МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет

Факультет електроніки та інформаційних технологій

Кафедра електроніки, загальної та прикладної фізики

«До захисту допущено»

Завідувач кафедри

Іван ПРОЦЕНКО

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня магістр

зі спеціальності 171 – Електроніка

освітньо-наукової програми «Електронні інформаційні системи»

на тему: **СИСТЕМИ КЕРУВАННЯ ТА КОНТРОЛЮ ФІЗИЧНОГО ЕКСПЕРИМЕНТУ НА ОСНОВІ МІКРОКОНТРОЛЕРІВ AVR**

Здобувача групи ЕП.м-11н Замятіна Дмитра Євгеновича

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Дмитро ЗАМЯТІН

Керівник, доцент кафедри
електроніки, загальної та прикладної
фізики, д.ф.-м.н., професор Ігор ШПЕТНИЙ

Суми – 2023

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра електроніки, загальної та прикладної фізики
Спеціальність 171 – Електроніка, освітньо-науковая програма
«Електронні інформаційні системи»

ЗАТВЕРДЖУЮ
Зав. кафедри ЕЗПФ
І.Ю. Проценко
«03» травня 2023 року

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ МАГІСТРА

Замятіна Дмитра Євгеновича

Тема роботи **Системи на основі мікроконтролерів AVR для керування та контролю фізичного експерименту**

затверджена наказом по університету від «26» квітня 2023 р., № 0426-VI

2. Термін здачі студентом закінченої роботи 24 травня 2023 року

3. Вихідні дані до роботи (актуальність, мета) Ідея того, що пристрої можуть обмінюватися інформацією один з одним без участі людини з'явилася досить давно. Ще наприкінці 70-х років розглядалась можливість повної автоматизації передачі даних. Тоді подібний підхід мав назву pervasive computing. Технологіям знадобилося кілька десятиліть розвитку для того, щоб нарешті це стало можливим. Такі технології знайшли застосування у реалізації технологій «Інтернету речей». Розвиток технологій автоматизації процесів, у тому числі керування та контролю фізичного експерименту став можливим завдяки розвитку мікроконтролерів AVR. Мета кваліфікаційної роботи здобувача полягала у розробці пристрою для керування системою обігріву та кондиціонування приміщення.

4. Зміст розрахунково-пояснювальної записки (перелік питань, що належить їх розробити)
1. У літературному огляді представити історію розробки мікропроцесорів AVR та їх архітектуру.
2. Показати особливості платформи Arduino. Описати програмні засоби, що дозволяють застосовувати сучасну апаратну платформу Arduino для реалізації

системи керування та контролю.

3. Навести результати експерименту, описати програмне забезпечення та вказати висновки.

4. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Слайд № 1. Актуальність і мета роботи.

Слайди № 2-3. Архітектура AVR мікроконтролерів.

Слайди № 4-5. Застосування апаратної платформи Arduino.

Слайди № 6-8. Експериментальні результати, отримані з допомогою електронної системи на основі мікроконтролера Arduino

Слайд № 9: Висновки.

Слайд № 10: Список публікацій. Подяка.

6. Дата видачі завдання 03.05.2023 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів виконання кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1.	Аналіз літературних даних	до 07.05.2023 р.	*вик.*
2.	Проведення експерименту, моделювання, розрахунків, обробка результатів	до 21.05.2023 р.	*вик.*
3.	Оформлення тексту кваліфікаційної роботи.	до 24.05.2023 р.	*вик.*
4.	Попередній захист роботи	25.05.2023 р., онлайн	*вик.*
5.	Захист кваліфікаційної роботи	30.05.2023 р., 10-05 – 13-00 онлайн	

Здобувач вищої освіти Замятін Д.Є.

Науковий керівник Шпетний І.О.

АНОТАЦІЯ

Обсяг роботи: 40 сторінок, 13 рисунків, 3 таблиці, 20 використаних джерел.

Актуальність теми роботи. Відомо, що електронні системи на основі мікроконтролерів знаходять все більше застосування як вбудовані системи для вирішення завдань керування об’єктом. Важливою особливістю даного застосування є робота в реальному часі, тобто забезпечення реакції на зовнішні події протягом певного часового інтервалу. Кожного року з розвитком розумних технологій зростає і попит на них. Розумні термостати не стали виключенням, тому що вони підвищують зручніть та комфорт проживання у власній оселі.Популярністю у розробників користуються 8-бітові мікроконтролери PIC від фірми «Microchip Technology» і «AVR» від фірми «Atmel». Мікроконтролери AVR, а особливо плати Arduino, мають доволі великий потенціал у використанні як у промислових проєктах, так і доволі малих, котрі доступні будь-якій людині, навіть якщо вона не знайома з основами програмування.

Мета роботи полягає в розробці пристрою на базі мікропроцесорної плати Arduino Nano, котра дозволить керувати системою обігріву та кондиціонування у приміщенні приватного будинку.

Методи: під час виконання роботи використовували програмне забезпечення Arduino IDE та прилад системи контролю температури та управління кондиціонуванням.

Отримані результати:

1. Змодельований проєкт розумного термостату на основі мікроконтролера Arduino. Показані необхідні схеми для розробки ідентичного.

2. Наведений приклад розводки виводів на платі Arduino.

3. Продемонстровані ключові частини коду необхідні для роботи приладу

Рекомендації щодо використання: Отримані результати можна використати для вдосконалення функціональності та забезпечення простоти у використанні саморобних термостатів на основі Arduino.

Апробація: Результати роботи опубліковані у Матеріалах І Міжнародна науково-теоретична конференція «Сучасні засоби та методи наукових досліджень» (26 травня 2023 р.; Антверпен, Бельгія).

Ключові слова: Arduino, AVR, IDE, Експеримент, Мікроконтролер, Термостат.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ ТА ТЕРМІНІВ

MCU - Micro Controller unit (Блок мікроконтролера)

RISC - Reduced Instruction Set Computing (Обчислення зі скороченим набором команд)

EEPROM - Electrically Erasable Programmable Read-Only Memory (Програмована пам'ять тільки для читання з електричним стиранням)

RAM - Random Access Memory (Пам'ять з довільним доступом)

ARM - Advanced RISC Machine (Удосконалена обчислювальна машина зі скороченим набором інструкцій)

JTAG - Join Test Action Group (Назва робочої групи з розробки стандарту IEEE 1149)

ALU - Arithmetic Logic Unit (Арифметико-логічний пристрій)

HVAC - Heating, Ventilation and Air Conditioning (Система опалення, вентиляції та кондиціонування повітря).

РОЗДІЛ 1
ТЕХНОЛОГІЧНІ ОСНОВИ МІКРОКОНТРОЛЕРІВ НА БАЗІ AVR

1.1 Історія розвитку AVR архітектури

Архітектура AVR була розроблена двома студентами Норвезького технологічного інституту (NTH) Альф-Егілем Богеном та Вегардом Волланом. Atmel стверджує, що назва AVR не є абревіатурою і не означає нічого конкретного. Творці AVR не дають однозначної відповіді, що саме означає термін "AVR". Однак прийнято вважати, що AVR означає RISC-процесор Альфа і Вегарда.

Оригінальний AVR MCU був розроблений в місцевому будинку ASIC в Тронхеймі, Норвегія, який в той час називався Nordic VLSI, тепер Nordic Semiconductor, де Боген і Воллан працювали студентами. Він був відомий як µRISC (Micro RISC) і був доступний як кремнієвий IP/building block від Nordic VLSI. Коли технологія була продана компанії Atmel від Nordic VLSI, внутрішня архітектура була доопрацьована Богеном і Волланом в Atmel Norway, дочірньою компанією Atmel. Розробники тісно співпрацювали з авторами компіляторів з IAR Systems, щоб гарантувати, що набір команд AVR забезпечує ефективну компіляцію мов високого рівня [1].

Одним з перших в лінійці AVR був AT90S8515, який в 40-контактному DIP-корпусі має таку ж розводку, як і мікроконтролер 8051, включаючи зовнішню мультиплексовану шину адреси і даних. Полярність лінії RESET була протилежною (8051 має активно-високий рівень RESET, тоді як AVR має активно-низький рівень RESET), але в іншому розводка була ідентичною, але першим комерційним мікроконтролером був AT90S1200.

8-розрядна архітектура мікроконтролерів AVR була представлена в 1997 році. До 2003 року компанія Atmel відвантажила 500 мільйонів флеш-мікроконтролерів AVR. Платформа Arduino, розроблена для простих

електронних проектів, була випущена в 2005 році і містила мікроконтролери ATmega8 AVR. В 2016 році Atmel придбала компанія Microchip Technology.

Мікроконтролер AVR забезпечує цифрове керування будь-якими електричними, автомобільними або механічними системами, промисловими установками, різними пристроями, електронними гаджетами тощо. Ці мікроконтролери доступні у 8, 16 та 32-розрядних мікросхемах [2].

