ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерія програмного забезпечення

Пояснювальна записка

до бакалаврської роботи

на ступінь вищої освіти бакалавр

на тему: **«РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ УПРАВЛІННЯ ІОТ ПРИСТРОЯМИ З ВИКОРИСТАННЯМ МІКРОКОНТРОЛЕРА ЗА ДОПОМОГОЮ МОВИ ПРОГРАМУВАННЯ PYTHON»**

Виконав: студент 4 курсу, групи ПД-44

спеціальності

121 Інженерії програмного забезпечення

(шифр і назва спеціальності)

Гуленко В.С.

(прізвище та ініціали)

Керівник Жебка В.В.

(прізвище та ініціали)

Рецензент_____

(прізвище та ініціали)

Київ – 2021

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерія програмного забезпечення

Ступінь вищої освіти - «Бакалавр»

Спеціальність – 121 Інженерії програмного забезпечення

ЗАТВЕРДЖУЮ

Завідувач кафедри ІПЗ

_____ О.В.Негоденко

“____” _____2021 року

З А В Д А Н Н Я

НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Гуленко Володимира Сергійовича

(прізвище, ім’я, по батькові)

1.Тема роботи: «Розробка програмного забезпечення для управління IoT пристроями з використанням мікроконтролера за допомогою мови програмування Python»

керівник роботи Жебка В.В., к.т.н, доц.

(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від “12” березня 2021року №65

2. Строк подання студентом роботи 01.06.2021 року

3. Вихідні дані до роботи:

1). Архітектура технології IoT

2). Вимоги до технічних характеристик сенсорів, датчикі та мікроконтролера IoT

3). Наукова література. Міжнародні стандарти.

4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити):

1. Дослідити концепцію технології Інтернет Речей (IoT)

2. Дослідити апаратне та програмне забезпечення технології Інтернет Речей (IoT)

3. Розробити програмне забезпечення для управління IoT пристроям використанням мікроконтролера за допомогою мови програмування Python

5. Перелік графічного матеріалу

1) Об'єкт, предмет та мета дослідження

2) Загальна концепція технології IoT в реалізації «розумний» будинок

3) Програмне та апаратне забезпечення IoT

4) Дослідження існуючих алгоритмів для управління додатками та каналами зв'язку в IoT

5) Розробка програмного забезпечення для управління IoT пристроями з використанням мікроконтролера за допомогою програмування Python

6) Розробка моделі управління пристроями IoT з використанням мікроконтролера та комбінації Blockly та Python

7) Висновки

6. Дата видачі завдання 19.04.2021 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської Роботи	Строк виконання етапів роботи	Примітка
1	Аналіз науково-технічної літератури	19.04.2021 р.	виконано
2	Особливості функціонування архітектури IoT	23.04.2021 р.	виконано
3	Дослідження програмного та апаратного забезпечення технології IoT	28.04.2021 р.	виконано

4	Дослідження існуючих алгоритмів для управління додатками та каналами зв’язку в IoT	10.05.2021 р.	виконано
5	Розробка програмного забезпечення для управління IoT пристроями з використанням мікроконтролера за допомогою мови Python	25.05.2021 р.	виконано
6	Реферат, вступ, висновки.	28.05.2021 р.	виконано
7	Підготовка презентації до захисту.	01.06.2021р.	виконано

Студент _____ Гуленко В.С.

(підпис) (прізвище та ініціали)

Керівник бакалаврської роботи _____ Жебка В.В.

(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина бакалаврської роботи 50 с., 47 рис., 33 джерела.

Об`єкт дослідження – технологія Інтернет Речей (IoT).

Предмет дослідження – управління апаратним та програмним забезпеченням IoT пристроїв.

Мета роботи – розробка програмне забезпечення для управління IoT пристроями з використанням мікроконтролера.

Методи дослідження – методи та алгоритми складних систем, методи системного аналізу, засоби моделювання, алгоритмічного і програмного забезпечення задач аналізу та синтезу складних розподілених у просторі гнучких комп'ютерно- інтегрованих систем.

Досліджено основні особливості побудови та функціонування архітектури технології Інтернет Речей (IoT) та підкреслено необхідність широкого дослідження особливостей даної технології. Проаналізовано особливості створення та розповсюдження технології IoT. Описано принцип роботи системи «Розумний» будинок при умові впровадження мікроконтролера Arduino, а також приведено головні переваги даного мікроконтролера.

Приведено особливості впровадження та використання базових датчикі та сенсорів, та підкреслено, що їх призначення полягає у реагуванні на вхідні фізичні властивості та перетворенні його в електричний сигнал, сумісний з електронними схемами. Зазначено, що система керується мікроконтролером, що в більшості випадків виконує функцію центрального процесора, який з'єднаний з портативним блоком плат.

Описано різні типові для «розумного» будинку алгоритми, які необхідні для налаштування головних датчиків та сенсорів. Дотримуючись алгоритмів користувач може здійснювати управління кількома побутовими електроприладами одночасно. Однак кількість контрольованих приладів буде залежати від кількості вихідних портів центрального або головного мікроконтролера.

Досліджено мову візуального програмування Blockly, яка дозволяє користувачам створювати програми, об'єднуючи блоки, які представляють різні

логічні структури мови, а не написанням фактичного коду. Blockly працює в веб-браузері та може відображати візуально створену програму в Python.

Розроблено модель управління пристроями IoT з використанням мікроконтролера та комбінації мов Blockly та Python.

Галузь використання – комп'ютеризовані системи управління технологічними процесами.

ІОТБ МІКРОКОНТРОЛЕР, ARDUINO, PYTHON, РОЗУМНИЙ БУДИНОК, БЕЗПРОВОДОВИЙ ЗВ'ЯЗОК, СТРУКТУРА, МЕРЕЖА, ПЕРЕДАЧА ДАНИХ, МОДЕЛЮВАННЯ, СИГНАЛ, СИСТЕМА УПРАВЛІННЯ.

ЗМІСТ

ВСТУП

Інформаційні технології настільки швидко розвиваються, що появою нових технологій, додатків чи систем вже нікого не вдасться здивувати. Однієї із найбільш перспективних технологій сьогодення є технологія Інтернет Речей (IoT).

Ця технологія швидко розвивається та використовується для забезпечення комфорту, зручності, якості життя та безпеки для мешканців. Нині більшість її елементів використовуються для зменшення людської праці. Як наслідок, вона може бути спроектована та розроблена за допомогою одного мікроконтролера, який має можливість керувати та контролювати різні пов'язані між собою IoT пристрої, такі як світильники, датчики температури та вологості, димові, газові та пожежні сповіщувачі, а також аварійні та охоронні системи.

Одна з найбільших переваг впровадження IoT полягає в тому, що можна легко керувати та управляти масивом пристроїв, таких як смартфон, планшет, настільний комп'ютер та ноутбук. На сьогодні практично будь-який IoT пристрій може працювати з будь-якою платформою завдяки синхронізації через віддалені сервіси, або поєднуючи в собі базові функції всіх інших, просто підключаючись до домашньої мережі.

Інтернет речей має величезний вплив як на робочі, так і на побутові сфери. Згідно з дослідженнями IT ринку, до кінця 2021 року кількість «розумних» пристроїв в мережах перевищить 30 мільярдів пристроїв. Це означає, що IoT буде проникати в усі аспекти повсякденного життя людини. Отже, найближчим часом Інтернет речей різко вплине на те, як живе та працює звичайний користувач. Це також відкриє величезні можливості для економіки та приватних осіб. Тому постійно розроблюються та пропонуються все нові системи автоматизації «Розумних» будинків, що здатні здійснювати дистанційне управління системою моніторингу та контролю за рівнем температури, вологості та газу. Також такі системи можуть контролювати світло за допомогою датчиків руху та світла всередині та зовні будинку.

Це дослідження має на меті запропонувати архітектуру домашньої автоматизованої мережі, яка об’єднує декілька IoT-пристроїв, які не лише поєднанні між собою за допомогою мікроконтролера, але і здатні бути запрограмовані на виконання певних команд та завдань за допомогою мови програмування Python.

Поряд з подібним управлінням «розумними» приладами, концепція IoT базується на системі управління споживанням енергії, завдяки якій споживачі можуть зменшити надмірне споживання енергії, віддалено керуючи пристроями. Це може заощадити надмірне використання будь-якого приладу енергії, часу та одночасно зменшити додаткові витрати багатства.

Об`єкт дослідження – технологія Інтернет Речей (IoT).

Предмет дослідження – управління апаратним та програмним забезпеченням IoT пристроїв.

Мета роботи – розробка програмне забезпечення для управління IoT пристроями з використанням мікроконтролера.

Методи дослідження – методи та алгоритми складних систем, методи системного аналізу, засоби моделювання, алгоритмічного і програмного забезпечення задач аналізу та синтезу складних розподілених у просторі гнучких комп’ютерно- інтегрованих систем.

1 ОСНОВНА КОНЦЕПЦІЯ ТЕХНОЛОГІЇ ІНТЕРНЕТ РЕЧЕЙ (IOT)

Взаємозв'язок між пристроями, підключеними до мережі Інтернет, додавання функції машинного навчання та використання безпроводових датчиків, які дистанційно управляються, контролюються та оптимізуються користувачем, називається технологією Інтернет речей (IoT). Термін «речі» означає фізичні пристрої, такі як мікросхеми, мікроконтролери, камери, датчики та ін. Такі пристрої відповідають за зв'язок, збір та обробку інформації, обмін даними шляхом підключення до мережі. Вбудована технологія цих фізичних пристроїв робить можливим такий обмін інформацією.

Існує цілий набір пристроїв, який може допомогти зробити життя вдома вигіднішим та простішим. Наприклад, користувач здатен дистанційно керувати приладами, якими зазвичай користується щодня. За допомогою домашньої системи комп'ютеризації користувач може здійснювати управління побутовою технікою, коли знаходиться поза домом, і не запам'ятовувати, чи вимкнув світло чи ні, «розумна» система зможете дати відповідь на це питання.

Розвиток технології Інтернет речей та глибоке впровадження її у повсякдене життя людини дозволяє пов'язувати та синхронізовувати різні мережеві клієнтські пристрої за рахунок використання програмного забезпечення до мережі Інтернет Серед таких пристроїв розглядаються: настільні ПК, ноутбуки, смартфони, планшети тощо [1].

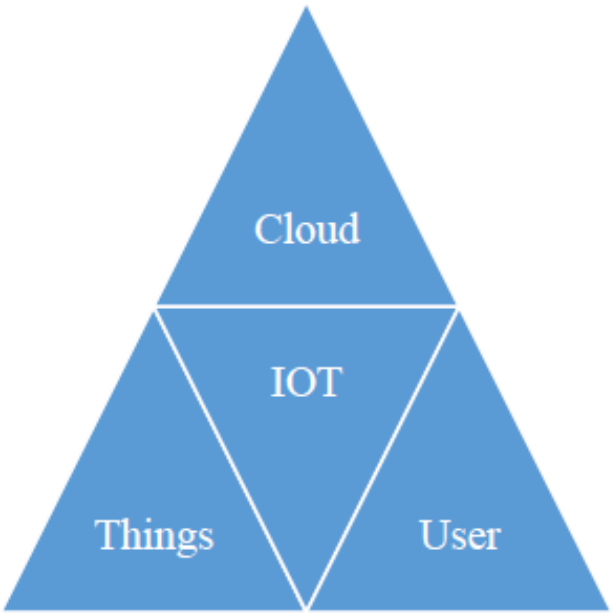

Рисунок 1.1 – Основна концепція IoT

1.1 Характеристики IoT пристроїв

Інтернет речей (IoT) поєднує в собі мережу різних пристроїв, таких як побутова техніка, офісні девайси, яка містить програмне забезпечення, електроніку, датчики та підключення, що дозволяє їм обмінюватися даними за допомогою проводових та безпроводових з'єднань. Кілька датчиків та виконавчих механізмів використовуються для підключення пристроїв та надання зв'язку для самостійних операцій.

Характеристика технології IoT включає синтез апаратного та програмного забезпечення, суворих вимог щодо дотримання виконання складних алгоритмів, завдяки яким може функціонувати внутрішній штучний інтелект в пристроях, що дозволяє їм поводитися та діяти відповідно до конкретних ситуацій, тобто навчатися, підбирати конкретний алгоритм, та виконувати поставлене завдання.

Завдяки таким можливостям можна підключати різні пристрої за технологією IoT та можна організовувати підключення різних об'єктів, створювати мережі та комплексні системи. Динамічна природа пристроїв IoT визначає стан пристрою, увімкнений він чи вимкнений. Пристрій IoT також збирає дані про динамічні зміни

із сусіднього середовища, здійснює первинну обробку, та передає до серверів чи виокремлених баз даних.

Суть IoT полягає у різноманітності та неоднорідності завдяки використанню різних платформ, мереж та мов програмування. Нарешті, питання безпеки IoT є дуже важливими через конфіденційну інформацію, яка передається в мережі. У майбутньому кількість компонентів IoT мережі збільшиться настільки, що виникнуть все нові завдання щодо обробки та передачі масивів даних.

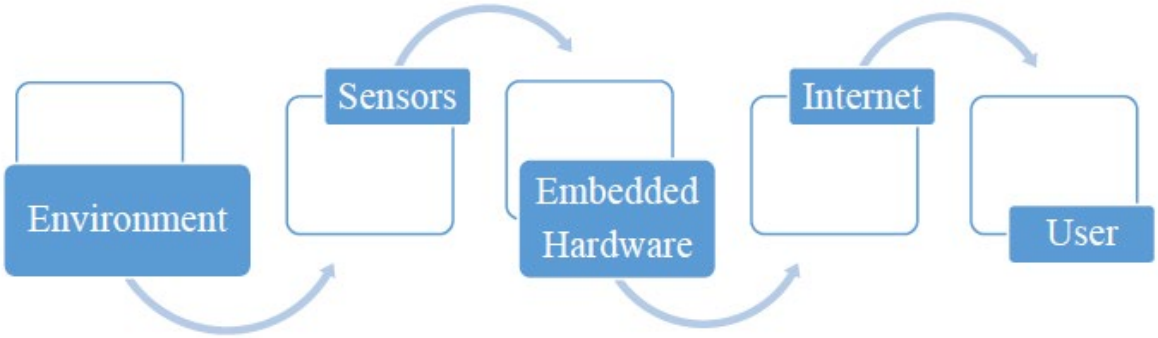

Рисунок 1.2 – Послідовність роботи IoT пристроїв

В останні роки IoT відіграє важливу роль у зменшенні використання людських ресурсів. Швидке зростання пристроїв IoT та додатків з кожним днем збільшується, що дисциплінується використанням мережі Інтернет. Сфера IoT базується на поєднанні фізичного світу з комп'ютерною базою шляхом посилення ефективності технологій, зменшення людських зусиль та збільшенням економічних вигод.

Основні характеристики пристроїв IoT дуже подібні, однак технологія, що лежить в основі кожного пристрою, визначається від одного пристрою до іншого. Завдяки атмосфері інтелекту та зрозумілого контролю робить IoT сьогодні більш популярним. Найближчим часом пристроїв IoT стануть більш розумними, автоматизованими, систематизованими, здатними до незалежної роботи та здатними передбачати умови та атмосферу заздалегідь.

Оскільки телекомунікаційний сектор стає все більш ефективним та розгалуженим, безпроводовий та широкосмуговий Інтернет-зв’язок зараз широко доступний. З появою більш досконалої технології комунікації тепер набагато дешевше виготовляти «розумні» пристрої та датчики з вбудованими можливостями підключенням до каналів Wi-Fi, що робить підключення пристроїв менш витратними [2]. Найголовніше, що використання смартфонів збільшено до такого рівня, що його сьогодні використовують у всіх можливих аспектах людського життя.

Що стосується систем, заснованих на IoT, немає необхідності в окремій системі зв'язку, і можна використовувати чинні технології за допомогою смартфонів, що робить систему дешевшою та досяжнішою для звичайного користувача.

Завдяки цьому зараз кожен може побудувати повністю автоматизовану систему «розумний» будинок або ціле «розумне» місто з постійним та безвідмовним моніторингом споживання енергії або системою моніторингу дорожнього руху для підвищення ефективності його функціонування.

1.2 Архітектура IoT

Основна архітектура IoT складається з декількох етапів, та включає пристрій підключення до шлюзу, обробку даних та хмарний або інтерфейс користувача. По-перше, фізичні пристрої, такі як датчики, прилади, пристрої та виконавчі механізми, збирають вихідні дані із сусіднього середовища та перетворюють їх у корисні дані. Сенсори та датчики працюють як перетворювачі, які перетворюють енергію із однієї (аналогової) форми в іншу форму (цифрову) і використовуються в архітектурі IoT.

Сенсори перетворюють енергію в рух, крім датчика, який є пристроєм, який приймає і реагує на сигнал. Датчики використовують специфічний протокол, такий як Modbus, ZigBee, Bluetooth, NFC, Wi-Fi або фірмовий протокол для підключення Edge-шлюзу.

Edge шлюз перетворює необроблені дані з аналогових у цифрові за допомогою системи збору даних, крім агрегування даних. Інтернет-шлюз отримав зведені дані як приклад попередньої обробки і забезпечує маршрутизацію, що затримує підключення до хмари, використовуючи систему, наприклад, веб-сокети, концентратор подій, аналіз кінцевого обладнання, протокол AMQP, протокол телеметрії MQTT, обмежений протокол програми (COAP) або обчислення хмари.

Далі проводиться детальний аналіз даних та обробка IT-системами (на місці) чи за межами мережі. Нарешті, дані передаються для подальшого опрацювання до БД або до хмари. Хмарний додаток обробляє всі запити, які надходять на кожному із етапів [3]. Приклад подібної послідовності опрацювання даних приведено на рис.1.3.

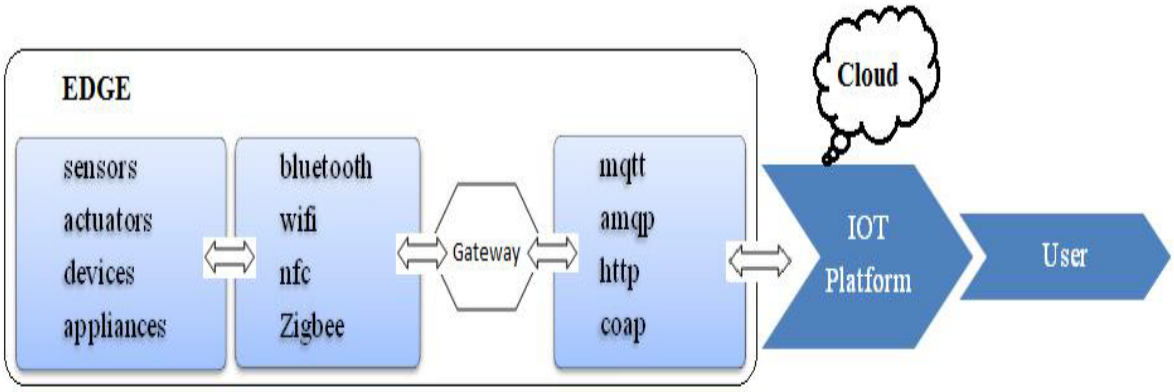

Рисунок 1.3 – Архітектура IoT

1.3 Особливості функціонування архітектури IoT

Технологія IoT змінює життя людства та робить його легким, комфортним та простим. Сфера, яку охоплює IoT, величезна. IoT має широке застосування в різних секторах (рис.1.4), наприклад, у комерції, промисловості, медицині та галузі споживання [4]. У всіх цих випадках IoT залишає свій слід: «розумний» будинок, «розумне» місто (паркування, управління процесом прибирання та утилізацією відходів), комунальні послуги (інтелектуальна мережа, «розумне» вимірювання використанних ресурсів), переносні гаджети, транспорт та логістика (підключення

трекерів до машин, управління автопарком, відстеження товарів), промислові (моніторинг та контроль процесів, виробництво, обслуговування), сільське господарство (моніторинг сільського господарства, клімат, відстеження стану худоби), телемедицина та охорона здоров'я, догляд за літніми людьми та за навколишнім середовищем (моніторинг навколишнього середовища) [5].

