

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ ТА ТЕЛЕКОМУНІКАЦІЙ

УДК 631.344

«Допущено до захисту»

Завідувач кафедри біомедичної інженерії
та телекомунікацій
к.т.н., доц. Никітчук Т.М.

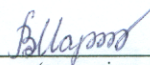
« 15 » 12 2020 р.

КВАЛІФІКАЦІЙНА РОБОТА

на тему Розробка та дослідження системи дистанційного керування розумною
теплицею

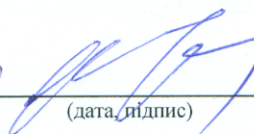
за спеціальністю 172 «Телекомунікації та радіотехніка»
освітній ступінь «магістр»

Виконавець роботи
студентка 2 курсу, гр.ТРМ-
19-1

15.12.20 
(дата, підпис)

Волошинська
М.П.

Керівник роботи
к.т.н., доц.

15.12.20 
(дата, підпис)

Ципоренко В.Г.

Житомир
2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ ТА ТЕЛЕКОМУНІКАЦІЙ

Пояснювальна записка
до кваліфікаційної магістерської роботи

на тему Розробка та дослідження системи дистанційного керування розумною
теплицею

Виконала: студентка 2 курсу, групи ТРм-19-1
спеціальності
172 «Телекомунікації та радіотехніка»

Волошинська М.П.

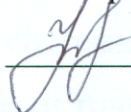
Керівник Ципоренко В.Г.

Рецензент Дубина О.Ф.

Житомир
2020

«Затверджено»

Завідувач кафедри біомедичної
інженерії та телекомунікацій

 Т.М. Нікітчук

« 15 » червня 2020 р.

Завдання

на кваліфікаційну магістерську роботу

Волошинської Маргарити Петрівни

1. Тема роботи Розробка та дослідження системи дистанційного керування розумною теплицею.

затверджено наказом по університету від «16» жовтня 2020 р. № 366/с.

2. Мета дослідження Розробка дистанційного керування розумною теплицею та дослідження точності системи.

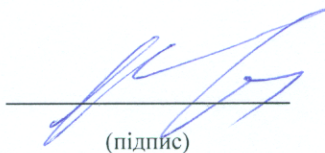
3. Вихідні дані (характеристики об'єкта, умов дослідження та ін.): Дані із датчиків та приборів.

4. Основні задачі дослідження: Створення і аналіз системи контролю для тепличного господарства, щоб покращити точність отриманих даних та взаємодіяти з ними

5. Термін подання роботи до захисту грудень 2020 р.

6. Дата видачі завдання «15» червня 2020 р.

Науковий керівник


(підпис)

Ципоренко В. Г.

Завдання прийняв


(підпис)

Волошинська М. П.

до виконання

АНОТАЦІЯ

Темою даної магістерської роботи є розробка системи дистанційного керування та її дослідження.

Метою даної роботи є створення системи контролю для тепличного господарства, щоб покращити змогу отримувати точні дані та взаємодіяти з ними, а також по можливості знизити собівартість системи в порівнянні з аналогами.

ABSTRACT

The topic of this master's thesis is the development of a remote control system and its research.

The purpose of this work is to create a control system for the greenhouse to improve the ability to obtain accurate data and interact with them, as well as reduce the cost of the system compared to analogues.

РЕФЕРАТ

Обсяг пояснювальної записки містить 58 сторінок, 35 рисунків, 5 таблиць, 16 джерел посилання.

Метою кваліфікаційної роботи є розробка та дослідження системи Інтернет речей-обладнання для тепличного господарства.

В першому розділі проведено аналіз нюансів розробки системи дистанційного контролю. Була поставлена задача дослідження контролю через Wi-Fi.

У другому розділі розробка системи віддаленого контролю теплиці.

У третьому розділі проведено дослідження точності вимірів: температури, вологості, тиску, освітленості та точність відклику системи. Дані отримані при моделюванні системи. Побудовано графіки залежностей.

У висновку було підведено підсумки досліджень.

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧКИ

Інтернет речей	– Internet of Things
МК	– Мікроконтролер
RTC	– Real-time clock
I2C	– Inter-Integrated Circuit
SPI	– Serial Peripheral Interface
HTTP	– Hyper Text Transfer Protocol
EEPROM	– Electrically Erasable Programmable Read-Only Memory

ЗМІСТ

АНОТАЦІЯ	4
ЗМІСТ	7
ВСТУП	8
1 АНАЛІЗ АСПЕКТІВ РОЗРОБКИ	10
1.1 Обґрунтування розробки системи контролю для розумної теплиці	10
1.2 Системи контролю мікроклімату	12
1.3 Сучасні системи	15
1.4. Постановка завдання на дослідження	16
2 РОЗРОБКА СИСТЕМИ	18
2.1 Розробка структурної схеми системи контролю для розумної теплиці	18
2.1.1 Вибір платформи Arduino	19
2.1.2 WiFi модуль	21
2.1.3 Побудова структурної схеми	24
2.2 Розробка алгоритму функціонування системи контролю для розумної теплиці	25
2.3 Обґрунтування вибору технології віддаленого з'єднання	28
2.4 Розробка принципової схеми системи контролю	28
2.5 Розробка програмного забезпечення для системи керування для розумної теплиці	31
2.6 Система розумна теплиця	45
3 ДОСЛІДЖЕННЯ ПРАЦЕЗДАТНОСТІ ТА ЕФЕКТИВНОСТІ СИСТЕМИ ТА ЇЇ КОНТРОЛЮ	47
3.1 Імітаційне моделювання системи розумна теплиця	47
3.2 Дослідження точності вимірів датчиків системи розумна теплиця	49
3.3 Моделювання та дослідження точності відклику системи контролю для розумної теплиці	54
ВИСНОВКИ	56
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ	57

ВСТУП

Актуальність теми. Розумні системи є однією з найбільш актуальних тем на сьогоднішній день. «Розумні» будинки, теплиці, та багато іншого всі ці об'єкти включають в себе Інтернет речей. Ця тема є основним питанням конференцій всіх наукових спільнот.

Актуальності розробки системи для розумної теплиці пов'язана з розвитком концепції Смарт речей. З'явилася різна кількість «розумних речей», які отримання інформацію, яка з достатнім ступенем деталізації описувала б певні об'єкти. Наприклад, скільки датчиків потрібно, щоб у повній мірі описати стан теплиці? На яку максимальну дальність можливо передавати дані з теплиці? Також не слід забувати що пристрої обробки даних мають обмежену кількість дротових з'єднань, а пристрої бездротового з'єднання збільшують собівартість систем.

Метою роботи є створення системи контролю для тепличного господарства, щоб покращити змогу отримувати точні дані та взаємодіяти з ними, а також по можливості знизити собівартість системи в порівнянні з аналогами.

Для досягнення цієї мети виконані такі **завдання**:

1. Обґрунтувати актуальність теми роботи по розробці системи контролю для розумної теплиці.
2. Проаналізувати основні вимоги до системи контролю для розумної теплиці та вибір необхідних датчиків.
3. Розробити структурну схему системи контролю для розумної теплиці.
4. Розробити алгоритми функціонування системи контролю для розумної теплиці.
5. Розробити принципову схему системи контролю для розумної теплиці.
6. Провести моделювання системи контролю для розумної теплиці.

По при все система має бути надійною, тобто виконувати свою роботу навіть коли немає доступу до інтернету. Крім того не менш важливими є вимоги до низької вартість пристроїв і їх мінімального енергоспоживання.

Об’єкт дослідження – система контролю для розумної теплиці.

Предмет дослідження – способи та алгоритми збору та передачі інформації від системи до користувача.

Наукова новизна – удосконалено способи та алгоритми збору та передачі інформації.

Практична значення роботи полягає в тому що застосування розроблених алгоритмів дозволяє покращити процеси збору та передачі інформації в системах та проводити їх обробку що в цілому сприяє управлінню тепличним господарством.

1 АНАЛІЗ АСПЕКТІВ РОЗРОБКИ

1.1 Обґрунтування розробки системи контролю для розумної теплиці

Система управління - концепція мережі, що складається з взаємопов'язаних пристроїв та програмного забезпечення, яка буде передавати та обмінюватися даними між фізичними та комп'ютерними системами за допомогою базових протоколів зв'язку. Окрім датчиків, система має виконавчі механізми, які вбудовані у фізичні об'єкти і з'єднані між собою за допомогою бездротового або бездротового з'єднання. Взаємозв'язані пристрої мають здатність читати та активувати функції, а також зменшувати участь людини завдяки використанню автоматизованих інтелектуальних систем.

Розглянемо основні умови, які враховуються при виборі автоматичної системи контролю мікроклімату для теплиць. Вартість системи. Фактор, який завжди враховується в першу чергу і відіграє важливу роль у проектуванні системи.

1. Вартість системи управління. Цей фактор завжди враховується в першу чергу і, найчастіше, є вирішальним.
2. Сучасні системи автоматичного управління складаються з пристрою управління, виконавчих пристроїв і допоміжного обладнання.
 - Пристрій управління - мікроконтролери та комп'ютери з вбудованим і прикладним програмним забезпеченням, різні датчики і будь-який спосіб виведення інформації на екран або на друк;
 - Виконавчі пристрої - реле, комутуючі обладнання, електродвигуни, електроклапана.
 - Допоміжне обладнання - силові кабелі і кабелі для ланцюгів управління, монтажні коробки і корпусу, і монтажні матеріали.
3. Функціональні можливості системи управління.

Практично всі сучасні системи автоматичного управління мікрокліматом теплиць володіють достатнім набором функціональних можливостей, необхідних для управління мікрокліматом, але відрізняються деталями:

- можливість автоматичного управління наявними в теплиці технологічними системами обігріву, кватиркової вентиляції, поливу, електричного досвічування, підживлення вуглекислим газом;
- можливість забезпечення одночасного управління різними вузлами системи;
- якість підтримки необхідного клімату в теплицях або можливість наявними засобами домогтися мінімального відхилення від завдання підтримуваних кліматичних параметрів: температури, вологості, концентрації вуглекислого газу, освітленості;
- можливість коригування поставлених параметрів мікроклімату в теплиці в залежності від стану рослин, часу доби, року і метеоумов для підвищення врожайності в теплиці;

4. Ступінь автоматизації функцій управління.

Чим вище ступінь автоматизації, тим менше потрібно втручань оператора в процеси управління. За вказівкою, задаються графіки підтримки мікроклімату і режими роботи устаткування, які автоматично коригуються відповідно до процесів управління, станом рослин і зовнішніми метеоумовами.

5. Технічна база системи управління.

Технічна база або технічне забезпечення систем управління включає в себе дві основні складові: засоби контролю і введення даних, перетворення, обробки і видачі керуючих впливів (Інтелектуальна складова) і засоби електричного живлення, комутації, захисту та підключення засобів управління (силова складова).

6. Надійність елементної бази.

Як правило: чим нижче надійність елементів, тим нижче капітальні витрати при створенні системи і вище вартість експлуатаційних витрат на

енергоносії та ремонт обладнання. І навпаки, надійне обладнання дозволяє менше уваги приділяти його працездатності і більше - вирощування врожаю;

7. Зручність експлуатації та обслуговування.

Як відомо, ніщо не вічне під Місяцем. Це відноситься і до систем управління. Згодом їх доводиться ремонтувати, а також бажано, для запобігання збоїв в роботі, виконувати профілактичне обслуговування. Зручність експлуатації та обслуговування визначається наявністю і повнотою технічної і технологічної документації, включаючи електричні принципові схеми, паспорти на обладнання, опису та інструкції по експлуатації. Дані умови і їх актуальність залежать, в свою чергу, від інших факторів: ступеня автоматизації, структури, надійності технічного забезпечення, наявності проектної та експлуатаційної документації.

8. Міжмашинна взаємодія.

Тема більшості запитань та дискусій щодо інтелектуальних систем полягає в наступному: чим інтелектуальні системи відрізняються від взаємодії між машинами (M2M). До того, як розумні системи стали популярною темою для розмов, основна увага приділялася взаємодії між машинами. Технологія сумісності та розумні системи дуже схожі.

Підтримання мікрокліматичних умов у теплицях є складним завданням, оскільки мікрокліматичні умови в приміщеннях створюються тепловими системами, вентиляцією, опаленням, кондиціонуванням та впливом навколишнього середовища. Інтелектуально-адаптивні автоматичні системи управління повинні використовуватися для поліпшення умов для підтримання необхідних кліматичних параметрів. Однак у цьому випадку існує проблема опису кімнатної температури як об'єкта регулювання.

Щоб змодельовати приміщення як об'єкт регулювання, необхідно побудувати розумно поєднану автоматичну систему мікрокліматичної підтримки. Це можна додатково описати, описуючи елементи системи автоматичного управління окремими деталями. Кожна частина системи характеризується початковим значенням або спеціальним параметром. Усі дії системи будуть поділятися на виконавчі та управлінські. Виконавчий вплив відхиляє

регульований параметр від встановленого значення. Ефект управління використовується для повернення регульованого параметра до бажаного значення. Цикл управління - це послідовний ряд з'єднань між об'єктом і системним контролером, які забезпечують зворотний зв'язок. Схема управління включає ряд елементів системи, які поєднуються за загальним принципом роботи "об'єкта управління", первинного перетворювача (датчика), вторинного перетворювача та виконавчої системи. Ці три елементи об'єднані поняттям «автоматичне регулювання». Отже, для термостата вхідним значенням є температура, а вихідним - приміщення виконавчого механізму ($y\%$). Контролер у задачі вибору конфігурації вважатиметься інерційною ланкою.

Схема автоматичних систем управління, як правило, розташована поруч, враховуючи напрямок руху повітря. Тому окремі частини систем утворюють механізми, розташовані послідовно. Одноіменні блоки в зонах автоматичної стабілізації будуть паралельними.