Принцип роботи мікроконтролера можна порівняти з персональним комп'ютером (ПК), який має материнську плату всередині. У материнській платі використовується мікропроцесор (мікросхеми AMD, Intel), який забезпечує інтелектуальні функції, пам'ять EEPROM і оперативну пам'ять RAM для взаємодії з системою, наприклад, послідовні порти, інтерфейси дисплея і драйвери дисків. Мікроконтролер має всі або більшість цих функцій, вбудованих в одну мікросхему, тому він не потребує материнської плати та інших компонентів.

Мікроконтролери AVR знаходять багато застосувань в якості вбудованих систем. Вони особливо поширені в аматорських та освітніх вбудованих додатках, популяризовані завдяки їх включенню в багато лінійок плат для розробки апаратного забезпечення Arduino з відкритим вихідним кодом.

Мікроконтролер AVR поставляється в різних конфігураціях, деякі з них призначені для поверхневого монтажу, а деякі - для монтажу в отвори. Він доступний з 8-контактними до 100-контактними виводами, будь-який мікроконтролер з 64-контактними або більше виводами призначений тільки для поверхневого монтажу [3].

Мікроконтролери AVR доступні в 3 категоріях: TinyAVR, MegaAVR і XmegaAVR.

Мікроконтролери **TinyAVR** доступні в невеликих розмірах, мають менший обсяг пам'яті і підходять лише для простих додатків.

Мікроконтролери **MegaAVR** дуже відомі тим, що мають до 256 КБ пам'яті, містять максимальну кількість периферійних пристроїв і використовуються в середніх і складних додатках.

XmegaAVR часто використовується для складних додатків, де потрібна висока швидкість і велика пам'ять програм.

AVR підтримує широкий набір інструкцій, включаючи оригінальний набір інструкцій THUMB від ATmel, а також звичайний набір інструкцій ARM. Плати Arduino використовують мікроконтролери ATMEL серії ATmega в якості центрального процесора.

Таблиця 1.1 Категорії мікроконтролерів та їх особливості. Адаптовано з роботи [3]

Назва серії	Кількість контактів	Флеш-пам’ять	Особливість
TinyAVR	Від 6 до 32	Від 0,5 до 8 КБ	Мініатюрний розмір
MegaAVR	Від 28 до 100	Від 4 до 56 КБ	Розширені периферійні пристрої
XmegaAVR	Від 44 до 100	Від 16 до 384 КБ	Включено DMA та систему подій

Мікроконтролери AVR доступні в чотирьох типах, таких як ATmega8, ATmega16, ATmega32 і ATmega328, де кожен мікроконтролер і його функції будуть описані нижче.

Мікроконтролер Atmega8 AVR

Цей мікроконтролер являє собою 28-контактну мікросхему з внутрішньою пам'яттю SRAM - 1 Кбайт, флеш-пам'яттю - 8 Кбайт і підтримкою двох зовнішніх переривань. Цей мікроконтролер заснований на RISC-архітектурі і розроблений компанією Microchip. Цей мікроконтролер доступний у трьох корпусах PDIP, TQFP та MLF, де перший корпус має 28 виводів, а решта два корпуси будуть доступні з 32 виводами на кожному модулі.

Програмна пам'ять або флеш-пам'ять цього мікроконтролера становить 8 КБ, яка використовується для зберігання програмного коду та постійних

налаштувань. Цей мікроконтролер використовується для створення електричних та електронних проектів [3].

Мікроконтролер Atmega16 AVR

Це високопродуктивний 8-розрядний мікроконтролер з сімейства Mega AVR від Atmel. Цей мікроконтролер має 40 виводів на основі вдосконаленої RISC-архітектури, що містить 131 потужну інструкцію.

Він має програмовану флеш-пам'ять - 16 Кб, 1 Кб статичної оперативної пам'яті та 512 байт EEPROM. Цей тип мікроконтролерів містить 32 регістри загального призначення та набір інструкцій, які підключаються безпосередньо до ALU і дозволяють отримати доступ до 2 окремих регістрів в межах однієї інструкції, яка виконується за один цикл CLK.

Цикл витривалості флеш-пам'яті, як і EEPROM, становить близько 10K та 100K відповідно. Він може працювати на максимальній частоті 16 МГц. Мікроконтролер Atmega16 AVR використовується в комерційних продуктах і невеликих промислових машинах, цей мікроконтролер також може бути використаний для вимірювання робочого циклу і частоти зовнішнього пристрою [15].

Мікроконтролер Atmega32 AVR

Мікроконтролер Atmega32 AVR - це малопотужний, високопродуктивний мікроконтролер на базі RISC. Цей контролер працює від 1,8 до 5,5 вольт.

Цей мікроконтролер має різні функції, наприклад, він поєднує 32 Кб флеш-пам'яті ISP з 1 Кб EEPROM, 2 Кб SRAM, можливості читання під час запису, універсальні лінії вводу-виводу, універсальні робочі регістри - 32, інтерфейс JTAG для налагодження або програмування на кристалі, 3 гнучких таймера/лічильника, включаючи режими порівняння, послідовний програмований USART, послідовний порт SPI, внутрішні та зовнішні переривання, USI (універсальний послідовний інтерфейс) з детектором умов запуску, програмований сторожовий таймер, включаючи внутрішній генератор, 10-розрядний 8-канальний АЦП. Використовується в системах управління двигунами, DSP, системах периферійних інтерфейсів, системах контролю

температури, вимірювання аналогових сигналів і різних вбудованих системах, таких як торгові автомати, кавомашини і т.д. [15].

Мікроконтролер Atmega328 AVR

Мікроконтролер Atmega328 AVR - це високопродуктивна і малопотужна 8-розрядна мікросхема на базі RISC, яка просто поєднує в собі 32 КБ флеш-пам'яті ISP з можливістю читання і запису. Робоча напруга цього AVR-мікроконтролера становить від 1,8 до 5,5 вольт.

Основні характеристики мікроконтролера Atmega328 AVR містять EEPROM-1 КБ, SRAM-2 КБ, 32 регістри загального призначення, 23 лінії вводу/виводу загального призначення, гнучкий таймер або лічильники, включаючи режими порівняння, послідовний програмований USART, внутрішні та зовнішні переривання, послідовний порт SPI, 2-провідний послідовний інтерфейс, 10-розрядний 6-канальний АЦП, програмований сторожовий таймер, включаючи внутрішній генератор, і п'ять програмних режимів енергозбереження, які можна вибрати.

Застосовується в автономних системах і різних електронних проєктах, де потрібен малопотужний, простий, недорогий мікрочіп. Це найпопулярніший AVR-контролер, тому він використовується в платформах розробки Arduino, таких як Arduino Uno, Arduino Nano і Arduino Pro Mini [3].

1.2 Архітектура та виводи мікроконтролерів AVR

Розглянемо більш детально конфігурацію виводів на прикладі мікроконтролера AVR Atmega32. Цей мікроконтролер має чотири порти: Порт-A, порт-B, порт-C, порт-D. Порт-A в основному містять виводи від PA7 до PA0, порт-B - від PB7 до PB0, порт-C - від PC7 до PC0 і порт-D - від PD7 до PD0.

Порт-A (PA7-PA0)

У наведеному вище мікроконтролері AVR, виводи порту-A в основному містять PA7-PA0, який працює як 8-бітний двонаправлений порт вводу/виводу, а також аналогові входи для аналого-цифрового перетворювача. Вихідні дані

буферів порту А в основному містять симетричні характеристики накопичувача, включаючи високу ємність джерела та стоку. Коли виводи порту А використовуються як входи від PA0 до PA7, вони замикаються ззовні на низький рівень. Виводи цього порту мають тристадійний стан, коли стан скидання вмикається, навіть якщо CLK не працює.

Порт B (PB7-PB0) і Порт D (PD7-PD0)

Виводи цих двох портів в основному містять в собі PB7-PB0 і PD7-PD0. Ці порти є 8-бітними двонаправленими портами вводу/виводу з внутрішніми підтягуючими резисторами. Вихід цих двох буферів портів в основному містить симетричні характеристики накопичувача, в тому числі високі можливості як приймача, так і джерела. Як і входи, виводи цього порту, які ззовні затягнуті на низький рівень, забезпечуватимуть струм, якщо резистори активовані. Виводи цих двох портів переходять у тристадійний стан щоразу, коли активується умова скидання, навіть якщо CLK не працює.

Порт C (PC7-PC0)

Виводи цього порту С в основному містять в собі від PC7 до PC0 і є 8-бітним двонаправленим портом вводу/виводу. Якщо інтерфейс JTAG дозволено, підтягуючі резистори на виводах PC3 (TMS), PC2 (TCK) і PC5 (TDI) спрацьовують, навіть якщо відбувається скидання. Порт С також виконує функції інтерфейсу JTAG та інші спеціальні функції ATmega32[19]

VCC - це вивід цифрової напруги живлення.

GND - це вивід заземлення.