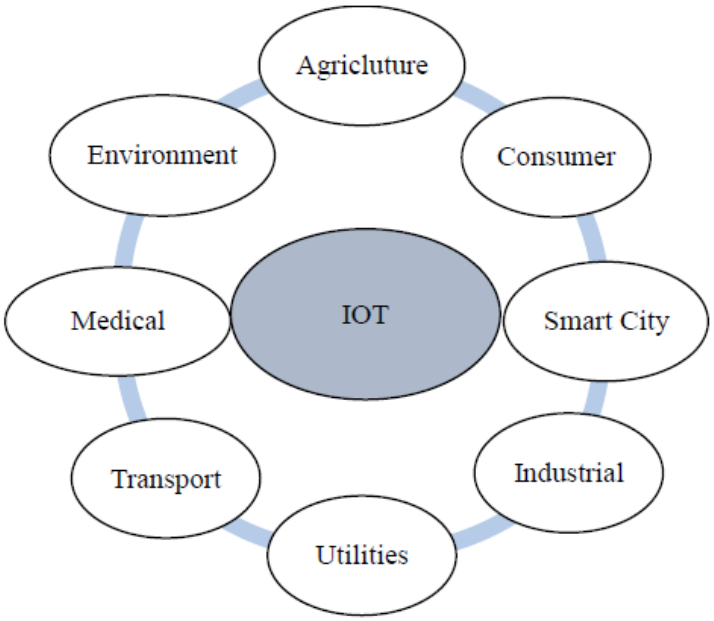

Рисунок 1.4 – Програми IoT

Поєднання декількох IoT пристроїв в «Розумний» будинок серед них є найбільш досліджуваним сектором, який корисутвачі намагаються модернізувати щодня.

«Розумний» будинок – це автоматизований будинок, який може автоматично обслуговуватися, або ж керуватися вручну користувачем. Складається з такої кількості актуальних компонентів, як: освітлення, опалення, кондиціонування повітря, термостат, безпроводова колонка, безпека та моніторинг, детектор диму, детектор води, пральна машина та холодильник (рис.1.5).

Датчики та безпроводові сенсорні мережі використовуються в рішеннях для «Розумних» будинків, вимірюючи з точністю умови навколишнього середовища. Їх вдосконалені функціональні можливості зондування та зростальна точність дозволяють розробляти додатки, які пропонують вдосконалену автоматизацію.

Звичайні резиденції перетворюються на цілковитий «Розумний» будинок, що включає вбудовані датчики та виконавчі механізми [6]. Автоматизація процесів включає централізоване управління освітленням, кондиціонуванням, вентиляцією та опаленням, а також управління «розумними» пристроями. Також спрямована на підвищення ефективності споживання енергії та безпеки у побутових сценаріях. Будинки оснащені автономними пультами управління всіма системами та пристроями, наявними у приміщенні.

Основна мета «Розумного» будинку - зосередити управління всіма пристроями в блок управління, який може бути запрограмований на виконання спеціальних завдань, що підходять для власника та відповідного будинку. Мета «Розумного» будинку - зменшення споживання таких ресурсів, як електроенергія, газ тощо. Завдяки сучасним тенденціям щодо ціноутворення на енергію, енергозбереження стало частиною повсякденного життя людини. Якщо людина має можливість контролювати свою домашню автоматизацію, вона може віддалено зменшити споживання енергії і тим самим скоротити власні витрати.

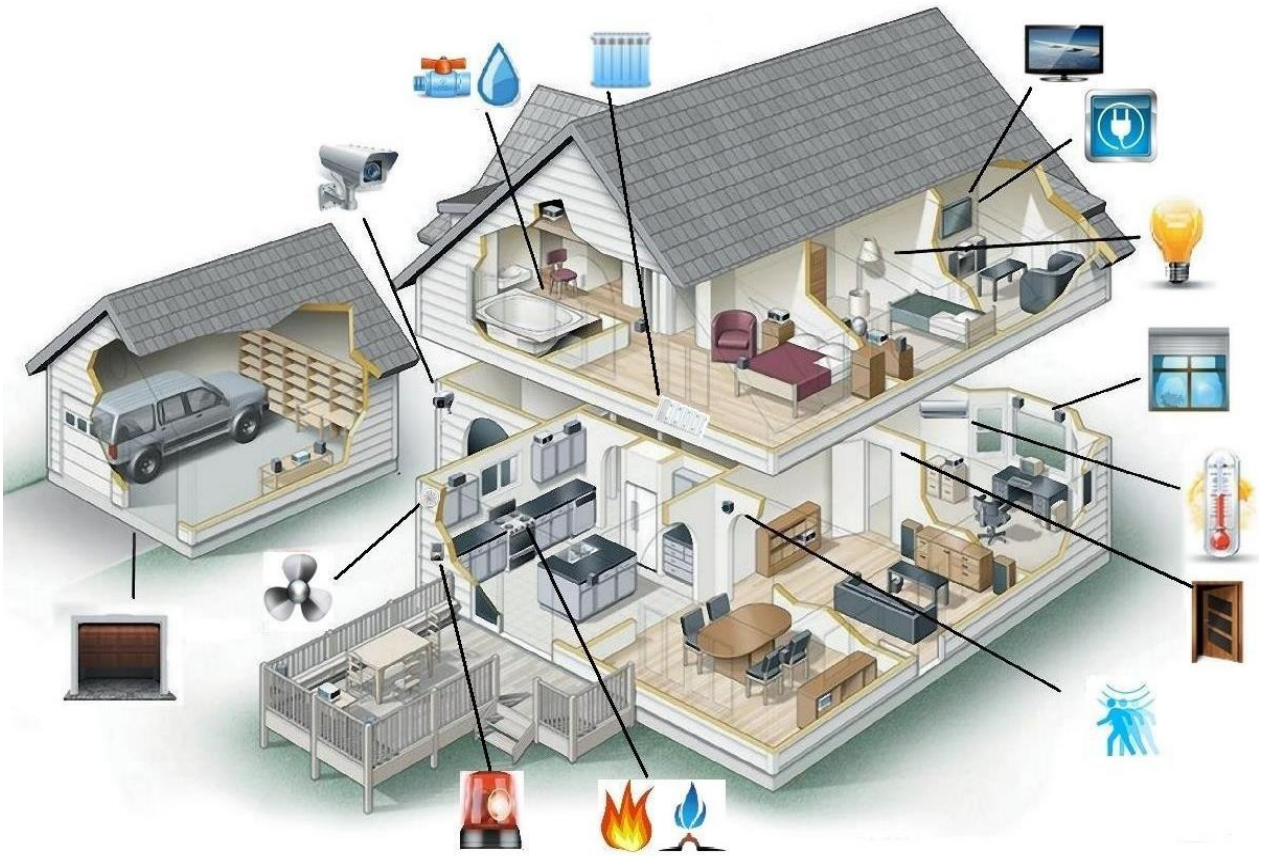

Рисунок 1.5 – Автоматизація будинку

Деякі системи розумних будинків призупиняють роботу пристроїв, поки вони знову не знадобляться. Крім того, існує кілька різних технологій для дослідження такого «розумного» будинку. Деякі стандарти використовують складні протоколи зв'язку та контрольну проводку; інші покладаються на вбудовані сигнали в запропонований ланцюг живлення будинку. Частини покладаються на радіочастотні (РЧ) сигнали, а інші стають гібридами, поєднуючи декілька методів, усі контрольні завдання виконуються за допомогою мікропроцесора [7].

При розробці системи дистанційного управління важливо підкреслити, що вона повинна складатися з двох частин: апаратної та програмної. Апаратне забезпечення може складатися з декількох блоків, таких як: смартфони, персональні комп’ютери, Arduino, Ethernet, реле, датчики температури, вологості, руху, світла, зумер та пожежна сигналізація. Arduino є основним апаратним блоком, підключеним до Ethernet Shield, відповідальним за надсилання та отримання запитів через Інтернет. Програмне забезпечення є другою частиною

такої системи (код Arduino IDE), та може працювати для управління та моніторингу.

Виконуючи оптимальну навичку планування, користувач може заощадити більше енергії та додаткові витрати грошей одночасно. IoT система управління енергією будинку забезпечує максимальне задоволення від мінімальних витрат. Модель попиту, що використовує відповідь на попит, дає більше задоволення за допомогою оптимізації часових інтервалів. Іноді планування застосовується не для всіх користувачів, тому потрібен баланс. Цей тип переваг не тільки може заощадити енергію, але і зробити наш час більш ефективним для роздумів про більше речей. Таким чином, ми знаходимо концепцію енергозберігаючої системи домашньої автоматизації, засновану на IoT, яка обговорюється в цьому документі простим та витратним способом. Ми зосереджуємось на реакції на ціни, ціноутворенні за час використання (TOUP), ціноутворенні в реальному часі (CPP), критичному ціновому рівні (CPP) та управлінні попитом (DSM).

Мікроконтролерний блок. MCU – це невеликий комп'ютер, побудований на системі на мікросхемі (SoC). Він схожий на SBC, але містить меншу потужність процесора та функціональність. Він містить ядро процесора, пам'ять та програмовану периферію введення/виводу. Мікроконтролери призначені для вбудованих додатків або додатків, які потребують невеликих ресурсів комп'ютера. Прикладами програм, які покладаються на мікроконтролери, є системи керування двигунами автомобілів, імплантаційні медичні пристрої, пульти, офісні машини та прилади.

Мікроконтролери змішаного сигналу також поширені, інтегруючи аналогові компоненти, необхідні для керування нецифровими електронними системами. PT 7.0 підтримує емулятор MCU. Користувач може програмувати PT MCU для виконання завдань, подібних до реальних MCU. Для спрощення процесу PT MCU також можна запрограмувати за допомогою Python.

PT MCU має один порт USB, шість цифрових портів вводу/виводу та чотири аналогові порти вводу/виводу Цифрові порти вводу/виводу на PT MCU дозволяють користувачеві підключати цифрові датчики та виконавчі механізми. Аналогові

порти вводу/виводу дозволяють користувачеві підключати аналогові датчики та виконавчі механізми.

На рис.1.6 показано скріншот вікна PT MCU. З точки зору симуляції, PT SBC і PT MCU дуже мало відрізняються один від одного. А саме, PT SBC може розміщувати файли та має вкладку Робочий стіл (Desktop) з низкою програм, таких як веб-браузер та клієнт електронної пошти. PT MCU не розміщує файли і не має вкладки Робочий стіл. Хоча в реальному житті Python зазвичай не підтримується на MCU, PT 7.0 реалізує підтримку Python для зручності користувача [8].

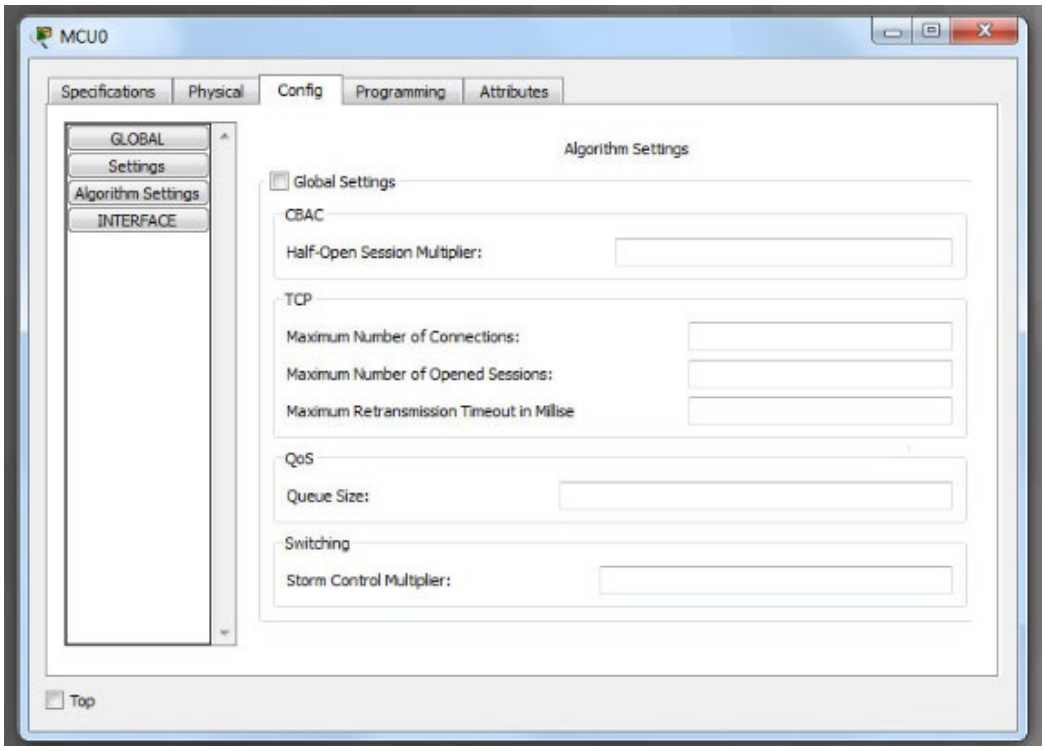

Рисунок 1.6 - Скріншот вікна налаштування MCU емулятора Cisco Packet Tracer

Висновки до першого розділу

Проаналізовано основні особливості побудови та функціонування архітектури технології Інтернет Речей (IoT) та підкреслено необхідність широкого дослідження особливостей даної технології

Виокремлено головні напрямки, які потребують більш глибокого аналізу, а саме автоматизація системи управління на використання мікроконтролерів для

здійснення первинного об‘єднання та налаштування Іо-пристроїв в окремі сегменти мережі.

Зазначено, що в емуляторі Cisco Packet Tracer можна налаштовувати функціонування мікроконтролера, однак в реальному житті Python зазвичай не підтримується на MCU. Cisco Packet Tracer (версія 7.0 і більше) реалізує підтримку Python для зручності користувача.

2 ДОСЛІДЖЕННЯ АПАРАТНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТЕХНОЛОГІЇ IOT

Сьогодні використання датчиків та сенсорів IoT здатне замінити різні системи, які раніше проектувались з набором складних електронних схем. Зазвичай серцем будь-якої вбудованої системи є мікроконтролер. У випадку рішень IoT таким мікроконтролером виступає Arduino.

Arduino - це платформа прототипування з відкритим кодом, яка використовується для виявлення та управління фізичними IoT пристроями.

2.1 Апаратне забезпечення технології IoT

У реальному житті життєво важливо забезпечити швидку та цілісну обробку та передачу даних у конкретних сценаріях. Arduino являє собою плату мікроконтролера, яку можна розглядати як спрощений шлюз IoT. При чому, Arduino – це також електронна платформа з відкритим кодом, заснована на простому у використанні апаратному та програмному забезпеченні [9]. Розробники можуть писати код на програмній платформі для управління компонентами на апаратній платформі, та виокремили три основні переваги.

Рисунок 2.1 – Arduino Uno

По-перше, це добре налаштована апаратна платформа. Arduino Uno - це плата мікроконтролера, що містить все, що потрібно мікроконтролеру, включаючи 14 цифрових входів/виходів, 6 аналогових входів, роз'єм живлення, USB-з'єднання, кнопка скидання та заголовок ICSP (In-Circuit Serial Programming) [10]. І лише за допомогою кабелю USB розробники можуть легко підключити його до власного ПК для програмування та спілкування з мікроконтролером, не турбуючись про з'єднання або інші інтерфейси.

По-друге, це програмна платформа з відкритим кодом. Простий у використанні та інтегрований IDE ідеально підходить для написання коду та завантаження його на плату мікроконтролера. Він доступний майже для всіх платформ, включаючи Windows, Mac OS X та Linux. Окрім ряду прикладних кодів, представлених в IDE, існує також великий асортимент включених бібліотек, що зменшує складність та складність програмування для розробників.

По-третє, це недорого. У порівнянні з іншими платформами мікроконтролера, плати Arduino значно дешевші за інші. Отже, розробники дозволяли робити все, що завгодно, не турбуючись про ціну, як тільки чіпи будуть зламані.

При варіанті реалізації системи «Розумний» будинок з використанням мікроконтролера Arduino можна створити систему розумного контролю мікрокліматичних параметрів з вбудованими функціями енергозбереження та прогнозу енергоспоживання. Розширюваність системи забезпечується за рахунок сумісних с Arduino модулів та датчиків, однак для підхоплення нових сенсорів потрібно встановити новий скетч на Arduino через скріптогенератор.

Система «розумний» будинок складається з трьох частин: модулі управління кліматотехніки на основі Arduino, центральний контролер і мобільний додаток для управління системою та відображення інформації у візуально-зрозумілому вигляді. Відмінність від інших рішень полягає у використанні тільки вільних компонентів під вільною ліцензією як у випадку з апаратним забезпеченням, так і з програмним. Абсолютно будь-який користувач може налаштувати систему під себе, використовуючи недорогі компоненти для Arduino і малопотужний пристрій в якості центрального контролера, не написавши при цьому жодного рядка коду.

2.1.1 Фізичні компоненти мікроконтролера Arduino

Arduino - це інструмент з відкритим кодом для розробки комп'ютерів, які можуть відчувати і контролювати більший фізичний світ, ніж настільний комп'ютер. Це фізична обчислювальна платформа з відкритим кодом, заснована на простій платі мікроконтролера та середовищі розробки для написання програмного забезпечення для плати.

Особливості:

1. 8-бітний мікроконтролер

2. Висока продуктивність, низька потужність

3. Розширена архітектура RISC (135 потужних інструкцій, виконання найбільш частотного циклу 2, 32 × 8 робочих реєстрів загального призначення, повністю статична робота, пропускна здатність до 16 MIPS на частоті 16 МГц, вбудований двоколісний множник)

4. Енергонезалежні сегменти пам'яті високої витривалості (64K/128, /256KBite внутрішньосистемного самопрограмованого спалаху, 4 Кбайт EEPROM

внутрішня SRAM на 8 Кбайт, цикли записування/стирання: 10000 Flash/100 000 EEPROM, необов'язковий розділ завантажувального коду з незалежними блокувальними бітами).

Рисунок 2.2 – Arduino IDE

2.1.2 Arduino Shields

Shields - це плати, які можна підключати поверх плати Arduino, розширюючи її можливості. Зображення екрану Arduino Ethernet, представлене на рис.2.3.

Різні екрани відповідають тій самій філософії, що і оригінальний набір інструментів: їх легко монтувати та дешево виготовляти [11]. Arduino Shields призначені для поліпшення універсальності простої плати. Майже кожна модель Arduino сумісна з призначеними для неї щитами. Щити не тільки покращують

Arduino, надаючи йому більше підключених датчиків або схем, але також містять бібліотеки коду, створені для конкретного використання.

Найбільш поширеною причиною придбання плати для Arduino є те, що проект вимагає більше пристроїв введення або виводу, а кількість портів за замовчуванням не може задовольнити всі потреби. Після того, як спільнота почала самостійно розробляти різні щити, виробники Arduino почали вбудовувати додаткові компонети безпосередньо в плати Arduino.

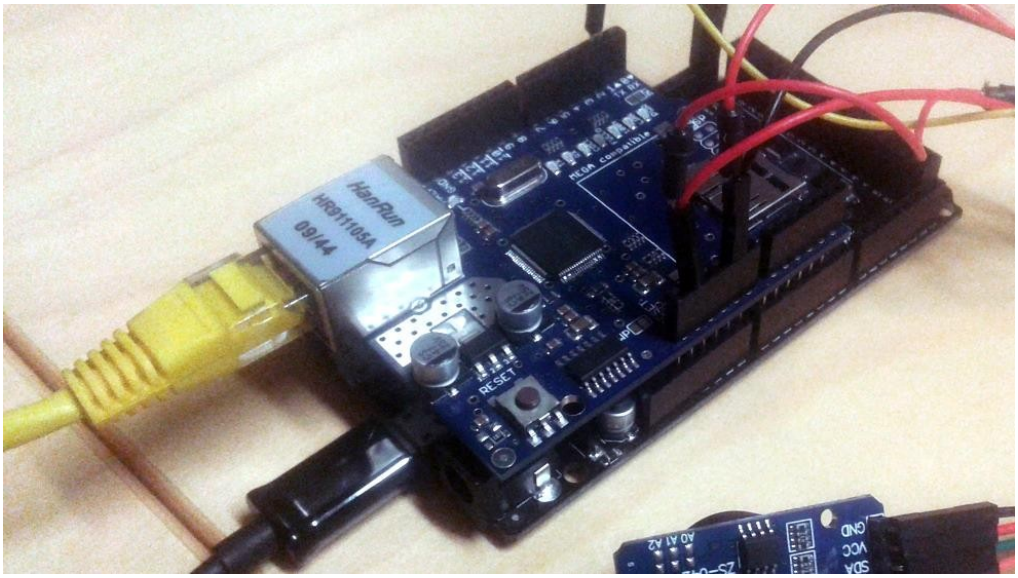

Рисунок 2.3 – Поєднання Ethernet Shield з мікроконтролером Arduino

2.2 Особливості використання сенсорних датчиків та «розумних» компонентів IoT

Призначення датчика полягає у реагуванні на вхідні фізичні властивості та перетворенні його в електричний сигнал, сумісний з електронними схемами. Датчики – це електронні пристрої, які вимірюють фізичну якість, таку як світло або температура, і перетворюють її у напругу. Приклад цифрового датчика температури та вологи представлений на рис.2.4.

Існує два типи датчиків: цифровий та аналоговий. Вихід цифрового датчика коливається від одиниці до нуля, що означає діапазон напруги датчиків. Аналоговий датчик може виводити будь-яке значення між діапазонами напруги.

Його вихідна напруга змінюється відповідно до показань з датчика. Вихід цифрового датчика ввімкнено (1), часто 5 В, або вимкнено (0), 0 В.