Для точної роботи інтелектуально-адаптивної системи, яка автоматично підтримує параметри мікроклімату, необхідно оцінити статичні та динамічні характеристики території як об'єкта для стабілізації температури. Математичний опис процесу ускладнюється багатьма факторами. Розглянемо їх докладніше. Джерела тепла розподіляються хаотично по об'єму приміщення; пункти подачі повітря також розташовані довільно; подача електроенергії та потік повітря в деяких частинах кімнати не збалансовані. Як результат, температура повітря як регульований параметр всебічно розподіляється в обсязі приміщення. Особливо значні нерівності спостерігаються в тій частині, куди потрапляє неізотермічна струмінь (як правило, у верхню зону). Джерела тепла в приміщенні є променево-конвективними, тоді як конвективне тепло потрапляє в повітря і сяє на поверхні огорож та обладнання. Щільність теплового потоку на різних поверхнях неоднакова. Поряд зі зміною температури повітря відбувається зміна температури огорож та обладнання, що залежить від розміру та теплофізичних характеристик матеріалів, періоду температурних коливань (у періодичному процесі). Обладнання може бути досить складної форми, і його подання

найпростішим корпусом (пластиною, кулею або циліндром) може призвести до значної помилки. При деяких порушеннях, таких як потік повітря в приміщенні та температура зовнішнього повітря, змінюється як променисте, так і конвективне тепло. Для інших впливів - регулювання, наприклад, температури припливного повітря та масивного повітрообміну - вводиться лише конвективне тепло. Опис конвекційного теплообміну ускладнюється тим, що його інтенсивність, оцінена за коефіцієнтом тепловіддачі АК, є змінною і може бути визначена лише приблизно, оскільки вона залежить від кількох факторів. З них найбільш значуща схема повітрообміну, тип струменів по відношенню до поверхні огорож (наземний, знімний), форма струменів (плоскі, конусоподібні, симетричні), частота повітрообміну в кімнатах. Ми будемо розглядати приміщення як об'єкт з точки зору концентрованих параметрів, тому рівняння теплового балансу буде складено відносно температури вхідного повітря, яка, як правило, відрізняється від температури в робочій (населеній) зоні. Основними огорожами є плити, в яких тепловий потік у поздовжньому напрямку відсутній. Коефіцієнт тепловіддачі вважається середньою площею огорож. Теплофізичні параметри огорож та обладнання у розглянутому діапазоні температур є постійними. Залежно від фізичної природи скандальних (із відкритим питомим коефіцієнтом поглинання) та контрольних (із закритим питомим коефіцієнтом поглинання) ефектів, ефекти можуть вимагати перехідних та амплітудних частотних характеристик приміщення.

1.2 Системи контролю мікроклімату

Для отримання гарних результатів при вирощуванні рослин, необхідно дотримуватись оптимальних умов розвитку. Для забезпечення цього можуть використовуватись різні прилади, які дозволяють контролювати мікроклімат теплиці:

- система обігріву. Призначена для дотримання заданої температури. Як правило, опалення теплиці складається з декількох роздільних контурів (систем). Регулювання температури повітря здійснюється за допомогою

зміни температури води в контурах, що здійснюється за допомогою змішувального клапана, який змішує в необхідній пропорції воду від джерела тепла (прямий теплоносій) з водою, яка повернулася з теплиці (зворотним теплоносієм). Також для теплиць малих розмірів використовують нагрівачі повітря, якщо вини із спіралі їхній недолік в тому що вони підсушують повітря.

- система провітрювання. Для вентиляції повітря можна використовувати кватирки. Відкривання та закривання можна здійснити за допомогою спеціальних фоторедукторів, які відкривають та закривають вентиляцію в вказаний період дня;
- система зашторювання. Практично всі нові проекти теплиць комплектуються системою зашторювання, тому що вона дозволяє економити до 30% теплоресурсів, а також захистити рослини від сонячних опіків. Принцип її роботи полягає в тому, що спеціальний полімерний матеріал згортається і розгортається над рослинами, відсікаючи тим самим покрівлю. Згортання та розгортання проводиться за допомогою спеціальних моторедукторів;
- система досвічування. Використовують для забезпечення рівня освітленості тоді, коли природного світла недостатньо. Наприклад для квітів це є обов'язковою умовою для вирощування. Так як ми вирощуєм печериці, то освітленням можна знехтувати. Грибам не потрібне світло, тому що вони позбавлені можливості фотосинтезу;
- система рециркуляції повітря. Зазвичай монтують в верхній частині теплиці для забезпечення течії повітря і вирівнювання температурних зон.

1.3 Сучасні системи

Вперше про створення розумної теплиці стали говорити К. Малишевський і Н. Курдюмов. Вони також є авторами своєї книги «Розумна теплиця».

Що це таке? Курдюмов і Малишевський були досвідченими городниками. За всі роки своєї практики, вони намагалися внести корективи у створення та забезпечення теплиці.

Вони вносили автоматизовані процеси в конструкції автоматичного поливу, провітрювання та обігріву.

Примітка. Всі ці процеси допомагають заощадити не тільки час городника, але й сили. Якщо раніше всі дії здійснювалися вручну, то на даний момент кожен бажає автоматизувати роботу теплиці.

Ці системи програмуються на певний показник і тільки після цього спрацьовує автоматика. Звичайно, без допомоги фахівців у цій галузі в деяких випадках просто не обійтися. Вони допоможуть налаштувати обладнання або відремонтувати його у разі потреби. Але, не дивлячись на це, будь-хто може створити розумну теплицю своїми руками.

Розглянемо декілька фірм, які займаються розповсюдженням автоматизованих теплиць. Компанія Green IQ займається розповсюдженням розумних теплиць які базуються на аеропонних трубах. В одній трубі знаходиться 70 комірок. Конструкція може бути горизонтальної для вирощування, наприклад, базиліка або руколи і вертикальної. На 1м² поміщається 4 вертикальних аеропонних труби. В середині знаходиться системи для автоматичного догляду за рослинами. Компанія CyberGrow теж займається розповсюдженням автоматизованих теплиць також способом аеропонних труб. Компанія T-lux займається будівництвом і обладнанням аркових теплиць.

Основним принципом автоматизації в теплицях є управління електрообладнанням, а також зняття та корегування мікроклімату. Мозком теплиці є контролер, він виконує основну частину: знімає показники датчиків, порівнює їх із діапазоном даних потрібного клімату, контролює полив.

1.4. Постановка завдання на дослідження

Призначення роботи: Створення системи контролю для розумної теплиці.

Метою створення є дослідження технологій передачі даних для аналізу та здійснення контролю системою за допомогою віддаленого доступу.

Технічні параметри об'єкту розробки:

- Збір даних із теплиці
- Початок роботи при подачі живлення
- Передача даних на хмарній сервіс з використанням протоколів Wi-Fi

Умови експлуатації:

- Діапазон робочих температур: -40 до +70 С
- Відносна вологість: від 0 до 100%

Вимоги до конструкції:

- Габаритні розміри, ШхДхВ, мм, не більше: 100х150х50.
- Тип використання: універсальний
- Живлення: від мережі та автономно, за допомогою акумуляторів 5-12В

2 РОЗРОБКА СИСТЕМИ

Система контролю «Розумної теплиці» призначена для локального та віддаленого доступу до теплиці. Це дозволить в будь-який момент перевірити стан теплиці та при необхідності корегувати налаштування теплиці.

2.1 Розробка структурної схеми системи контролю для розумної теплиці

Система теплиці складається із:

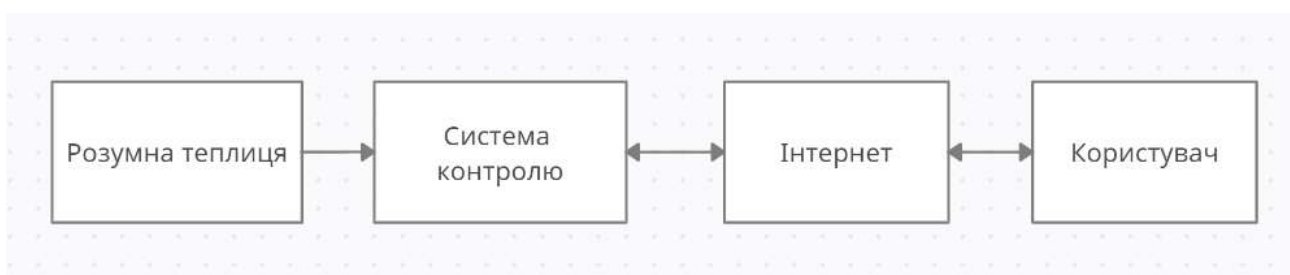


Рисунок 2.1 – Схема складових Розумної теплиці.

У цьому випадку роль розумної теплиці - це система, розроблена раніше. Система управління діє як комунікація між користувачем та розумною теплицею.

Система складається з таких вузлів:

Мікроконтролер - це спеціалізований мікроелектронний програмований пристрій, призначений для використання в пристроях управління, системах передачі даних та системах управління процесами.

Мікроконтролери використовуються в побутовій техніці, медичних приладах, системах управління ліфтами, телефонах, раціях та інших засобах зв'язку, електронних музичних інструментах та автомагнітолах, комп'ютерній периферії (клавіатури, джойстики, принтери тощо), світлофорах, автоматичних воротах та бар'єри, інтерактивні дитячі іграшки, машини, локомотиви та літаки, роботи та промислові машини. Мікроконтролери можна знайти на нових сучасних приладах, таких як телефони, праймери, вони відповідають за роботу двигунів та гальмівних систем сучасних автомобілів завдяки їх створенню систем

управління та систем збору інформації. На основі мікроконтролерів проектуються і створюються вимірювальні прилади, системи управління об'єктами і процесами, вони є основою безпеки, систем протипожежного захисту, домофонів, сигналізації та ін. [1]

Більшість процесорів у світі є мікроконтролерами. Неповний список периферії, яка може бути присутня у мікроконтролерах, включає:

- Різні інтерфейси вводу-виводу, такі як UART, I2C, SPI, CAN, USB;
- Аналого-цифрові і цифро-аналогові перетворювачі;
- Компаратори;
- Широтно-імпульсні модулятори;
- Таймери.

Програми для мікроконтролерів створюються в спеціальних інтегрованих середовищах інструментів (IDE) в Assembler (машинні інструкції) або на C ++. Зазвичай один і той же контролер може бути запрограмований на декілька мов на вибір користувача. Для цього існують інструментальні програмні комплекси, які дозволяють не тільки розробляти програму, але й налаштовувати її за допомогою програмної моделі контролера («на стимуляторі») або в режимі моніторингу (коли виконується програма користувача реальним контролером). його робота). Програмні стимулятори використовуються для налагодження програм. Спеціальні програми для персональних комп'ютерів, що імітують роботу мікроконтролера.

2.1.1 Вибір платформи Arduino

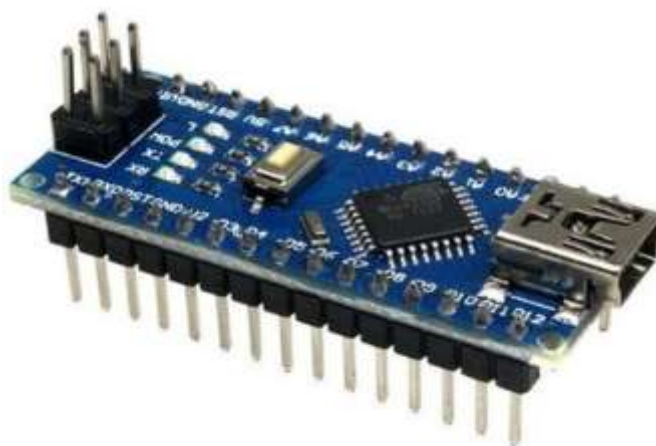


Рисунок 2.2 - Плата Arduino

Arduino - це електронний конструктор і зручна платформа швидкої розробки електронних пристроїв для новачків і професіоналів. Платформа користується величезною популярністю в усьому світі завдяки зручності і простоті мови програмування, а також відкритій архітектурі і програмного коду. Пристрій програмується через USB без використання програматорів.

Arduino дозволяє комп'ютеру вийти за рамки віртуального світу в фізичний і взаємодіяти з ним. Пристрої на базі Arduino можуть отримувати інформацію про навколишнє середовище за допомогою різних датчиків, а також можуть управляти різними виконавчими пристроями.

Мікроконтролер на платі програмується за допомогою мови Arduino (заснований на мові Wiring) і середовища розробки Arduino (заснована на середовищі Processing). Проекти пристроїв, засновані на Arduino, можуть працювати самостійно, або ж взаємодіяти з програмним забезпеченням на комп'ютері (напр .: Flash, Processing, MaxMSP). Плати можуть бути зібрані користувачем самостійно або куплені в зборі. Програмне забезпечення є для безкоштовного скачування. Вихідні креслення схем (файли CAD) є загальнодоступними, користувачі можуть застосовувати їх на свій розсуд. Ардуіно і Ардуіно-сумісні плати спроектовані таким чином, щоб їх можна було при необхідності розширювати, додаючи в пристрій нові компоненти («shields»). Ці плати розширень підключаються до Ардуіно за допомогою встановлених на них штирових роз'ємів. Існує ряд уніфікованих плат, що допускає конструктивно жорстке з'єднання процесорної плати та плат розширення в стопку через штирові лінійки. Крім того, випускаються плати зі зменшеним (наприклад, Nano, Lilypad) і спеціальним (для задач робототехніки) форм-фактором.

Програмування ведеться цілком через власну програмну оболонку (IDE), безкоштовно доступну на сайті Arduino. У цій оболонці є текстовий редактор, менеджер проектів, препроцесор, компілятор і інструменти для завантаження програми в мікроконтролер. Оболонка написана на Java на основі проекту Processing, працює під Windows, Mac OS X і Linux.

