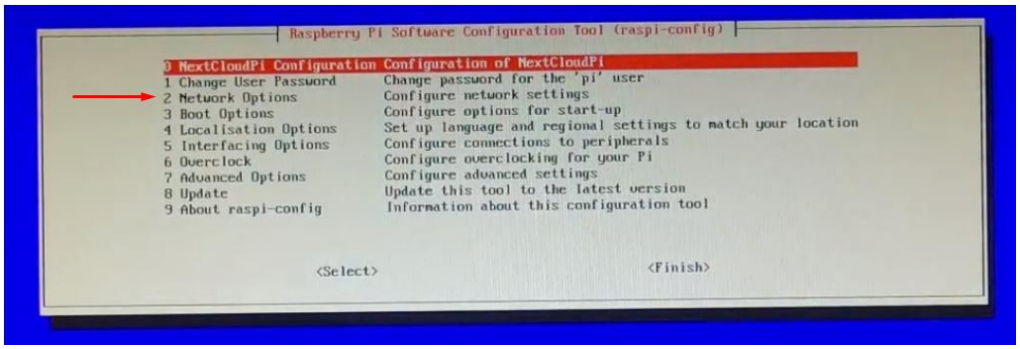

Рисунок 5.7 – Вікно raspi-config

Переходимо у пункт Network Options. Далі вибираємо підпункт Wi-Fi (рис. 5.8.) та обираємо нашу країну (рис. 5.9.). Потрібно це, щоб налаштувати передавач на одноплатному комп’ютері на необхідні частоти роботи, так як існують частоти заборонені для користування простими громадянами і в кожній країні ці частоти відрізняються.