Аналоговий датчик використовується для вимірювання точної числової інформації, такої як температура або швидкість. Аналогові датчики можуть виводити майже нескінченний діапазон значень. Датчики використовуються для розширення можливостей Arduino.

Вихід датчика підключається до вхідного виводу Arduino і дані перетворюються в цифрову форму. Деякі датчики мають вбудований в датчик аналого-цифровий перетворювач, тому дані виводяться як цифрові дані. Датчики, які не мають вбудованого аналого-цифрового перетворювача, передають дані аналогом Arduino, який потім використовує свій вбудований перетворювач для перетворення даних у цифрові. Після обробки даних у цифрову форму їх можна обробити на мікроконтролері [12].

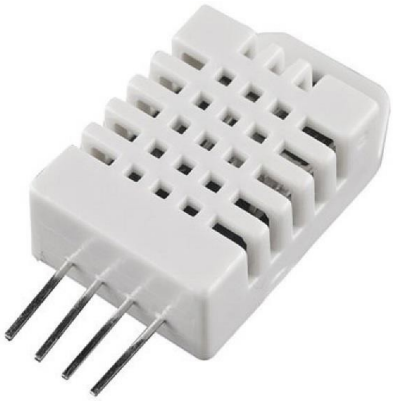

Рисунок 2.4 – Цифровий датчик вологи та температури (DHT22)

Ультразвуковий датчик. Ультразвуковий датчик – це пристрій, який може вимірювати віддаленість від об’єкта за допомогою звукових хвиль (рис.2.5). Він вимірює відстань, посилаючи звукову хвилю при певному повторенні та налаштовуючи її на пропуск назад. Зрозуміло, що кілька об’єктів, можливо, не будуть розпізнані ультразвуковими датчиками.

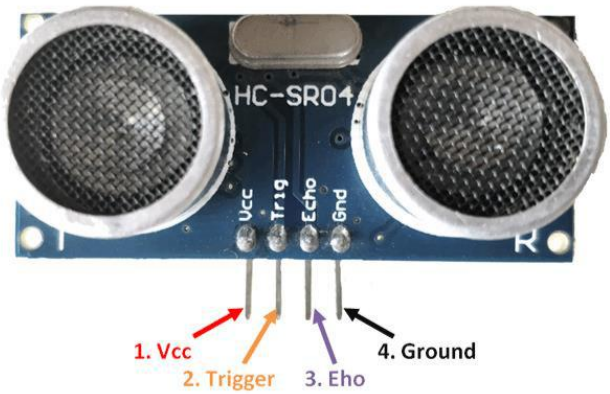

Рисунок 2.5 – Ультразвуковий датчик

Датчик температури LM35. LM35 – це прилад з температурою, що координуються з точністю, з напругою текучості, яка безпосередньо відповідає температурі за Цельсієм (рис.2.6). Низький опір, прямий вихід і точне власне регулювання пристрою LM35 роблять взаємодію з пристроєм зчитування або управління особливо простим.

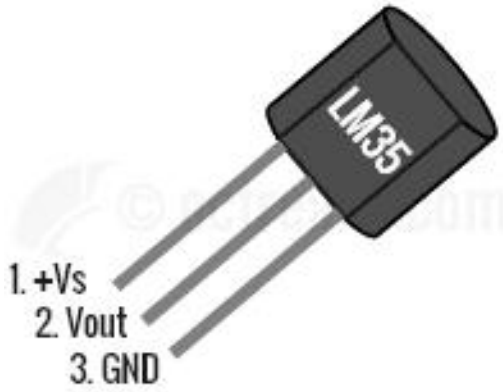

Рисунок 2.6 – Датчик температури LM35

Датчик температури. Датчик температури – це пристрій, для кількісної оцінки температури за допомогою електричного показника. Потрібна термопара або RTD (детектори температури опору). Термопара складається з двох металів, що розходяться, та створюють електричну напругу за умови, що відповідає зміні температури. RTD – це змінний опір, він змінить електричну опозицію круговим шляхом відносно змін температури точно і майже безпосередньо.

Оцінка датчика температури стосується високої або низької температури виробу. Робочою базою датчиків є напруга, яка проглядається над діодом. У випадку, коли напруга наростає, в цей момент температура підвищується і відбувається падіння напруги між клемами транзистора бази і виробника, вони реєструються датчиками. У випадку, коли різниця в напрузі посилюється, пристрій видає простий знак, і це прямолінійно відносно температури.

Світлозалежний резистор. Світлозалежний резистор (LDR) або фоторезистор – це пристрій, опір якого є складовою електромагнітного випромінювання (рис.2.7). Вони легкі у застосуванні. Їх ще називають фотопровідниками, фотопровідними елементами або, по суті, фотоелементами. Складаються з напівпровідникових матеріалів, що мають велику перешкоду.

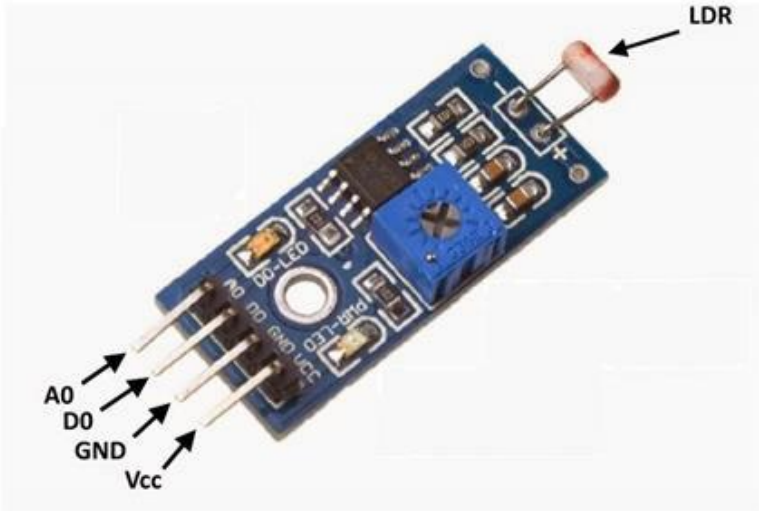

Рисунок 2.7 – Датчик LDR

Світлозахищений резистор відповідає настановам щодо фотопровідності. У той момент, коли падає світло, наприклад коли фотони падають на пристрій, електрони у валентній зоні напівпровідникового матеріалу направляються до зони провідності [13].

Наслідком цієї процедури є постійно зростаюча кількість струмів, які починають рухатись через пристрій, коли ланцюг замикається, і відтепер перешкода гаджету зменшена. Це найвідоміше робоче правило LDR.

Сервомотор 9g. Це пристрій малого розміру, підходить для управління (рис.2.8). Гаджет консолідує заряд акумулятора і веде точне вимірювання часу, коли порушується основна ємність гаджета. Включення самоцвітного резонатора покращує точність приладу на далекі відстані, а також зменшує кількість деталей, що входять в лінію складання. Доступний у бізнесі та за механічних температурних режимів, і пропонується у комплекті з 16-міліметровим 300-міліметровим SO.

Рисунок 2.8 – Сервомотор

Реле. Реле – це електромагнітний пристрій, який використовується для електричного розділення двох ланцюгів та взаємодії з ними. Являється цінним датчиком і дозволяє одній схемі перемикати іншу, поки вони абсолютно дискретні (рис.2.9).

Часто використовується для взаємодії електронної схеми (що працює при низькій напрузі) з електричним ланцюгом, який працює при винятково високій напрузі. Наприклад, реле може створити ланцюг постійного струму 5 В для перемикання мережі живлення змінного струму 220 В.

Таким чином, невелика схема датчика може приводити в дію, наприклад, вентилятор або електричну лампочку.

Рисунок 2.9 – Реле

Перемикач реле може бути розділений на дві секції: вхідну та вихідну. Вхідна секція має котушку, яка створює магнітне поле, коли до неї підключена невелика напруга від електронної схеми. Ця напруга відома як робоча напруга. Загальновживані передачі доступні в різних конструкціях робочих напруг, таких як 6В, 9В, 12В, 24В тощо. Вихідна секція складається з контакторів, які точно взаємодіють або від'єднуються. У основній передачі є три контактори: нормально розімкнутий (NO), нормально закритий (NC) і нормальний (COM). При відсутності вхідного стану COM пов'язаний з NC. У той момент, коли робоча напруга підключена, котушка реле посилюється, і COM змінює контакт на NO. Доступні різні пристрої реле, такі як SPST, SPDT, DPDT тощо, які мають відмінну кількість контактів перемикання.

Датчик газу MQ2. Для ідентифікації витоків газу цей газовий датчик MQ2 використовується в побутовій та промисловій частинах (рис.2.10). Завдяки високій чутливості та швидкій реакції вимірювання можна зробити дуже швидко. Його чутливість можна регулювати за допомогою потенціометра.

Рисунок 2.10 – Датчик газу MQ2

Цей датчик може зафіксувати пари та викиди гідрогену, монооксиду вуглецю, вуглецю, паливних газів, диму і т.д. Широкий діапазон виявлення, вища чутливість та довговічні характеристики стабільності роблять цей датчик широко використовуваним у різних професійних та домашніх секторах.

Специфікації:

- Робоча напруга від 4,9-5,1 В;
- Споживання опалення 0,5-800 мВт;
- Опір зондування 3-30 кОм.

Модуль цифрового зумера MOD-00055. Цей простий цифровий зуммерний модуль, модель MOD-00055 використовується для того, щоб попередити користувача та інших, коли цей зуммер отримує команду від датчика газового детектора MQ2 (рис.2.11).

Рисунок 2.11 – Модуль цифрового зумера MOD-00055

Цей зуммер має 3 висновки із землею і VCC, який працює при напрузі 5 В. Інший штифт - вимикач для з'єднання з датчиком.

Модуль годинника DS3231. Модуль годинника реального часу DS3231 використовується для запам'ятовування реального часу та дати за відсутності зовнішнього джерела живлення (рис.2.12). Він має власну установку акумулятора для автоматичного оновлення часу та дати, навіть якщо пристрій вимкнено. Реалізація цього в цьому проекті з тієї ж причини. Щоб отримати значення в реальному часі, він підключений до Arduino для збереження даних реального часу та дати, коли зовнішнє виробництво електроенергії вимкнено.

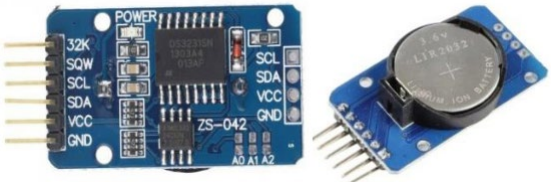

Рисунок 2.12 – Модуль годинника DS3231

Всього 10 штифтів, включаючи VCC та GND. 32 кГц є вихідним генератором для отримання вихідного сигналу. Цей модуль RTC збирає послідовні дані у штирі SDA. Штифт SCL призначений для збору даних вимірювання часу. Щоб отримати вихід прямокутної хвилі, інший висновок SQW розміщений безпосередньо біля вихідного виводу 32 кГц.

Специфікація

- RTC підраховує години, хвилини, секунди та роки;
- Цифровий датчик температури з точністю +/- 3 градуси
- Точність: + 2ppm до -2ppm для 0 до 40 градусів Цельсія, + 3,5ppm до -3,5ppm для -40 до +85 градусів Цельсія;
- Інтерфейс I2C 400 кГц;
-Автоматичний ланцюг перемикання акумулятора;
- Низьке споживання енергії, питний розмір та підтримка протягом 3 років;
- Робота постійного струму: 2,3-5,5 В;
- Споживає резервну батарею на 500 нА;

- Максимальна напруга: VCC +0,3 В.

Цифровий датчик вологості та температури DHT22. Цифровий датчик вологості та температури DHT22 (рис.2.12) підключений до цифрового виводу 12 мікроконтролера.

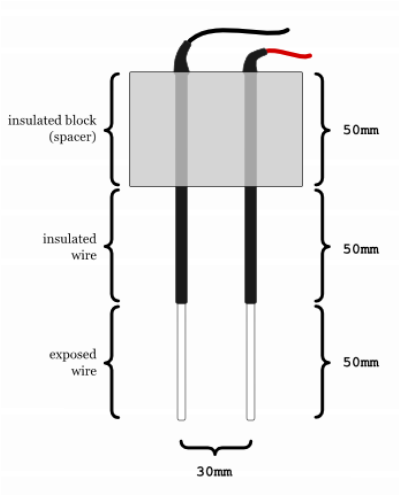

Рисунок 2.12 – Датчик вологості ґрунту

Два датчики вологості ґрунту, показані на рис.2.13, підключені до виводів A0 і A1. Струм подається від Arduino до одного з полюсів датчика вологості ґрунту. Другий полюс датчика підключений до аналогового штифта Arduino. Аналоговий вихід використовується для вимірювання провідності ґрунту. Волога в ґрунті збільшує провідність ґрунту. Схема підключення датчика вологості ґрунту показана на рис.2.14.

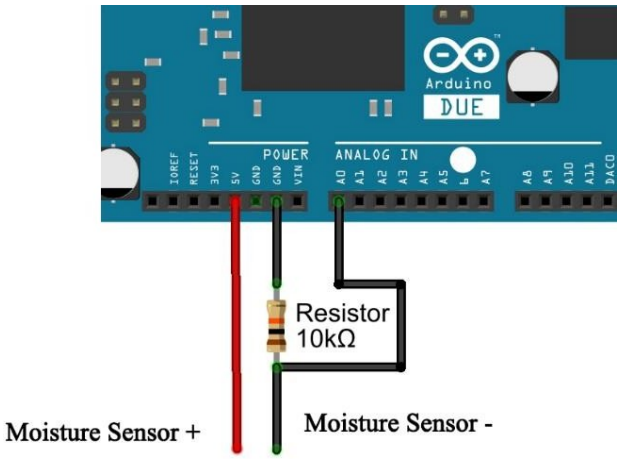

Рисунок 2.13 – Схема підключення датчика вологи

Модулі реле необхідні для відокремлення пристроїв високої струму та високої напруги від пристроїв логічного рівня. Водяний насос та нагрівач – це пристрої високої напруги та сильного струму, які управляються за допомогою релейних модулів. Модулі підключені до цифрових виводів 6 і 7. Блок-схема відображає зв'язок між Arduino, датчиками та модулями. Графічний варіант схеми показаний на рис.2.14.

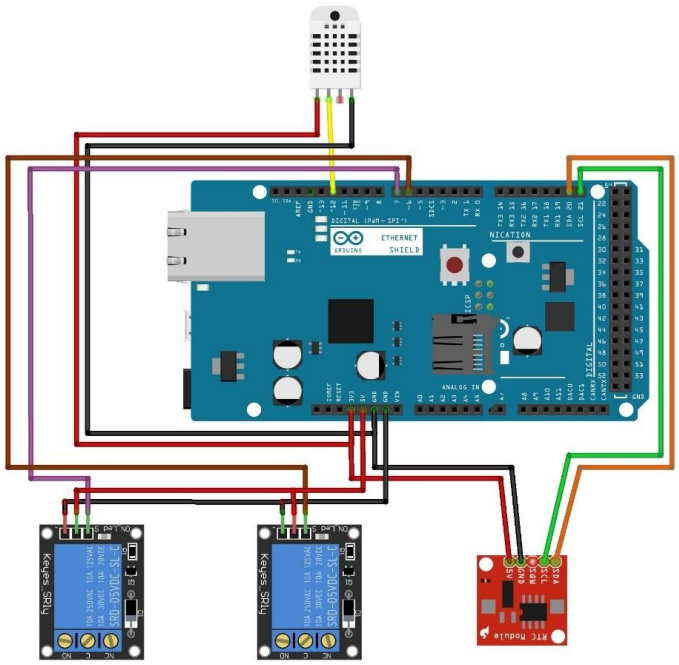

Рисунок 2.14 - Схема підключення обладнання

2.3 Програмне забезпечення технології IoT

HOMER. Професійний симулятор HOMER – це імітаційна модель, яка імітує життєздатну структуру для кожного окремого поєднання обладнання, яке необхідно розглянути. Залежно від того, як користувач налаштовує схему, HOMER може відтворити сотні або навіть тисячі фреймворків. HOMER здатен змоделювати діяльність від однієї хвилини до 60 хвилин.

Texmaker. Texmaker – це сучасний крос-платформний редактор LaTeX, який безкоштовно використовується для систем Linux, Mac OS та Windows. Він інтегрує безліч інструментів в одному додатку для розробки документів за допомогою

LaTeX. Texmaker включає підтримку Unicode, виведення коду, перевірку правопису, автоматичне заповнення та вбудований переглядач PDF. Цей додаток використовується для таких завдань, як генерування нових документів, створення таблиць, відображення табличних та графічних середовищ, експортування даних в документ LaTeX через TeX4ht. Він автоматично визначає помилки та створює попередження, виявлені у журналі, після компіляції. Цей редактор дуже простий у використанні та налаштуванні.

ThingSpeak. ThingSpeak – це програма IoT та API, яка є відкритим вихідним кодом для зберігання та отримання даних з апаратних та сенсорних пристроїв. Використовує протокол HTTP мережі Інтернет або LAN для зв'язку з ним. Бази даних MATLAB включені для оцінки та імітації отриманої інформації про апаратні засоби або датчики. Може бути використана як хмара з відкритим кодом, у випадку, коли у розробника немає купленого домену, а веб-сторінка не відкрита, тому для спостереження використовується сервер ThingSpeak.

IDE Arduino для програмування контролера та ESP. Інтегроване середовище розробки Arduino або Arduino Software (IDE) – містить диспетчер вмісту для складання коду, область повідомлення, зручність вмісту, панель інструментів з кнопками для нормальної ємності та прогресу меню. Він асоціюється з обладнанням Arduino та Genuine для передачі програм та розмови з ними.

Проекти, складені з використанням програмного забезпечення Arduino (IDE), називаються контурами. Записані в текстовому середовищі та можуть бути пошкоджені при збільшенні записів. Менеджер редакції має основні можливості для вирізання/склеювання та пошуку/заміни вмісту [13].

Arduino (IDE) показує ймовірні помилки та дає можливість їх виправити. У нижньому правому куті вікна показано розроблену плату та послідовний порт. Фіксатори на панелі інструментів дозволяють підтверджувати та передавати програми, робити, відкривати та резервувати макети, а також відкривати послідовний екран.

У програмуванні Arduino є дві основні функції. Основними функціями є setup () та loop (). Функція Setup () працює лише один раз, коли пристрій завантажується,

вона в основному використовується для налаштування параметрів ініціювання. Цикл () запускається після завершення функції setup (), функція loop () буде працювати кілька разів, поки не буде натиснута кнопка вимкнення або скидання (рис.2.15).

Програмування Arduino підтримується великою кількістю бібліотек. Велика кількість бібліотек з відкритим кодом доступна від спільноти Arduino. Блок-схема функції Setup () описує процес налаштування системи.

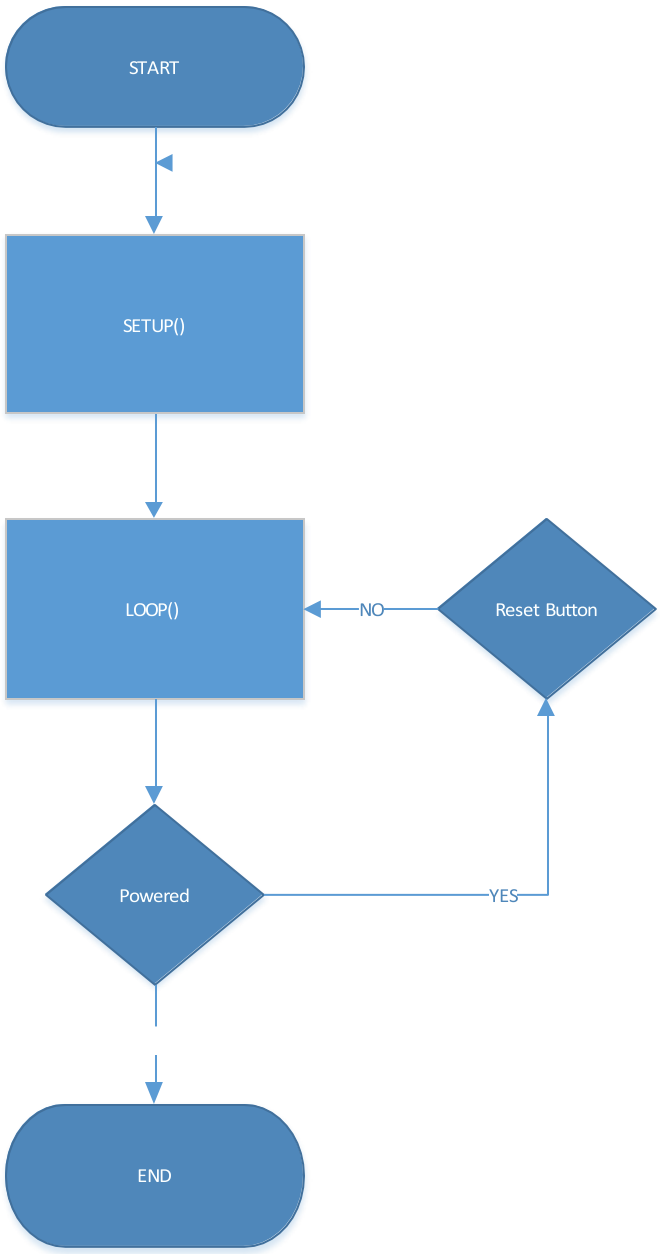

Рисунок 2.15 - Блок-схема програмного забезпечення Arduino

Налаштування програмного забезпечення спочатку було розпочато з датчиків та модуля годинника реального часу. Arduino IDE надав бібліотеки для читання даних з модулів DHT22 та RTC. Дані датчика вологості ґрунту (рис.2.16) зчитувались безпосередньо з аналогових виходів. Дані з датчиків та RTC були роздруковані на серійному моніторі IDE для цілей тестування.