Використання платформи Arduino більш чим достатньо для обробки, керування, та збору інформації, а також це дає змогу швидко замінити несправну частину пристрою, швидко запрограмувати, та удосконалювати пристрій в майбутньому.

2.1.2 WiFi модуль

Wi-Fi - технологія бездротової локальної мережі з пристроями, що базуються на стандартах IEEE 802.11. Під аббревіатурою Wi-Fi (від англійської фрази Wireless Fidelity, що можна дослівно перекласти як "бездротове вкладення") в даний час розробляється сімейство стандартів цифрової передачі даних по радіоканалам. Wi-Fi працює в неліцензійному діапазоні частот ISM (2402-2480 МГц).

Для підключення Arduino до бездротової мережі та обміну інформацією використовуйте модуль wifi. Спочатку такі модулі використовувались як wi-fi-екран arduino. Існує багато бібліотек, включаючи власні, і вона підтримує роботу через послідовні шини та найпростіші команди AT і AT +. Через це ніякої семантики горезвісного C99, як це часто буває з іншими сторонніми мікроконтролерами, не потрібно вивчати..



Рисунок 2.3 - ESP8266

ESP8266 - мікроконтролер з інтерфейсом Wi-Fi. Крім Wi-Fi, мікроконтролер відрізняється відсутністю флеш-пам'яті в SoC, програми користувача виконуються із зовнішнього флеш-пам'яті з інтерфейсом SPI.

Даний мікроконтролер має наступні характеристики:

- 80 MHz 32-bit процесор Tensilica Xtensa L106;
- IEEE 802.11 b / g / n Wi-Fi. Підтримується WEP і WPA / WPA2;
- 14 портів вводу-вивода(из них возможно использовать 11), SPI, I2S, UART, 10-bit АЦП.
- Живлення 2,2 ... 3,6 В. Споживання до 215 мА в режимі передачі, 100 мА в режимі прийому, 70 мА в режимі очікування. Підтримуються три режими зберігання, без збереження з'єднання з точкою доступу: Modem sleep (15 мА), Light sleep (0.4 мА), Deep sleep (15 мкА).

Модуль реального часу.



Рисунок 2.4 – Модуль реального часу

ТС (real time clock, рус. Годинник реального часу) - це вид мікросхем, призначених для відліку часу в «реальних» одиницях (секунди, хвилини, години і т.д.).

Вони залежні від джерела живлення, який може бути як зовнішнім, у вигляді змінної батареї або літєвий акумулятор, так і вбудованим в корпус мікросхеми (див. Фото нижче). Тактові сигнали для звіту часу можуть бути отримані з зовнішнього кварцового резонатора, а рідше - з електромережі.

Точність відліку якраз і залежить від якості і точності настройки внутрішнього генератора або зовнішнього кварцового резонатора. При цьому точність кварцу і RTC відповідно, вказується не в герцах і не у відсотках, а в «ppm», наприклад $\pm 12 \text{ ppm}$, $\pm 50 \text{ ppm}$. Це розшифровується, як Parts Per Million, т. Е. Кількість мільйонних частин від якоїсь середньої величини.

Годинник реального часу можуть бути реалізовані на базі мікроконтролерів, однак, використання спеціальних чіпів дозволяє знизити

енергоспоживання, оскільки більшість мікроконтролерів навіть в сплячому режимі (або режимі зниженого енергоспоживання) споживають більше енергії ніж спеціальні інтегральні мікросхеми (IMC). RTC можуть бути і вбудованими в сам мікроконтролер (як в STM32).

Саме завдяки годинах реального часу на вашому комп'ютері не збивається час і дата після його відключення від мережі, в цьому випадку вони працюють від батарейки CR2032, встановленої в роз'ємі на материнській платі, вона ж і живить мікросхему BIOS, щоб не збивалися виставлені в ньому настройки. класифікація

Класифікація мікросхем RTC може відрізнятися від виробника до виробника. Найбільш поширені годинник реального часу таких виробників, як: Maxim Integrated і STMicroelectronics. На ринку є мікросхеми та інших компаній:

Intersil Corporation (д.к. Renesas Electronics);

Symbet (лінійка EnerChip TM RTC, відмінна риса - вбудована твердотільна батарея);

NXP (RTC з календарем, з підтримкою протоколів I2C або SPI)

Zilog;

Epson;

ON semiconductor.

Робоча напруга, В	2...5,5
Елемент живлення	CR2032
Доступна пам'ять модуля, байт	31
Максимальний споживаний струм при 2,5 В, мА	300
Робоча температура, ° С	0 ... +70
Розміри датчика, мм	44 x 23 x 12

Модуль годин реального часу - це електронна схема, призначена для обліку хронометричних даних (поточний час, дата, день тижня і ін.), Являє собою систему з автономного джерела живлення.

Модуль можливо використовувати в якості годинника або будильника, побудованих на базі плат Arduino. Або ж як оповіщення для різних систем, наприклад в «Розумному будинку».

В платах Arduino вже є вбудований датчик часу Millis, проте він працює тільки при живленні плати. При відключенні і надалі включенні Arduino відлік часу Millis скинеться до нуля. А модуль годин реального часу має вбудовану батарейку, яка навіть при відключеному плати Arduino продовжує «живити» модуль, дозволяючи йому вимірювати час.

Джерело живлення – живить всі вузли системи.

Джерела живлення характеризуються значенням електрорушійної сили і внутрішнього опору.

До джерел живлення належать гальванічні елементи, електрохімічні батареї, акумулятори, термопари, сонячні батареї, електричні генератори тощо.

Залежно від виду електрорушійної сили джерела живлення поділяють на джерела живлення постійного струму і джерела живлення змінного струму.

В якості джерела живлення можливо використовувати батарейку типу «Крона» або блок живлення.

Блок живлення перетворює змінну напругу величиною 220 В і частотою 50 Гц, в задану постійну напругу.

2.1.3 Побудова структурної схеми

На основі елементів пристрою розробимо структурну схему.

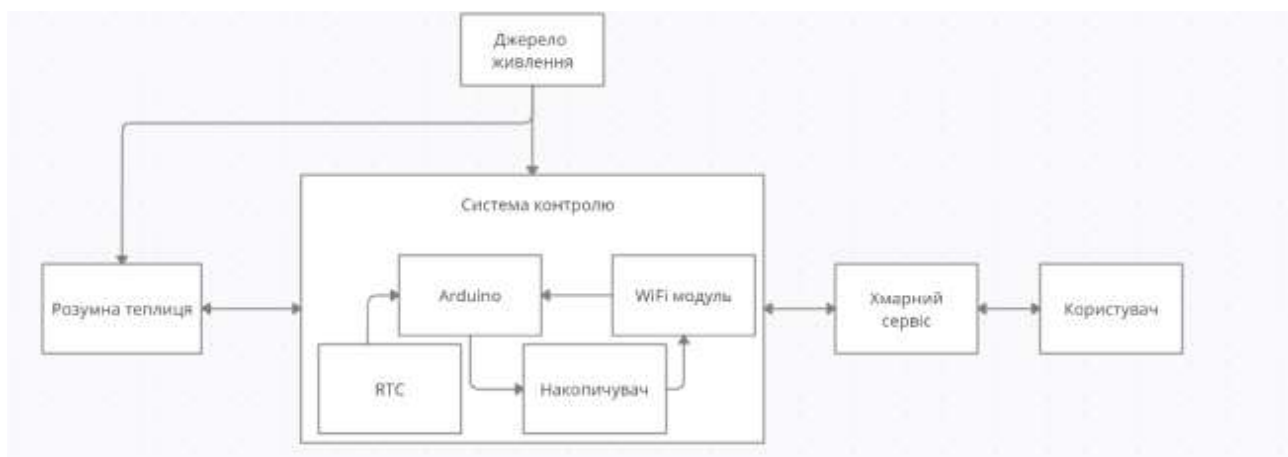


Рисунок 2.5 Структурна схема системи

Як видно зі структурної схеми основний блок - мікроконтролер, цей блок використовується для відправки оброблених даних а також контролю роботи всієї системи, адже завдяки мікроконтролеру можна програмувати та керувати пристроєм, в якості мікроконтролера використаємо готову платформу Arduino .

На Arduino подається напруга від джерела живлення тим самим забезпечує роботу всього модулю. Для нормального функціонування системи потрібно подати напругу від 5 В до 20 В, тобто в якості джерела живлення можливе використання USB – роз'єму, батарейки типу «Крона» або блока живлення що на виході має від 5 В до 20 В.

Отримана інформація передаватиметься на інтернет ресурс завдяки Wi-Fi модулю. Для зручного перегляду відправленої інформації потрібно її структурувати, в цьому допоможе модуль реального часу який в свою чергу фіксує дату та час в який було отримано інформацію.

2.2 Розробка алгоритму функціонування системи контролю для розумної теплиці

Алгоритм програми - це точне розпорядження (сукупність послідовних кроків, схема дій), яке визначає процес переходу від первинних даних до бажаного результату. При створенні програм алгоритм описує логічну послідовність кроків виконання операцій, для завершення та отримання бажаного результату. Для візуального відображення алгоритмів використовують блок-схеми.

Алгоритм роботи системи:

Початок - Початок роботи програми при подачі живлення

Завантаження параметрів - Завантаження параметрів (стану системи та параметри точок доступу)

Пошук Wi-Fi сигналу – Шукає сигнал Wi-Fi

Найдено сигнал - При умові, якщо знайдено сигнал переходимо до підключення, в іншому випадку створюємо точку доступу

Підключення здійснено - При успішній спробі підключення до знайденого сигналу проводимо отримання даних, в іншому випадку повертаємось до створення точки доступу

Створення точки доступу - Створення точки доступу на основі отриманих з Arduino даних

Запуск веб-інтерфейсу - Запуск веб-інтерфейсу для відображення інформації локально

Отримання даних - Отримання оброблених даних з Розумної теплиці

Доступ до віддаленого сервера - При умові доступу до мережі інтернет відправляємо дані на хмарне сховище, у випадку відсутності доступу до інтернет мережі повертаємось до блоку отримання даних

Відправлення на хмарне сховище - Відправка даних на хмарне сховище, якщо операція не відбулась повертаємось до перевірки доступу до мережі.



Рисунок 2.6 – блок схема алгоритму роботи системи

2.3 Обґрунтування вибору технології віддаленого з'єднання

В даній роботі я вибрала реалізацію через веб-інтерфейс через WiFi з'єднання.

Даний метод дозволяє підтримувати безперервне з'єднання з веб сервером для аналізу даних та виконання корекції роботи теплиці. Порівнянно із більшістю видів з'єднань (радіочастотний, Bluetooth) він має необмежену дальність та дозволяє використовувати доступ з будь-якої точки світу, а зручний веб інтерфейс дозволить налаштувати систем в пару кліків. До переваг можна додати що даний метод мультиплатформний, тобто доступ можливо отримати як із смартфона, ПК та інших пристроїв які мають змогу вийти в мережу інтернет. Із недоліків слід відмітити що при використанні системи в багатоквартирному будинку при знеструмлені будинку система буде працювати в автономному режимі без змоги підключитися віддалено, рішенням є використовувати GSM технологію, але це збільшить фінансові затрати на обслуговування системи.

2.4 Розробка принципової схеми системи контролю

Вибір мікроконтролерів дуже різноманітний. Вони відрізняються якістю, характеристиками, а також ціною.

Atmega 328p - 8-розрядний мікроконтролер. Має низьке енергоспоживання, що було досягнуто за рахунок зменшення техпроцесу. Може виконувати багато різних інструкцій одночасно.



Рисунок 2.7 – Мікроконтролер Atmega328p

Таблиця 2.1 Характеристики Arduino Nano

32 kB Flash
2 kB ОЗУ
1 kB EEPROM
два 8-бітних таймера / лічильника з модулів порівняння і дільниками частоти
16-бітний таймер / лічильник з модулем порівняння і дільником частоти, а також з режимом запису
Лічильник реального часу з окремим генератором
Шість каналів PWM
6-канальний ЦАП з вбудованим датчиком температури
Програмований послідовний порт USART
Послідовний інтерфейс SPI; інтерфейс I2C
Програмований сторожовий таймер з окремим внутрішнім генератором
Внутрішня схема порівняння напруг
Блок обробки переривань і пробудження при зміні напруги на висновках мікроконтролера
Програмне розпізнавання зниження напруги живлення
Внутрішньо калібруємий тактовий генератор
Обробка внутрішніх і зовнішніх переривань
6 режимів сну (знижене енергоспоживання і зниження шумів для більш точного перетворення АЦП)
1.8 - 5.5 В при частоті до 4 МГц
2.7 - 5.5 В при частоті до 10 МГц
4.5 - 5.5 В при частоті до 20 МГц

9. Було обрано Arduino Nano v3 на мікроконтролері ATmega328 – готова до використання та має малі розміри.

Мініатюрний модуль на основі популярного мікроконтролера ATMEGA328P. Підходить для макетування без пайки. Контакти знаходяться по 2 сторони плати, відстань між ними 2,54мм, відстань між лінійками 15мм.

Вбудований bootloader і перетворювач USB $\leftrightarrow$ COM на базі мікросхеми CH340, дозволяє оновлювати прошивку без використання програматора, єдиним натисканням кнопки на комп'ютері. Однак, при необхідності, може бути "прошитий" і будь-яким стандартним програматором зі стандартним 6-вивідним інтерфейсом ISP.