RESET - це вивід RESET, який використовується для встановлення основного значення мікроконтролера ATmega32. Під час запуску програми цей вивід повинен бути встановлений на високому рівні протягом двох циклів.

XTAL1 - це вхідний вивід для підсилювача інвертуючого генератора, а також для внутрішньої робочої схеми CLK.

XTAL2 - цей вивід є виходом з підсилювача інвертуючого генератора.

AVCC - це вивід живлення для порту А, а також для АЦП. ПІдключення цього виводу повинно бути зроблено до VCC ззовні, навіть якщо АЦП не

використовується, то він повинен бути підключений до VCC через низькочастотний фільтр.

AREF - це аналоговий опорний вихід, який використовується для аналого-цифрового перетворювача [4].

Архітектура мікроконтролерів AVR базується на розширеній RISC і містить 32 8-бітних регістрів загального призначення. За один цикл CLK цей мікроконтролер може отримати входи з двох регістрів, підключити їх до ALU для виконання необхідної операції і перемістити результат назад у довільний регістр. Тут ALU виконує арифметичні та логічні операції над входами з регістра [17].

AVR може виконувати однотактне виконання, що означає, що цей мікроконтролер може виконувати 1 мільйон інструкцій за кожну секунду, якщо частота циклу становить 1 МГц. Якщо робоча частота контролера вища, то і швидкість його роботи буде вищою. Таким чином, енергоспоживання потрібно оптимізувати зі швидкістю обробки і, отже, потрібно вибирати робочу частоту відповідно [18].

Архітектура мікроконтролера AVR містить різні структурні блоки, і кожен блок пояснюється на блок-схемі мікроконтролера AVR, показаний нижче.

До переваг мікроконтролера AVR можна віднести наступні:

- Ці мікроконтролери мають високу швидкість, високу продуктивність і менше енергоспоживання;
- Він скасовує машинний цикл для виконання операції транспортування як з циклом CLK, так і з циклом команд;
- Дуже простий у використанні і дешевший;
- Менша вага;
- Легкий у програмуванні та налаштуванні.

До недоліків відноситься наступне:

- Він має єдиного виробника, тому доступний тільки від Atmel;
- Ці мікроконтролери мають досить низьку потужність;

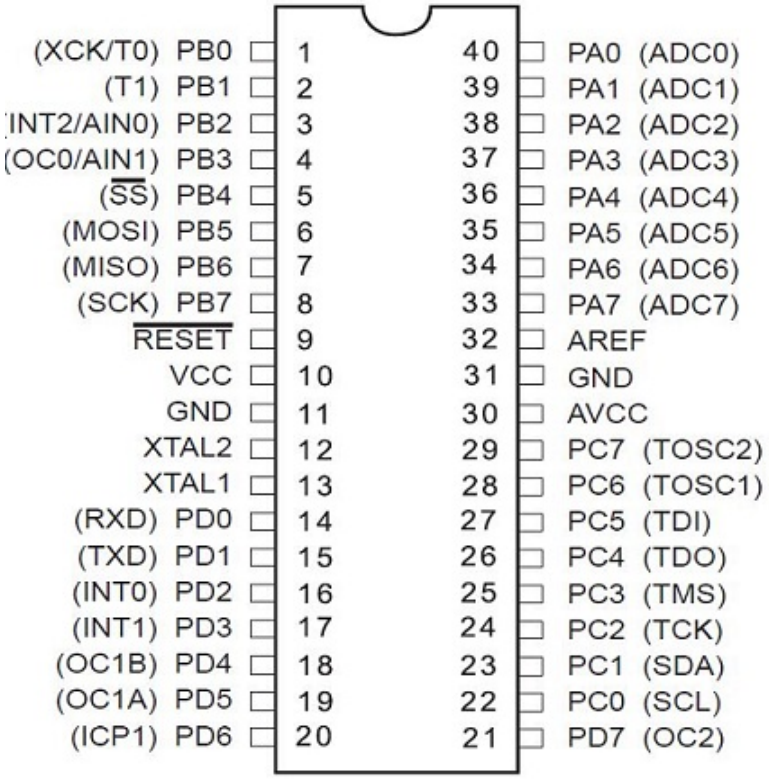

Рисунок 1.1 – Конфігурація виводів мікроконтролера AVR [5]

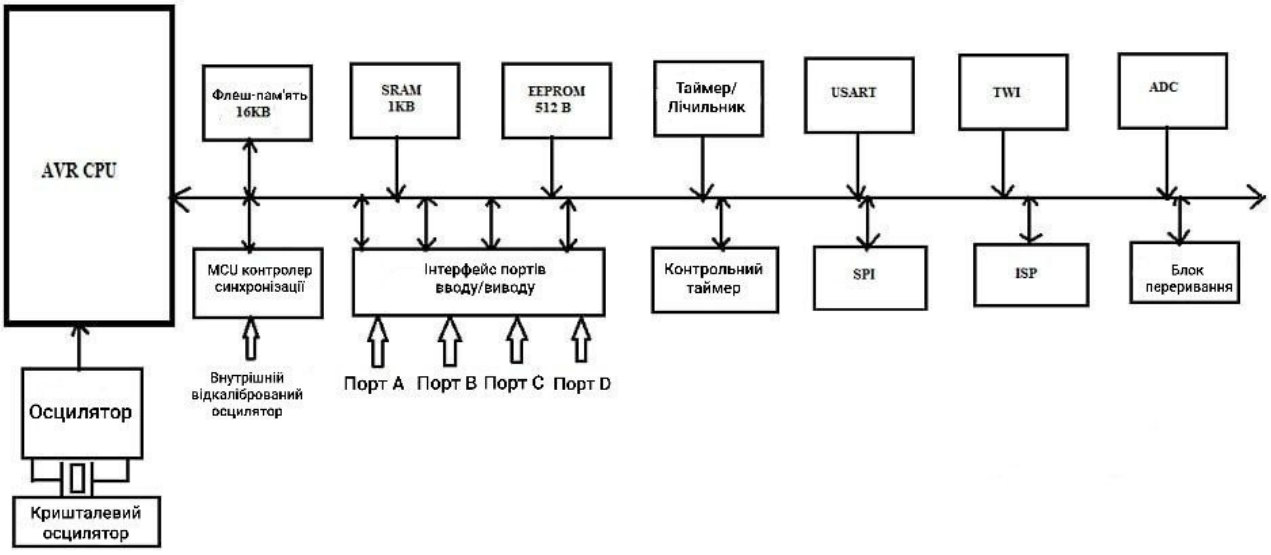

Рисунок 1.2 – Архітектура мікроконтролерів AVR. Адаптовано з роботи [2]

- Його максимальна швидкість становить близько 20 MIPS;
- Обчислювальна потужність і пам’ять обмежені, тому він не потрібен для кожної програми.

1.3 Розвиток плат Arduino на основі мікроконтролерів AVR

У 2005 році, спираючись на роботу Ернандо Баррагана (творця Wiring), Массімо Банзі та Девід Куартіеллес створили Arduino, простий у використанні програмований пристрій для інтерактивних проектів арт-дизайну, в Інституті інтерактивного дизайну Івреа в Івреа, Італія.

Девід Мелліс розробив програмне забезпечення Arduino, яке базувалося на Wiring. Незабаром до проекту приєдналися Джанлука Мартіно і Том Ігое, і ці п'ятеро відомі як перші засновники Arduino. Вони хотіли, щоб пристрій був простим, легко підключався до різних речей (наприклад, реле, двигунів і датчиків) і легко програмувався. Він також мав бути недорогим, оскільки студенти та митці не мають багато вільних грошей. Вони обрали сімейство 8-розрядних мікроконтролерів AVR (MCU або µC) від Atmel, розробили автономну друковану плату з простими у використанні з'єднаннями, написали прошивку завантажувача для мікроконтролера і упакували все це в просте інтегроване середовище розробки (IDE), яке використовувало програми, що називаються "скетчі". Результатом став Arduino [5].

Відтоді Arduino розвивався в кількох різних напрямках, деякі версії ставали меншими за оригінал, а деякі - більшими. Кожна з них має певну призначену нішу, яку потрібно заповнити. Спільним елементом для всіх них є бібліотека AVR-GCC для середовища виконання Arduino, яка постачається разом із середовищем розробки Arduino, а також вбудоване програмне забезпечення завантажувача, яке попередньо завантажується в мікроконтролер кожної плати Arduino.

Плати сімейства Arduino використовують процесори, розроблені корпорацією Atmel з Сан-Хосе, Каліфорнія. Більшість проектів Arduino

використовують 8-розрядні мікроконтролери серії AVR, основним винятком є Due з 32-розрядним процесором ARM Cortex-M3. Я не розглядатиму Due в цій роботі, оскільки він радикально відрізняється від пристроїв AVR за багатьма фундаментальними параметрами.