Однією з вимог системи було те, що для збереження діаграм зберігатимуться довгострокові дані. Arduino надає бібліотеку карт SD, яка може бути використана для створення функції sdCardDatalog ().

Функція зберігає дані датчика та час на SD-карті на мережевому екрані W5100. Щоб надати користувачеві можливість віддалено контролювати свій проект, наприклад через веб-сторінку, необхідно створити веб-сервер. Бібліотеки веб-серверів були створені для Arduino, але вони не відповідали вимогам системи [14].

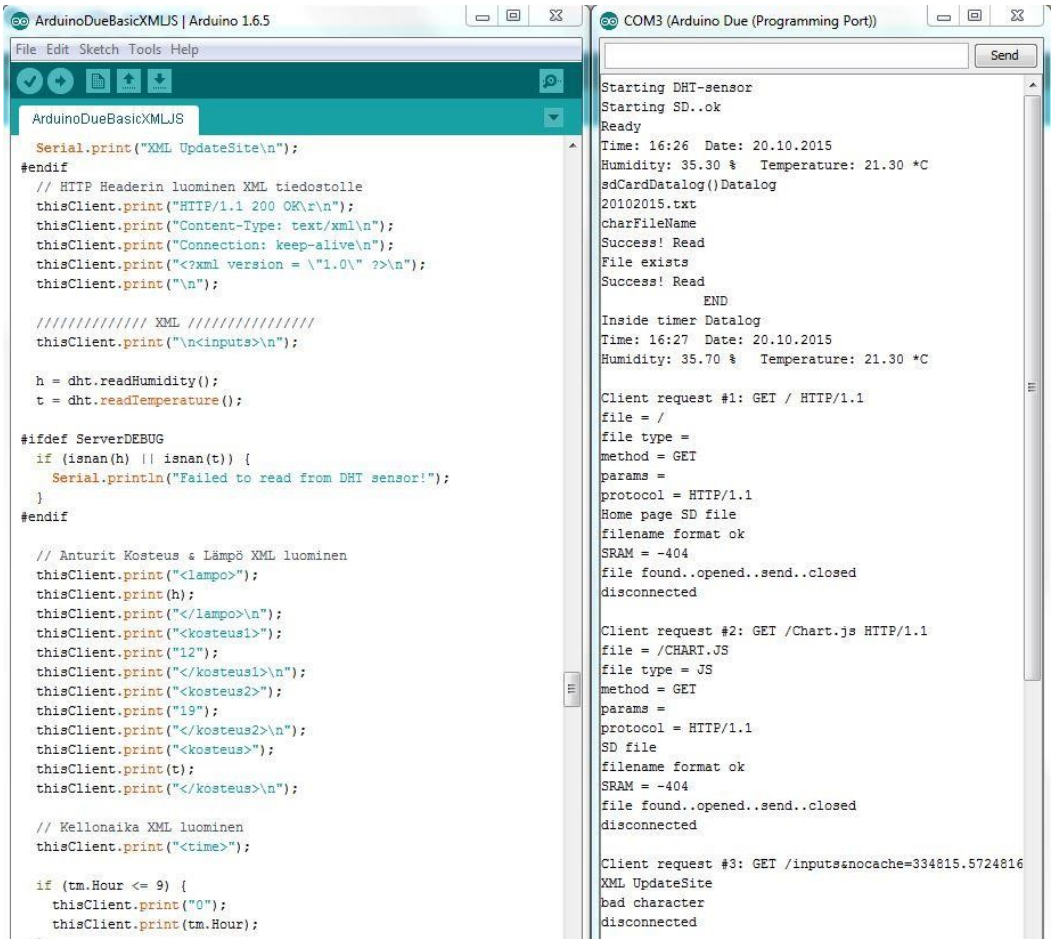

Рисунок 2.15 – IDE Arduino з послідовним монітором

На рис.2.16 приведений приклад вихідного коду системи. Асинхронний JavaScript та XML (AJAX) використовувались для передачі даних датчиків з веб-сервера на клієнта. Клієнт надсилає запит об'єкта XMLHttpRequest для зв'язку зі сценаріями на стороні сервера. Сервер відповідає на запит, надсилаючи файл XML, що містить дані сенсора в заголовку HTTP. Запит може надсилати, а також отримувати інформацію в різних форматах, включаючи XML, HTML і навіть текстові файли. Запит від клієнта та сервера представлені на рис.2.16. Якщо веб-сервер не може отримати доступ до даних модуля RTC, система не зберігає дані датчика на SD-карті та не надсилає його веб-клієнту.

Рисунок 2.16 – Відповідь XML для передачі даних між веб-сервером та клієнтом

Arduino пропонує інструменти для сприяння впровадженню системи. Arduino IDE забезпечує послідовну комунікацію з мікроконтролером Arduino через USB. Друк на послідовній консолі в IDE допомагає відстежувати функціональні події між апаратним та програмним рівнем.

Бібліотеки спільноти Arduino з відкритим кодом допомагають реалізації датчиків та мережевого екрану для Arduino. У цій реалізації бібліотеки, які використовуються, мають спростити отримання даних датчиків та мережеве підключення. Бібліотеки надають доступ до даних датчиків у реальному часі з однією функцією [15].

Після того як датчик, веб-сервер, SD-карта та функції годинника реального часу були протестовані та діагностовані як повністю функціонуючі, було зібрано

програмне забезпечення системи. Можливість друку послідовних даних Arduino в процесі розробки допомогла налагодити систему. Для перевірки передачі даних між веб-сервером і клієнтом було створено кілька фіктивних змінних для заповнення лінійної діаграми на веб-сайті. Приклад послідовного коду, що використовується під час тестування, видно на рис.2.17.

```
86
87  #ifdef ServerDEBUG
88    Serial.print(F("Starting SD.."));
89  #endif
90
91    if(!SD.begin(4)) {
92  #ifdef ServerDEBUG
93      Serial.println(F("failed"));
94  #endif
95    }
96    else {
97  #ifdef ServerDEBUG
98      Serial.println(F("ok"));
99      sdStatus = 1;
100 #endif
101   }
```

Рисунок 2.17 – Фрагмент коду послідовного друку, що використовується для тестування

Висновки до другого розділу

Проаналізовано особливості створення та розповсюдження технології IoT. Описано принцип роботи системи «Розумний» будинок при умові впровадження мікроконтролера Arduino, а також приведено головні переваги даного мікроконтролера.

Проведено дослідження головних завдань, які здатна виконувати система «Розумний» будинок при умові підключення пристроїв IoT.

Приведено особливості впровадження та використання базових датчикі та сенсорів, та підкреслено, що їх призначення полягає у реагуванні на вхідні фізичні властивості та перетворенні його в електричний сигнал, сумісний з електронними схемами. Датчики – це електронні пристрої, які вимірюють фізичну якість, таку як світло або температура, і перетворюють її у напругу.

3 ДОСЛІДЖЕННЯ ІСНУЮЧИХ АЛГОРИТМІВ ДЛЯ УПРАВЛІННЯ ДОДАТКАМИ ТА КАНАЛАМИ ЗВ'ЯЗКУ В ІОТ

Система має мікроконтролер, що в більшості випадків виконує функцію центрального процесора, який з'єднаний з портативним блоком плат, що складається з веб-сервера та датчиків.

Мікроконтролер також може бути підключеним до побутових електроприладів за допомогою електричних реле та деяких інших різних елементів, таких як двигуни та дальні датчики. Центральний мікроконтролер слідує різним алгоритмам для збору інформації, порівняння та прийняття рішень, показує результати та контролює прилади.

Після підключення, мікроконтролер готовий показувати стан електричних навантажень та різні результати, а також приймати команду від клієнта для управління приладами. Для виконання бажаних додатків використовуються різні типові алгоритми [16].

3.1 Управління одним електричним навантаженням змінного струму

Якщо користувач натискає кнопку "ON load-1" з веб-сторінки, тоді esp-8266 посилає сигнал 3,3 В на мікроконтролер, і він зберігає значення напруги під змінною, що називається load-1, а потім порівнює значення, яке більше ніж 3 В чи ні.

Якщо воно дорівнює або більше 3 В, тоді мікроконтролер посилає сигнал 5 В на відповідне реле для навантаження-1. Якщо значення напруги, що зберігається в навантаженні-1, менше 3 В, тоді контролер посилає 0 В на відповідне реле для навантаження-1. Коли реле отримує 5 В на вході, воно замикає лінію живлення 220 В з навантаженням-1 на його вихідній стороні. І коли реле отримує 0 В на вході, воно розмикає лінію живлення 220 В від навантаження-1, отже навантаження-1 перестає працювати.

Алгоритм 1 – Контроювання сигналу Electrical AC Load приедений нижче:

Require: Voltage signal from esp-8266
Load-1 = incoming voltage value from esp-8266
if (load-1≥3V) then
return 5V to the relay of load-1
else
return 0V to the relay of load-1
end if

Дотримуючись алгоритму, користувач може керувати кількома побутовими електроприладами одночасно. Однак кількість контрольованих приладів буде залежати від кількості вихідних портів центрального або головного мікроконтролера.

3.2 Вимірювання температури та контроль роботи кондиціонера

Для вимірювання температури може бути використаний датчик температури (lm35), який підключений до мікроконтролера. Мікроконтролер збирає значення датчика з lm35 і зберігає його під змінною з назвою temp-Celsius. Це значення сигналу напруги, що зберігається під температурою (Цельсій), яке надходило від датчика lm35. Щоб перетворити його у відповідний контролер температури, множиться temp-Celsius на 500, а потім ділиться на 1023 і, нарешті, зберігає результат під змінною з назвою temp (500 і 1023 - це значення специфікації мікроконтролера) [17].

Тепер збережене значення в temp є виміряною кімнатною температурою. Потім мікроконтролер порівнює, що збережене значення в temp дорівнює або перевищує 25 градусів чи ні (тут ми приймаємо 25 градусів як температуру планування кондиціонера, але користувач може встановити будь-яку температуру відповідно до власних потреб). Якщо значення temp дорівнює або перевищує 25 градусів, тоді контролер надішле 5 В на відповідне реле кондиціонера. Якщо

значення температури не дорівнює або перевищує 25 градусів, тоді контролер надішле 0В на відповідне реле.

Однак для управління кондиціонером необхідно тримати ручний перемикач між лінією живлення змінного струму та реле кондиціонера, щоб запобігти тому, що температура в корпусі перевищує 25 градусів, але в кімнаті нікого немає. У цьому випадку користувач повинен вимкнути ручний вимикач перед виходом з приміщення, щоб контролер надіслав 5 В на відповідне реле, оскільки температура перевищує 25 градусів, але кондиціонер не отримуватиме напруги живлення, оскільки ручний вимикач вимкнений. Знову ж таки, користувач повинен увімкнути ручний перемикач, щоб подати напругу живлення, і тоді контролер буде контролювати кондиціонер відповідно до температури.

Алгоритм 2 – Заходи щодо контролю температурними режимами на кондиціонері.

Require:LM-35 sensor value
temp-celsium = LM-35 sensor value
*temp = [temp-celsium * 500]/1023*
if (temp≥25 degree) then
return 5V to the relay of the ac
else
return 0V to the relay of the ac
end if

3.3 Управління датчиком освітлення

Якщо в приміщенні є три ліхтарі, які віддалені від ультразвукового датчика на 357 см, 528 см та 954 см. Три вогні та датчик підключені до мікроконтролера. Він обчислює відстань між вхідними або вихідними об'єктами та датчиком, а потім відповідно до положення об'єкта вмикає відповідне світло коридору. Для розрахунку відстані об’єкта можна використовували ультразвуковий датчик.

Ультразвуковий датчик постійно передає звукову хвилю і приймає відбиту звукову хвилю, якщо перед нею є предмет. Нарешті, датчик визначає загальний час, що пройшов між відправкою та прийомом хвилі, і передає час як значення датчика контролеру.

Мікроконтролер збирає значення часу з ультразвукового датчика і зберігає його під змінною назвою duration1. А потім необхідно розділити тривалість1 на 33 (оскільки швидкість звукової хвилі в повітрі становить 330 м/с; обирається 33 так, щоб відстань була розрахована в см) і знову необхідно поділити на 2, оскільки хвиля проходить ту ж відстань два рази. Нарешті, мікроконтролер зберігає обчислене значення відстані під назвою змінної distance1.

Алгоритм 3 - Управління датчиком освітлення.

Require: Ultrasonic Sensor Value
duration1 = ultrasonic sensor value
distance1 = [duration1 / 33] /2
if (distance1≤357 cm) then
    return 5v to the relay of light2
else
    return 0v to the relay of light2
end if
if (357<distance1≤528 cm) then
    return 5v to the relay of light3
else
    return 0v to the relay of light3
end if
if (528<distance1≤954 cm) then
    return 5v to the relay of light1
else
    return 0v to the relay of light1
end if

3.4 Управління механізмом автоматичних дверей

Автоматичні двері можуть бути встановлені скрізь в будинку, наприклад в кімнаті для гостей. Дверцята будуть відкриватися та закриватися сервомотором постійного струму. Над дверима є датчик ехолота. Якщо хтось стоїть перед дверима, тоді його/її відстань буде приблизно від 5 см до 45 см від датчика ехолота.

Якщо положення людини знаходиться від 5 см до 45 см від датчика ехолота, тоді мікроконтролер приводить ротор двигуна в положення 1 (припускається, що положення ротора одне визначається для відкривання дверей, і буде залежати від розміру та ваги дверей). В іншому випадку ротор двигуна залишиться в положенні два (припускається, що положення два визначено для закриття дверей) [18].

Однак, щоб встановити автоматичні двері, необхідно встановити гідроакустичні датчики по обидва боки від дверей, щоб кожен міг увійти в кімнату, а також вийти з неї. Ще одна головна річ полягає в тому, що двигун дверей буде рухатися від напруги постійного струму, що зберігається в батареї. Отже, користувач повинен знати, що акумулятор має достатню напругу чи ні. Низька напруга вплине на функціональність двигуна.

Алгоритм 4 – Управління механізмом автоматичних дверей

Require: Ultrasonic Sensor2 Value
duration2 = ultrasonic sensor2 value
distance2 = [duration2 / 33] /2
if (5 cm≤distance2≤45 cm) then
    drive the motor for rotor position one
else
    drive the motor for rotor position two
end if

3.5 Управління вхідними дверцятами з веб-сторінки

На веб-сторінці є кнопка для відкриття та закриття головних дверей будинку. Якщо користувач натискає кнопку, щоб відкрити головні дверцята, тоді esp-8266 надсилає 3,3 В на центральний мікроконтролер. Потім мікроконтролер приводить серводвигун дверей у положення відкривання. Якщо користувач натискає кнопку, щоб закрити головну дверцята, тоді esp-8266 посилає 0V мікроконтролеру. А мікроконтролер приводить серводвигун у положення закриття.

Алгоритм 5 – Управління вхідними дверцятами з веб-сторінки

Require: voltage signal from web server (esp-8266)
door-voltage = incoming voltage from esp-8266
if (door-voltage ≥ 3V) then
    drive the motor2 for opening position
else
    drive the motor2 for closing position
end if

3.6 Управління водяним насосом відповідно до рівня води

Оскільки водяний насос підключений до мікроконтролера за допомогою електричного реле, можна налаштувати його, виходячи з кількості води, зарезервованої в резервуарі для води, та зміни ціни на електроенергію за часом. Для вимірювання рівня води в резервуарі можна використати гідроакустичний датчик.

Типова висота резервуара для води 5000 л становить близько 230 см. Наприклад можна обрати 225 см за умови 100% заповнення. Отже, на кожен см резервуар буде заповнений 0,44%. Коли рівень води нижче 15%, мікроконтролер вмикає водяний насос.

Коли рівень води знаходиться в межах від 15% до 50%, мікроконтролер перевірить, пікові години це чи ні. Якщо це пікова година, мікроконтролер не

вмикає водяний насос. Якщо це не пікова година, тоді мікроконтролер увімкне водяний насос. Нарешті, коли рівень води перевищує 95%, мікроконтролер вимикає водяний насос.

Щоб відстежувати час і дату та планувати роботу пристрою залежно від годин пік або не пікових годин, можна використовували годинник реального часу (RTC) DS-3231 [19]. Після підключення DS-3231 до мікроконтролера та завантаження асоційованої бібліотеки можна планувати електричні навантаження, використовуючи цей час. Припускається, що години піку з 6 вечора до 23.59 вечора.

Алгоритм 6 - Управління водяним насосом відповідно до рівня води

Require: Sonar sensor value, real time from DS-3231
duration3 = sonar sensor value
time = incoming DS-3231 value
distance3 = [duration3 / 33] / 2
*water-level = distance3 *0,44*
if (water-level ≤ 15 percent) then
    return 5V to the relay of the water pump
else (15 percent < water-level ≤ 50 percent)
    if (18:00 ≤ time ≤ 23:59) then
        return 0V to the relay of the water pump
    else
        return 5V to the relay of the water pump
    end if
end if
if (water-level ≥ 95 percent) then
    return 0V to the relay of the water pump
else
    return 0V to the relay of the water pump
end if

3.7 Управління автоматичним смітником

Смітник має гідролокаційний датчик, який підключений до мікроконтролера. Мікроконтролер приймає значення датчика ехолота і обчислює положення об'єктів всередині сміттєвого ящика. Припускається, що типова висота кошика для сміття становить 90 см. Отже, на кожен сантиметр сміттєвого відра заповнюється 1,11%.

Алгоритм 7 – Управління автоматичним смітником

Require: Sonar sensor value
duration1 = sonar sensor value
distance4 = [duration4 / 33] / 2
*wastage-level = distance4 * 1,44*
Show wastage level in the display

3.8 Управління інтенсивністю освітлення

Для контролю інтенсивності освітлення спочатку потрібно виміряти фактичну інтенсивність світла в приміщенні. Отже, для вимірювання інтенсивності світла в приміщенні використовується світлозалежний резистор (LDR). Відповідно до значення LDR, спочатку надсилається на світло інший модульований по ширині імпульсу (ШІМ) сигнал.

Алгоритм 8 – Управління інтенсивністю освітлення

Require: Idr signal value
intensity = Idr signal value
if (intensity ≤800 mV) then
    return 33 percent PWM signal to the light
else (800 mV < (intensity < 900 mV)
    return 66 percent PWM signal to the light
end if

if (intensity ≥ 900 mV) then

return 100 percent PWM signal to the light

end if

3.9 Виявлення диму та легкозаймистого газу

Для виявлення диму та легкозаймистого газу можна використати датчик MQ-2. Датчик MQ-2 чутливий до диму та наступних легкозаймистих газів: LPG, бутан, пропан, метан, етан, алкоголь та водень. MQ-2 виявляє концентрацію диму/газу в атмосфері і відповідно видає сигнал напруги [20].

Якщо рівень диму/газу високий, то вихідна напруга висока. Коли рівень диму/газу низький, тоді вихідна напруга низька. Тут порогова напруга для датчика становить 400 мВ. Якщо значення датчика перевищує 400 мВ, тоді центральний контролер направить на зумер 5 В, і зумер буде гудіти, щоб повідомити користувача про підвищення рівня диму/газу.

Алгоритм 9 – Виявлення диму та легкозаймистого газу

Require: MQ-2 sensor value

threshold-voltage = 400 mV

smoke/gas-level = MQ-2 sensor value

if (smoke/gas-level ≤ threshold-voltage)

then

return 0V to the buzzer

else

return 5V to the buzzer

end if

Висновки до третього розділу

Зазначено, що система керується мікроконтролером, що в більшості випадків виконує функцію центрального процесора, який з'єднаний з портативним блоком плат.

Підкреслено, що мікроконтролер також може бути підключеним до побутових електроприладів за допомогою електричних реле та деяких інших різних елементів, таких як двигуни та дальні датчики. Центральний мікроконтролер слідує різним алгоритмам для збору інформації, порівняння та прийняття рішень, показує результати та контролює прилади.

Виокремлено, що після підключення, мікроконтролер готовий показувати стан електричних навантажень та різні результати, а також приймати команду від клієнта для управління приладами.