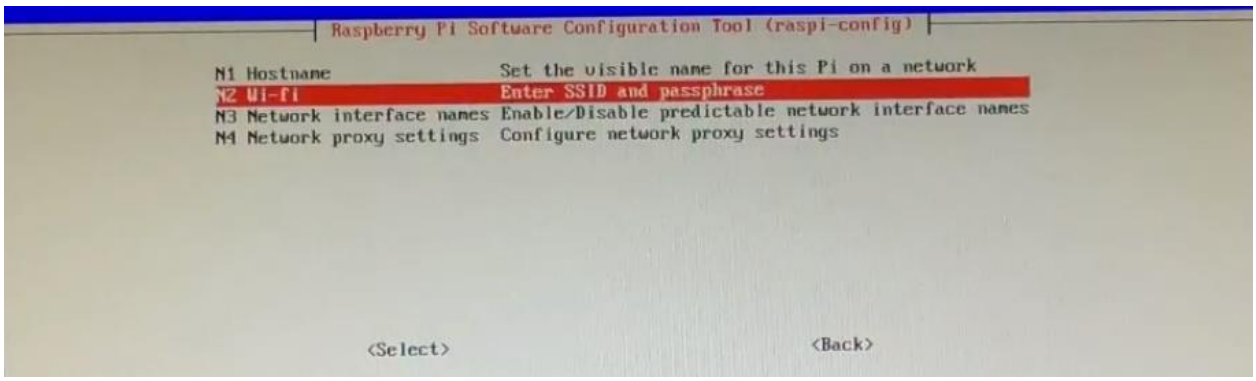

Рисунок 5.8 – Вибір налаштуваннь Wi-Fi

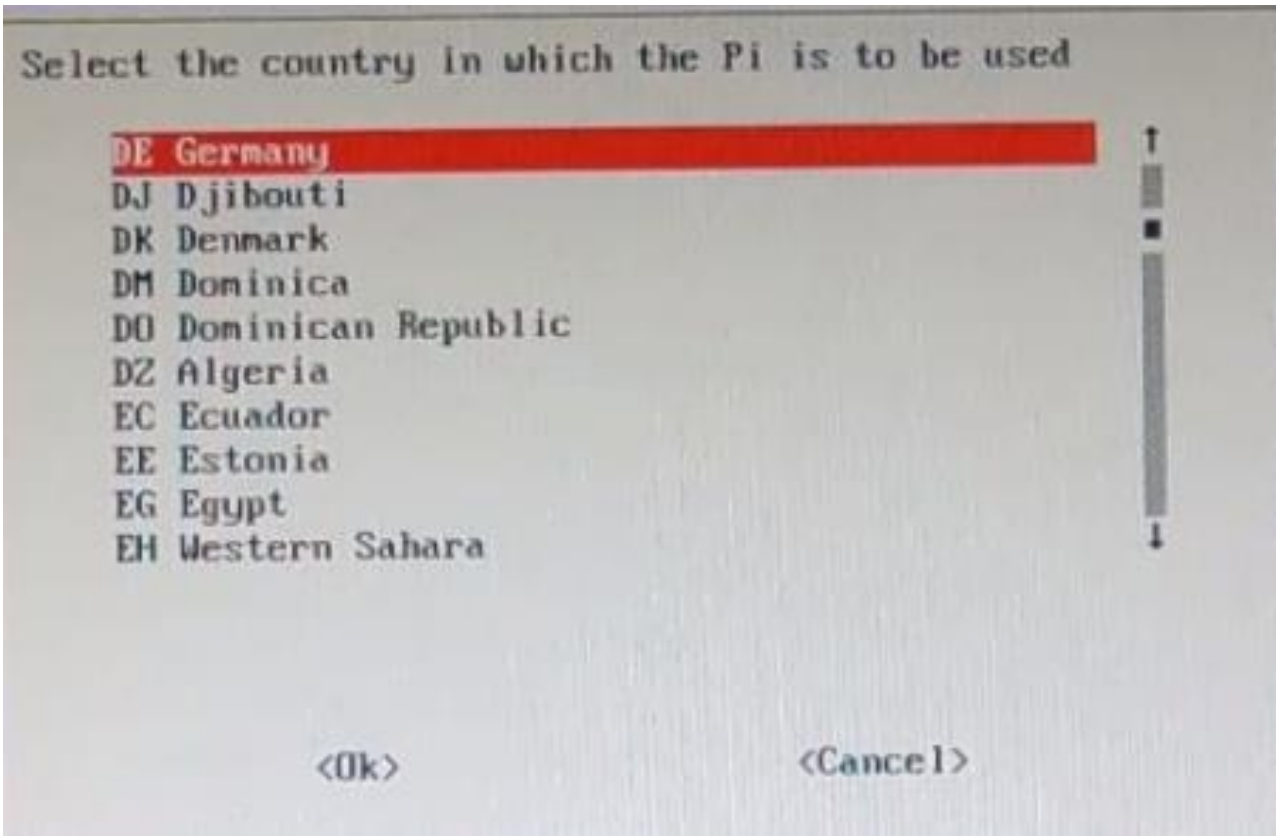

Рисунок 5.9 – Вибір країни

Далі потрібно налаштувати підключення до власної Wi-Fi мережі класичним методом, ввести назву власної мережі та ввести пароль від неї.

Тепер переходимо у пункт Interfacing Options, потім у пункт SSH де і вмикаємо дану функцію. Тепер ми можемо віддалено підключатися до нашого Raspberry Pi і нам більше не потрібен монітор, клавіатура.

SSH технологія. SSH, або Secure Shell – мережевий протокол, який дозволяє одному комп'ютеру безпечно підключатися до іншого комп'ютера через незахищену мережу інтернет, завдяки спільної згоди про те, як буде проводитися

обмін пакетами. SSH це протокол прикладного рівня, який є 7-м рівнем моделі OSI.

SSH корисний у нашій роботі, оскільки нам не потрібно мати фізичний доступ до керованого пристрою, тому можна просто підключитися до неї через інтернет, це дозволяє нам керувати сервером віддалено **[18]**.

SSH вперше з'явився в середині 90-х років і був розроблений як заміна Telnet, який також є протоколом прикладного рівня, що передає дані без шифрування. Без шифрування дані передаються через Інтернет у вигляді звичайного тексту.

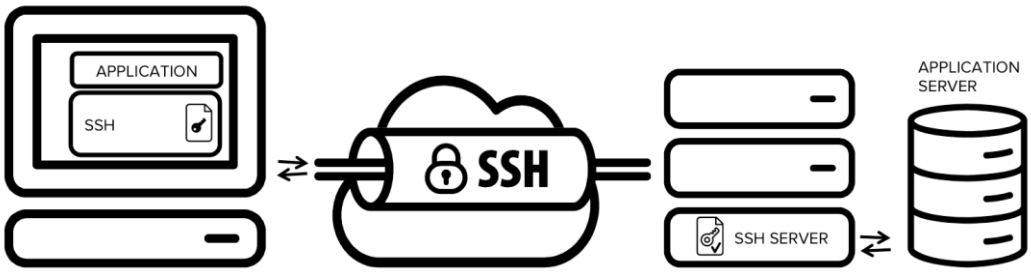

Рисунок 5.10 – SSH тунель

Шифрування – спосіб приховати частину даних так, щоб вони були незчитувані, якщо тільки ви не знаєте, як розшифрувати або декодувати дані. SSH був створений як безпечний спосіб зв'язку, який шифрує дані через тунель, щоб зловмисники не могли отримати дані під час передачі. При використанні SSH ви можете бачити, що дані передаються і скільки даних передається, але ви не можете бачити, що це за дані.