Протягом багатьох років дизайнери Arduino.cc розробили низку конструкцій плат. Перша широко розповсюджена плата Arduino, Diecimila, була випущена в 2007 році, і з моменту її першого випуску сімейство Arduino розвивалося, щоб використовувати переваги різних типів пристроїв Atmel AVR MCU. Due, випущена в 2012 році, є першим Arduino, який використовує 32-розрядний процесор ARM Cortex-M3, і відрізняється від решти сімейства як за обчислювальною потужністю, так і за конфігурацією розведення плати. Інші плати, такі як LilyPad і Nano, також не мають такої ж розводки, як інші члени сімейства, і призначені для різних сфер застосування - для носіння у випадку LilyPad; для портативних пристроїв у випадку Esplora; і для компактних розмірів у випадку Mini, Micro і Nano [5].

З кожним роком з'являються нові типи плат Arduino, новіші версії мають більш досконалі процесори з більшим об'ємом пам'яті та розширеними можливостями вводу/виводу, але здебільшого вони використовують ту саму схему розведення і працюють з існуючими додатковими платами, які називаються шильдами, та різними додатковими компонентами, такими як датчики, реле та виконавчі механізми. У таблиці 1.2 перераховані типи Arduino, які з’явилися з 2005 року. Новіші версії Arduino також працюватимуть з більшістю скетчів, створених для старих моделей, можливо, з деякими незначними змінами та новими бібліотеками, але скетчі, написані для найновіших версій, можуть працювати зі старими моделями, а можуть і не працювати з ними [6].

Таблиця 1.2 не включає Arduino Robot, який являє собою друковану плату з прикріпленими до неї двигунами і колесами. Однією з найновіших плат в лінійці Arduino є Yún, цікавий звір, який має як мікроконтролер ATmega32U4, так і модуль Linino з процесором Atheros AR9331 MIPS, здатний працювати під

управлінням версії операційної системи OpenWrt на базі Linux. Я не буду вдаватися в деталі OpenWrt в кінці Yún, але сторона Arduino - це, по суті, просто стандартний Arduino (Leonardo, якщо бути точним).

Таблиця 1.2 Хронологія розвитку продуктів Arduino. Адаптовано з роботи [5]

Назва плати	Рік	Мікроконтролер	Назва плати	Рік	Мікроконтролер
Diecimila	2007	ATmega168V	Mega 2560	2010	ATmega2560
LilyPad	2007	ATmega168V /ATmega328 V	Uno	2010	ATmega328P
Nano	2008	ATmega328/ ATmega168	Ethernet	2011	ATmega328
Mini	2008	ATmega168	Mega ADK	2011	ATmega2560
Mini Pro	2008	ATmega328	Leonardo	2012	ATmega32U4
Duemilanove	2008	ATmega168/ ATmega328	Esplora	2012	ATmega32U4
Mega	2009	ATmega1280	Micro	2012	ATmega32U4
Fio	2010	ATmega328P	Yún	2013	ATmega32U4 +Linino

Якщо в таблиці 1.2 вказано більше одного типу мікроконтролера, це означає, що певна версія плати Arduino спочатку була зроблена з одним мікроконтролером, а пізніше - з іншим (зазвичай більш потужним) пристроєм. Наприклад, старіша версія Duemilanove буде мати ATmega168, тоді як новіші моделі мають ATmega328. Функціонально ATmega168 і ATmega328 ідентичні, але ATmega328 має більше внутрішньої пам'яті.

Останні доповнення до сімейства Arduino, Leonardo, Esplora, Micro та Yún, використовують ATmega32U4. Хоча ця частина схожа на ATmega328, вона також містить в собі інтегрований компонент інтерфейсу USB-послідовний, який усуває одну з частин інтегральної схеми (IC), знайдених на таких платах, як Uno і Duemilanove [10].

Інтерфейс програмування також дещо відрізняється від плат, що використовують ATmega32U4, але для більшості користувачів він має бути в основному прозорим.

Фізично Arduino не є великою друкованою платою. Базові плати, які мають фізичне розташування виводів, що зазвичай використовується для додаткових плат, мають розмір приблизно 2,1 на 2,7 дюйма (53,3 на 68,6 мм). На рисунку 1.3 показано вибір плат Arduino з лінійкою для масштабування [5].

Рисунок 1.3 – Відносні розміри плат Arduino [5]

Зауважте, що незважаючи на малий розмір, плата типу Nano має всі ті ж можливості, що і Duemilanove, за винятком зручних штирьових роз'ємів і звичайного (тип B) USB-роз'єму. Вона ідеально підходить для додатків, де її не будуть турбувати після встановлення, і де невеликий розмір є вимогою. Деякі застосування, які спадають на думку, - це автономні пристрої для збору даних про навколишнє середовище (наприклад, автоматизовані метеорологічні станції на сонячних батареях або буї для збору океанських даних), хронометраж і збір

даних для моделей ракет, системи безпеки, і, можливо, навіть "розумна" кавоварка [5].

На рисунку 1.4 показано Nano, встановлений на безпаяльній макетній платі.

На додаток до різних типів плат, розроблених або схвалених Arduino.cc, існує багато пристроїв, які є або апаратно-сумісними, або програмно-сумісними. Сумісність цих пристроїв з Arduino полягає в тому, що вони містять завантажувач Arduino (або щось схоже на нього), і їх можна програмувати за допомогою Arduino IDE, вибравши відповідний сумісний тип плати Arduino з випадаючого списку IDE [7].

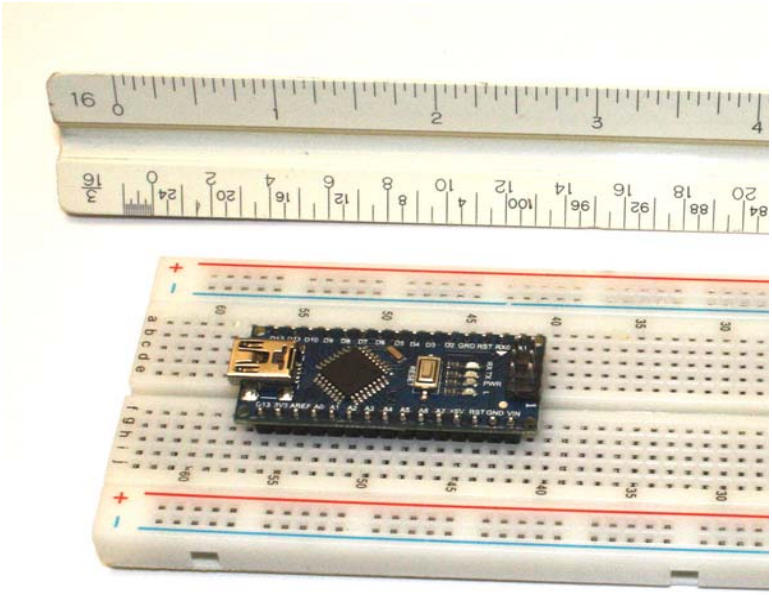

Рисунок 1.4 – Arduino Nano на безпаяльній макетній платі [5]

У більшості випадків апаратно-сумісні плати виглядають як будь-яка інша плата Arduino, за винятком відсутності офіційного логотипу Arduino та шовкографії. Інші апаратно-сумісні продукти можуть виглядати зовсім не так, як типова плата Arduino, але мають гнізда для контактів у правильному розташуванні для використання стандартної захисної плати типу Arduino. Деякі апаратно-сумісні продукти мають додаткові роз'єми, наприклад, версія Uno від

SainSmart з додатковими роз'ємами для функцій вводу/виводу. На рисунку 1.5 зображені плати SainSmart UNO, SainSmart Mega2560 та Diavolino. Існує набагато більше, ніж показано нижче, але це повинно дати деяке уявлення про те, що доступно.

Рисунок 1.5 – Апаратно-сумісні з Arduino пристрої [5]

Варто зазначити, що Diavolino - це набір і потребує збірки.

Існує багато програмно-сумісних з Arduino плат. Вони використовують завантажувач і середовище розробки Arduino, але не мають повністю сумісного з Arduino фізичного форм-фактору. Програмно-сумісні пристрої можна програмувати за допомогою інструментів розробки Arduino, але вони можуть використовувати інше розташування контактів вводу/виводу, або, можливо, використовувати інші типи роз'ємів замість штирьових гнізд, які можна знайти на стандартних платах Arduino. Користувацькі схеми на основі мікроконтролера AVR, вбудовані в якийсь більший пристрій або систему, потрапляють в категорію програмно-сумісних, якщо в мікроконтролері встановлений завантажувач Arduino [9].

РОЗДІЛ 2

ЗАСТОСУВАННЯ АПАРАТНОЇ ПЛАТФОРМИ ARDUINO

2.1 Завантажувач Arduino IDE

Як і для більшості одноплатних мікроконтролерних систем, програми для Arduino і його мікроконтролера AVR розробляються в іншій системі певного типу (Windows, Linux або Mac), а потім переносяться або завантажуються в мікроконтролер AVR на платі Arduino. Система розробки називається хостом розробки, а Arduino або інший пристрій на основі мікроконтролера - цільовим. Цей тип процесу розробки існує вже досить давно і використовувався, коли потрібно було створити програмне забезпечення для цільової машини, яка не мала можливості компілювати код для себе.