Досліджено різні типові для «розумного» будинку алгоритми, які необхідні для налаштування головних датчиків та сенсорів. Дотримуючись алгоритмів користувач може здійснювати управління кількома побутовими електроприладами одночасно. Однак кількість контрольованих приладів буде залежати від кількості вихідних портів центрального або головного мікроконтролера.

4 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ УПРАВЛІННЯ IOT ПРИСТРОЯМИ З ВИКОРИСТАННЯМ МІКРОКОНТРОЛЕРА ЗА ДОПОМОГОЮ ПРОГРАМУВАННЯ PYTHON

4.1 Програмування IoT-пристроїв візуальною мовою програмування (Blocky)

Подібно до людських мов, існує багато різних комп'ютерних мов. Деякі комп'ютерні мови кращі за інші при виконанні певних завдань. JavaScript, інтерпретована комп'ютерна мова, призначена для створення веб-додатків. За допомогою JavaScript програміст може створювати веб-додатки, які можуть взаємодіяти з користувачами та іншими програмами.

Python, інша інтерпретована мова, дозволяє простіші конструкції. Python дуже простий у використанні, потужний та універсальний, обирається багатьма розробниками IoT. Однією з основних причин популярності Python є спільнота розробників. Розробники Python створили та надавали доступ до багатьох спеціальних модулів, які можна імпортувати до будь-якої програми, щоб негайно запозичити доданий функціонал.

Blockly - це мова візуального програмування, яка дозволяє користувачам створювати програми, об'єднуючи блоки, які представляють різні логічні структури мови, а не написанням фактичного коду. Blockly працює в веб-браузері та може відображати візуально створену програму на JavaScript, PHP або Python. Приклад Blockly показаний на рис.4.1.

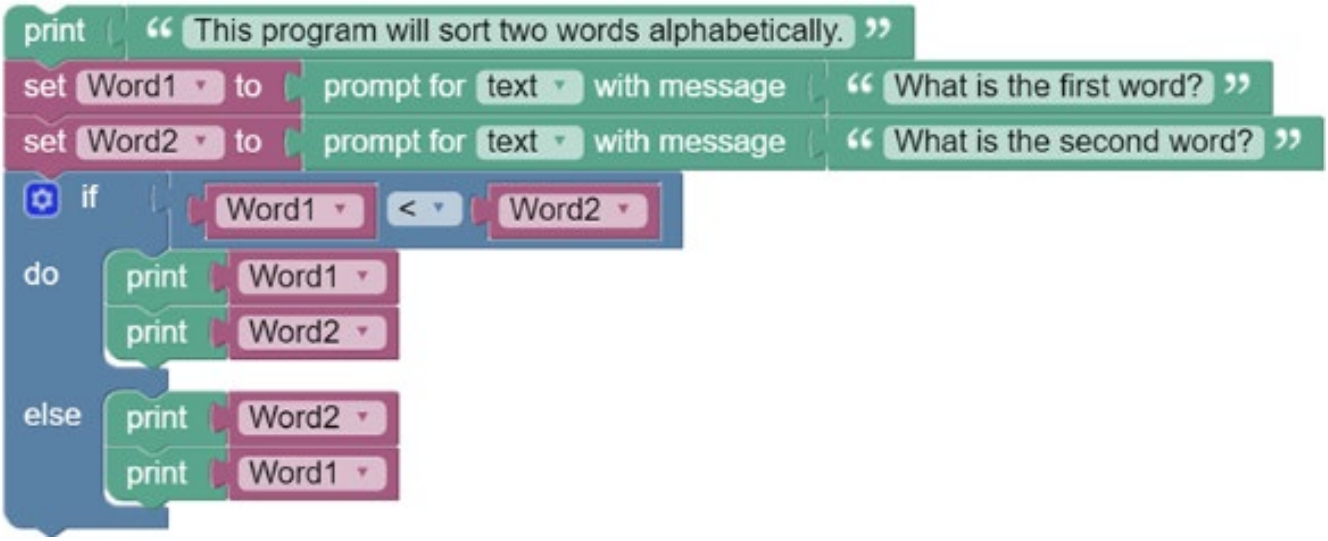

Рисунок 4.1 – робочий простір Blockly

C, компільована мова, чудово підходить для створення складних та швидких програм, але суворі правила та синтаксис ускладнюють її розвиток. Створений на початку 70-х років, C стала однією з найбільш широко використовуваних мов програмування усіх часів. Операційна система Linux написана на C.

Java є компільованою мовою, до якої можна застосувати твердження “write once, run anywhere” (WORA). Попри те, що назви схожі, Java і JavaScript не пов'язані між собою. Java призначена для роботи на будь-якій платформі без необхідності перекомпіляції. Для додатків на базі Java потрібна платформа JVM (Java Virtual Machine), встановлена на комп'ютері. JVM - середовище, в якому виконується компільований код Java.

4.2 Змінні та основні вирази

Blockly - це інструмент для графічного програмування, сворений для того щоб пояснити новачкам в програмуванні, основні концепції. Використовуючи декілька типів блоків, Blockly дозволяє користовачу створити програму без написання коду.

Blockly реалізує графічне програмування, призначаючи різним структурам програмування різні кольорові блоки. Блоки також містять комірки та пропуски,

які дозволяють програмістам ввести значення, яке потребує структура. Програмісти можуть звязувати між собою структури програмування, перетягуючи і прикріплюючи відповідні блоки. Структури програмування такі як, умови, цикли та змінні доступні для використання.

Змінна - це об'єкт програмування, який використовується для зберігання даних. По суті, змінна - це частина пам'яті, якій можна присвоїти зрозуміле для людини ім'я для зберігання та пошуку даних. Також є можливість змінювати значення змінної в процесі виконання програми.

Створення нової змінної в Blockly - це проста задача перетягування блоку змінної до області робочого простору та заповнення комірки значень. На малюнку показана змінна Blockly.

Blockly також підтримує функції. Подібно до змінних, блоклі має певні блоки для представлення функцій. Подібно до змінних, програмісти просто вибирають та перетягують функціональні блоки до робочої області та заповнюють необхідні комірки. Блок змінної та блок виводу значення на екран мають виступ у нижній частині та проріз у верхній. Це означає що два блоки можуть бути з'єднані разом для створення послідовності в програмі. Спершу Blockly виконає команду верхнього блоку, а потім перейде на блок знизу.

Доступні також інші блоки, такі як блок IF THEN, блок WHILE та блок FOR. Існують також спеціальні блоки для датчиків та виконавчих механізмів. Врешті, Blockly можна використовувати для перекладу коду на основі блоків в Python або JavaScript.

Умова IF-THEN. Структура умови IF-THEN використовується для прийняття рішень. Після перевірки достовірності виразу програма перейде до наступного твердження, якщо вираз буде оцінено як хибний (false). Якщо вираз є істинним (true), то дія виконується перед тим, як перейти до наступного оператора в коді.

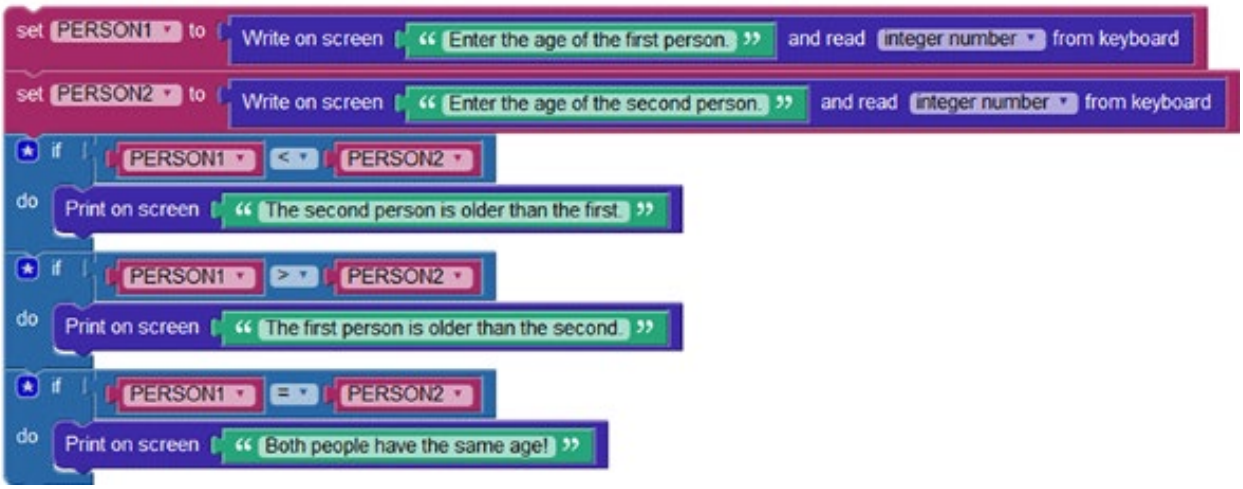

Рисунок 4.2 – Приклда створення блокової програми

Програма Blockly (рис.4.2) являє собою програму, яка визначає, хто з двох людей є старшим. Програма просить користувача ввести вік першої особи та зберігає її у змінній під назвою PERSON1. Потім програма запитує вік другої людини і зберігає її в іншій змінній під назвою PERSON2. Нарешті, програма порівнює вік двох осіб та виводить на екран, хто старший з двох осіб. На рис.4.3 показано еквівалентний код Python.

```
PERSON1 = None
PERSON2 = None


PERSON1 = int(input ('Enter the age of the first person.'))
PERSON2 = int(input ('Enter the age of the second person.'))
if PERSON1 < PERSON2:
  print('The second person is older than the first.')
if PERSON1 > PERSON2:
  print('The first person is older than the second.')
if PERSON1 == PERSON2:
  print('Both people have the same age!')
```

Рисунок 4.3 – Еквівалентний код Python

Цикли FOR. Цикли FOR використовуються для повторення виконання певного блоку коду протягом певної кількості разів. Цикл FOR зазвичай використовують тоді, коли кількість ітерацій відома заздалегідь.

Програма Blockly, показана на рис.4.4, використовує цикл FOR для визначення всіх простих чисел між 1 і 100. Прості числа - це числа, які діляться лише на 1 або самі на себе, і виводяться на екран, тоді як складні числа ігноруються.

Програма використовує два цикли FOR, щоб обчислити остачу ділення між числом, яке перевіряється та числами від 2 та до самого числа. Якщо залишок цього ділення дорівнює нулю, число не є простим. Якщо жодна операція ділення не дає залишку, що рівний нулю, то число є простим. Програма виводить прості числа на екрані і переходить до перевірки наступного числа, зупиняючись лише тоді, коли вона дійде до 100.

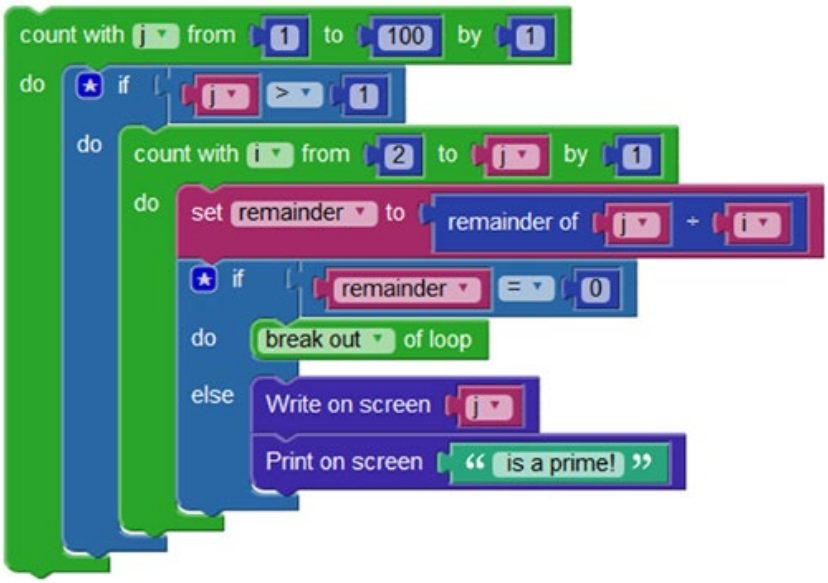

Рисунок 4.4 – Приклад створення блокчейн програми для вибірки від 1 до 100

Цикли WHILE. Цикли WHILE використовуються для виконання блоку коду поки умова істинна. Код в циклі WHILE часто модифікує змінні та об'єкти, щоб зрештою зробити вираз помилковим. Якщо вираз, перевірений WHILE, ніколи не стає хибним, цикл працює вічно і називається нескінченним циклом. Нескінченні цикли зазвичай небажані в програмуванні.

Програма Blockly, показана на рис.4.5, використовує цикл WHILE для створення гри-вгадування. Програма вибирає випадкове число від 1 до 10 і просить користувача відгадати його. Програма продовжуватиме просити відповідь допоки користувач не вгадає правильне число [21].

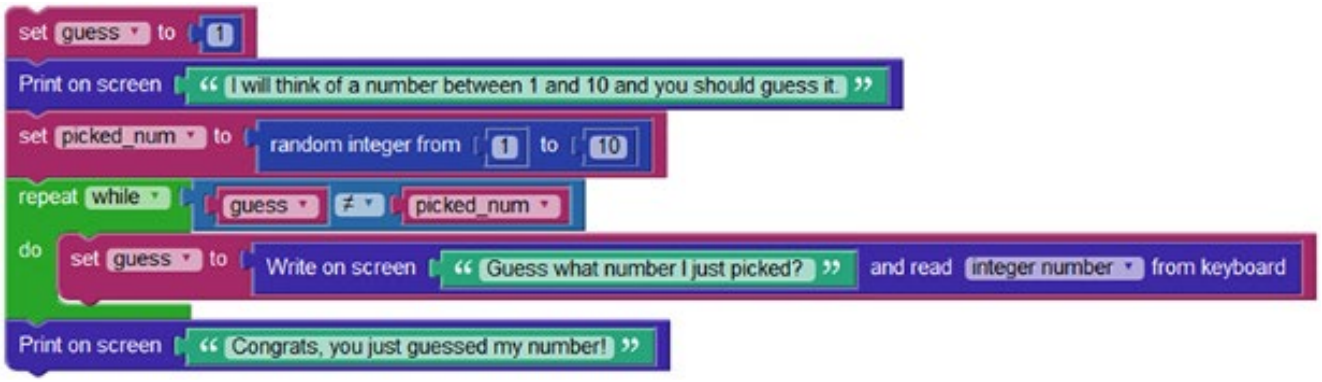

Рисунок 4.5 – Приклда створення гри в Blockly

4.3 Використання Blockly для роботи з Python

Python - дуже популярна мова програмування, яка розроблена для простого читання та написання. Спільнота розробників Python, збільшує корисність мови, створюючи різні типи модулів, і роблячи їх доступними для інших програмістів.

Не дилячись на те що Python був розроблений щоб бути легким, все одно залишається крива навчання. Для того щоб зробити вивчення Python легше, можна використовувати Blockly для кращого розуміння коду на Python.

Хоча різні мови програмування мають різну семантику та синтаксис, всі вони мають однакову логіку програмування. Початківці можуть використовувати Blockly, щоб легко сторювати незалежну від мови програму, та експортувати її код на мову Python і використовувати цей створений код, для того щоб краще розібратися в синтаксисі, структурі і семантиці мови програмування Python [22].

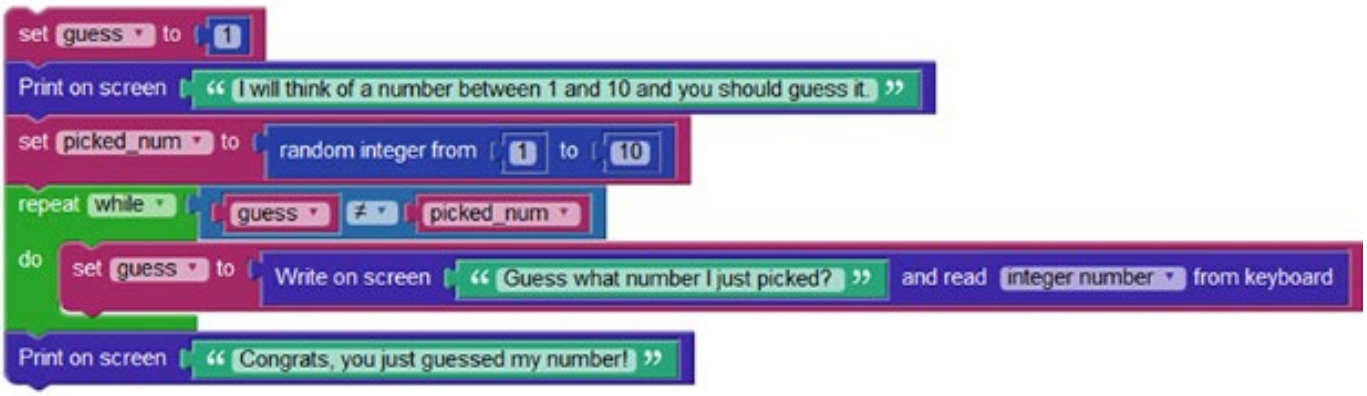

Рисунок 4.6 – Приклад гри вгадування в Blockly

На рис.4.6 та 4.7 показано гру на вгадування написану на мові Blockly та мові Python відповідно.

```
import random

guess = None
picked_num = None


guess = 1
print('I will think of a number between 1 and 10 and you should guess it.')
picked_num = random.randint(1, 10)
while guess != picked_num:
  guess = int(input ('Guess what number I just picked? '))
print('Congrats, you just guessed my number!')
```

Рисунок 4.7 – Експертований код в Python

4.4 Інтерпритатор Python

Python це інтерпритуєма мова програмування; це означає що для аналізу та виконання коду на Python, потрібен інтерпритатор. Інтерпритатор Python, розуміє та виконує код написаний на мові Python. Той факт, що код Python можна створити в будь-якому текстовому редакторі та що інтерпретатори Python доступні для багатьох операційних систем, дозволяє розробникам Python створювати та розгортати програми Python практично в будь-якій операційній системі. Сторонні інструменти, такі якPy2exe і Pyinstaller, можуть використовуватися для упакування вихідного коду на Python, у виконуваний код, така процедура усуває необхідність в інтерпритаторі Python, при запуску програмного кода на Python.

В операційній системі Linux інтерпретатор Python зазвичай встановлюється в /usr/bin/python або /usr/bin/python3(залежно від доступних версій Python у системі). З новим інсталятором Windows, Python стандартно встановлений у домашній каталог користувача На старіших версіях Windows, Python розміщується в директиві C:\PythonXX(де XX версія Python). Після того як інтерпритатор був

встановлений, він працює схоже як оболонка Linux. Це означає, що при виклику без аргументів він читає та виконує команди інтерактивно; коли він викликається аргументом імені файлу або файлом як стандартним входом, він зчитує та виконує сценарій із цього файлу.

Для того, щоб запустити інтерпретатор, просто можна ввести python або python3 в командному рядку. Деякі застарілі системи досі працюють на на більш ранній версії Python 2, але багато нових систем переходять до використання нової версії Python 3. Версія Python виводиться в першому рядку при запуску інтерпретатора (рис.4.8.).

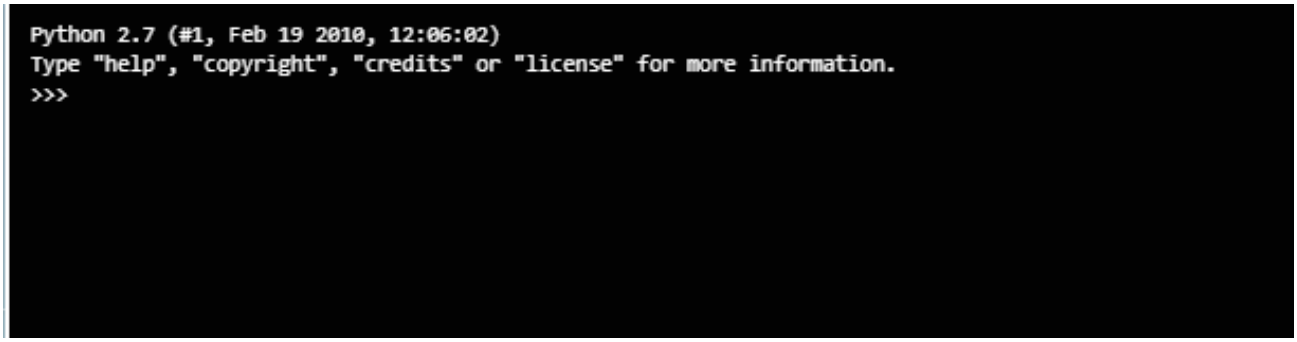

Рисунок 4.8 – Привітальне повідомлення інтерпритатора Python, перша варіація

Коли інтерпритатор викликається без аргументів, а команди вводяться з клавіатури, інтерпритатор знаходиться в інтерактивному режимі. В цьому режимі інтерпритатор очікує команди. Первинне запрошення представлено трьома знаками більше (>>>). Лінії продовження представлені трьома крапками (...). Продовження - це вторинне запрошення за замовчуванням.