SSH зазвичай реалізується за моделлю клієнт-сервер. Один комп'ютер називається клієнтом SSH, а інший виступає в ролі сервера SSH або його ще називають хостом.

Коли ви підключаєтеся до сервера SSH, ви потрапляєте в оболонку. Ця оболонка може бути термінальної оболонкою Linux або командним рядком Windows, в якій ви можете виконувати команди на машині, до якої ви підключені. Коли ви використовуєте термінал або командний рядок, ви спілкуєтеся з операційною системою. За допомогою SSH ви можете спілкуватися і з віддаленими операційними системами.

Для того, щоб під’єднатися до нашого одноплатного комп’ютера за технологією SSH, потрібно дізнатися його IP-адресу, для супер-користувача ця процедура доволі легка, нам потрібно лише в терміналі ввести команду – **ip a**. Результат виконання команди зображений на рис. 5.11.

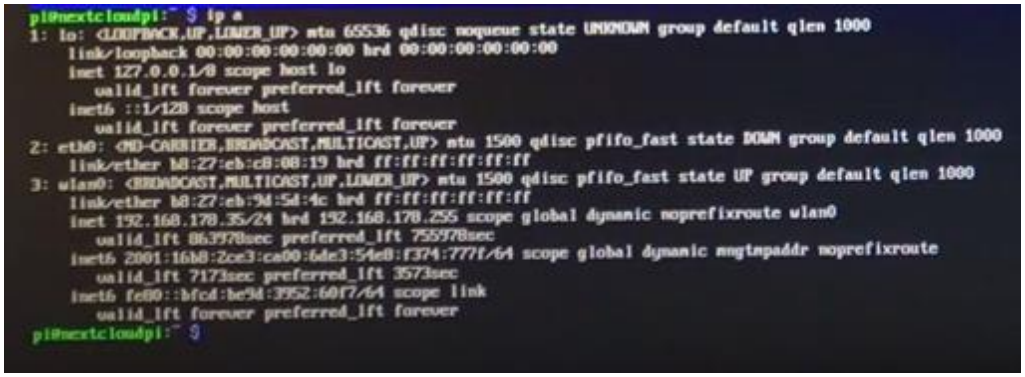

Рисунок 5.11 – Результат виконання команди ip a

Налаштування NextCloudPi. Вводимо в адресний рядок IP адресу, яку записали в попередньому пункті і переходимо налаштування NextCloudPi. Далі нам потрібно переписати паролі від конфігурації та самого NextCloud (рис. 5.12) та натискаємо активацію, після чого NextCloud робить необхідні налаштування.

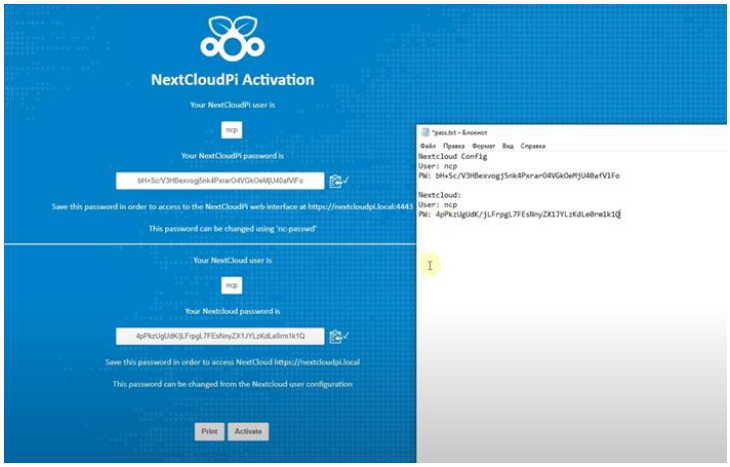

Рисунок 5.12 – Паролі від конфігурації та самого NextCloud

Процес авторизації, де ми вводимо записані логін і пароль від конфігурації, зображений на рис. 5.13.