Таблиця 2.2 Характеристики блока ATmega328P

Напруга ядра (логічні рівні)	+5 вольт
Напруга живлення входу VIN (рекомендовано)	+7 ... + 12 вольт
Гранично допустимий потенціал напруги живлення входу VIN	+6 ... + 20 вольт
Цифрових входів / виходів	14 (6 з яких дозволяють апаратно виводити PWM / ШІМ)
Світлодіоди	Живлення, Rx (прийом), Tx (Передача)
Входів АЦП	8
Максимально допустимий впадає / витікаючий струм висновків	40 мА
ППЗУ (Flash Memory)	32 кбайт, з яких 2 кБ використовується під завантажувач (bootloader)
ОЗУ (SRAM)	2 кБайт (вбудовано в ATmega328P)
EEPROM:	1 Кб (вбудовано в ATmega328P)
Тактова частота 16 МГц стабілізована кварцом	
Готовий модуль	
Мініатюрні розміри: 45 x18 мм	
Вага модуля 6 гр	

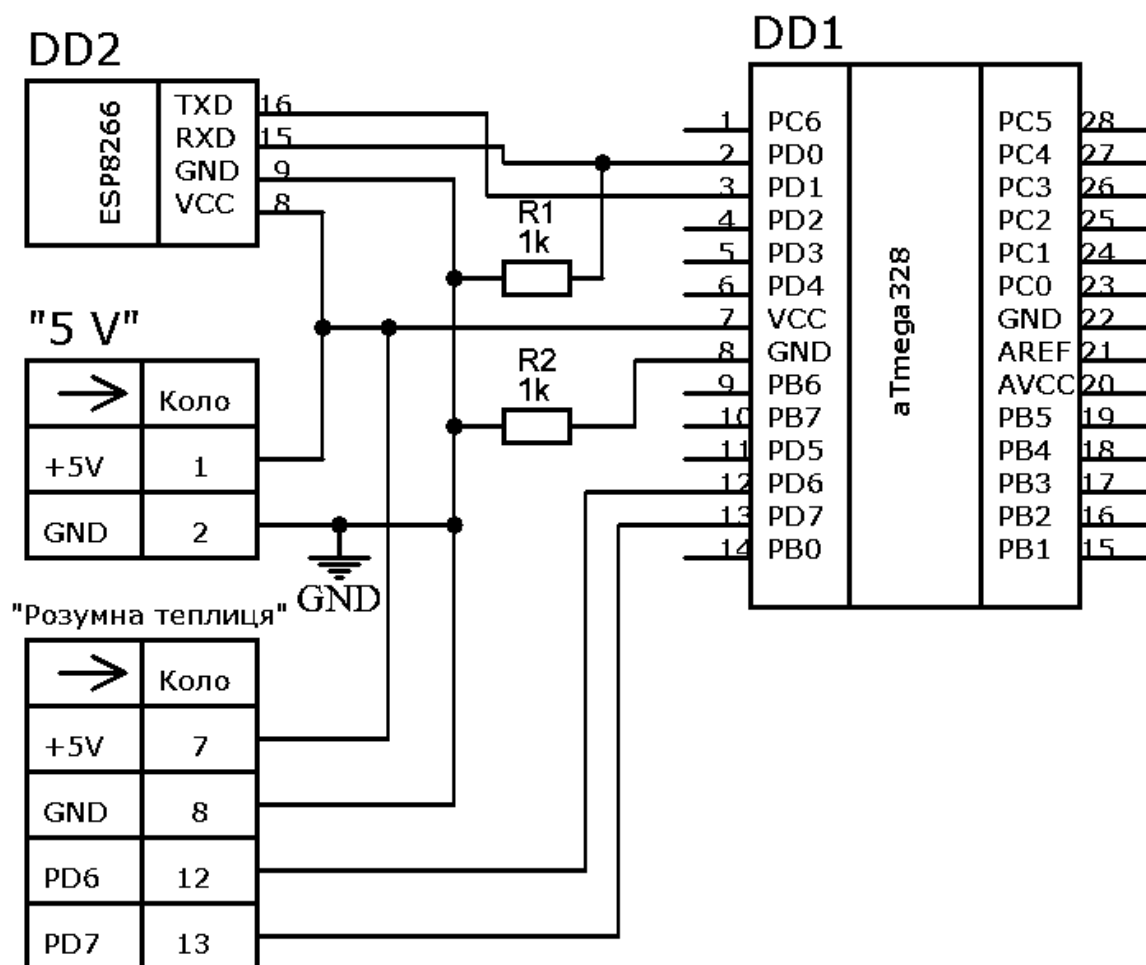


Рисунок 2.8 – блок схема алгоритму роботи системи

10. Мікроконтролер Atmega 2560 керує системою, та всіма елементами підключеними до його контактів. Збережені дані відправляються на Wi-Fi модуль ESP8266 який підключено до контактів RX та TX що виконує зв'язок між мікроконтролером та ESP модулем.

Дані із розумної теплиці відправляються на PD6, PD7.

2.5 Розробка програмного забезпечення для системи керування для розумної теплиці

Вибір середовища програмування.

Для написання програм під платформу Arduino існує декілька середовищ програмування, кожна з яких дає змогу програмувати на різних мовах, наприклад Arduino IDE; Programino; B4R; FLProg.

Розглянемо основну програму, з якою будемо працювати - Arduino IDE. IDE розшифровується як інтегроване середовище розробки, і в нашому випадку є блокнот, в якому ми пишемо код, препроцесор і компілятор, які перевіряють і компілюють код, і інструменти для завантаження, які завантажують код обраним способом. IDE написана на java, тому не дивуйтеся її довгому запуску, великому обсягу займаної оперативки і повільної роботи. Незважаючи на величезний досвід роботи з ардуіно я до сих пір продовжую писати код в Arduino IDE, тому що я до неї звик. Крім перерахованих проблем стандартна IDE виділяється на тлі "дорослих" середовищ розробки відсутністю дерева / файлової структури проекту (що не потрібно для простих проектів), відсутністю рефакторинга, відсутністю автоматичного доповнення коду (хоча його обіцяють ось-ось додати і вже є beta) і дуже повільною компіляцією коду. Цих недоліків позбавлені аналоги Arduino IDE, про них поговоримо в кінці уроку. Крім відсутності озвучених недоліків там є деякі корисні фішки, наприклад все `define`-и файлу з кодом відображаються в окремому блоці і з ними дуже зручно працювати.

Програма легка у освоєнні та розробці додатків для Arduino завдяки простій структурі.

Одразу після запуску програми ви бачите чотири області програми:

- меню;
- панель швидкого доступу;
- редактор;
- панель повідомлень і статусу.

У пункті меню:

- створити,
- зберегти
- роздрукувати вихідний код.

Розділи «Файл» і «Правка» містять стандартні параметри більшості програм.

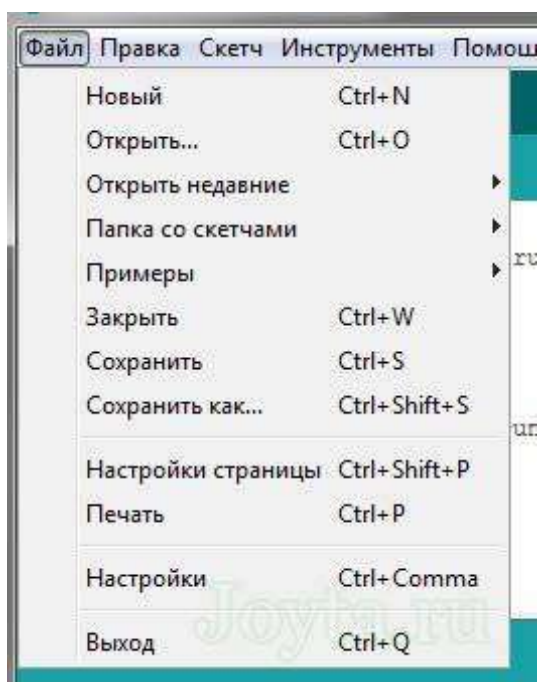


Рисунок 2.10 Меню «Файл»

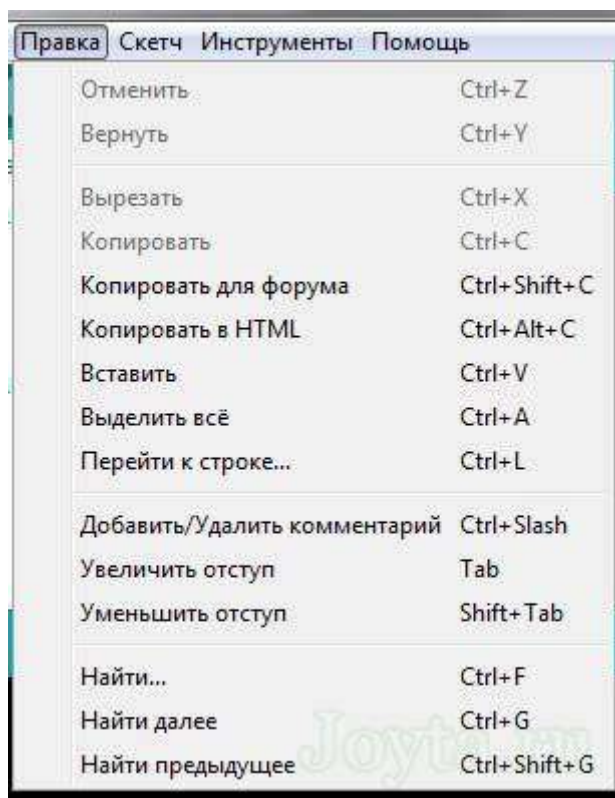


Рисунок 2.11 Меню «Правка»

Меню «Скетч» містить параметри для збору проекту і імпорту необхідних бібліотек програми.

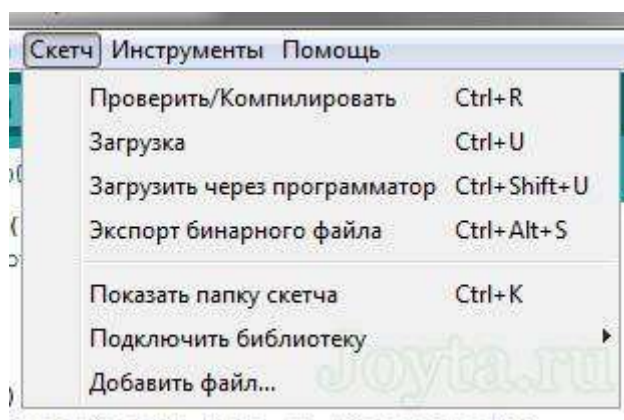


Рисунок 2.12 Меню «Скетч»

Надзвичайно корисним елементом програми є меню «Інструменти», яке включає в себе функції автоматичного форматування коду, архівування проекту, включення монітора послідовного порту (USB в Arduino розглядається як звичайний послідовний порт).

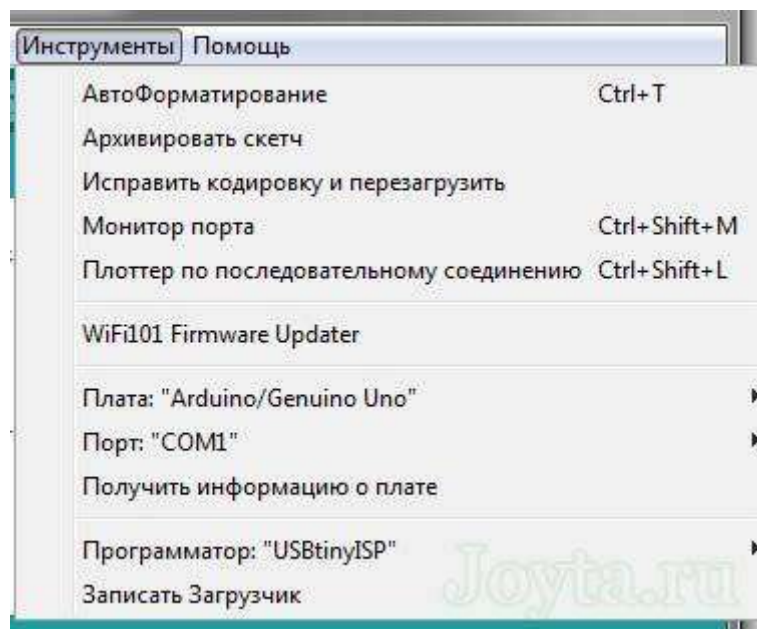


Рисунок 2.13 Меню «Інструменти»

Найбільш важливим елементом цього меню є можливість вибору відповідної плати підключеної до комп'ютера. У списку є всі офіційні версії

Arduino. Якщо ваш тип плати відсутня в списку, то ви можемо додати її, змінивши один з файлів програми.

У меню «Інструменти» ви можете також встановити порт, до якого підключена плата Arduino. Пакет Arduino IDE визначає сам порт, але іноді вручну встановити номер порту в налаштуваннях потрібно.

За допомогою Arduino IDE також можна завантажити, тобто запрограмувати Bootloader (завантажувач) для нового, чистого мікроконтролера Atmega, що дозволяє клонувати чіпи або просто замінити несправний мікроконтролера в Arduino.

Для нормальної роботи з Arduino IDE використовується панель швидкого доступу, яка оснащена важливими кнопками. Це полегшує роботу з пакетом IDE, дає прямий доступ до всім необхідним параметрам при написанні і тестуванні програми.

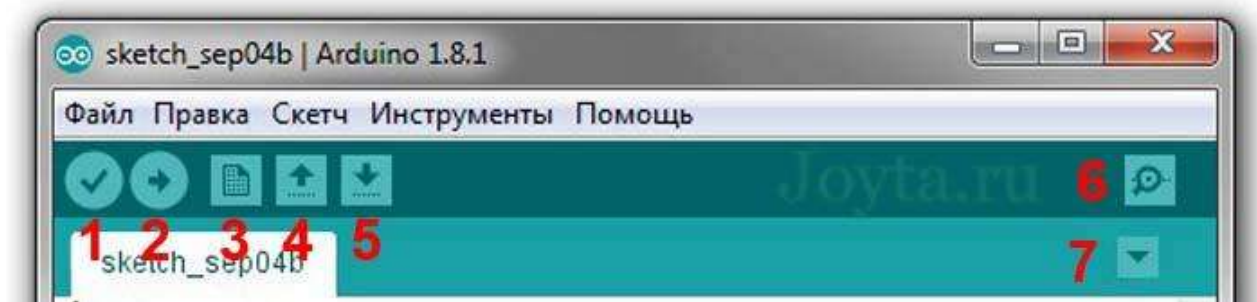


Рисунок 2.14 Панель швидкого доступу

Вони дозволяють компіляція коду без його завантаження в плату. Тобто код можна написати і перевірити на помилки навіть не підключаючи плату до комп'ютера компіляція і завантаження прошивки в плату; створити проєкт; відкрити проєкт; зберегти проєкт; відкриває монітор послідовного порта для спілкування з платою; робота з вкладками.