В підказці >>> вказано що інтерпритатор готовий та очікує команди.

Рядки продовження необхідні при введенні коду який складається з декількох рядків. На рис.4.9 зображено як конструкція IF-THEN пишеться на Python.

Інший спосіб використання інтерпретатора є python -c command [arg] ..., яка виконує інструкцію(ї), вказану(і) в команді. Оскільки оператори Python часто містять пропуски, та інші символи характерні для оболонки, то пропонується заключати всю команду між одинарними лапками [23].

```
>>>
>>> the_world_is_flat = True
>>> if the_world_is_flat:
...     print "Be careful not to fall off!"
...
Be careful not to fall off!
```

Рисунок 4.9 – Привітальне повідомлення інтерпритатора Python, друга варіація

4.5 Змінні та основні вирази в Python

Інтерпритатор приймає і виконує операції інтерактивно. Інтерпритатор працює як простий калькулятор. Можна ввести вираз в ньому та записати значення. Синтаксис виразів простий. Оператори +, -, * і / працюють так само, як і в багатьох інших мовах програмування (наприклад Pascal або C). Дужки (()) можуть бути використані для групування, як показано на рисунку 1.

Інтерактивний режим Python реалізує спеціальну зміну "_", для того щоб запам'ятовувати результат останнього виразу, як показано на рисунку 2.

Змінні - це мітки областей пам'яті, які використовуються для зберігання данних в процесі виконання програми. Щоб присвоїти значення змінним в Python, використовуйте знак рівності (=). Жодний результат не відображається до наступної інтерактивної підказки. Спроби використовувати невизначену змінну (без присвоєного значення), призведуть до помилки.

Рядки, визначені як послідовність символів, також можна обробляти в інтерактивному режимі. Можна використовувати символ зворотнього слешу (\) для використання керуючих символів. В якості прикладу рядок використовує подвійні лапки, але повинна містити подвійні лапки в середині рядка. Якщо рядок набраний наступним чином: "Мені дійсно "потрібно" це.". Python заплутається і подумає, що перші подвійні лапки, в середині рядку фактично завершує рядок. Якщо розміщується зворотній слеш (\) перед подвійними лапками всередині рядку

наступним чином: "I really \"need\" this", тоді Python ігнорує наступний символ після зворотнього слешу (\)[21].

Одинарні лапки чи подвійні, можуть використовуватися для обертання рядку. Функція print, виводить на екран результат виразу який був отриманий. Вона відрізняється від простого напису виразу (як це було в прикладах з калькулятором раніше) в тому що він оброблює декілька виразів і рядків. Рядки друкуються без лапок, і між елементами є відступ, так щоможна гарно відформатувати їх.

Функції є важливою частиною багатьох мов програмування. Функції дозволяють дати блоку коду ім'я та повторно використовувати його коли потрібно.

Корисні функції та типи даних в Python. Python підтримує дуже багато корисних функцій та типів даних. Ось деякі найважливіші з них. Функція range() генерує послідовність чисел, зазвичай використовуваних для ітерації з циклами.

range(stop)- число цілих чисел для генерації, починаючи з нуля.

range([start], stop[, step] - перше число послідовності, число, до якого генеруватиметься послідовність, не включаючи це число, різниця між кожним числом у послідовності.

Кортежі. Кортеж (turple) являє собою послідовність, незмінних об'єктів в Python. Кортежі представляють собою послідовності розділені дужками.

Списки. Списки (lists) представляють собою послідовності змінюваних об'єктів в Python. Списки можуть бути створені шляхом розміщення різних значень розділених комами, поміж квадратних дужок.

Набори. Набори (sets) преставляють собою невпорядковані колекції унікальних елементів. Звичайне використання включає тестування членства, видалення дублікатів із послідовності, і обчислення стандартних математичних операцій над наборами такими як перетин, об'єднання, різниця, та математична симетрія.

Словник. Словник - це список елементів розділений комами. Кожен елемент якого представляє собою комбінацію значення і унікального ключа. Кожен ключ відділений від значення двокрапкою. Весь словник записано у фігурних дужках. Елементи словника, можуть бути доступними, зміненими, чи видаленими. Існує

також багато вбудованих функцій словника, таких як функція порівняння елементів в різних словниках, функція яка підраховує кількість елементів в словнику [22].

4.6 Імпорт модулів у користувацький код

Вихід із інтерпретатора Python і введення його знову видаляє всі попередні визначення, функції та змінні будуть втрачені. Через це краще використовувати текстовий редактор, створюючи більш довгу або складну програму. Коли програма готова, просто можна перенести її у інтерпретатор для виконання. Використання текстового редактора таким чином створює тип програми, що називається скриптом. Оскільки програми стають довшими та складнішими, можливо, буде необхідно розбити їх на більш дрібні програми для легшого обслуговування. При такому підході програміст створив би основну програму, яка імпортує всі необхідні менші програми. Ці менші програми називаються модулями.

Навіть якщо програма не є довгою або складною, програміст може все-таки скористатися імпортом існуючого модуля в свою програму. Програміст може повторно скористатися необхідною функцією, що була написана раніше. Інший поширений сценарій: програмісти імпортують модулі, написані кимось іншим, щоб розширити функціональність свого коду або покращити програму У Python є велика і дуже активна спільнота розробників, які вже написали та поділилися модулями для багатьох завдань.

Поширені модулі, які часто зустрічаються в IoT включають: модулі обробки даних (математичні функції), модулі самоперевірки (рівень батареї та тест пам'яті), модулі захисту (хеші, шифрування/дешифрування) та мережеві серверні та клієнтські модулі (Веб, FTP, NTP тощо).

IF-THEN в Python. Подібно до інших мов програмування Python, підтримує структуру IF-THEN (якщо - то). Структура IF-THEN може використовуватися для того, щоб дозволити коду приймати рішення на основі результату виразу [23].

Код виконує декілька тестів і виводить повідомлення в залежності від результату тесту. Python також реалізує дві структури з іменем ELSE (інакше) і ELIF (інакше - якщо). ELSE дозволяє програмісту вказувати інструкції які повинні виконуватися якщо вираз хибний. Структури ELSE IF, ELIF виконуються в тому випадку якщо перший вираз хибний, і потрібно перевірити іншу умову. Там може бути нуль чи більше блоків ELIF, і частина ELSE є необов'язковою.

```
>>>
>>> x = int(input("Please enter an integer: "))
Please enter an integer: 42
>>> if x < 0:
...     x = 0
...     print ('Negative changed to zero‘)
... elif x == 0:
...     print ('Zero‘)
... elif x == 1:
...     print ('Single‘)
... else:
...     print ('More‘)
...
More
```

Рисунок 4.10 – Приклад if-then, else if та elif

Цикли FOR в Python. Цикл FOR в Python виконує ітерацію елементів будь-якої послідовності (списку чи рядку) [24] в порядку їх розташування в послідовності, як показано на рис.4.11

```
>>>
>>> # Measure some strings:
... words = ['cat', 'window', 'defenestrate']
>>> for w in words:
...     print (w, len(w))
...
cat 3
window 6
defenestrate 12
```

Рисунок 4.11 – Приклад циклу FOR

Цикли Loops в Python. Цикл WHILE виконує блок коду якщо умова повертає істину. Програма, наведена в рис.4.12, використовує цикл WHILE для обчислення та друку початкової підпослідовності ряду Фібоначчі, як показано на малюнку.

На третьому рядку використовується оператор множинного присвоювання. Змінні a і b отримують нові значення 0 і 1, в одному виразі.

Цикл WHILE обчислює наступне число Фібоначі, тоді коли умова b < 10 істинна. Як і в мові C, Python приймає будь яке ненульове ціле число як true (істина), і нуль як false (хибне). Тест використаний на рисунку представляє собою просте порівняння. Тіло циклу має відступ від краю. Відступ - це спосіб групування операторів Python [25].

В інтерактивному режимі, необхідно ввести табуляцію чи пробіл, для кожного рядку з відступом. Більш важке введення для Python повинно виконуватися за допомогою текстового редактору. Коли складений оператор вводиться в інтерактивному режимі, за ним повинний йти пустий рядок для вказання завершення (тому що парсер не може вгадати який рядок буде останнім). Кожний рядок в базовому блоці повинна мати відступи на одне і те саме значення.

```
>>>
>>> # Fibonacci series:
... # the sum of two elements defines the next
... a, b = 0, 1
>>> while b < 10:
...     print (b)
...     a, b = b, a+b
...
1
1
2
3
5
8
```

Рисунок 4.12 – Приклад циклу WHILE

4.7 Розробка моделі управління пристроями IoT з використанням мікроконтролера та комбінації Blockly та Python

Використовуючи візуальну мову програмування, необхідно зробити так, щоб пристрої IoT реагувати на сигнали від перемикача. Для цього, спочатку необхідно

створи комбінацію пристроїв, що працюватимуть так: при ввімкненні перемикача вмикається лампа та вентилятор [26]. Тепер необхідно додати всі необхідні пристрої до проекту в Cisco Packet Tracer.

1) Щоб знайти лампу та вентилятор, необхідно звернутися до [End devices] - [Home] (рис.4.13).

2) Щоб знайти плату MCU, необхідно звернутися до [Components] - [Boards] (рис.4.14).

3) Щоб знайти Rocker Switch, необхідно звернутися до [Components] - [Sensors] (рис.4.15).

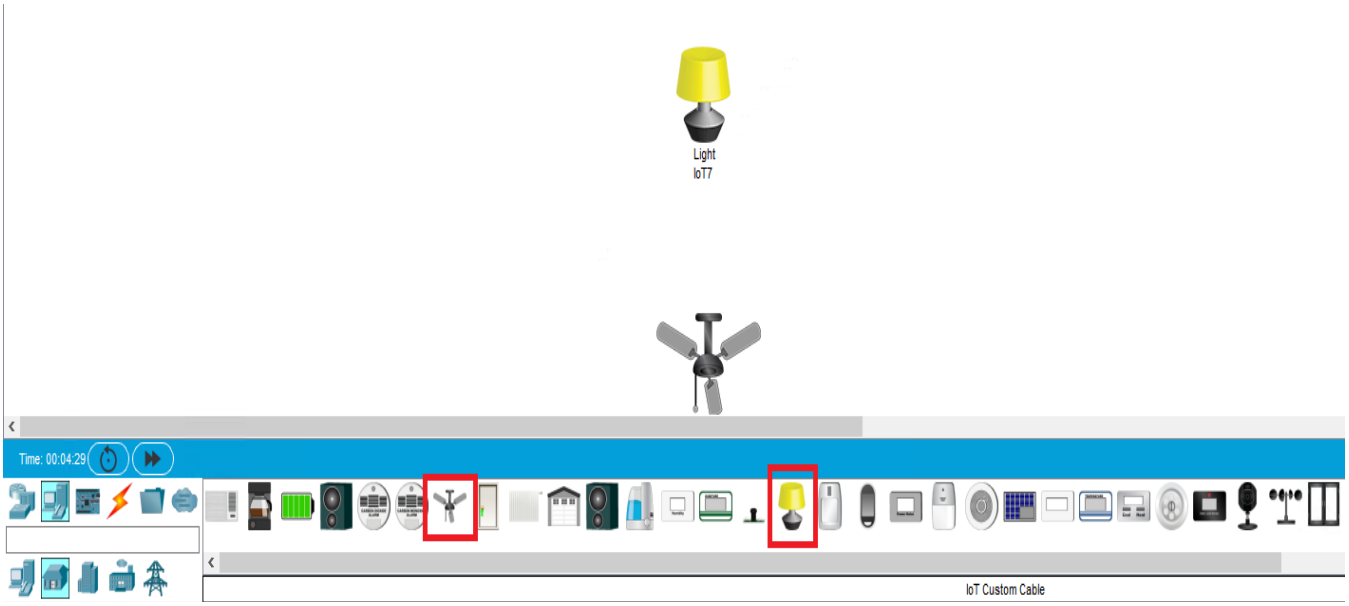

Рисунок 4.13 - IoT девайси в Cisco Packet Tracer

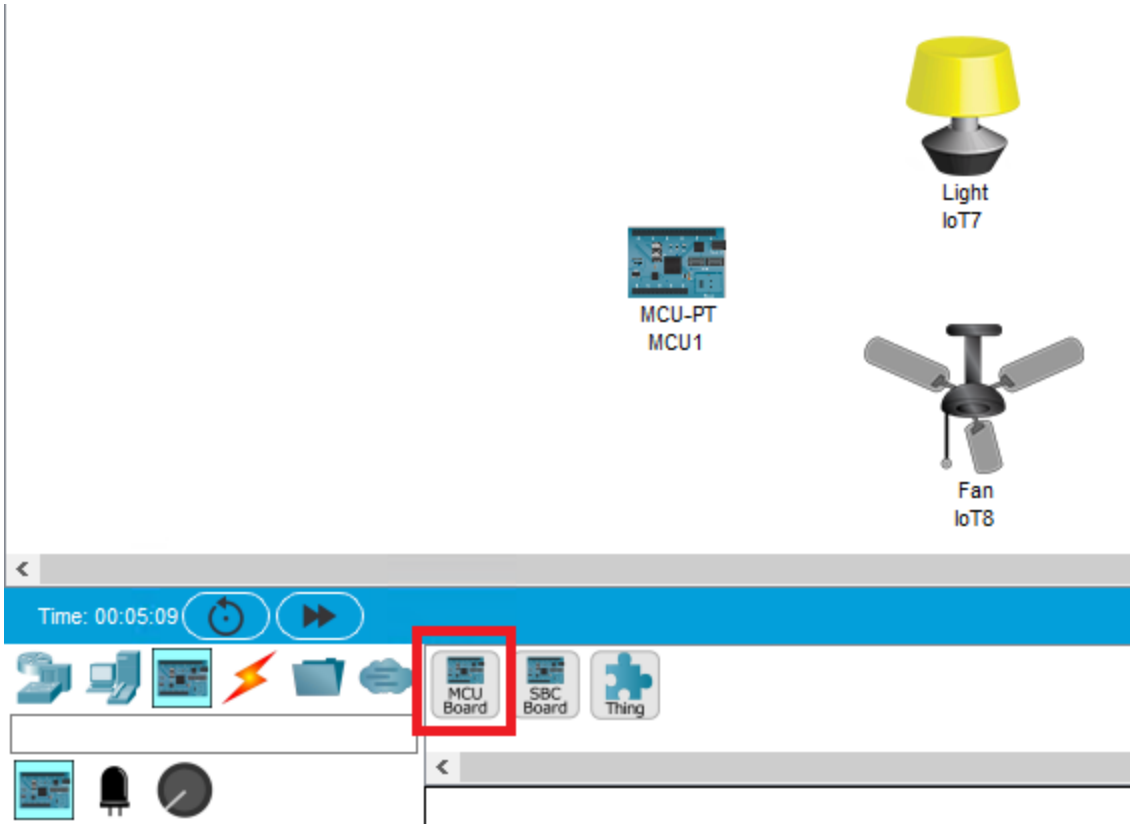

Рисунок 4.14 – Плата мікроконтролера

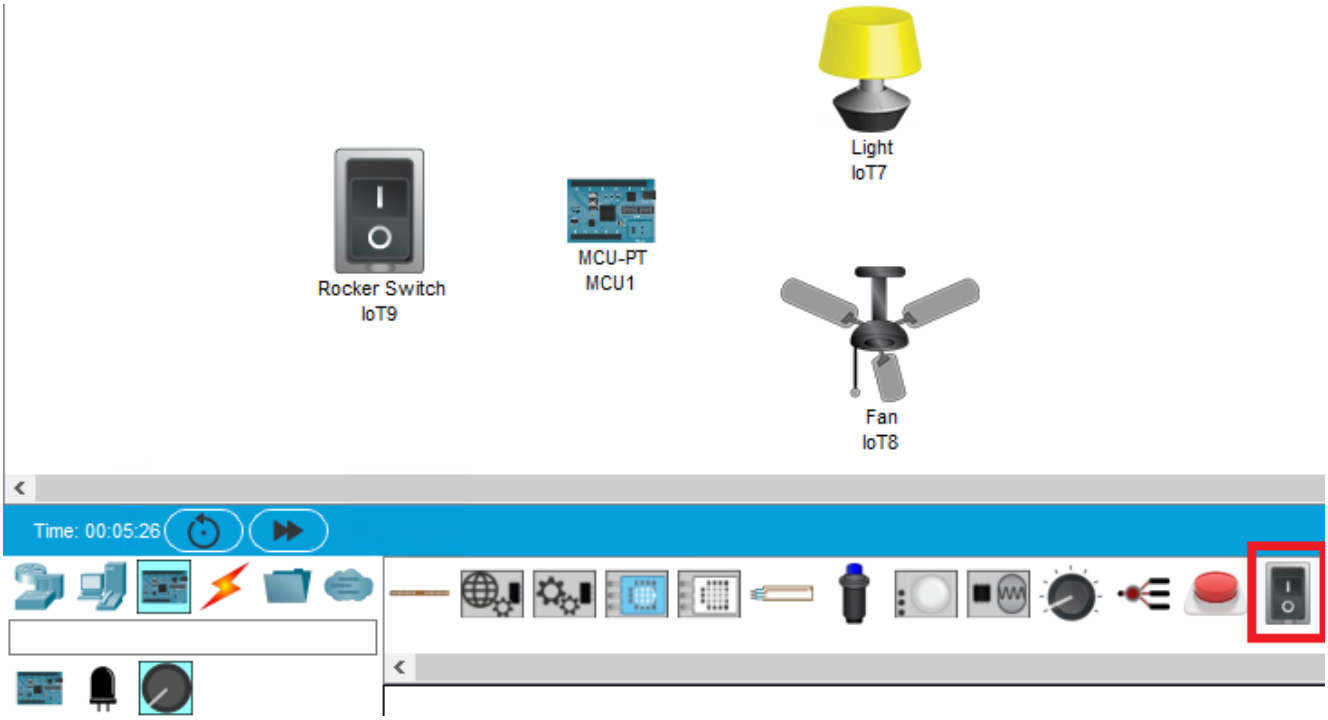

Рисунок 4.15 – Rocker Switch

Тепер необхідно підключити обрані пристрої за допомогою IoT Custom Cable [27]. Також потрібно запам'ятати номер порту, до якого будуть підключені кожен пристрій. Перевірити їх можна наведенням курсору миші на кільця на кабелі.

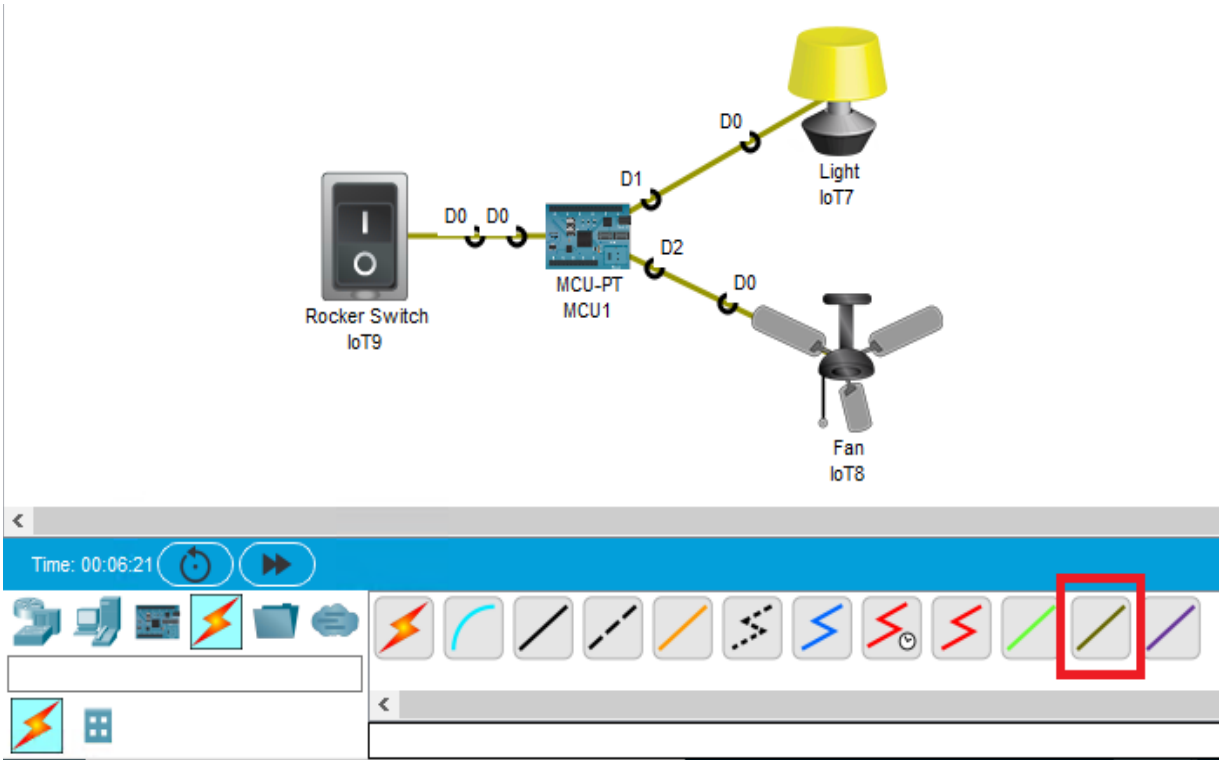

Рисунок 4.16 – Підключенні девайси

Наступним етапом є програмування мікроконтролера MCU [28]. Спочатку потрібно видалити скрипт за замовчуванням. Алгоритм наступний:

1) Клікнути лівою кнопкою миші на платі.
2) Перейти на вкладку Programming.
3) Обрати дефолтний скрипт.
4) Видалити його.