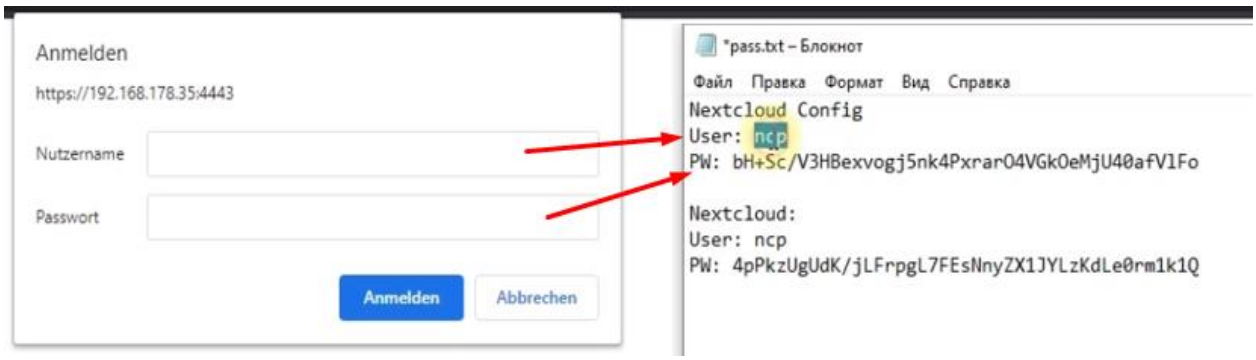

Рисунок 5.13 – Вхід в конфігурацію NextCloud

Це перший запуск інтерфейсу NextCloudPi, він запропонує провести початкові налаштування.

Перший пункт налаштування – USB configuration (рис. 5.14). NextCloud запропонує підключити флешку або жорсткий диск на якому будуть зберігатися дані користувача. При виборі пристрою для збереженні даних, система попередить, що всі дані з диску будуть втрачені у зв'язку з переформатування пристрою в потрібну файлову систему.

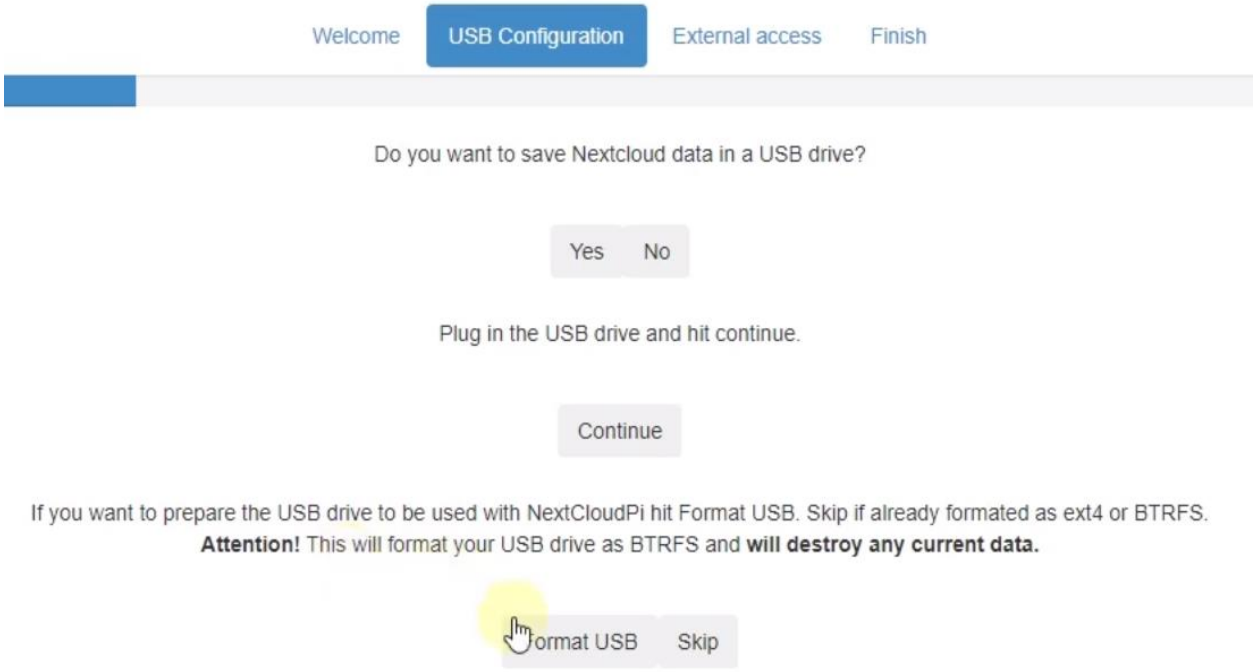

Рисунок 5.14 – Підготовка пристрою збереження даних

Другий пункт налаштування – External access (рис. 5.15). В цьому пункті налаштувань програма запитає, чи будемо використовувати домашнє сховище за

межами нашого будинку. В майбутньому вважаю потрібним залишити таку можливість. Пункті Port forwarding дає можливість обрати чи потрібно нам захищати весь потік інформації, який буде перенаправлятися на захищений порт по протоколу HTTP.