Техніка створення програмного забезпечення для одного типу процесора на іншому типі системи називається крос-компіляцією, і це дійсно єдиний спосіб створити скомпільоване програмне забезпечення для мікроконтролерів. Невеликі мікроконтролери, такі як AVR (або будь-який інший невеликий 8-розрядний пристрій), просто не мають ресурсів для компіляції та компонування чогось на кшталт програми на C або C++. Для компіляції та компонування використовується більш потужна машина з такими ресурсами, як швидкий процесор, дискові накопичувачі великої ємності та багато пам'яті, а потім готова програма передається на ціль для виконання [10].

У деяких випадках хост-система може навіть мати емулятор для цільового пристрою, який дозволяє розробнику завантажувати і тестувати програми в змодельованому середовищі. Емулятор може не забезпечити 100% ідеальну симуляцію цільового мікроконтролера та його реального оточення, але він все одно може бути корисним інструментом для перевірки базової функціональності програмного забезпечення до того, як воно буде завантажене

в реальний мікроконтролер. Для операційної системи Linux доступно декілька емуляторів AVR, зокрема simavr та GNU AVR Simulator.

Можна помітити, що терміни "об'єкт", "образ", "виконуваний файл" і "вихідний код" часто вживаються, коли йдеться про компіляцію, компонування і завантаження програмного забезпечення. Це все дуже старі терміни, які походять з часів перфокарт, барабанів пам'яті та магнітної стрічки, а також комп'ютерів, які заповнювали цілі кімнати.

Вихідний код, як і слід було очікувати, відноситься до вхідних даних, що подаються на вхід компілятора або асемблера. Вихідний код є читабельним для людини і створюється за допомогою певного типу текстового редактора. Для Arduino це зазвичай означає мову C або C++, створену за допомогою Arduino IDE або іншого редактора чи IDE [13].

Результат роботи компілятора або асемблера називається об'єктним кодом, але насправді це просто машинний код для конкретного процесора. Іншими словами, він складається з двійкових значень кодів операцій процесора і будь-яких пов'язаних з ними даних у вигляді так званих буквених значень, також у двійковій формі. Це зазвичай називають машинною мовою. У сучасних компіляторах і асемблерах це те, що знаходиться всередині *.o і *.obj файлів, знайдених після завершення збірки чи компіляції.

Об'єктний код може бути не відразу виконуваним. Якщо програма складається з двох або більше модулів або потребує об'єктного коду із зовнішньої бібліотеки, то вихідні дані компілятора або асемблера будуть містити заповнювачі, які посилаються на зовнішнє програмне забезпечення.

У деяких випадках зовнішній код може бути у вигляді вихідного коду (як, наприклад, більшість коду, що постачається з Arduino) і буде скомпільовано одночасно з вашим скетчем, або ж він може бути вже скомпільований в об'єктну бібліотеку, що містить один або декілька окремих модулів коду. Попередньо скомпільовані бібліотеки об'єктів зазвичай мають розширення *.a.

Інструмент, який називається компонувальник, використовується для заповнення прогалин і з'єднання різних частин у виконуваний образ. У випадку

програм для Arduino це зазвичай передбачає включення підтримки часу виконання у вигляді функції main() та основних функцій бібліотеки часу виконання AVR для вводу/виводу та інших низькорівневих операцій.

Кінцевим результатом є виконуваний образ з усім необхідним для запуску програми, все в одному пакеті. Після перетворення в ASCII-файл, що складається з рядків шістнадцяткових символів, він завантажується в мікроконтролер AVR, перетворюється назад в двійковий формат і зберігається у вбудованій флеш-пам'яті для виконання.

2.2 Arduino IDE, бібліотеки, що постачаються з IDE

Завантаження програми в сучасний мікроконтролер може відбуватися одним з декількох різних способів, але найпростіший - це дозволити самому мікроконтролеру допомогти з цим процесом. Це досягається за допомогою невеликого попередньо завантаженого біта мікропрограми, який називається завантажувачем.

У мікроконтролерах сімейства AVR зарезервовано місце у вбудованій флеш-пам'яті для завантажувача. Після того, як мікроконтролер налаштований на використання завантажувача, адреса цієї спеціальної області пам'яті є першим місцем, де мікроконтролер AVR буде шукати інструкції при увімкненні (або перезавантаженні). Доки завантажувач не буде перезаписаний завантаженою програмою, він зберігатиметься в пам'яті між циклами увімкнення та вимкнення живлення.

Резервування місця розташування завантажувача зазвичай передбачає увімкнення внутрішнього перемикача, або Juse, за допомогою спеціального пристрою програмування, який зв'язується з MCU через інтерфейс ICSP або JTAG. Під час запуску мікроконтролер перевіряє конфігурацію запобіжника (енергонезалежні біти конфігурації), щоб визначити, як організована флеш-пам'ять і чи зарезервовано місце для певного типу завантажувача або іншого коду запуску. Плати Arduino - як офіційні продукти, так і програмно-сумісні

плати - постачаються з завантажувачем типу Arduino, вже завантаженим в MCU. Завантажувач Arduino реалізує спеціальний протокол, який дозволяє йому розпізнавати Arduino IDE і виконувати передачу даних програми з хоста розробки на цільову плату.

Що робить Arduino особливим, а не просто ще однією друкованою платою з припаяним до неї AVR, так це Arduino IDE, прошивка Arduino, код виконання, програмні бібліотеки, розроблені Ardunio.cc, і, звичайно ж, програми, надані вами, розробником. Arduino IDE - це швидкий і простий спосіб створення і завантаження програмного забезпечення для мікросхеми AVR. Це досягається шляхом ефективного приховування більшої частини того, що відбувається під час компіляції, компонування та передачі коду до пристрою AVR.

На момент написання даної роботи останньою версією Arduino IDE є 2.1.0. На кожній платформі вона виглядає майже однаково. На рисунку 2.1 показано початковий екран, який з'являється при запуску IDE.

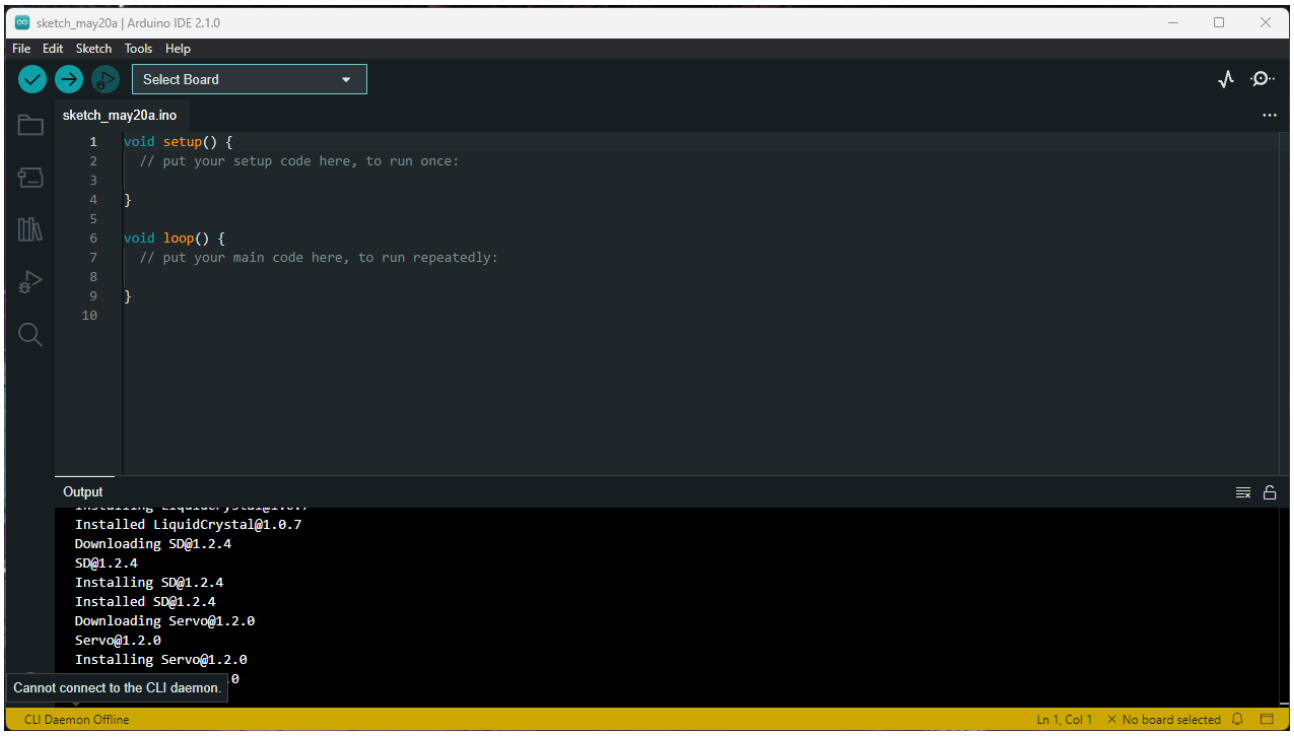

Рисунок 2.1 – Головний екран Arduino IDE

Якщо використовується стара плата Arduino (наприклад, Diecimila, Duemilanove або Uno), то відсутність останньої версії IDE не є великою проблемою. Більшість людей, які використовують старішу версію IDE зі старими типами плат Arduino, ймовірно, навіть не помітять, що у них немає останніх змін. [16]

Процедура встановлення Arduino IDE та бібліотек варіюється від надзвичайно простої до дуже складної, залежно від вашої платформи і того, скільки зусиль ви хочете докласти до цього.