Додатковим корисним елементом, що знаходяться під кнопкою включення монітора послідовного порта - це меню для управління вкладками (7). Виберіть в Arduino IDE написання складних проєктів, а так само дозволяють працювати з декількома проєктами одночасно.

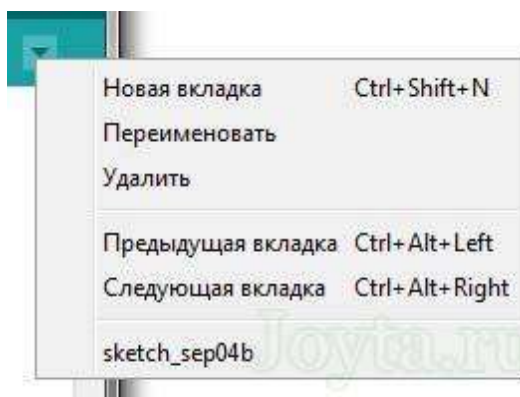


Рисунок 2.15 Меню роботи з декількома вкладками

Найбільша частина вікна програми призначена для написання безпосередньо самого коду програми. Arduino IDE не дуже просунутий, але має найважливіші елементи, що дозволяють полегшити написання простих програм, це підсвічування синтаксису і блоків (дужки). Це не багато, але достатньо для простих проектів.

Останнім елементом програми є вікно повідомлень і статусу. Видима там інформація дозволяє користувачеві знайти помилки в програмному коді і отримати підтвердження про завершення компіляції і завантаження програми в мікроконтролер.

Підводячи підсумок можна сказати, що Arduino IDE - це простий програмний пакет, який дозволяє запрограмувати будь-яку відому плату Arduino, спілкуватися з послідовним портом і легко управляти проектами.

Створення програми функціонування системи.

В процесі розробки було обрано середовище Arduino IDE тому програма для керування написана на мові C++.

Використання бібліотек полегшує написання програмного коду, адже не потрібно виділяти час на написання додаткового коду, значно зручніше підключити певну бібліотеку та звертаючись до неї виконувати поставлені завдання.

Оголошення бібліотек (рис. 2.16).

```
#include "Icdgfx.h" // бібліотека для взаємодії OLED дисплеєм
#include <EEPROM.h> // бібліотека для взаємодії з енергонезалежною пам'яттю
#include "Encoder.h" // бібліотека для зручного використання енкодера
#include <SPL.h> // бібліотека SPI інтерфейса
#include <SD.h> // бібліотека для взаємодії з модулем SD-карти
#include <iarduino_RTC.h> // бібліотека для взаємодії з модулем реального часу
```

Рисунок 2.16 – Оголошення бібліотек

Під час роботи система постійно оброблює велику кількість даних, тому для структурування та зберігання в пам'яті використовуються змінні, але в платформі Arduino змінні можливо використовувати не тільки для зручного зберігання даних, але й також для більш зручного звертання до певної ноги мікроконтролер.

Оголошення змінних та контактів (рис. 2.17).

```
const byte CLK
const byte DT = 3;
const byte SW = 4;
unsigned long millisValSensor = 30000;
String REAL_TIME::;
```

Рисунок 2.17 – Оголошення змінних та контактів

Окрім змінних для звернення до певного контакту мікроконтролера можливе використання директиви `#define`, вона дозволяє надати константі ім'я перед тим як вона буде скомпільована.

Використання змінних або директиви `#define` дуже зручне, адже присвоєння імені контакту виконується лише один раз, і надалі звернення до нього проходить по унікальному імені, це дозволяє швидко за необхідністю змінити номер контакту мікроконтролера не переписуючи весь код програми.

Використання функцій виклику спрощує написання програмного коду, а також дозволяє покращити його в майбутньому працюючи лише с тою частиною коду яка потрібна. Створення та виклик функції дозволяє розбити програмний код на блоки, це дозволяє полегшити сприйняття коду, та його наповнення.

`powerUP ();`- Виклик функції відображення часу роботи приладу.

Після того як виконання програми підходить і виконує функцію, починається виконуватись та частина яка знаходиться в середині викликаної функції (рис. 2.18)

```
void powerUP() {
  char bufchar [2];
  display.drawWindow(0, 0, 0, 0, " Power UP", true);
  display.printFixed (10, 52, "<Back", STYLE_NORMAL);
  while ( EEPROM.read(5) != 1) {
    if (millis() - millisVal >= 1000) { // якщо пройшло 5000 мс (5секунд)
      dtostrf(hour(), 2, 0, bufchar);
      display.printFixedN (10, 20, bufchar, STYLE_NORMAL, FONT_SIZE_2X);
      dtostrf(minute(), 2, 0, bufchar);
      display.printFixedN (50, 20, bufchar, STYLE_NORMAL, FONT_SIZE_2X);
      dtostrf(second(), 2, 0, bufchar);
      display.printFixedN (90, 20, bufchar, STYLE_NORMAL, FONT_SIZE_2X);
      millisVal = millis(); // сброс таймера }
    if (encoder() == 11) EEPROM.update(5, 1); }
  display.clear();
  EEPROM.update(5, 0);}

```

Рисунок 2.18 Функція powerUP та її інструкція

Оператор вибору switch дозволяє створювати зручні конструкцію розгалуження дії в залежності від значення однієї змінної.

```
switch (encoder()) {
  case 11: calibration(); break;
  case 12: main_menu(); break;
  case 3: if (EEPROM.read(9) == 0) value(0);else freeze(); break;
  case 1: value(1); break;
  case 2: value(2); break;
  case 4: sleep(); break;
  default: break; }

```

Рисунок 2.19 – Оператор вибору switch

Використання switch зручніше ніж використання оператора умови if, але для виконання певної умови яка має лише два можливих рішення, або виконання дії в залежності від виконання умови, доцільніше використання саме оператора if.

```
if (EEPROM.read(4) == 1) display.drawBitmap1(110, 8, 8, 8, Checkbox_on);
else display.drawBitmap1(110, 8, 8, 8, Checkbox_off);

```

Для взаємодії з пінами використовуються команди:

1. `analogRead` – перевіряє рівень напруги на аналоговому порті;
2. `analogWrite` – вказує який рівень напруги подати на аналоговий порт;
3. `digitalRead` – перевіряє рівень напруги на цифровому порті;
4. `digitalWrite` – вказує який рівень напруги подати на цифровий порт;
5. `pinMode` – вказує в якому режимі буде працювати певний пін.

Найшвидший спосіб спілкування з модулем ESP8266 - це дати йому команду AT і отримати відповідь.

Набір AT-команд - це спеціальний набір інструкцій, які модуль `esp8266` «знає» і може виконувати певні дії при їх отриманні та видавати терміналу результат їх виконання.

Програма, звана процесором команд AT, вже встановлена в модулі ESP8266 і готова приймати їх на послідовному порту. Ці команди починаються з символів "AT".

При запуску системи команда перевіряє наявність сигналу від мережі Wi-Fi і намагається підключитися, якщо не знайдена відома точка доступу, модулю надсилається команда для переходу в режим точки доступу, в цьому режимі система відображає локальна IP-адреса для підключення до веб-інтерфейсу системи.

Завдяки великому обсягу пам'яті модуль Wi-Fi ESP8266 дозволяє використовувати його як WebServer для управління HTTP та як FTP-сервер для доступу до файлової системи SPIFFS. Це дозволить вам використовувати ESP8266 не тільки як модуль для підключення до мережі, а й для побудови веб-інтерфейсу для управління системою віддалено або локально через веб-інтерфейс. Запити HTTP та GET використовуються для передачі даних у хмарне сховище.

Повний формат запису HTTP - це протокол передачі гіпертексту. HTTP пропонує набір правил і стандартів, що визначають спосіб передачі будь-якої інформації у Всесвітній павутині. По суті, HTTP є прикладним протоколом і забезпечує стандартні правила взаємодії трьох категорій - веб-браузерів (клієнтів), серверів та проксі-серверів.

Сам HTTP побудований поверх набору протоколів TCP. HTTP використовує структурований гіпертекстовий текст, який встановлює логічний зв'язок між вузлами, що містять текст. Він також відомий як "протокол без стану", оскільки кожна команда виконується окремо, без використання посилання на попередню команду запуску.

Основною метою протоколу HTTP є передача веб-сторінок (текстових файлів із розміткою HTML), хоча він успішно передає інші файли, які пов'язані з веб-сторінками (зображеннями та програмами) і не пов'язані з ними (в яких HTTP конкурує з більш складними FTP).

Основним об'єктом маніпуляцій у HTTP є ресурс, зазначений URI (уніфікований ідентифікатор ресурсу) у запиті клієнта. Як правило, такими ресурсами є файли, що зберігаються на сервері, але вони можуть бути логічними об'єктами або чимось абстрактним. Особливістю протоколу HTTP є можливість вказати в запиті та відповіді спосіб представлення одного і того ж ресурсу для різних параметрів: формату, кодування, мови тощо (зокрема, для цього використовується заголовок HTTP). Це обумовлено можливістю вказівки методу кодування повідомлення, що клієнт і сервер можуть обмінюватися двійковими даними, хоча цей протокол є текстовим.

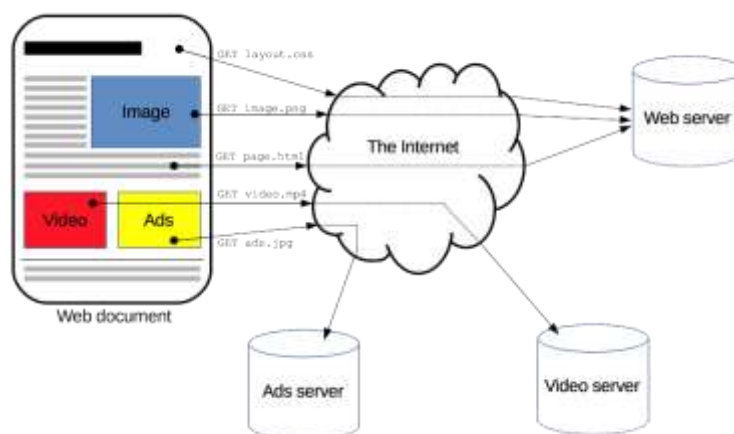


Рисунок 2.20 Принцип роботи протоколу HTTP

Обмін повідомленнями йде за звичайною схемою «запит-відповідь». Для ідентифікації ресурсів HTTP використовує глобальні URI. На відміну від багатьох інших протоколів, HTTP не зберігає свого стану. Це означає відсутність

збереження проміжного стану між парами «запит-відповідь». Компоненти, що використовують HTTP, можуть самостійно здійснювати збереження інформації про стан, пов'язаний з останніми запитами та відповідями. Браузер, котрий посилає запити, може відстежувати затримки відповідей. Сервер може зберігати IP-адреси та заголовки запитів останніх клієнтів.

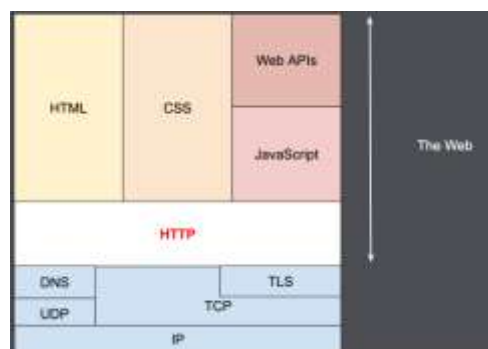


Рисунок 2.21 Структура мережі та веб-серверу

Хоча HTTP був розроблений на початку 1990-х років, завдяки своїй розширюваності він постійно вдосконалювався. HTTP - це протокол прикладного рівня, який найчастіше використовує можливості іншого протоколу - TCP (або TLS - захищений TCP) - для пересилання своїх повідомлень, але будь-який інший надійний транспортний протокол теоретично може бути використаний для доставки таких повідомлень. Завдяки своїй розширюваності він використовується не тільки для отримання клієнтом гіпертекстових документів, зображень та відео, але й для передачі вмісту серверів, наприклад, за допомогою HTML-форм. HTTP також може бути використаний для отримання лише частин документа з метою оновлення веб-сторінки на вимогу (наприклад, за допомогою запиту AJAX).

Пізніше природна розширюваність HTTP дозволила отримати більше контролю та функціональності Мережі. Кеш і методи автентифікації були першими функціями в історії HTTP. Натомість можливість ослаблення початкових обмежень була додана в 2010 році.

Нижче наведено загальні функції, якими керує HTTP.

Готівкою. Сервер може проінструктувати проксі-серверів та клієнтів: що і як довго кешувати. Клієнт може доручити кешу проксі-сервера ігнорувати збережені документи.

Послаблення обмежень джерела. Щоб запобігти шпигунському програмному забезпеченню та іншим порушенням конфіденційності, веб-браузер забезпечує чітке розділення між веб-сайтами. Лише сторінки з одного джерела можуть отримати доступ до інформації на веб-сторінці. Хоча такі обмеження навантажують сервер, заголовки HTTP можуть зменшити суворе розділення на стороні сервера, дозволяючи документу стати частиною інформації з різних доменів (з міркувань безпеки).

Аутентифікація. Деякі сторінки доступні лише для спеціальних користувачів. Базова автентифікація може бути надана через HTTP або шляхом налаштування спеціального сеансу за допомогою файлів cookie.