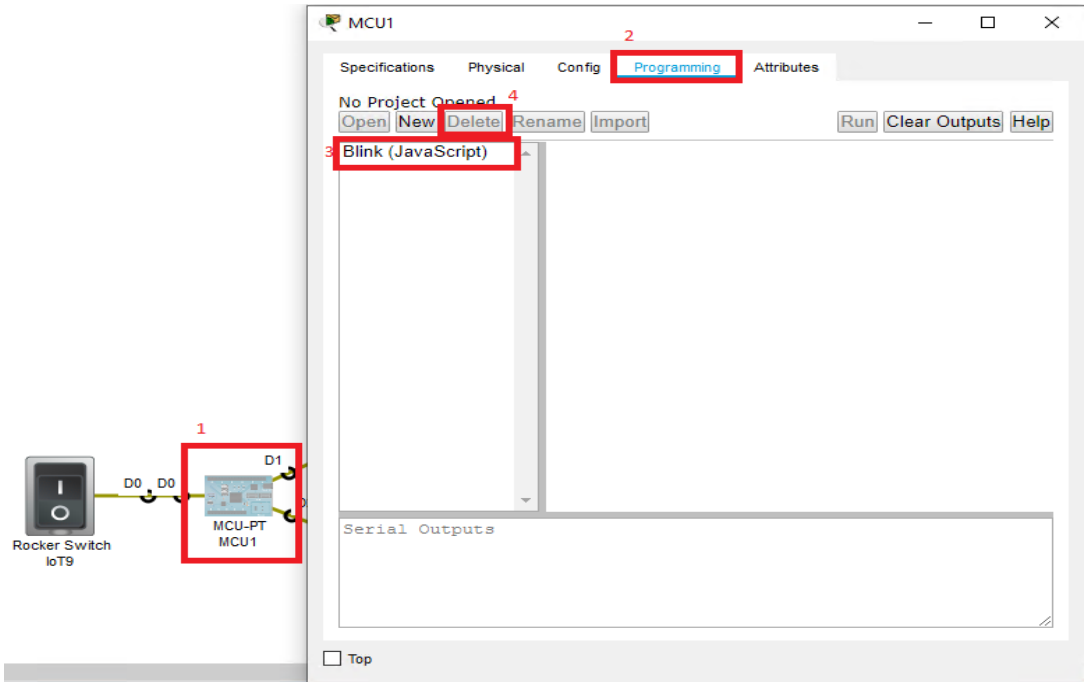

Рисунок 4.17 – Видалення скрипта за замовчуванням

Після цього потрібно створити новий скрипт [29]. Натиснувши кнопку Add, обирається назва проекта та обирається «Empty – Visual», оскільки в цій роботі планується програмування плати за допомогою візуальної мови програування.

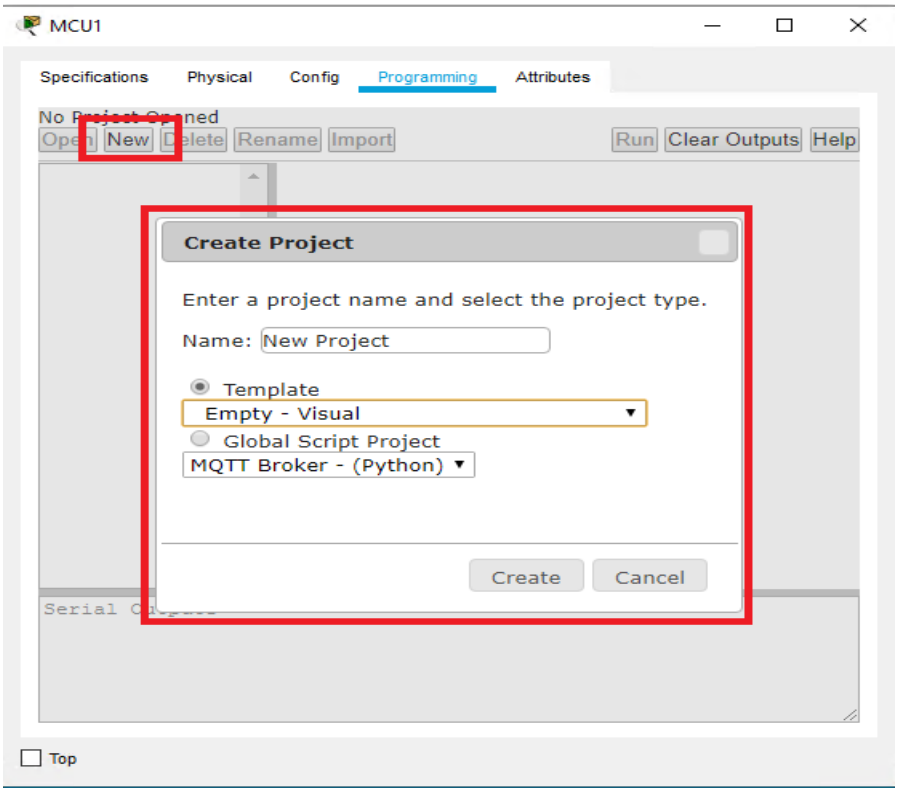

Рисунок 4.18 – Створення нового проекту

Після того, як новий проект створено, можна відкрити його та перейти до файлу в ньому [30].

1) Потрібно додати функцію. Для цього необхідно перейти до [Program] - [Functions]. І перетягнути наступний елемент до проекту.

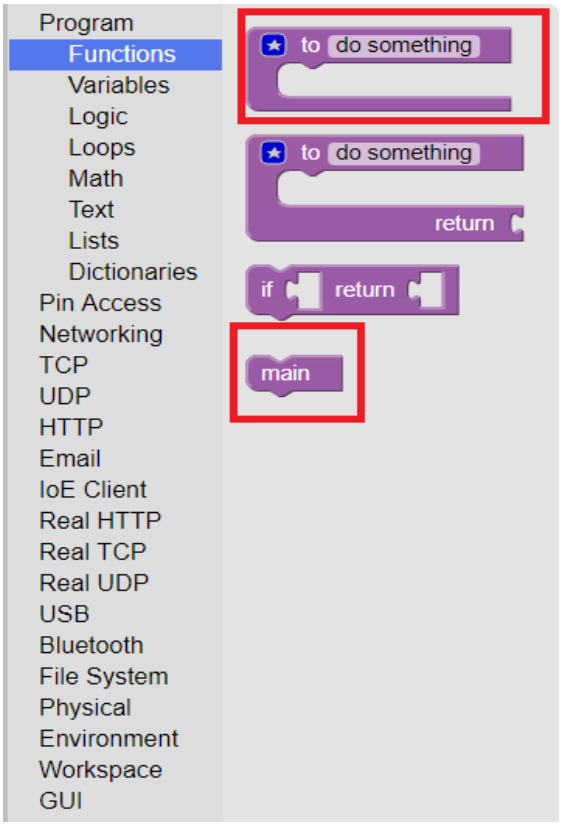

Рисунок 4.19 – Функції

2) Тепер, щоб мати можливість читати вхід з перемикача та надсилати інформацію до лампи та вентилятора, потрібно додати наступні елементи:

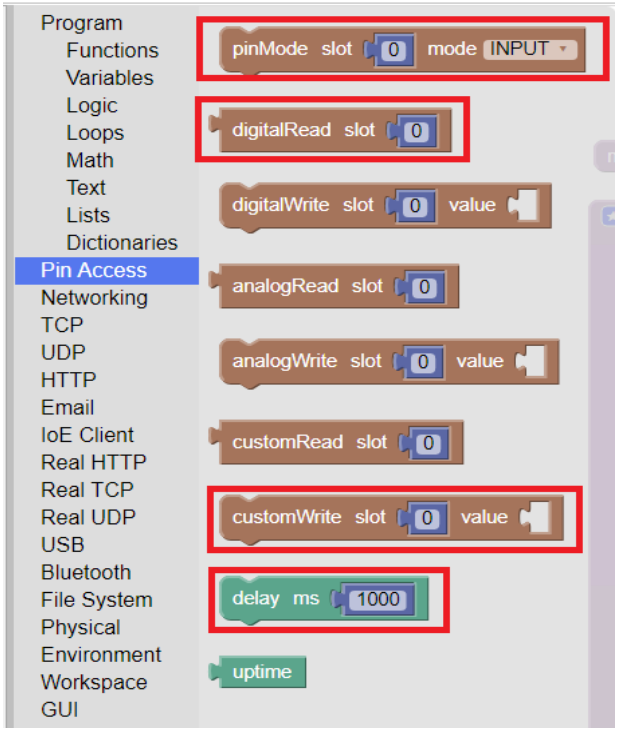

Рисунок 4.20 – Pin Access

3) Тепер, щоб зробити цикл у програмі, щоб вона могла постійно читати інформацію, що надходить з перемикача, потрібно додати компонент циклу:

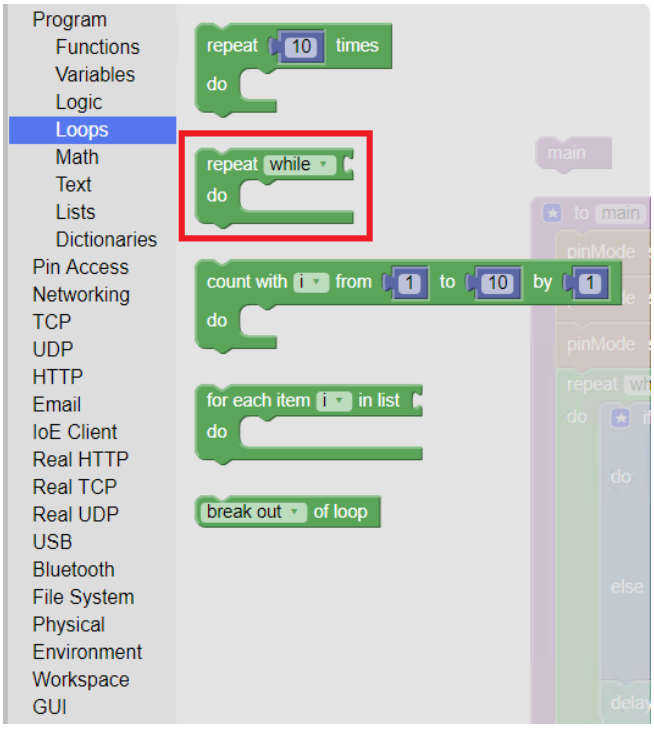

Рисунок 4.21 – Цикли

4) І щоб додати функцію для порівняння інформації яка надходить, необхідно додати наступні елементи до проекту:

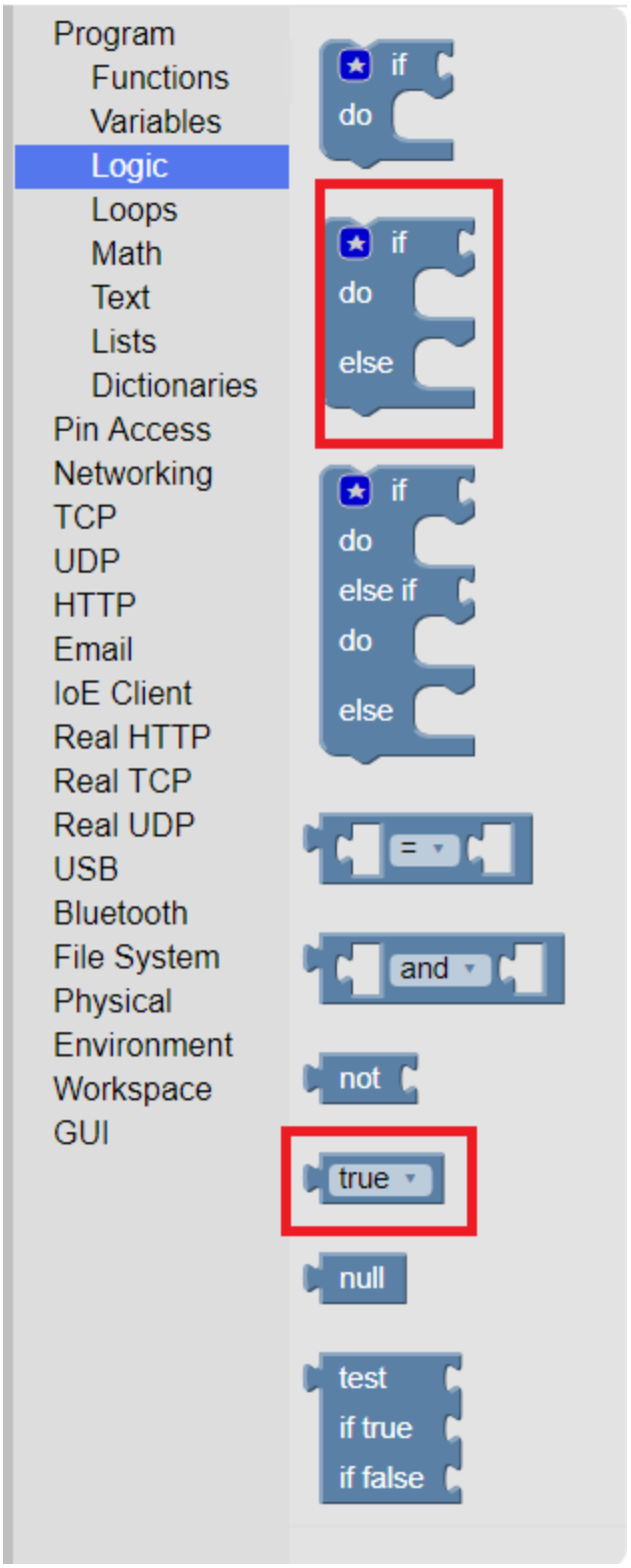

Рисунок 4.22 – Logic

Тепер, коли є всі елементи, потрібно створити логіку для програми, та можна почати їх поєднувати [31].

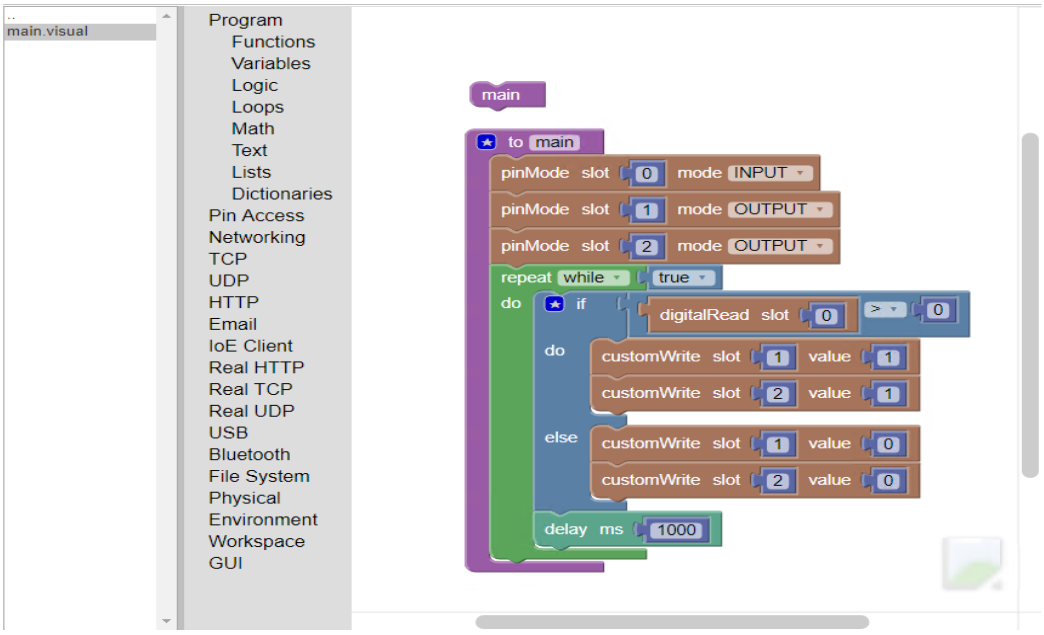

Рисунок 4.23 – Фінальний результат

Як результат, було:

1) Створено функцію і названо її «main».

2) Передано платі, що вона повинна зчитувати дані з контакту під номером 0, а також надсилати інформацію до контактів 1 і 2. Це ті порти, які запам’ятали раніше в цій роботі, коли було підключено пристрої кабелями.

3) Для постійного отримування інформації від перемикача використовується функцію Repeat.

4) Далі додано було оператор для перевірки сигналу від комутатора. Якщо сигнал> 0, це означає, що перемикач увімкнено.

5) Якщо це так, то надсилається значення = 1 на лампу та вентилятор, щоб увімкнути їх. В іншому випадку надсилається 0, щоб вимкнути їх [32].

Тепер можна перевірити, чи працює програма належним чином.

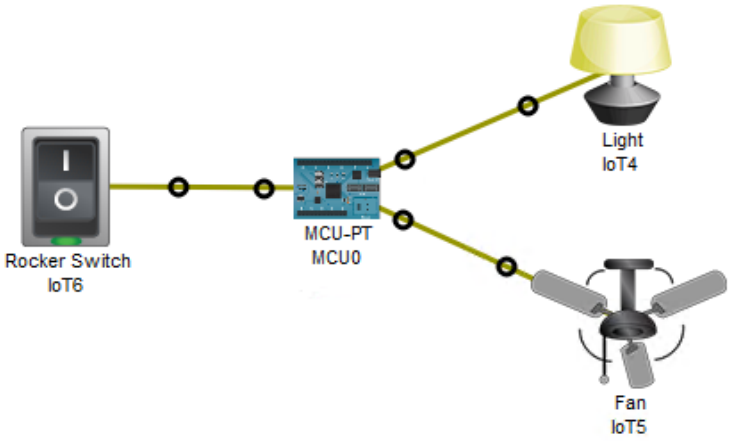

Рисунок 4.24 – Перевірка результату

Висновки до четвертого розділу

Зазначено, що Python дуже проста у використанні, потужна та універсальна мова, що обирається багатьма розробниками IoT. Однією з основних причин популярності Python є спільнота розробників. Розробники Python створили та надавали доступ до багатьох спеціальних модулів, які можна імпортувати до будь-якої програми, щоб негайно запозичити доданий функціонал.

Досліджено мову візуального програмування Blockly, яка дозволяє користувачам створювати програми, об'єднуючи блоки, які представляють різні логічні структури мови, а не написанням фактичного коду. Blockly працює в веб-браузері та може відображати візуально створену програму в Python

Розроблено модель управління пристроями IoT з використанням мікроконтролера та комбінації мов Blockly та Python.

ВИСНОВКИ

У ході виконання бакалаврської роботи отримано наступні теоретичні та науково-практичні результати:

1. Досліджено основні особливості побудови та функціонування архітектури технології Інтернет Речей (IoT) та підкреслено необхідність широкого дослідження особливостей даної технології

2. Проаналізовано особливості створення та розповсюдження технології IoT. Описано принцип роботи системи «Розумний» будинок при умові впровадження мікроконтролера Arduino, а також приведено головні переваги даного мікроконтролера.

3. Приведено особливості впровадження та використання базових датчикі та сенсорів, та підкреслено, що їх призначення полягає у реагуванні на вхідні фізичні властивості та перетворенні його в електричний сигнал, сумісний з електронними схемами. Датчики – це електронні пристрої, які вимірюють фізичну якість, таку як світло або температура, і перетворюють її у напругу.

4. Зазначено, що система керується мікроконтролером, що в більшості випадків виконує функцію центрального процесора, який з'єднаний з портативним блоком плат.

5. Підкреслено, що мікроконтролер також може бути підключеним до побутових електроприладів за допомогою електричних реле та деяких інших різних елементів, таких як двигуни та дальні датчики. Центральний мікроконтролер слідує різним алгоритмам для збору інформації, порівняння та прийняття рішень, показує результати та контролює прилади.

6. Виокремлено, що після підключення, мікроконтролер готовий показувати стан електричних навантажень та різні результати, а також приймати команду від клієнта для управління приладами.

7. Досліджено різні типові для «розумного» будинку алгоритми, які необхідні для налаштування головних датчиків та сенсорів. Дотримуючись алгоритмів користувач може здійснювати управління кількома побутовими

електроприладами одночасно. Однак кількість контрольованих приладів буде залежати від кількості вихідних портів центрального або головного мікроконтролера.