Розглянемо встановлення Arduino IDE на систему Windows, оскільки саме вона є найбільш застосованою. Інсталяційний пакет arduino ide_2.1.0_Windows_64bit.exe подбає про всі деталі. Якщо у вас вже встановлена стара версія IDE, він видалить її перед встановленням нової версії (але не видалить ваші існуючі скетчі). Виконавчі файли та бібліотеки Arduino знаходяться у папці Program Files\Arduino. Ви також можете знайти приклади тут. У вашому домашньому каталозі створюється каталог з назвою Arduino. Це місце, де будуть розміщені всі створені користувачем скетчі та бібліотеки, що надаються користувачем [9].

Після завершення інсталяції варто витратити кілька хвилин і переглянути, що було встановлено у вашій системі. У каталозі встановлення Arduino ви знайдете вихідний код численних прикладів, вихідні коди та приклади для стокових бібліотек, hex-файли (двійкові завантажувальні образи) для різних версій завантажувача, а також документацію у вигляді набору HTML-сторінок, які можна переглянути за допомогою веб-браузера або відкрити за допомогою пункту меню Help в IDE.

Можна налаштувати IDE відповідно до власних потреб за допомогою діалогового вікна Preferences, яке знаходиться у пункті File-> Preferences головного меню. Цей діалог дозволяє вказати каталог скетчів, вказати зовнішній редактор (якщо вам не подобається той, що постачається з IDE), а також змінити різні параметри поведінки IDE.[20]

Файл налаштувань, на який у діалозі посилається /home/jmh/.arduino/preferences.txt, містить набагато більше налаштувань, які не показано у діалозі. Але, не редагуйте цей файл, поки IDE активна.

Файл налаштувань - це файл даних у форматі ASCII KVP (ключ/значення), який містить налаштування редактора, початкову геометрію відображення програми (розмір вікна вікна Arduino IDE), параметри послідовного інтерфейсу (швидкість передачі даних, розмір даних тощо), а також браузер, який слід використовувати для перегляду файлів довідки у форматі HTML, що постачається разом з IDE [9].

Файл налаштувань також містить інформацію про останній відредагований скетч та останній розмір вікна IDE. Оскільки цей файл динамічно змінюється під час роботи IDE, його слід редагувати вручну лише тоді, коли IDE не активна. Для старих версій IDE це може бути єдиним способом змінити такі параметри, як розмір відступів (за замовчуванням 2 пробіли), розмір і тип шрифту, що використовується у редакторі (за замовчуванням Monospaced з розміром 12), і розгортання вкладок.

Мова програмування, яку використовує Arduino IDE, зазвичай C++, хоча вона також може використовувати C, оскільки інструментарій AVR-GCC підтримує обидві мови.

Окремі файли вихідного коду відображаються у IDE у так званих "вкладках". Ви можете переходити від одного файлу до іншого, просто вибравши відповідну вкладку у верхній частині вікна редактора. Коли вихідний код буде скомпільовано, IDE пройде по кожній вкладці, створюючи об'єктний файл для кожної з них [8].

РОЗДІЛ 3

ФІЗИЧНИЙ ЕКСПЕРИМЕНТ НА БАЗІ ARDUINO

3.1 Опис експерименту

“Розумні” термостати є різновидом програмованих цифрових регуляторів температури. Деякі з них дозволяють змінювати налаштування за допомогою Bluetooth або іншого методу бездротового з'єднання, а також відповідного додатку для смартфона або планшета. Інші пропонують можливості збору даних за допомогою бездротового завантаження, що може бути корисним, якщо є мета дізнатися, коли використовується найбільше енергії для обігріву або охолодження будинку. Звісно, є й такі, які не роблять набагато більше, ніж біметалеві котушки старого зразка, за винятком того, що замість циферблата та деяких перемикачів використовується ЛЕД-дисплей [13].

Існує багато способів вдосконалити класичний термостат з біметалевою котушкою, подібний до того, що показаний на рисунку 3.1. Цей тип пристроїв існує вже близько століття. У ньому використовується котушка з двох різних металів, кожен з яких має різний коефіцієнт теплового розширення, так що при зміні температури котушка злегка затягується або послаблюється. Цей рух потім використовується для зближення деяких контактів для замикання або розмикання ланцюга, або в деяких версіях невелика герметична скляна трубка, що містить два контакти і краплю ртуті, переміщується так, що ртуть заповнює проміжок між контактами і замикає ланцюг. Керуюча дія або увімкнена, або вимкнена; між ними немає нічого середнього [12].

У цьому розділі буде розглянуто компоненти, що є необхідними для проєктування, створення та програмування розумного термостата, використовуючи лише легкодоступні плати Arduino та звичайні додаткові компоненти.

Рисунок 3.1 – Електромеханічний термостат [5]

Спочатку потрібно отримати базове розуміння того, що саме треба вимірювати і контролювати. Основна ідея зміни температури в структурі полягає в тому, щоб додати або відвести тепло. Інакше кажучи, холод - це просто відсутність тепла, і ми можемо змінювати кількість тепла в системі, щоб досягти певної температури. Додавання тепла в навколишнє середовище можна здійснити за допомогою печі (природний газ, пропан, нафта, дрова або вугілля), електричних нагрівальних елементів або сонячної енергії. У системі теплового насоса тепло зовнішнього повітря витягується і повертається назад в будівлю, що схоже на роботу кондиціонера в зворотному напрямку. Для відведення тепла з будівлі його витягують за допомогою системи охолодження, яка використовує цикл конденсації та випаровування холодоагенту (аміаку, фреону або будь-якого іншого сучасного холодоагенту). Внутрішнє тепло поглинається холодоагентом, який потім передає його назовні в системі із замкнутим циклом [11].

Всю систему, яка включає в себе опалювальний блок, охолоджувальний блок і термостат, можна назвати системою опалення, вентиляції та

кондиціонування повітря, або ОВіК (HVAC). (Насправді, термін ОВіК найчастіше можна почути, коли йдеться про великі комерційні системи, але я буду використовувати його тут, оскільки це зручніше, ніж писати "система опалення та кондиціонування повітря"). За винятком функції вентиляції з активним повітрообміном, житлова система ОВіК виконує все те саме, що й велика система, тільки в менших масштабах.

Регулювання температури передбачає керування підсистемами, які нагрівають або охолоджують середовище для досягнення певної температури (заданого значення). У більшості випадків це робота в режимі "все увімкнено" або "все вимкнено"; у більшості побутових систем опалення, вентиляції та кондиціонування немає проміжних значень нагріву або охолодження. "Більшість", тому що цілком можливо, що хтось десь має систему резистивного нагрівача зі змінною потужністю. Принцип "все або нічого" системи HVAC означає, що фактична температура в приміщенні ніколи не буде точно відповідати заданому значенню, за винятком коротких періодів, коли вона або падає, або підвищується.

3.2 Розробка схеми термостату на базі Arduino

Термостат на основі плати Arduino має три основні функції: обігрів, охолодження та вентилятор. На рисунку 3.2 показано блок-схему з основними компонентами. Секрет успіху полягає в тому, як ці основні функції використовуються для досягнення найбільш ефективної роботи.

Поворотний датчик буде використовуватися для різних функцій, таких як встановлення температури, перемикання між днями тижня і так далі. На ЛЕД-дисплеї також є набір кнопок, але вони не будуть використовуватися. Багато старих систем опалення, вентиляції та кондиціонування мають лише чотири дроти, тоді як деякі новіші системи також мають допоміжну силову проводку.

Термостат Arduino може використовувати це джерело змінного струму, якщо воно доступне, якщо в корпусі також встановлено невеликий блок

живлення. Для цієї версії термостата буде використовуватися зовнішній настінний блок живлення, або настінна розетка.