Проксі та тунелювання. Сервери та / або клієнти часто знаходяться в Інтернеті і приховують свої реальні IP-адреси від інших. Запити HTTP проходять через проксі-сервер, щоб перетнути цей мережевий бар'єр. Не всі проксі-сервери є HTTP-проксі-серверами. Наприклад, протокол SOCKS працює на нижчому рівні. Інші, такі як ftp, можуть бути оброблені цими довіреними особами.

Сесії. Використання файлів cookie HTTP дозволяє пов'язати запит із статусом сервера. Це створює сеанс, хоча ядро HTTP не має стану. Це корисно не тільки для кошиків в Інтернет-магазинах, але і для будь-яких сайтів, які дозволяють користувачеві налаштувати висновок.

Одним із способів відправити HTTP-запит на сервер є його запит за допомогою GET. Цей метод є найпоширенішим, і запити до сервера найчастіше надходять із його використанням.

На ранніх стадіях Інтернету були лише запити GET. Вони представляють передачу даних безпосередньо в адресному рядку браузера, маючи такий синтаксис:

`http://домен/сторінка?[параметр1=значення1][&параметр2=значення2]...`

Тут набір переданих даних на сервер починаються з символу '?' і розділяються символом '&'. Самі дані представляють собою пари параметр=значення

Наприклад, якщо потрібно передати ім'я і прізвище користувача на сторінці реєстрації (наприклад, `register.php`) сайту `mysite.com`, то це буде виглядати так:

`http://mysite.com/register.php?fname=Маргарита&lname=Волошинська`

Слід зазначити, що браузери застарілих версій можуть неправильно сприймати кирилицю, а передача російських букв буде неправильною. У GET-запитах краще передавати лише службову інформацію у вигляді цифр та слів латиницею.

Недоліком запитів GET є обмеженість переданих даних. На стороні сервера рядок запиту обмежений деяким максимальним значенням. Наприклад, якщо максимальний розмір запиту може становити 1024 символи, то все, що перевищує це значення, буде видалено, і тоді частина переданої інформації не буде оброблена вказаною сторінкою сайту. Друге істотне обмеження - це можливість передавати строго визначені набори символів. Наприклад, символи '?' і '&' вже зарезервовані і не можуть бути передані як значення параметрів. Однак це правило можна обійти, якщо в рядок запиту ви не передасте сам символ, а його значення коду. Тут використовується символ '%', а потім код символу,

наприклад:

<http://mysite.com/register.php?fname=%CC%DF%AD%1F%DS&lname=%DD>

Тут кодові значення вказані в шістнадцятковому вигляді з метою економії довжини запиту.

Незважаючи на зазначені недоліки, створення сайтів без GET-запитів було б украй важко. Наприклад, вони незамінні у випадках початкової ініціалізації сторінки сайту для конкретного користувача, коли в запиті вказується не тільки сайт і поточна сторінка, але і його id, як це зроблено в соціальній мережі Вконтакте:

<http://vk.com/profile.php?id=12345678>

Також GET-запити часто застосовуються для перевірки коректності email-адреси при реєстрації користувача. В цьому випадку користувачеві на зазначений email приходить лист з посиланням активації і ця посилання являє собою GET-запит.

Приклад GET-виклику продемонстровано на рисунку 2.22:

<http://greenhouse.iotsystem.ua/home.php?date=20191013&time=112320&temperature=20>

Рисунок 2.22 – GET-виклик

В цьому прикладі відбувається відправка даних за адресом «<http://greenhouse.iotsystem.ua/home.php>», а дані які відправляються це дані про дату, час, результат виміру, а саме:

- date=20191013 – дата, 2019 рік, 10 місяць, 13 число;
- time=112320 – час, 11 годин, 23 хвилини, 20 секунд;
- temperature=20 – дані виміру, 449 грамів.

Відображення відправлених в мережу даних відбувається на Web сторінці, на ній надається можливість переглянути отримані дані.

Вимірювання	Дата	Час	Дані
Температура, °C	21.11.2020	07:58:09	22.75
Тиск, Р	21.11.2020	07:58:11	993.67
Вологість, RH	21.11.2020	07:58:11	67.29
Освітленість, lx	21.11.2020	07:58:11	45

Рисунок 2.23 – Головна сторінка веб ресурсу

На рисунку 2.23 зображено динамічну сторінку Web ресурсу на який здійснюється відправка зібраних даних, надалі ці данні заносяться в таблицю де ми можемо спостерігати за мікрокліматом теплиці або гроубоксу.

2.6 Система розумна теплиця

Для управління теплицею потрібні датчики температури і вологості, щоб підтримувати умови для росту рослин.

Виберемо один із варіантів:

EE07 (Датчик вологості та температури з цифровим виходом):

- Діапазон вимірювання: 0...100% RH, -40...80 °C
- Точність: $\pm 2\%$ RH $\pm 0.1^\circ\text{C}$
- Виходи: Цифровий E2 (2-провідний)
- Живлення: 3.8 - 5.5V

IS42R1.01 (Датчик температури та вологості) :

- Струм споживання: 50 мА
- Діапазон вимірювання: 0%...80% RH, -40 +85 °C
- Похибка: 2%°C, 3,5% RH
- Живлення: 6-36 V

AM2320 (Датчик температури і вологості)

- Живлення: 3,1 - 5,5 В;
- Діапазон виміру: -40 – 80 °С, 0 - 99,9% відносної вологості;
- Точність: температура: $\pm 0,5$, вологість: $\pm 3\%$ вологості (10 ... 90% RH);

Найбільш привабливий EE07, тому що в нього цифровий вихід, низька вартість і висока точність. І для контролю температури і вологості повітря вибираємо AM320.

Контролер важлива частина для автоматичного контролю, він буде обробляти сигнал з датчиків, керувати зволоженням і температурою ґрунту, а також температурою і вологістю повітря. Також можна використовувати платформи такі як Arduino, RoMeo, Rapsberry Pi, вони значно спрощують підключення та використання контролера. Arduino Nano найбільш підходить так як має самі маленькі розміри, а частоти в 16 МГц повністю достатньо, кількість входів і виходів теж достатня.

Для контролю температури буде використано РТС нагрівач і кулер, розміри кулера будуть вибрані відповідно до розмірів нагрівача для максимальної ефективності нагрівача. При низькій температурі він може підігріти повітря, а при високій – буде можливість включити кулер без підігріва.

Полив буде здійснений крапельним поливом, для виконання буде достатньо низько потужного насоса. QR30E – насос 240 л/г, напруга живлення 12В.

Щоб контролювати було простіше, в систему доданий дисплей який буде відображати температуру і вологість ґрунту, повітря. Виберемо найкращий із LCD 1602, TFT 2.4, LCD 2004.

Для зберігання звітності, а також щоб вмістити програмну частину потрібна додаткова пам'ять, використаємо накопичувач. Виберемо модуль TF SD Card Adapter, тому що він дозволяє використовувати до 16Гб карту пам'яті, цього цілком достатньо для завантаження програмного коду, а також запису звітів про стан в теплиці, для порівняння або досліджень.

З ДОСЛІДЖЕННЯ ПРАЦЕЗДАТНОСТІ ТА ЕФЕКТИВНОСТІ СИСТЕМИ ТА ЇЇ КОНТРОЛЮ

В даному розділі проведено дослідження працездатності та ефективності системи. Дослідження даної системи в першу чергу проводиться з моделювання. Розробка макету системи передбачає написання коду роботи системи, що виконується у програмному середовищі Arduino IDE, та безпосередньо із збору макетної моделі приладу. Дослідженням точності вимірів буде виступати порівняння зібраних даних із технічними характеристиками приладу та технічним завданням, таким чином можна визначити точність приладу та допустимі відхилення в порівнянні з технічними характеристиками. На останок рекомендації до застосування, області найбільшої необхідності та регіони рекомендовані для використання в зв'язку з неблагополучними умовами.

3.1 Імітаційне моделювання системи розумна теплиця

Перед безпосередньою зборкою макету було прийнято рішення змоделювати фрагменти системи, це дасть можливість запобігти певних труднощів під час збору макету системи.

Для моделювання віртуальних систем на базі Arduino існує певна кількість сервісів, а саме:

1. Yenska
2. Proteus
3. Virtual Bread Bord
4. Cisco Packet Tracer
5. Tngercad Circuits

Найкращим вибором серед перелічених сервісів є Tngercad Circuits, він дає можливість моделювання в середовищі браузера, а також має найбільшу базу потрібних компонентів, це дає змогу майже повного моделювання системи.

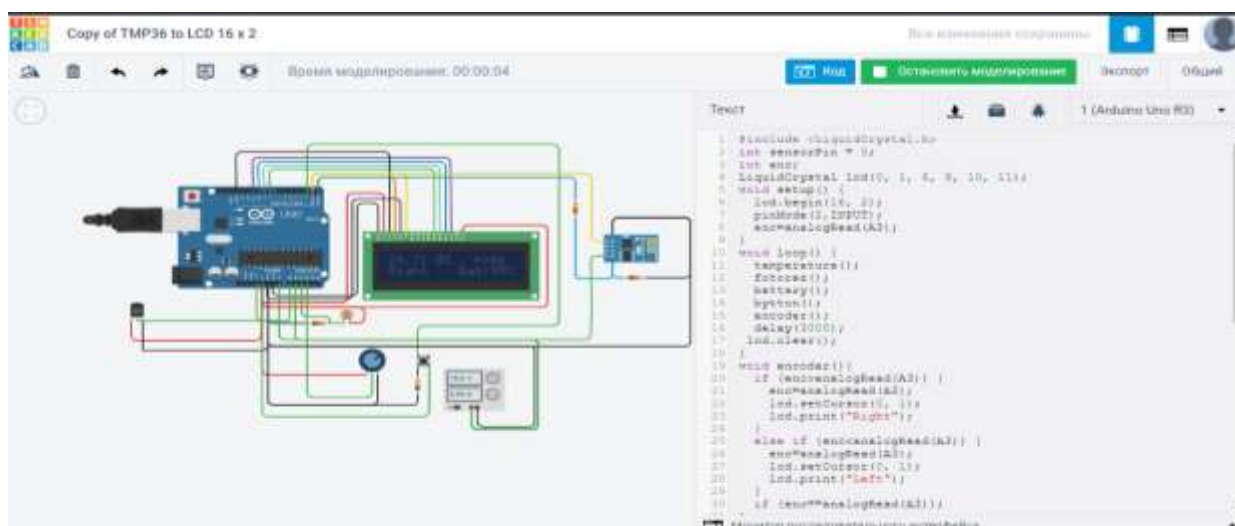


Рисунок 3.1 – вигляд системи розумна теплиця в програмі для моделювання

В якості платформи Arduino використано Arduino Uno до якої підключено такі компоненти:

1. Символьний дисплей 1604
2. Енкодер
3. Тактова кнопка
4. Блок живлення
5. Фоторезистор
6. Датчик температури TMP36

Обрані при моделюванні компоненти відрізняються від обраних компонентів системи, адже компоненти системи відсутні в середовищі моделювання.

Символьний дисплей 1604 підключений до Arduino даний дисплей потребує значно більшої кількості контактів для підключення ніж обраний нами з типом підключення I2C. На дисплей відображається інформація про стан кнопки, оберти енкодера, температура, рівень освітленості та процент заряду акумулятора.

Використання модулю енкодера при моделюванні дозволяє нам зрозуміти в який бік було виконано оберт, результат цієї дії відображається на екрані.

Тактова кнопка розширює можливість керування, при натисканні на кнопку здійснюється фіксація натискання.

Блок живлення виступає в ролі імітації акумулятора та виводить на дисплей рівень заряду акумулятора змінюючи напругу на блоці живлення – змінюємо показник рівня заряду.

3.2 Дослідження точності вимірів датчиків системи розумна теплиця

Для дослідження точності вимірів з допомогою моделювання було проведено вимірювання мікроклімату кімнати з взаємодією різних збудників (лампочки, обігрівача, провітрювання приміщення та інших). Частину отриманих даних про температуру, вологість, тиск та освітлення занесемо в таблиці 3.1 та побудуємо графіки.