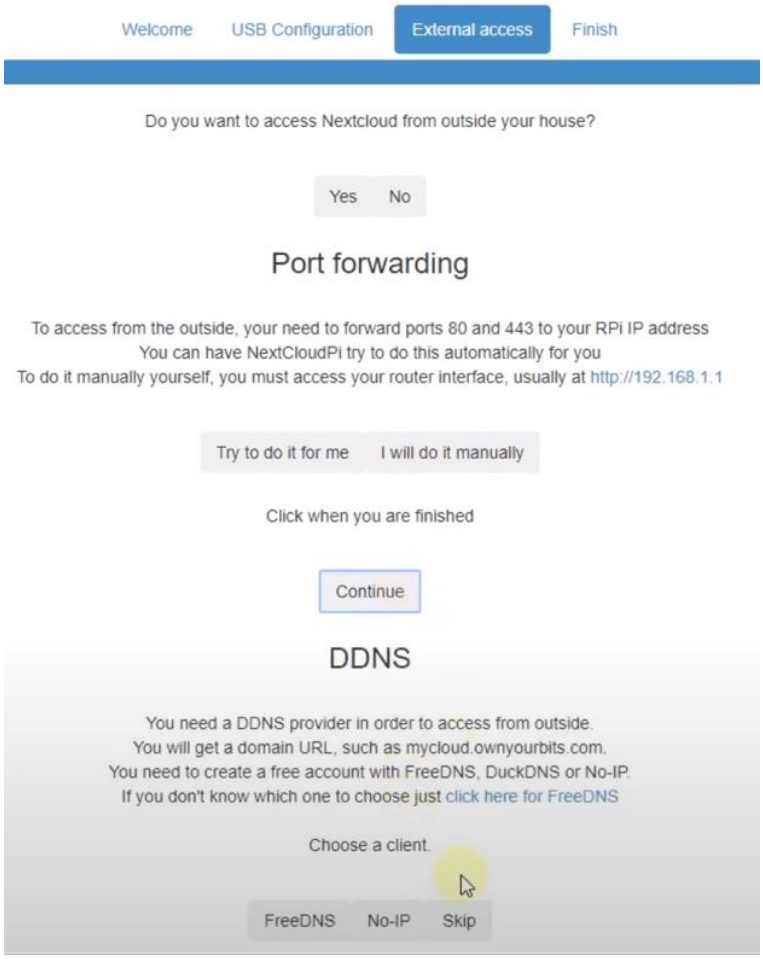

Рисунок 5.15 – Налаштування захисту даних

Протокол передачі гіпертексту (HTTP) – це метод кодування і передачі інформації між клієнтом (наприклад, веб-браузером) та веб-сервером. HTTP є основним протоколом для передачі інформації через інтернет. Обмін інформацією між клієнтами і серверами здійснюється у формі гіпертекстових документів, від яких HTTP і отримав свою назву. Гіпертекст – це структурований текст, в якому використовуються логічні зв'язки, або гіперпосилання, між вузлами, що містять текст. Гіпертекстовими документами можна маніпулювати за допомогою мови розмітки гіпертексту (HTML). Використовуючи HTTP і HTML, клієнти можуть запитувати різні види вмісту (наприклад, текст, зображення, відео і дані додатків) у веб-серверів і серверів додатків, на яких це

вміст розміщено. HTTP використовує парадигму "запит-відповідь", в якій клієнт робить запит, а сервер видає відповідь, що включає не тільки запитуваний вміст, а й відповідну інформацію про стан запиту. Така автономна конструкція дозволяє враховувати розподілену природу інтернету, де запит або відповідь може проходити через безліч проміжних маршрутизаторів і проксі-серверів. Вона також дозволяє проміжним серверам виконувати додаткові функції, такі як балансування навантаження, кешування, шифрування і стиснення **[19]**.

HTTP є протоколом прикладного рівня і для своєї роботи опирається на протокол мережевого рівня, такий як Transmission Control Protocol (TCP). Ресурси HTTP, такі як веб-сервери, ідентифікуються в інтернеті за допомогою унікальних ідентифікаторів, відомих як уніфіковані покажчики ресурсів (URL).

На цьому завершуємо налаштування нашого NAS, переходимо в налаштоване сховище, попередньо ввівши пароль і бачимо стандартні файли які вже зберігаються на системі хмарного сховища (рис. 5.16).