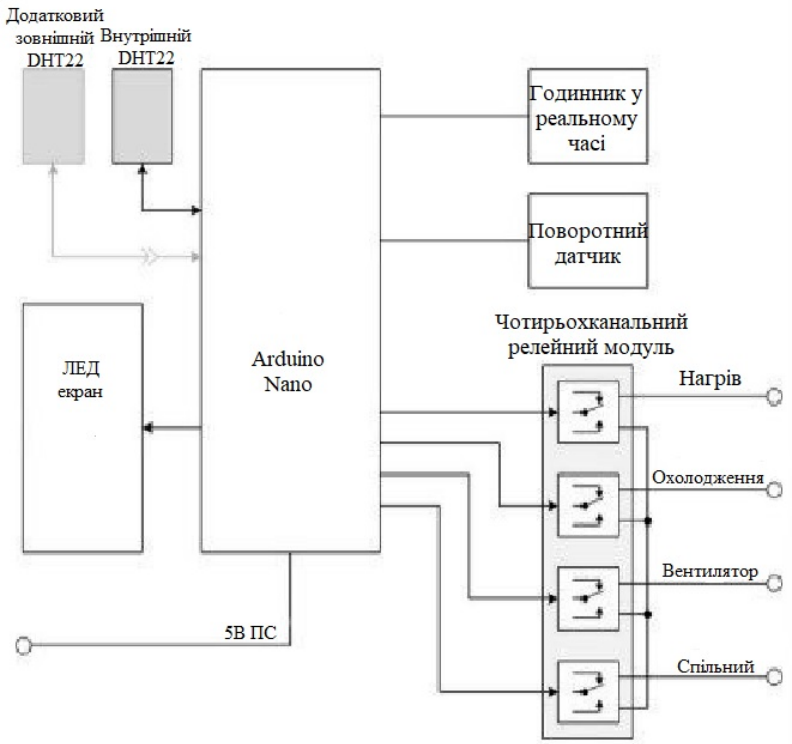

Рисунок 3.2 – Блок-схема термостату на Arduino. Адаптовано з [5]

В ідеалі було б краще брати живлення від проводки системи опалення, вентиляції та кондиціонування, але багато старих систем не мають запасної лінії 24 В змінного струму.

Елементи керування прості, з ЛЕД-дисплеєм і поворотним датчиком. Все це зручно розміститься на передній панелі корпусу розподільчої коробки.

Модуль реле буде встановлений в нижній частині корпусу, а існуючі дроти для термостата будуть виведені через отвір в нижній частині корпусу. Модуль ГРЧ(Годинник у Реальному Часі) буде встановлений всередині корпусу, а DHT22 буде прикріплений до дна так, що він буде піддаватися впливу навколишнього середовища. Я вирішив, що DHT22 варто розмістити саме так, що б не доводилося свердлити безліч отворів у корпусі для потоку повітря.

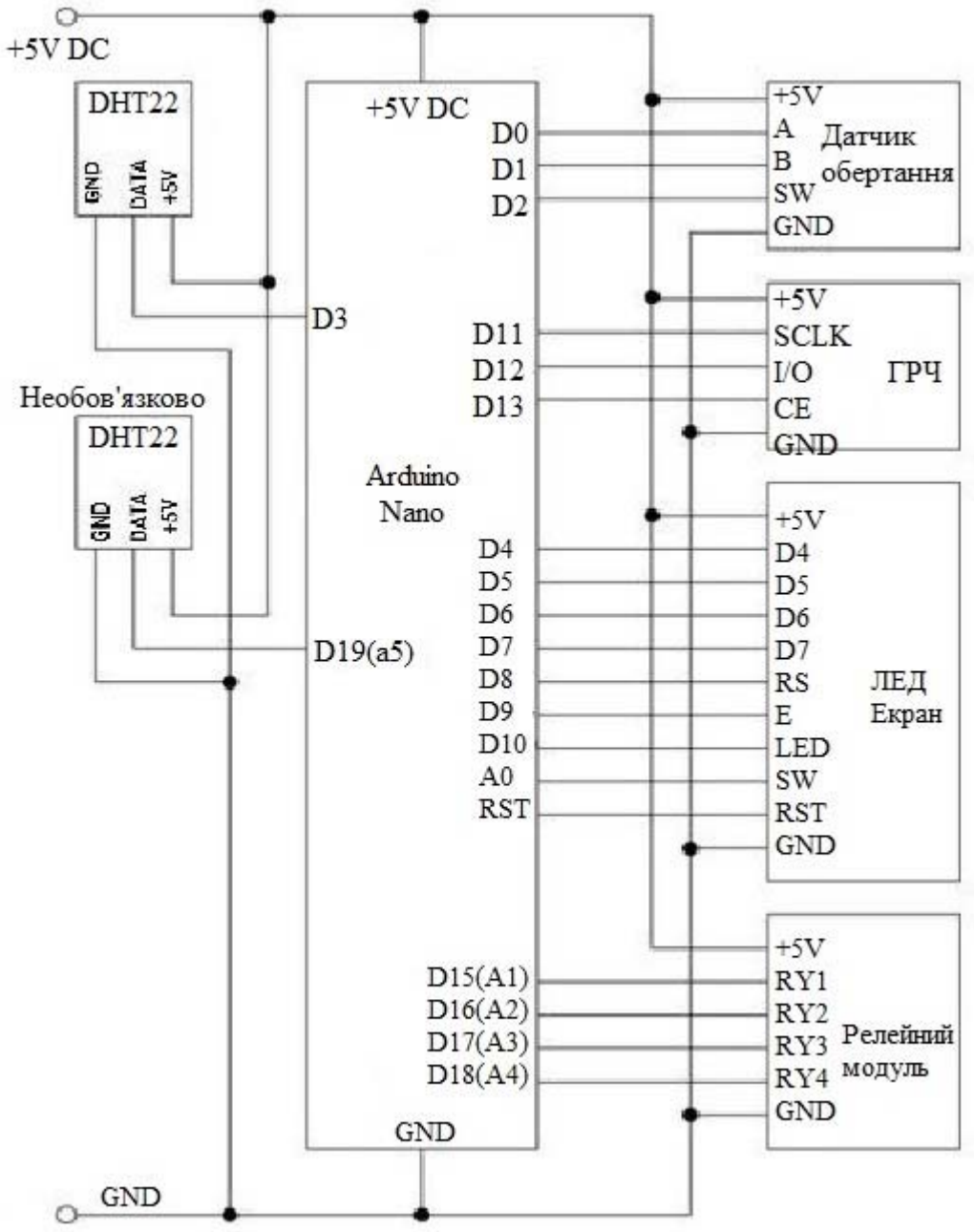

Рисунок 3.3 – Схема термостату на Arduino

Альтернативою було б зробити квадратний отвір, достатньо великий для DHT22, щоб його можна було встановити всередині, але при цьому він мав доступ до зовнішнього повітря.

Варто звернути увагу, що кожен вивід на мікроконтролері використовується. Інтерфейс програмування Arduino через D0 і D1 використовується спільно з поворотним датчиком. Поки датчик обертання не використовується, це не створить конфлікту. Також можна помітити, що аналогові входи були приєднані до цифрових контактів вводу/виводу. Це цілком нормально, і якщо ви повернетесь до Розділу 1, то побачите, що порт ATmega328, який використовується для аналогового вводу (порт C), також є стандартним дискретним цифровим портом вводу/виводу. Позначивши ці виводи як D14 - D19, я отримую доступ до них, як і до будь-яких інших цифрових виводів вводу/виводу.

Звичайно, аналогові входи зараз недоступні, але термостат не використовує жодного аналогового входу (за винятком контакту A0, який використовується екраном ЛЕД-дисплея), тому це не є проблемою. Це програмно-місткий проект, але більша частина програмного забезпечення пов'язана з користувацьким інтерфейсом та активним профілем.

На кожній ітерації в основному циклі програма перевіряє, чи вводяться дані з датчика обертання; оновлює поточні дані про температуру та вологість; визначає, чи потрібно перемикатися між екранами, переходити до різних налаштувань або змінювати значення. Обробники переривань використовуються для захоплення вхідних даних від поворотного датчика.

Переходячи до логічної системи термостату на Arduino треба показати принцип контролю логіки. Дана схема зображена нижче на рисунку 3.4.