Таблиця 3.1 Дані вимірів температури

№	Дата	Час	Температура
1.	21.11.2020	23:32:56	22,00
2.	21.11.2020	23:33:10	33,00
3.	21.11.2020	23:33:25	32,00
4.	21.11.2020	23:33:39	32,00
5.	21.11.2020	23:33:54	26,00
6.	21.11.2020	23:34:09	24,00
7.	21.11.2020	23:34:24	24,00
8.	21.11.2020	23:34:40	23,00
9.	21.11.2020	23:34:55	23,00
10.	21.11.2020	23:35:11	23,00
11.	21.11.2020	23:35:26	22,00
12.	21.11.2020	23:35:41	22,00
13.	21.11.2020	23:35:57	23,00
14.	21.11.2020	23:36:12	29,00
15.	21.11.2020	23:36:27	32,00

Таблиця 3.1 Продовження

16.	21.11.2020	23:36:41	34,00
17.	21.11.2020	23:36:55	34,00
18.	21.11.2020	23:37:09	35,00
19.	21.11.2020	23:37:24	35,00
20.	21.11.2020	23:37:38	35,00

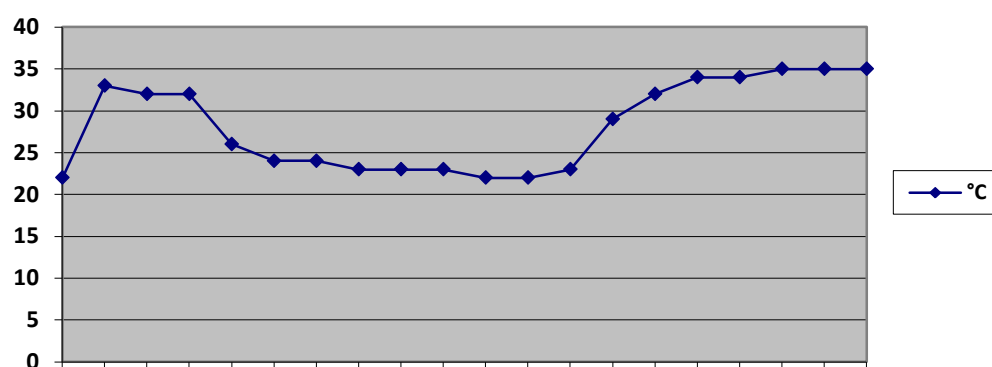


Рисунок 3.2 – Графік вимірів температури

Таблиця 3.2 Дані вимірів вологості

№	Дата	Час	Вологість
1.	21.11.2020	23:32:56:48	13
2.	21.11.2020	23:33:10:30	7
3.	21.11.2020	23:33:25:30	9
4.	21.11.2020	23:33:39:29	68
5.	21.11.2020	23:33:54:36	5
6.	21.11.2020	23:34:09:38	17
7.	21.11.2020	23:34:24:39	21
8.	21.11.2020	23:34:40:40	51
9.	21.11.2020	23:34:55:41	27
10.	21.11.2020	23:35:11:42	0
11.	21.11.2020	23:35:26:41	85

Таблиця 3.2 Продовження

12.	21.11.2020	23:35:41:42	11
13.	21.11.2020	23:35:57:42	93
14.	21.11.2020	23:36:12:33	86
15.	21.11.2020	23:36:27:30	87
16.	21.11.2020	23:36:41:29	9
17.	21.11.2020	23:36:55:28	4
18.	21.11.2020	23:37:09:27	57
19.	21.11.2020	23:37:24:27	6
20.	21.11.2020	23:37:38:27	6

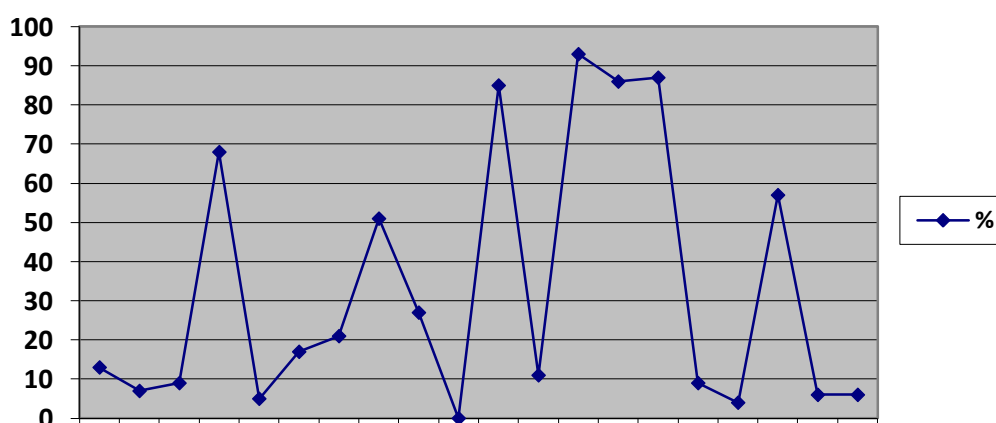


Рисунок 3.3 – Графік вимірів вологості

Таблиця 3.3 Дані вимірів тиску

№	Дата	Час	Тиск
1.	21.11.2020	23:32:56	987,50
2.	21.11.2020	23:33:10	987,49
3.	21.11.2020	23:33:25	987,60
4.	21.11.2020	23:33:39	987,60
5.	21.11.2020	23:33:54	987,67
6.	21.11.2020	23:34:09	987,70

Таблиця 3.3 Продовження

№	Дата	Час	Тиск
7.	21.11.2020	23:34:24	987,67
8.	21.11.2020	23:34:40	987,76
9.	21.11.2020	23:34:55	987,86
10.	21.11.2020	23:35:11	987,71
11.	21.11.2020	23:35:26	987,84
12.	21.11.2020	23:35:41	987,99
13.	21.11.2020	23:35:57	988,16
14.	21.11.2020	23:36:12	987,98
15.	21.11.2020	23:36:27	988,14
16.	21.11.2020	23:36:41	988,15
17.	21.11.2020	23:36:55	988,12
18.	21.11.2020	23:37:09	988,10
19.	21.11.2020	23:37:24	988,08
20.	21.11.2020	23:37:38	988,02

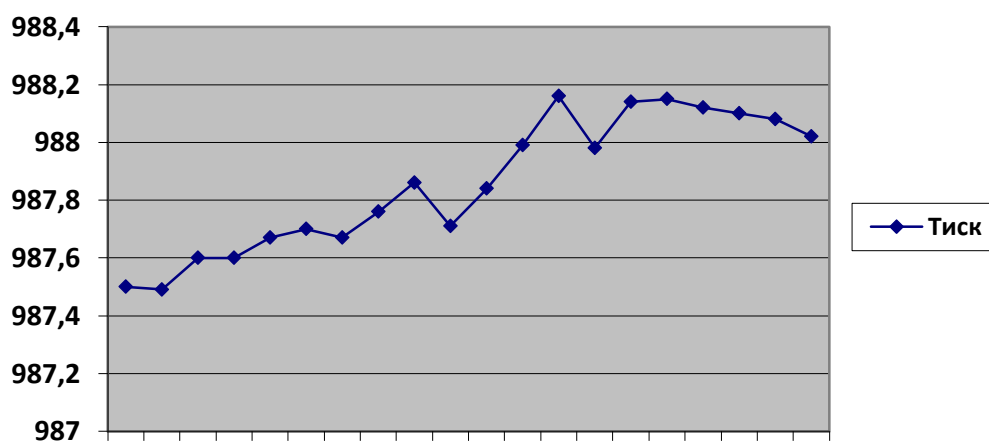


Рисунок 3.4 – Графік вимірів тиску

Таблиця 3.4 Дані вимірів освітленості

№	Дата	Час	Освітленість
1.	21.11.2020	23:32:56	4,00
2.	21.11.2020	23:33:10	61,00
3.	21.11.2020	23:33:25	35,00
4.	21.11.2020	23:33:39	22,00
5.	21.11.2020	23:33:54	19,00
6.	21.11.2020	23:34:09	7,00
7.	21.11.2020	23:34:24	6,00
8.	21.11.2020	23:35:11	5,00
9.	21.11.2020	23:35:26	6,00
10.	21.11.2020	23:35:57	5,00
11.	21.11.2020	23:36:27	4,00
12.	21.11.2020	23:36:41	5,00
13.	21.11.2020	23:37:38	4,00
14.	21.11.2020	23:37:52	5,00
15.	21.11.2020	23:38:06	3,00
16.	21.11.2020	23:39:36	2,00
17.	21.11.2020	23:30:53	2,00
18.	21.11.2020	23:31:09	1,00
19.	21.11.2020	23:32:27	1,00
20.	21.11.2020	23:32:42	0,00

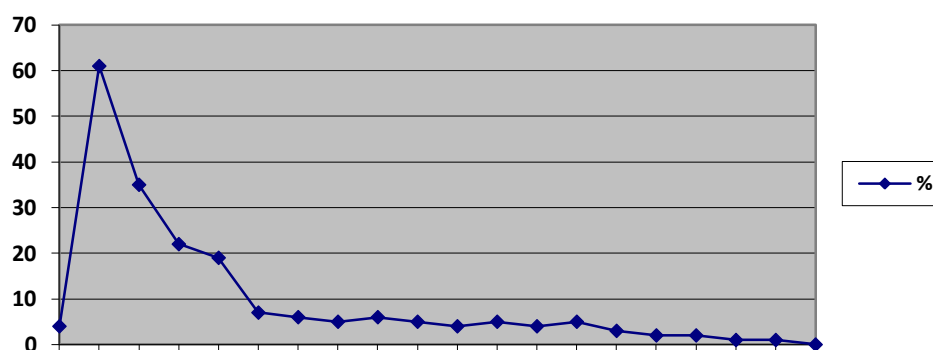


Рисунок 3.5 – Графік вимірів освітленості

Аналіз даних для дослідження показує що температура в приміщенні змінюється повільно. Але при створенні умов підвищеної температури (включення нагрівача) температура зростає. Це свідчить про правильне функціонування датчика.

Вологість змінюється стрибкоподібно що пов'язане з умовами розміщення датчика. При відкритті вікон вологе повітря потрапляє до датчика тим самим змінюючи показники, а під дією нагрівача вологість повітря зменшується.

3.3 Моделювання та дослідження точності відклику системи контролю для розумної теплиці

Зберемо систему для моделювання на базі Arduino UNO та модуля WiFi ESP 8266-01.

Завдання моделювання полягає в точності відправлених даних на віддалений сервер, а також швидкість відклику системи на корекційні завдання користувача(корекція температури/вологості) на різних швидкостях інтернету.

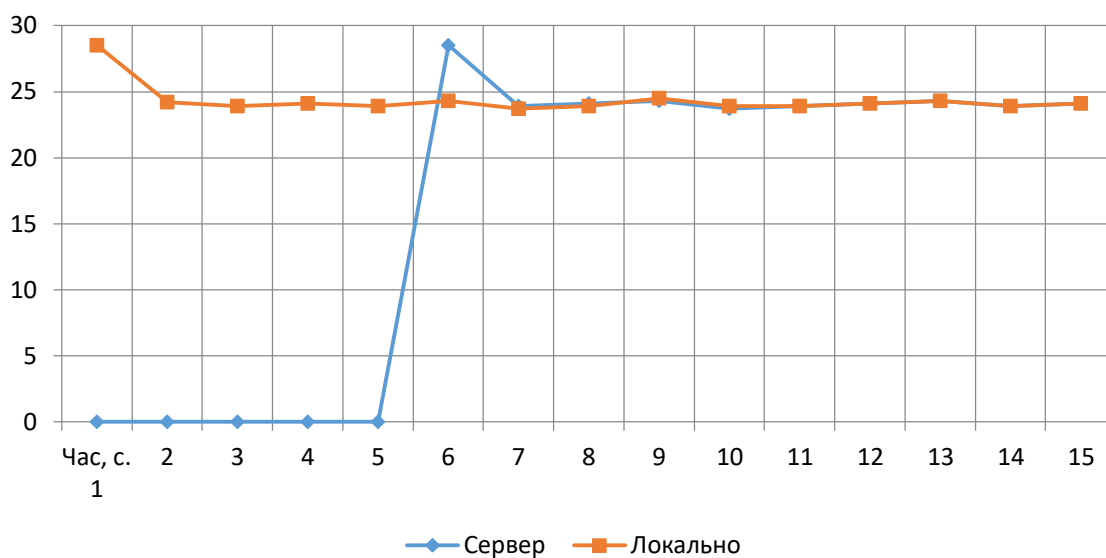
В налаштування маршрутизатора обмежим швидкість інтернет з'єднання до 5Кбіт/с, 32Кбіт/с, 64Кбіт/с, 128Кбіт/с, 256Кбіт/с. Знімимо показники при моніторингу відображення локально та на сервері:

Таблиця 3.5 Дані вимірів відклику системи

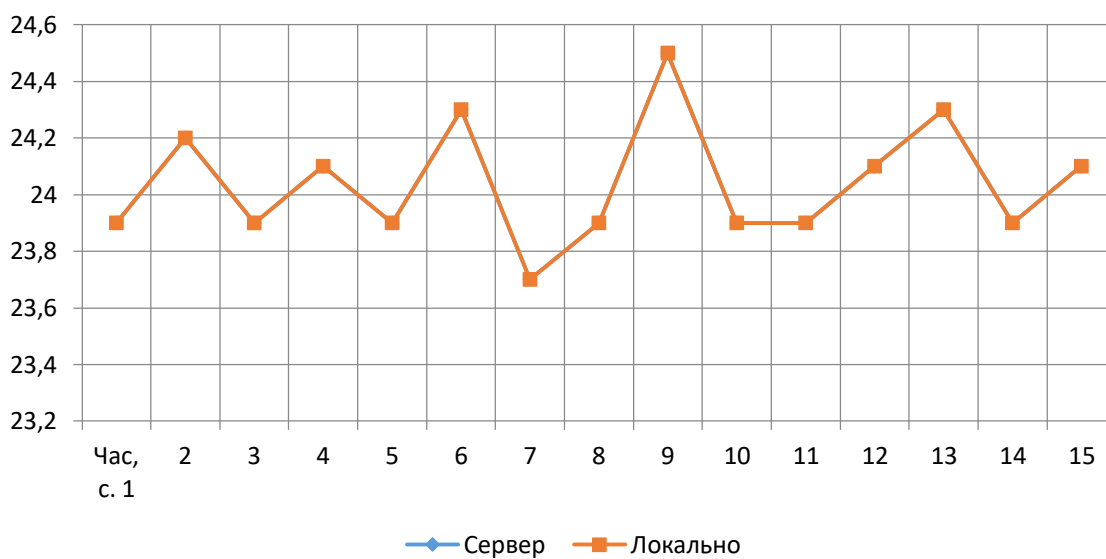
	5 Кбіт/с	32 Кбіт/с	64 Кбіт/с	128 Кбіт/с	256 Кбіт/с
Зміна показників на сервері	7 секунд	1 секунд	0,6 секунди	0,1 секунди	<0,1 секунди
Відклик на корегування користувача	10 секунд	3 секунд	1 секунда	0,3 секунди	<0,1 секунди

Для кращої картини складемо графіки моніторингу температури на сервері та локально, на швидкостях 5Кбіт/с, 64Кбіт/с:

5 Кбіт/с



64 Кбіт/с



Зрівнюючи графіки стає зрозуміло що на швидкості 5 Кбіт/с йдуть втрати інформації, а на 64 Кбіт/с графіки повністю співпадають.