8. Зазначено, що Python дуже проста у використанні, потужна та універсальна мова, що обирається багатьма розробниками IoT. Однією з основних причин популярності Python є спільнота розробників. Розробники Python створили та надавали доступ до багатьох спеціальних модулів, які можна імпортувати до будь-якої програми, щоб негайно запозичити доданий функціонал.

9. Досліджено мову візуального програмування Blockly, яка дозволяє користувачам створювати програми, об'єднуючи блоки, які представляють різні логічні структури мови, а не написанням фактичного коду. Blockly працює в веб-браузері та може відображати візуально створену програму в Python

10. Розроблено модель управління пристроями IoT з використанням мікроконтролера та комбінації мов Blockly та Python.

ПЕРЕЛІК ПОСИЛАНЬ

1. Evans, D. (2011). The internet of things: How the next evolution of the internet is changing everything. CISCO white paper, 1(2011), 1-11.

2. Buyya, R., Broberg, J., & Goscinski, A. M. (Eds.). (2010). Cloud computing: Principles and paradigms (Vol. 87). John Wiley & Sons.

3. Xiong, G., Chen, C., Kishore, S., & Yener, A. (2011, January). Smart (in-home) power scheduling for demand response on the smart grid. In ISGT 2011 (pp. 1-7). IEEE.

4. Bharath,S., Pasha, M.Y., & Deepth, J.(2017, April). IoT-Home Automation. International Journal of Computer Technology and Research, 5, 4-6.

5. Li, B., & Yu, J. (2011). Research and application on the smart home based on component technologies and Internet of Things. Procedia Engineering, 15, 2087-2092.

6. Kim, J. (2016). HEMS (home energy management system) base on the IoT smart home. Contemporary Engineering Sciences, 9(1), 21-28.

7. Kodali, R. K., Jain, V., Bose, S., & Boppana, L. (2016, April). IoT based smart security and home automation system. In 2016 international conference on computing, communication and automation (ICCCA) (pp. 1286-1289). IEEE.

8. Pan, J., Jain, R., Paul, S., Vu, T., Saifullah, A., & Sha, M. (2015). An internet of things framework for smart energy in buildings: designs, prototype, and experiments. IEEE Internet of Things Journal, 2(6), 527-537.

9. Jain, S., Kumar, V., Paventhan, A., Chinnaiyan, V. K., Arnachalam, V., & Pradish, M. (2014, March). Survey on smart grid technologies-smart metering, IoT and EMS. In 2014 IEEE Students' Conference on Electrical, Electronics and Computer science (pp. 1-6). IEEE.

10. Aarons Creek Farms. 2012. Greenhouse Buying Guide. Available: http://www.littlegreen- house.com/guide.shtml. Accessed: 26 April 2016.

11. Arduino http://www.arduino.cc/en/Guide/Introduction.

12. Banzi, M. 2011. Arduino – Getting Started with Arduino. Maker Media, Inc. Banzi, M. 2012. Available: http://www.ted.com/talks/massimo_banzi_how_arduino_is_open_sourcing_imagination

13. Evans, B. 2011. Beginning Arduino Programming. 2-3. New York: Apress.

14. Fraden, J. 2010. Handbook of Modern Sensors: Physics, Designs, and Applications. Pringer Science & Business Media.

15. Fritzing. 2013. Fritzing: a tool for advancing electronic prototyping for designers. Available: http://fritz- ing.org/home/.

16. Frueh, A. 2012. Soil Moisture Sensor. Available: http://gardenbot.org/howTo/soilMoisture/how-to_moisture-sensor_big.png.

17. Karvinen, T. & Karvinen, K. 2014. Getting Started with Sensors: Measure the World with Electronics, Arduino, and Raspberry Pi. Maker Media, Inc.

18. SparkFun Electronics – DHT22. 2010. Digital-output relative humidity & temperature sensor/module DHT22 (DHT22 also named as AM2302). Available: https://www.sparkfun.com/datasheets/Sen- sors/Temperature/DHT22.pdf.

19. Timmerman, G. J. & Kamp, P. G. H. 2003. Computerized Environmental Control in Greenhouses, PTC.

20. Zhao, Z., Lee, W. C., Shin, Y., & Song, K. B. (2013). An optimal power scheduling method applied in home energy management system based on demand response. Etri Journal, 35(4), 677-686.

21. Haque, K. F., Saqib, N., & Rahman, M. S. (2019, March). An Optimized Stand-alone Green Hybrid Grid System for an Offshore Island, Saint Martin, Bangladesh. In 2019 International Conference on Energy and Power Engineering (ICEPE) (pp. 1-5). IEEE.

22. Singh, R. S. S., Ibrahim, A. F. T., Salim, S. I. M., & Chiew, W. Y. (2009, November). Door sensors for automatic light switching system. In 2009 Third UKSim European Symposium on Computer Modeling and Simulation (pp. 574-578). IEEE.

23. Vaghela, M., Shah, H., Jayswal, H., & Patel, H. (2017). Arduino based auto street light intensity controller. Invention Rapid: Embedded Systems, 2013(3), 1-4.

24. Jewel, M. H., Islam, M. N., & Hasan, M. J. (2017). Automatic Room Light Control Using Bidirectional Visitor Counter and Gas Detection (Doctoral dissertation, East West University).

25. Pavithra, D., & Balakrishnan, R. (2015, April). IoT based monitoring and control system for home automation. In 2015 global conference on communication technologies (GCCT)(pp. 169-173). IEEE.

26. Khadem, T., Billah, S. B., Barua, S., & Hossain, M. S. (2017, September). Homer based hydrogen fuel cell system design for irrigation in Bangladesh. In 2017 4th International Conference on Advances in Electrical Engineering (ICAEE)(pp. 445-449). IEEE.

27. Adhya, S., Saha, D., Das, A., Jana, J., & Saha, H. (2016, January). An IoT based smart solar photovoltaic remote monitoring and control unit. In 2016 2nd international conference on control, instrumentation, energy & communication (CIEC) (pp. 432-436). IEEE.

28. Khan, M., Silva, B. N., & Han, K. (2016). Internet of things based energy aware smart home control system. IEEE Access, 4, 7556-7566.

29. Bedi, G., Venayagamoorthy, G. K., Singh, R., Brooks, R. R., & Wang, K. C. (2018). Review of internet of things (IoT) in electric power and energy systems. IEEE Internet of Things Journal, 5(2), 847-870.

30. Adila, A. S., Husam, A., & Husi, G. (2018, April). Towards the self-powered Internet of Things (IoT) by energy harvesting: Trends and technologies for green IoT. In 2018 2nd International Symposium on Small-scale Intelligent Manufacturing Systems (SIMS) (pp. 1-5). IEEE.

31. Kamilaris, A., "Enabling smart homes using web technologies," Ph.D. dissertation, Dept. of computer science, University of Cyprus, Cyprus, 2012.

32. Bergman E. Information appliances and beyond. Interaction design for consumer products. Morgan Kaufmann, pp.2-10, 2000.

33. Augusto, J.C., and Nugent, C.D. Smart homes can be smarter. Springer Berlin Heidelberg, pp. 5-12, 2006.

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Бакалаврська робота

на тему:

«РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ УПРАВЛІННЯ IOT ПРИСТРОЯМИ З ВИКОРИСТАННЯМ МІКРОКОНТРОЛЕРА ЗА ДОПОМОГОЮ МОВИ ПРОГРАМУВАННЯ PYTHON»

Виконав:
студент групи ПД-44
Гуленко В.С.
Науковий керівник: к.т.н., доцент
Жебка В.В.

1

ОБ'ЄКТ, ПРЕДМЕТ ТА МЕТА ДОСЛІДЖЕНЬ

- *Об`єкт дослідження* – технологія Інтернет Речей (IoT).
- *Предмет дослідження* – управління апаратним та програмним забезпеченням IoT пристроїв.
- *Мета роботи* – розробка програмне забезпечення для управління IoT пристроями з використанням мікроконтролера.

2

ЗАГАЛЬНА КОНЦЕПЦІЯ ТЕХНОЛОГІЇ IOT В РЕАЛІЗАЦІЇ «РОЗУМНИЙ» БУДИНОК

3

ПРОГРАМНЕ ТА АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ IOT

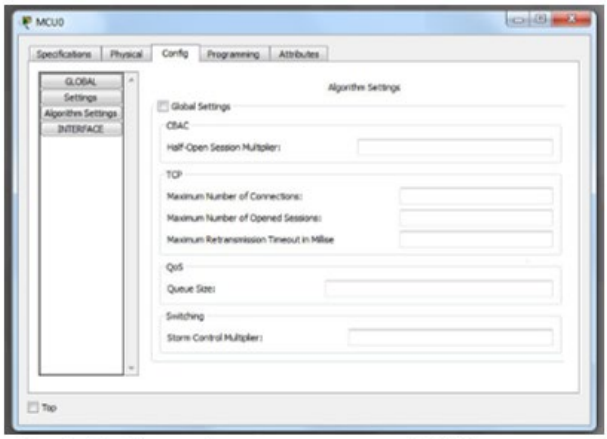

Рис.1. Скріншот вікна налаштування MCU емулятора Cisco Packet Tracer

Рис.2. Arduino Uno

4

ДОСЛІДЖЕННЯ ІСНУЮЧИХ АЛГОРИТМІВ ДЛЯ УПРАВЛІННЯ ДОДАТКАМИ ТА КАНАЛАМИ ЗВ'ЯЗКУ В IOT

Алгоритм 1 – Контролювання сигналу Electrical AC Load:

Require: Voltage signal from esp-8266
Load-1 = incoming voltage value from esp-8266
if (load-1≥3V) then
return 5V to the relay of load-1
else
return 0V to the relay of load-1
end if

Алгоритм 2 – Заходи щодо контролю температурними режимами на кондиціонері.

Require:LM-35 sensor value
temp-celsium = LM-35 sensor value
*temp = [temp-celsium * 500]/1023*
if (temp≥25 degree) then
return 5V to the relay of the ac
else
return 0V to the relay of the ac
end if

Алгоритм 3 - Управління датчиком освітлення

Require: Ultrasonic Sensor Value
duration1 = ultrasonic sensor value
distance1 = [duration1 / 33] /2
if (distance1≤357 cm) then
return 5v to the relay of light2
else
return 0v to the relay of light2
end if
if (357<distance1≤528 cm) then
return 5v to the relay of light3
else
return 0v to the relay of light3
end if
if (528<distance1≤954 cm) then
return 5v to the relay of light1
else
return 0v to the relay of light1
end if

5

ДОСЛІДЖЕННЯ ІСНУЮЧИХ АЛГОРИТМІВ ДЛЯ УПРАВЛІННЯ ДОДАТКАМИ ТА КАНАЛАМИ ЗВ'ЯЗКУ В IOT

Алгоритм 4 – Управління механізмом автоматичних дверей

Require: Ultrasonic Sensor2 Value
duration2 = ultrasonic sensor2 value
distance2 = [duration2 / 33] /2
if (5 cm≤distance2≤45 cm) then
drive the motor for rotor position one
else
drive the motor for rotor position two
end if

Алгоритм 5 – Управління вхідними дверцятами з веб-сторінки

Require: voltage signal from web server (esp-8266)
door-voltage = incoming voltage from esp-8266
if (door-voltage ≥ 3V) then
drive the motor2 for opening position
else
drive the motor2 for closing position
end if

Алгоритм 6 – Управління інтенсивністю освітлення

Require: Idr signal value
intensity = Idr signal value
if (intensity ≤800 mV) then
return 33 percent PWM signal to the light
else (800 mV < (intensity < 900 mV)
return 66 percent PWM signal to the light
end if
if (intensity ≥ 900 mV) then
return 100 percent PWM signal to the light
end if

Алгоритм 7 – Виявлення диму та легкозаймистого газу

Require: MQ-2 sensor value
threshold-voltage = 400 mV
smoke/gas-level = MQ-2 sensor value
if (smoke/gas-level ≤ threshold-voltage) then
return 0V to the buzzer
else
return 5V to the buzzer
end if

6

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ УПРАВЛІННЯ IOT ПРИСТРОЯМИ З ВИКОРИСТАННЯМ МІКРОКОНТРОЛЕРА ЗА ДОПОМОГОЮ ПРОГРАМУВАННЯ PYTHON

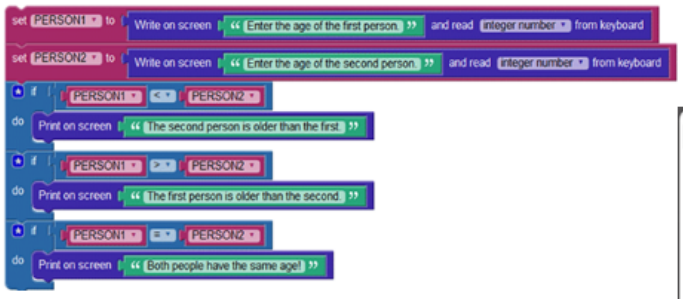

Рис.1. Приклад створення блокової програми

```
PERSON1 = None
PERSON2 = None


PERSON1 = int(input ('Enter the age of the first person.'))
PERSON2 = int(input ('Enter the age of the second person.'))
if PERSON1 < PERSON2:
  print('The second person is older than the first.')
if PERSON1 > PERSON2:
  print('The first person is older than the second.')
if PERSON1 == PERSON2:
  print('Both people have the same age!')
```

Рис.2. Еквівалентний код Python

7

РОЗРОБКА МОДЕЛІ УПРАВЛІННЯ ПРИСТРОЯМИ IOT З ВИКОРИСТАННЯМ МІКРОКОНТРОЛЕРА ТА КОМБІНАЦІЇ BLOCKLY ТА PYTHON

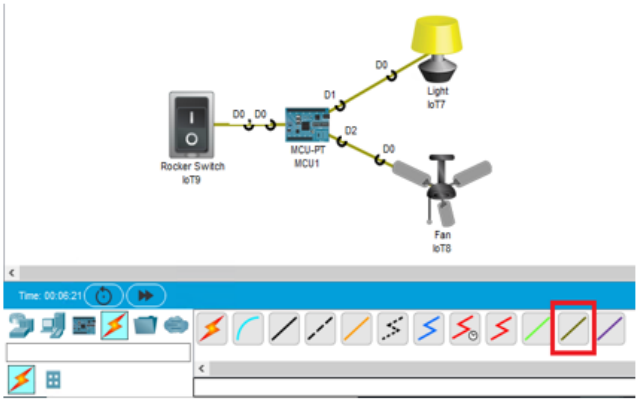

Рис.1. Підключення девайсів в емуляторі Cisco Packet Tracer

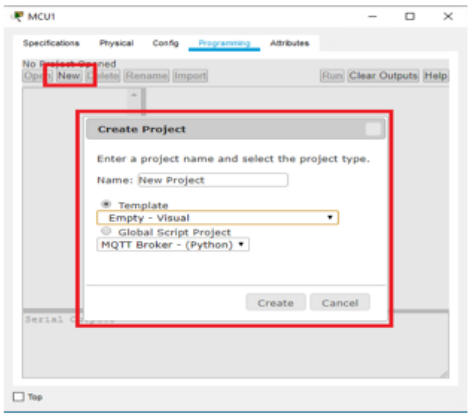

Рис.2. Створення нового проекту в емуляторі Cisco Packet Tracer

8

РОЗРОБКА МОДЕЛІ УПРАВЛІННЯ ПРИСТРОЯМИ IOT З ВИКОРИСТАННЯМ МІКРОКОНТРОЛЕРА ТА КОМБІНАЦІЇ BLOCKLY ТА PYTHON

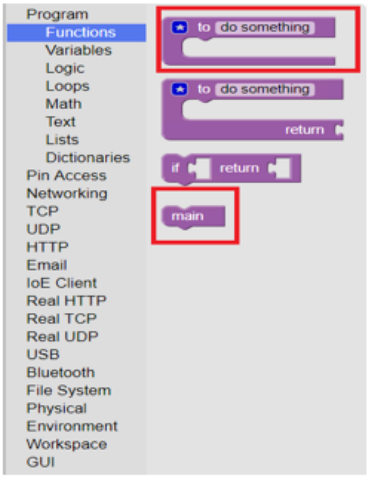

Рис.1. Приклад вибору функцій

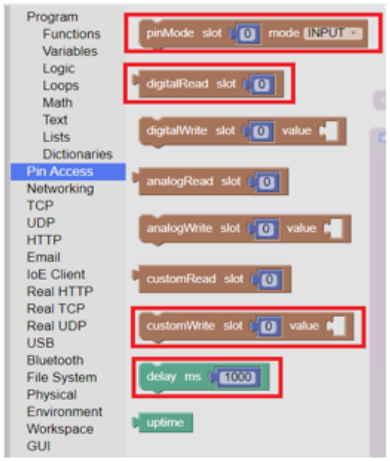

Рис.2. Pin Access

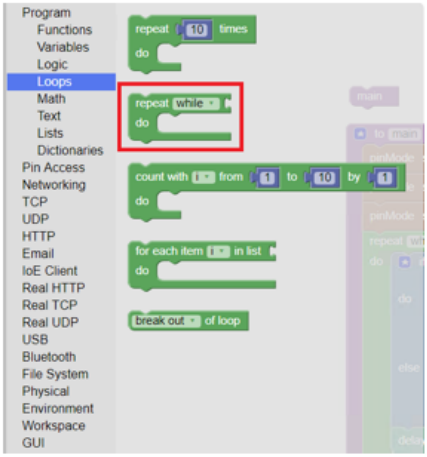

Рис.3. Цикли

РОЗРОБКА МОДЕЛІ УПРАВЛІННЯ ПРИСТРОЯМИ IOT З ВИКОРИСТАННЯМ МІКРОКОНТРОЛЕРА ТА КОМБІНАЦІЇ BLOCKLY ТА PYTHON

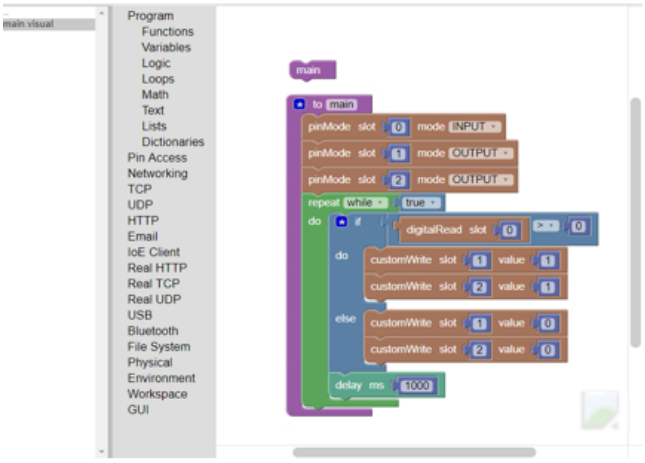

Рис.1. Фінальний результат

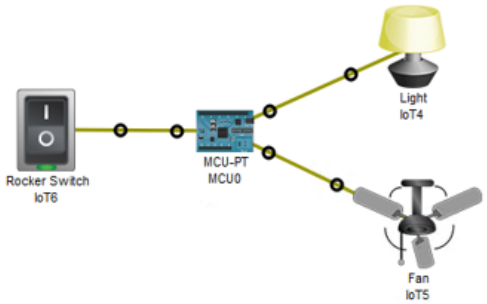

Рис.2. Перевірка результату

ВИСНОВКИ

1. Досліджено особливості створення та розповсюдження технології IoT. Описано принцип роботи системи «Розумний» будинок при умові впровадження мікроконтролера Arduino, а також приведено головні переваги даного мікроконтролера.
2. Зазначено, що система керується мікроконтролером, що в більшості випадків виконує функцію центрального процесора, який з'єднаний з портативним блоком плат. Підкреслено, що мікроконтролер також може бути підключеним до побутових електроприладів за допомогою електричних реле та деяких інших різних елементів, таких як двигуни та дальні датчики. Центральний мікроконтролер слідує різним алгоритмам для збору інформації, порівняння та прийняття рішень, показує результати та контролює прилади.
3. Досліджено різні типові для «розумного» будинку алгоритми, які необхідні для налаштування головних датчиків та сенсорів. Дотримуючись алгоритмів користувач може здійснювати управління кількома побутовими електроприладами одночасно. Однак кількість контрольованих приладів буде залежати від кількості вихідних портів центрального або головного мікроконтролера.
4. Зазначено, що Python дуже проста у використанні, потужна та універсальна мова, що обирається багатьма розробниками IoT. Досліджено мову візуального програмування Blockly, яка дозволяє користувачам створювати програми, об'єднуючи блоки, які представляють різні логічні структури мови, а не написанням фактичного коду. Blockly працює в веб-браузері та може відображати візуально створену програму в Python
5. Розроблено модель управління пристроями IoT з використанням мікроконтролера та комбінації мов Blockly та Python.

Дякую за увагу!