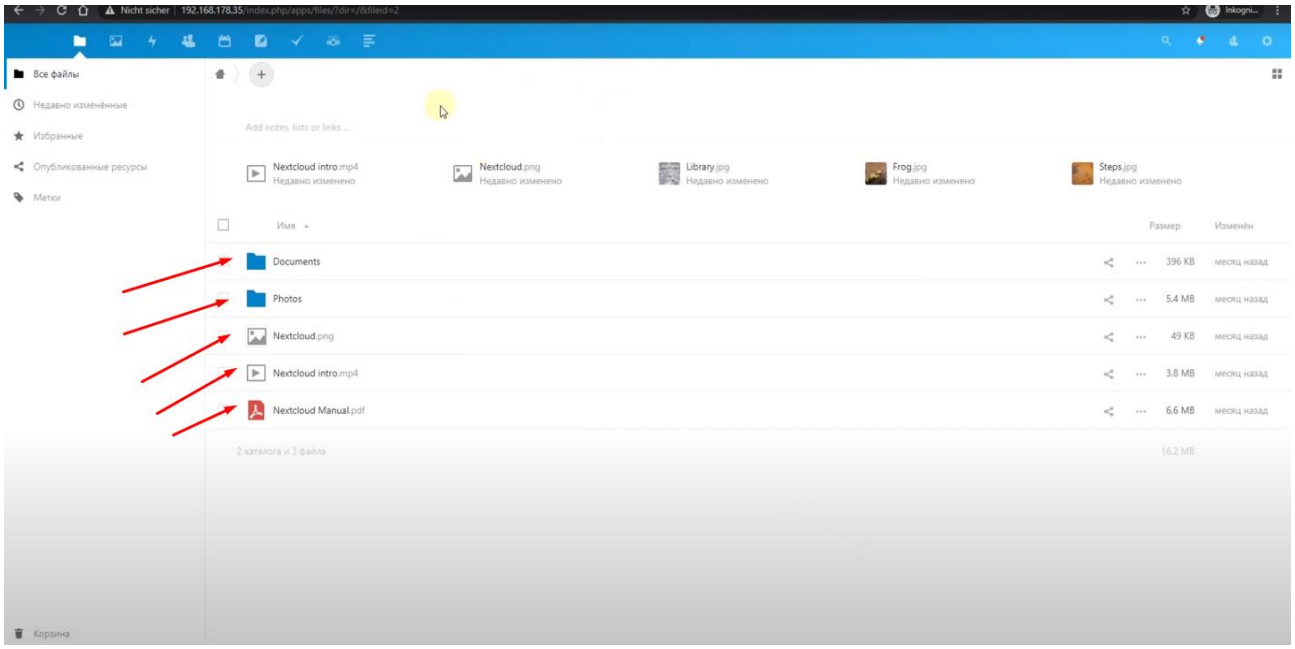

Рисунок 5.16 – Інтерфейс хмарного сховища

5.2 Розробка коду на Асемблері

Основна програма для нашого проекту написана на високорівневій мові програмування PHP. Але ми розглянемо приклад коду, для виводу текстової інформації на низькорівневій мові програмуванні assembler, спочатку на прикладі найпростішого коду, пізніше зі застосуванням циклів [20].

Наведений код, на рис. 5.17 виводить рядок " Entrance" на екран:

```
start:
        mov r7, #4
        mov r0, #1
        ldr r1, =string
        mov r2, #stringlen
        swi 0

        mov r7, #1
        swi 0

        .data
string:
        .ascii "Entrance\n"
        stringlen = . - string
```

Рисунок 5.17 – Код програми для виведення тексту

Перші два рядки коду не є інструкціями центрального процесора, а являються директивами асемблера і лінковщіка. Кожна програма повинна мати чітко задану точку входу під назвою «start», причому в нашому випадку вона виявилася в самому початку коду. Таким чином ми повідомляємо лінковщіку, що виконання коду має починатися з першої ж інструкції і ніяких додаткових дій не потрібно.

За допомогою наступної інструкції ми поміщаємо число 4 в регістр R7. Чіпи архітектури ARM надають розробникам велику кількість регістрів загального призначення, ми можемо використовувати до 16 регістрів з іменами від R0 до R15.

Хоча інструкція «mov» і дуже схожа на однойменну інструкцію архітектури x86, необхідно звернути увагу на символ решітки поруч з числом 4, який вказує на те, що далі розташоване цілочисельне значення, а не адреса в пам'яті. В даному випадку ми бажаємо використовувати системний виклик «write» ядра Linux для виведення нашого рядки. Для використання системних викликів слід заповнювати регістри необхідними значеннями перед тим, застосовувати ядро для виконання роботи. Номер системного виклику повинен поміщатися в регістр R7, причому число 4 є номером системного виклику «write».

За допомогою наступної інструкції «mov» ми поміщаємо дескриптор файлу, в який має бути записаний рядок "Entrance". Тобто, дескриптор стандартного потоку виводу, в регістр R0. Так як в даному випадку використовується потік стандартного виведення, в регістр поміщається його стандартний дескриптор, тобто 1. Далі нам потрібно помістити адресу рядка, яку ми хочемо вивести, в регістр R1 за допомогою інструкції «ldr» (інструкція "завантаження в регістр"). В кінці коду, а саме в секції даних ми оголошуємо цей рядок в формі послідовності символів ASCII. Для успішного використання системного виклику «write» нам також доведеться повідомити ядру операційної стеми про те, яка довжина рядка, що виводиться, тому ми поміщаємо значення «stringlen» в регістр R2. Значення stringlen розраховується шляхом вирахування адреси закінчення рядки з адреси її початку.