ВИСНОВКИ

В кваліфікаційній роботі виконане актуальне завдання по розробці системи контролю для розумної теплиці. Серед отриманих результатів слід визначити наступні:

Проведено обґрунтування актуальності теми роботи по розробці системи контролю для розумної теплиці, де вказано що такі системи дозволяють покращити контроль теплиць.

Аналіз існуючих рішень систем контролю для розумної теплиці підтвердив, що тема є актуальною, адже аналогів даної системи досить мало а їх ринкова вартість не дає змоги впроваджувати дані системи масштабно.

Проведено аналіз елементів системи та на основі цього розроблена структурна схема, алгоритм роботи системи та алгоритм роботи системи контролю для розумної теплиці.

Керуючись всіма обраними елементами та обравши найбільш підходящі було розроблено електричну принципову схему.

Найзручнішим середовища програмування для написання програми функціонування мікроконтролеру було обрано Ardunino IDE. Програма функціонування системи розроблена на мові C++ з використанням фреймворку wiring . Методом відправки даних на хмарне сховище обрано GET-запит протоколу HTTP.

Обрано сервіс для імітаційного моделювання та здійснено імітаційне моделювання з допомогою веб сервісу TinkerCAD Circuits.

В ході дослідження точності вимірів було виявлено, що зібраний пристрій має дуже малу похибку а зазначені у технічних характеристиках похибки датчиків відповідають дійсності. При дослідженні на відклик виявлено мінімальну рекомендовану швидкість інтернет з'єднання – 64 Кбіт/с.

Таким чином завдання на кваліфікаційну роботу виконано в повному обсязі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Іванов І. П. "Мікроконтролери і Arduino: проектування і застосування". – Харків: Вища школа, 2017. – 450 с.
2. Петров М. В., Бабич О. М. "Дистанційне керування теплицею за допомогою Arduino". Технічні науки та інженерія. 2018. № 5. С. 23-28.
3. Іваненко Л. Г. "Arduino та IoT в системах автоматизації теплиць". Журнал електроніки. 2016. № 3. С. 90-95.
4. De Luca, V., & Fiumara, M. "Arduino-Based Greenhouse Control System". International Journal of Engineering Research and Applications. 2014. № 4. С.1-6.
5. Смирнов М. О. "Програмування Arduino: основи і практичне застосування". – Київ: Політехніка, 2018. – 320 с.
6. Smaragdakis, A., & Skianis, H. "Remote Monitoring and Control of Greenhouse Using Arduino and IoT". International Journal of Advanced Research in Computer Science and Software Engineering. 2015. № 5(6). С. 40-45.
7. Шевченко В. О. "Системи автоматизації на базі Arduino". – Львів: Українська академія друкарства, 2015. – 295 с.
8. Жданов І. В. "Робототехніка та Arduino: від простого до складного". – Київ: Факт, 2016. – 280 с.
9. Вакуленко М. І. "Arduino для всіх: практичні проекти". – Одеса: Студія "Фоліо", 2017. – 312 с.
10. Дубовий І. П. "Проектування систем автоматизації теплиць на базі мікроконтролерів". Сучасні інформаційні технології. 2017. № 12. С. 45-52.
11. Карпенко В. П. "Автоматизація контролю клімату в теплиці на базі Arduino". Сільськогосподарські технології. 2015. № 7. С. 38-44.
12. Колесник О. І. "Система моніторингу теплиць на основі мікроконтролера Arduino". Технічна творчість. 2018. № 2. С. 16-22.
13. Іщенко О. М. "Автоматизовані системи на базі Arduino: застосування у сільському господарстві". Прикладна електроніка. 2017. № 8. С. 67-72.
14. Ковальчук М. В. "Моніторинг навколишнього середовища в теплиці за допомогою Arduino". Журнал сучасної техніки. 2016. № 4. С. 58-63.
15. Білозерова І. О. "Автоматизовані системи для теплиць: Arduino та IoT". Журнал сільськогосподарських технологій. 2015. № 9. С. 77-82.

- 16.Литвиненко П. В. "Інтелектуальні системи для теплиць на основі Arduino".
Технічні рішення. 2017. № 6. С. 35-40.



Рис. А1 – Алгоритм роботи системи



Рис.А2 – Структурна схема

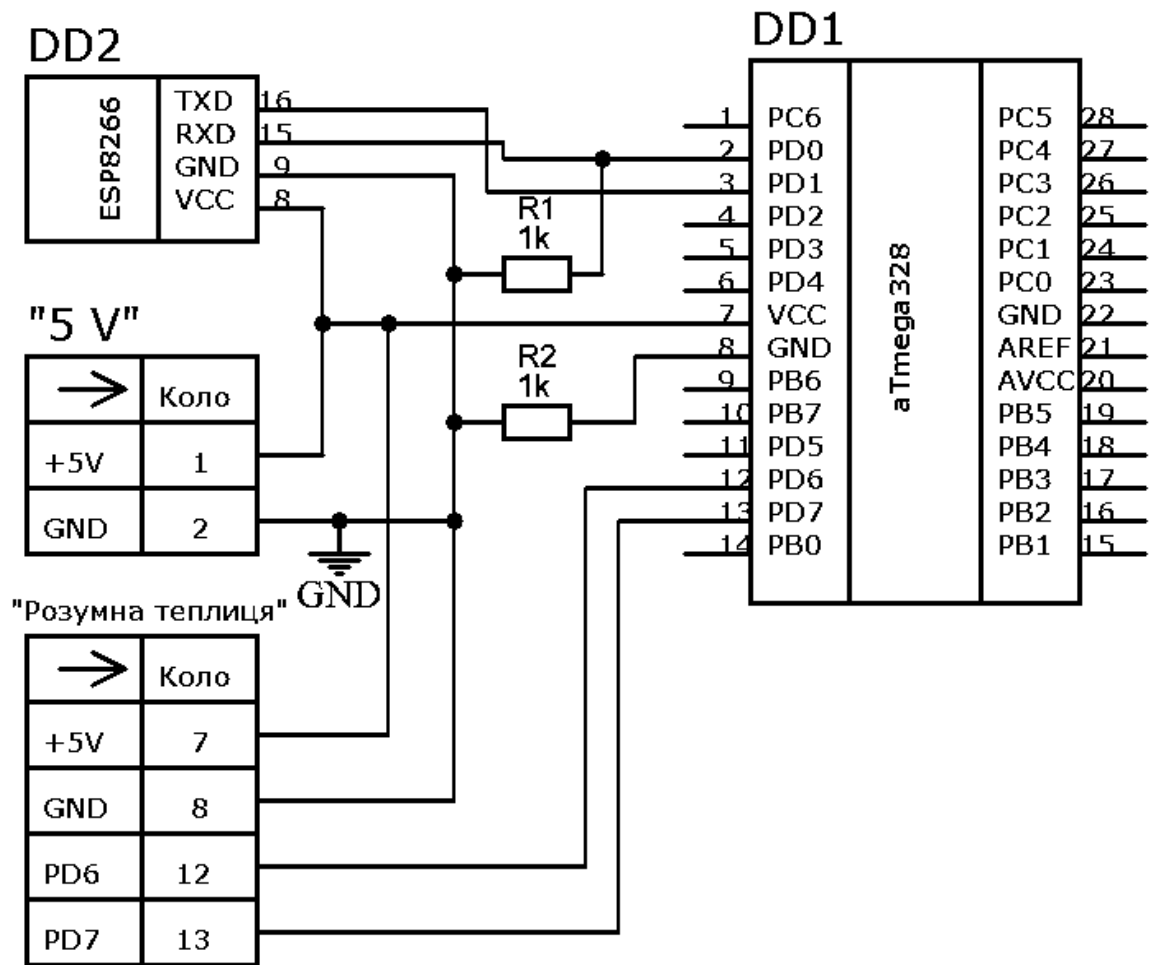


Рис. Б1 – Електрично-принципова схема контролю розумної теплиці

Програма мікроконтролера

```

#include "lcdgfx.h"
#include <EEPROM.h>
#include "Encoder.h"/
#include <SPI.h>
#include <SD.h>
#include <iarduino_RTC.h>
#include <EEWrap.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>
#include <TimeLib.h>
File myFile;
Adafruit_BME280 bme;
#define CLK 46
#define DT 44
#define SW 48
#define RST 5
#define DAT 6
#define KLC 7
#define SEALEVELPRESSURE_HPA (1013.25)
DisplaySSD1306_128x64_I2C display(-1);
iarduino_RTC time(RTC_DS1302, RST, KLC, DAT);
Encoder enc1(CLK, DT, SW);
const byte chipSelect PROGMEM = 4;
unsigned long millisVal;
void setup () {
  pinMode(A0, INPUT);
  attachInterrupt(0, interapt, RISING);
  Serial.begin(9600);
  enc1.setType(TYPE1);
  enc1.setTickMode(AUTO);
  time.begin();
  bme.begin();
  display.begin();
  display.clear();
  display.setFixedFont(ssd1306xled_font6x8);
  settings();
notSD: if (!SD.begin(chipSelect)) {
  EEPROM.update(2, 0);
  if (EEPROM.read(7) == 0) {
    display.clear();
    display.drawBitmap1(0, 0, 128, 64, NotSD);
    goto notSD;
  }
} else EEPROM.update(2, 1);
display.clear();
if (EEPROM.read(4) == 1) calibration();
switch (EEPROM.read(0)) {
  case 1: calibration() ; break;

```

```

    case 2: main_menu(); break;
    default: break;
}
}
void loop () {
    homeUi(); switch (encoder()) {
        case 11: calibration(); break;
        case 12: main_menu(); break;
        case 3: if (EEPROM.read(9) == 0) value(0);else freeze(); break;
        case 1: value(1); break;
        case 2: value(2); break;
        case 4: sleep(); break;
        default: break;
    }
    //Serial.println( 'EEPROM.read(5)');
}
void(* resetFunc) (void) = 0;
int8_t tempval;
byte encoder() {
    //if (enc1.isTurn()) display.fill(0x00);
    if (enc1.isLeft()) return 1; //left
    if (enc1.isRight()) return 2; //right
    if (enc1.isLeftH()) return 11; //leftH
    if (enc1.isRightH()) return 12; //rightH
    if (enc1.isClick()) return 3;
    if (enc1.isHoleded()) return 4;
    return 0;
}
void homeUi() {
    byte procent PROGMEM = 100;
    if (EEPROM.read(1) != 1) display.drawBitmap1(0, 0, 10, 8, Hotspot); else {
        display.drawBitmap1(0, 0, 10, 8, Wi_Fi);
    }
    if (EEPROM.read(2) == 1) display.drawBitmap1(11, 0, 6, 8, sd);
    if (EEPROM.read(3) == 0)display.drawBitmap1(115, 0, 13, 8, Not_battery);
    else if (procent <= 25) display.drawBitmap1(115, 0, 13, 8, battery_low);
    else if (procent <= 75) display.drawBitmap1(115, 0, 13, 8, battery_half);
    else display.drawBitmap1(115, 0, 13, 8, battery_full);
    if (millis() - millisVal >= 1000) {
        value(0);
        display.printFixed (30, 0, time.gettime("D.d.m.y"), STYLE_NORMAL);
        display.printFixed (45, 8, time.gettime("H:i:s"), STYLE_NORMAL);
        millisVal = millis();
    }
    display.printFixed (0, 60, "<Calibration", STYLE_NORMAL);
    display.printFixed (97, 60, "Menu>", STYLE_NORMAL);
}
unsigned long millisValSensor = 30000;
String REAL_TIME;
float SensorT;
float SensorP;
float SensorRH;

```

```

float SensoL;
void SD_Sensor() {
  // Serial.println(millisValSensor);
  if (millis() - millisValSensor >= 300000)
  {
    REAL_TIME = time.gettime("d-m-Y,H:i:s");
    Temperature_data();
    Pressure_data();
    Humidity_data();
    Lighting_data();

    millisValSensor = millis();
  }
}
void Temperature_data() {
  if (bme.readTemperature() != SensorT) {
    SensorT = bme.readTemperature();
    REAL_TIME = time.gettime("d-m-Y,H:i:s");
    myFile.close();
    myFile = SD.open("Sensor/Temper.txt", FILE_WRITE);
    myFile.print(REAL_TIME);
    myFile.print(";");
    myFile.println (SensorT);
    myFile.close();
    // Serial.println("OK");
  }
}
void Pressure_data() {
  // bme.readPressure() / 100.0F
  if (bme.readPressure() / 100.0F != SensorP) {
    SensorP = bme.readPressure() / 100.0F;
    myFile.close();
    myFile = SD.open("Sensor/Press.txt", FILE_WRITE);
    myFile.print(REAL_TIME);
    myFile.print(";");
    myFile.println (SensorP);
    myFile.close();
    // Serial.println("OK");
  }
}
void Humidity_data() {
  if (bme.readHumidity() != SensoRH) {
    SensoRH = bme.readHumidity();
    myFile.close();
    myFile = SD.open("Sensor/HR.txt", FILE_WRITE);
    myFile.print(REAL_TIME);
    myFile.print(";");
    myFile.println (SensoRH);
    myFile.close();
    // Serial.println("OK");
  }
}

```

```

void Lighting_data() {
  if (fotorez() != SensoL) {
    SensoL = fotorez();
    myFile.close();
    myFile = SD.open("Sensor/Light.txt", FILE_WRITE);
    myFile.print(REAL_TIME);
    myFile.print(";");
    myFile.println (SensoL);
    myFile.close();
    //Serial.println("OK");
  }
}

```

```

float mint = 100;
float maxt;
float minp = 100000;
float maxp;
float minrh=100;
float maxrh;
float minl = 100;
float maxl;

```

```

int fotorez(){
  int fot;
  fot = analogRead(A1);
  fot = map (fot, 1, 900, 0, 100);
  fot = constrain (fot, 0, 99);
  return(fot);
}

void clear_disp() {
  display.setColor(0);
  display.fillRect(0, 20, 128, 50);
  display.setColor(1);
}

```