На даний момент ми заповнили всі регістри необхідними даними і готові до передачі управління ядру Linux. Для цього ми використовуємо інструкцію «swi», назва якої розшифровується як «software interrupt», яка здійснює перехід в простір ядра ОС. Ядро ОС досліджує вміст регістра R7, виявляє в ньому цілочисельне значення 4 і робить висновок, що записаний код намагається вивести рядок. Після цього воно досліджує вміст інших регістрів, здійснює вивід рядка і повертає управління нашій програмі.

Таким чином ми бачимо на екрані рядок "Entrance", Після чого нам залишається лише коректно завершити виконання програми. Ми вирішуємо цю задачу шляхом периміщення номера системного виклику «exit» в регістр R7 з подальшим викликом інструкції програмного переривання – 0. І на цьому все - ядро ОС завершує виконання нашої програми і ми знову потрапляємо у командну оболонку.

Тепер приведемо приклад складнішої програми, вона повинна відслідковувати натискання користувача на клавіатурі та відповідно реагувати. Приклад коду зображений на рис. 5.18.

Наша програма починається з периміщення покажчика на початок рядка і занесення значення її довжини у відповідні регістри для подальшого здійснення системного виклику «write», причому відразу ж після цього здійснюється перехід до підпрограми «print_string», розташованої нижче в коді.

```
start:
        ldr r1, =string1
        mov r2, #string1len
        bl print_string
loop:
        mov r7, #3      @ read
        mov r0, #0      @ stdin
        ldr r1, =char
        mov r2, #2      @ два символи
        swi 0

        ldr r1, =char
        ldrb r2, [r1]
        cmp r2, #113    @ Код ASCII символа 'q'
        beq done

        ldr r1, =string2
        mov r2, #string2len
        bl print_string
        b loop

done:
        mov r7, #1
        swi 0

print_string:
        mov r7, #4
        mov r0, #1
        swi 0
        bx lr

        .data
string1:
        .ascii "Enter q to quit!\n"
        string1len = . - string1
string2:
        .ascii "Wrong button\n"
        string2len = . - string2
char:
        .word 0
```

Рисунок 5.18 – Код програми для виведення тексту

Для здійснення цього переходу використовується інструкція «bl», назва якої розшифровується як «branch and link», причому сама вона зберігає поточну адресу в коді, що дозволяє повернутися до неї згодом за допомогою інструкції «bx». Підпрограма «print_string» просто заповнює інші регістри для здійснення системного виклику «write» таким же чином, як і в нашій першій програмі перед

переходом в простір ядра ОС з наступним поверненням до збереженого адресою коду за допомогою інструкції «bx.»

Повернувшись до здійснення виклику коду, ми можемо виявити мітку під назвою «loop» – назва мітки вже натякає на те, що ми повернемося до неї через деякий час. Але спочатку ми використовуємо ще один системний виклик з ім'ям «read» (під номером 3) для читання символу, введеного користувачем за допомогою клавіатури. Тому ми поміщаємо значення 3 в регістр R7 і значення 0 (дескриптор стандартного потоку введення) в регістр R0, так як нам потрібно прочитати для користувача, те що ми ввели, а не дані з файлу.

Далі ми розміщуємо адресу, за якою ми хочемо зберегти символ, прочитаний і поміщений ядром ОС в регістр R1 – в нашому випадку це область пам'яті «char», описана в кінці секції даних.

Повернувшись до основного коду, ми побачимо, що в регістр R2 поміщається значення 2, що відповідає двом символам, які ми хочемо зберегти, після чого здійснюється перехід в простір ядра ОС для виконання операції читання. Користувач вводить символ і натискає клавішу Enter. Тепер нам потрібно перевірити, що це за символ. Ми поміщаємо адресу області пам'яті («char» в секції даних) в регістр R1, після чого за допомогою інструкції «ldrb» завантажуємо байт з області пам'яті, на яку вказує значення з цього регістра.

Квадратні дужки в даному випадку вказують на те, що дані зберігаються в цікавій для нас області пам'яті, а не в самому регістрі. Таким чином, регістр R2 тепер містить єдиний символ з області пам'яті «char» з секції даних, причому це саме той символ, який ввів користувач. Наше наступне завдання буде полягати в порівнянні вмісту регістра R2 з символом "q", який є 113 символом таблиці ASCII. Тепер ми використовуємо інструкцію «cmp» для виконання операції порівняння, після чого використовуємо інструкцію «beq», ім'я якої розшифровується як "branch if equal", для переходу до мітки done в тому випадку, якщо значення з регістра R2 рівне 113. Якщо це не так, то ми виводимо наший другий рядок, після чого здійснюємо перехід до початку циклу за допомогою інструкції b.

Нарешті, після мітки done ми повідомляємо ядру ОС про те, що ми хочемо завершити виконання програми, точно так же, як і в першій програмі.

ВИСНОВКИ

У кваліфікаційній роботі було розроблено власне мережеве сховище даних на базі одноплатного комп'ютера Raspberry Pi 3 B+. Воно зберігає в собі дані користувача та забезпечує доступ до них з будь-якої точки світу, де є підключення до мережі інтернет. Основні потреби, які покриває розроблена система:

– завантаження файлів через мережу;
– доступ до завантажених файлів через мережу;
– редагування файлів в хмарі;
– захищений доступ до даних.

Пристрій відповідає технічному завданню.

У першому розділі, було розглянуто тарифи найпоширеніших постачальників хмарних сервісів та готові рішення NAS. На основі розглянутої інформації було сформульовано та поставлено завдання на проектування пристрою.

У другому розділі був розроблений алгоритм роботи мережевого сховища даних. За цим алгоритмом в подальшому прийшли до структурної схеми. Після створення структурної схеми, на її основі була зроблена функціональна схема.

У третьому розділі був проведений вибір елементної бази для реалізації проекту. Сформований список елементів, за яким обиралися оптимальні частини, які відповідали технічному завданню.

У четвертому розділі була наведена методика налаштування NAS на одноплатному комп'ютері та перхід від терміналу linux до зручного графічного інтерфейсу.

СПИСОК ЛІТЕРАТУРИ

1. Кузнецов А. Тест: какое облачное хранилище самой быстрое. [Электронный ресурс]//iguides.ru:[сайт].[2018] https://www.iguides.ru/main/other/test_kakoe_oblachnoe_khranilishche_samoe_bystroe/.
2. https://www.microsoft.com/uk-ua/microsoft-365/onedrive/compare-onedrive-plans?activetab=tab:primaryr1
3. https://one.google.com/plans.
4. https://mega.io/pro
5. My Cloud™ Home Duo Personal Cloud Storage - Product Overview
6. Synology_DS220_Plus_Data_Sheet.
7. Щепетов А.Г., Основы проектирования приборов и систем; М.l Академия, 2016.
8. Погуляй О.Р. Пристрій реєстрації аналогових сигналів / Бережна О.В., Щокотова І.В., Романенко Є.С., Гагіна О.М., Погуляй О.Р. // Фізика, електроніка, електротехніка (ФЕЕ-2021). Матеріали та програма науково-технічної конференції. – Суми: СумДУ, 2021. - С.90.
9. https://elinux.org/RPi_SD_cards
10. ASUS Tinker Board development board Rockchip RK3288
11. Погуляй О.Р. Особливості раманівських спектрів полікристалічних плівок Cd1-xZnxTe, отриманих методом вакуумного термічного випаровування в квазізамкненому об’ємі. / Д’яченко О.В., Борисенко, Іващенко М.М., Опанасюк А.С. // Фізика, електроніка, електротехніка (ФЕЕ-2019). Матеріали та програма науково-технічної конференції. – Суми: СумДУ, 2019. - С.°90.
12. Погуляй О.Р. Оптимізація конструкції оптично-прозорих сонячних елементів на основі гетеропереходу n-Zn1-xMgxO / p-Cu2O. / Д’яченко О.В., Іващенко М.М., Опанасюк А.С. // Фізика, електроніка, електротехніка (ФЕЕ-2020). Матеріали та програма науково-технічної конференції. – Суми: СумДУ, 2020. - С.°90.
13. Погуляй О.Р. Коди Фібоначчі в системах обробки інформації. / Борисенко О.А., Бережна О.В., Романенко Є.С., Гагіна О.М., Погуляй О.Р. // Фізика, електроніка, електротехніка (ФЕЕ-2021). Матеріали та програма науково-технічної конференції. – Суми: СумДУ, 2021. - С.90.

14. Орлов П. А. Сравнительный анализ эффективности использования современных облачных хранилищ // Молодой ученый. — 2017.
15. https://ownyourbits.com/nextcloudpi/s
16. https://rufus.ie/ru/
17. Глибин Е.С., Прядилов А.В., Программирование электронных устройств; Тольятти: Изд-во ТГУ,2014.
18. https://levelup.gitconnected.com/what-is-ssh-103f89e3e4b8
19. https://seopressor.com/blog/http-vs-https/
20. http://rus-linux.net/MyLDP/algol/asm/asmschool_arm_assembly_language.html