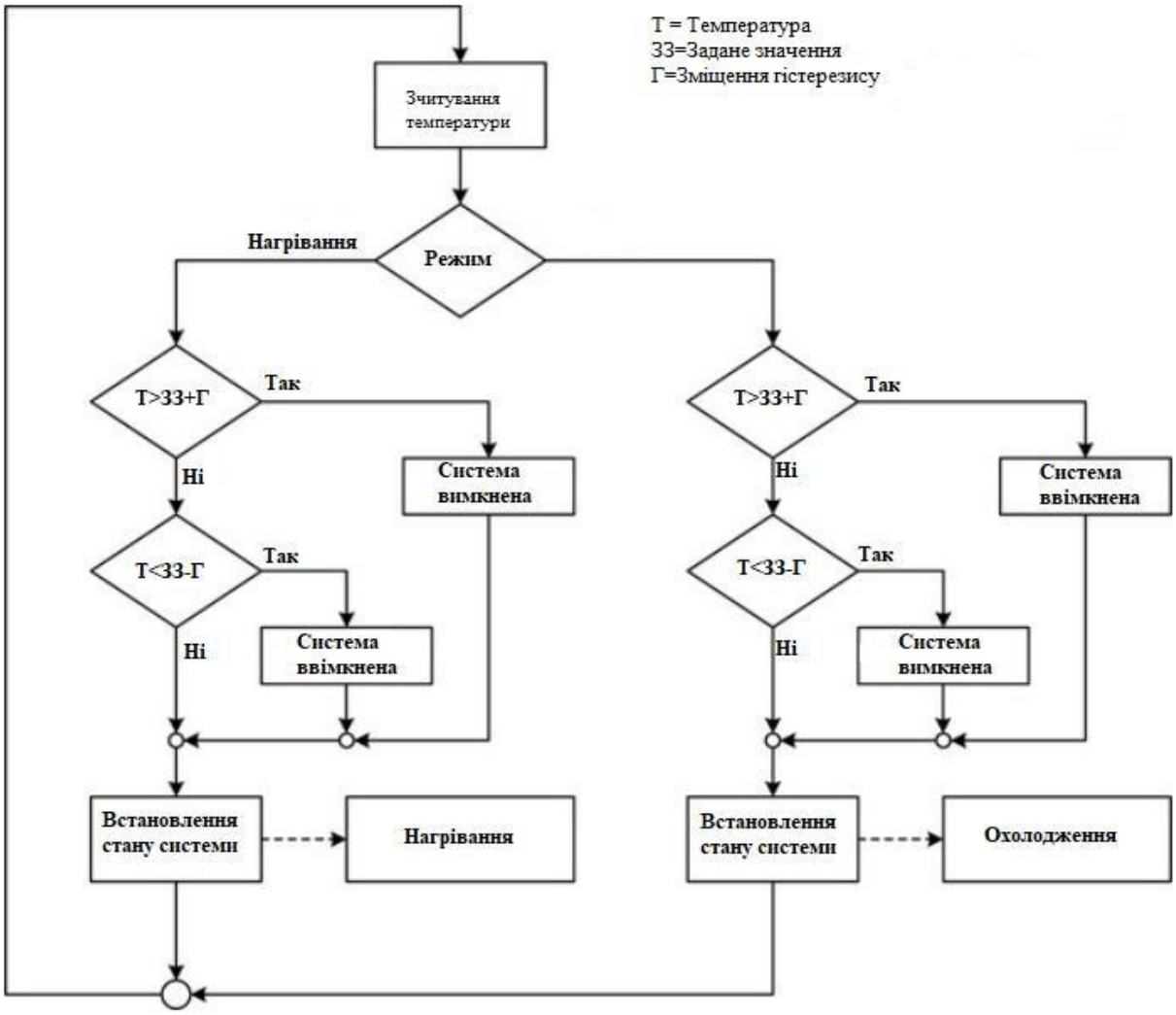

Рисунок 3.4 – Логіка керування термостатом для нагріву та охолодження

Вхід керування складається з одного поворотного датчика. Це може здатися дивним, але датчик також має вбудований перемикач, який вмикається, коли користувач натискає на ручку. Перемикач використовується для переміщення між конфігураціями дисплея (або екранами, як я їх називаю). Кожен екран містить поля, які відображають дані або налаштування, а поворотний регулятор також використовується для переміщення між полями на кожному екрані. Тривалість утримання кнопки перемикача на поворотному датчику визначає, яку дію виконає програмне забезпечення. Коротке натискання - це вибір, а тривале натискання - перехід до наступного екрану.

3.3 Опис програмного забезпечення

Програмне забезпечення можна розділити на три функціональні частини: інтерфейс користувача, логіка керування та керування дисплеєм.

Функції інтерфейсу користувача, що містяться у вихідному модулі tstat_face.cpp, забезпечують функціональність для зчитування даних з поворотного датчика та навігації по екранах дисплея. tstat_lcd.cpp містить функції роботи з ЛЕД-дисплеєм. Вихідний модуль tstat_ctrl.cpp забезпечує функціональність для вимірювання температури та вологості за допомогою датчика DHT22, логіки контролера та користувацьких програм.

У цьому проекті використовуються різні бібліотеки для DS1302 RTC, поворотного датчика, функцій часу і дати та периферійного пристрою таймера 1 AVR. Їх наведено у таблиці 3.1. Це одна з приємних особливостей роботи з Arduino, якщо вам потрібна бібліотека для використання певного модуля, датчика або екрана, швидше за все, хтось, десь, витратив час на її створення, щоб вам не довелося цього робити [13, 14].

Ці бібліотеки можна знайти в Arduino Playground та на GitHub. Обов'язково прочитайте включену документацію та вихідний код, щоб краще зрозуміти, що робить код і як його налаштувати, якщо це необхідно. Файл tstat.h містить глобальні визначення, які використовуються іншими модулями. Вони відповідають призначенню виводів, показаному на рисунку 3.3.

Функція setup() у файлі Thermostat.ino, показана на рисунку 3.6, ініціалізує ЛЕД-екран, відображає деякі повідомлення про запуск, перевіряє модуль ГРЧ та відображає перший екран.

Функція loop() у Thermostat.ino не є складною, як ви можете бачити з рисунку 3.7. Більша частина роботи відбувається, коли використовується поворотний датчик і коли програма отримує дані про температуру та вологість і виконує функції керування у tstat_ctrl.cpp.

Таблиця 3.1 Зовнішні бібліотеки, що використовуються з термостатом

Назва	Функціонал	Автор
Час	Функції часу	Майкл Марголіс
DS1302RTC	Клас RTC	Тимур Максімов
ClickEncoder	Клас поворотних датчиків	Пітер Даннеґґер
TimerOne	Клас таймерів 1	Лекс Таліоніс

```
#define ROTENC_A 0
#define ROTENC_B 1
#define ROTENC_SW 2
#define LCD_D4 4 // Визначається  екраном ЛЕД-екрану
#define LCD_D5 5 //
#define LCD_D6 6 //
#define LCD_D7 7 //
#define LCD_RS 8 //
#define LCD_E 9 //
#define LCD_LED 10 //
#define LCD_SW A0 //
#define RTC_SCLK 11
#define RTC_IO 12
#define RTC_CE 13
#define RY1 15 // A1
#define RY2 16 // A2
#define RY3 17 // A3
#define RY4 18 // A4
#define DHT1 3 // Внутрішній DHT22
#define DHT2 19 // A5, Зовнішній DHT22
```

Рисунок 3.5 Визначення вводу/виводу tstat.h

```
void setup()
{
  lcd->begin(16, 2); // Встановлення розмірів ЛЕД-екрану
  TitleDisp("Ініціалізація...", "", 1000);
  lcd->clear();
  if (rtc->haltRTC())
  lcd->print("Годинник зупинено!");
  else
  lcd->print("Годинник працює.");
  lcd->setCursor(0,1);
  if (rtc->writeEN())
  lcd->print("Редагування дозволено.");
  else
  lcd->print("Захист від редагування.");
  delay (2000);
  // Бібліотека налаштування часу
  lcd->clear();
  lcd->print("Синхронізація ГРЧ");
  setSyncProvider(rtc->get); // Функція отримання часу з ГРЧ
  lcd->setCursor(0,1);
  if (timeStatus() == timeSet)
  lcd->print("Ок!");
  else
  lcd->print("ПОМИЛКА!");
  delay (1000);
  TitleDisp("Ініціалізація", "Завершено", 1000);
  curr_screen = 0;
  Screen1();
  disptime = millis();
}
```

Рисунок 3.6 – Функція setup()

```
void loop()
{
  // Отримання поточної дати та часу з ГРЧ
  RTCUpdate();
  if (input_active) {
    HandleInput();
  }
  else {
    // Перемикання між екраном 1 і екраном 2
    if ((millis() - disptime) >
    MAX_DISP_TIME) {
      if (curr_screen) {
        Screen1();
        curr_screen = 0;
        disptime = millis();
        }
      else {
          Screen2();
          curr_screen = 1;
          disptime = millis();
          }
          }
      }
GetTemps();
SystemControl();
}
```

Рисунок 3.7 – Основна функція loop() термостату

Коли датчик не використовується, дисплей буде чергуватися між екранами 1 і 2, які показують поточні умови і робочий стан, а також дату і час, відповідно. Коли датчик використовується, прапорець input_active встановлюється в true, і він залишатиметься таким доти, доки користувач не повернеться до екрана 1, натиснувши на ручку датчика, щоб увімкнути кнопковий перемикач.

На цьому етапі можна вважати розробку завершеною. Головною задачею цього експерименту було продемонструвати розроблену схему і програмну частину.

ВИСНОВКИ

1. Встановлено, що AVR являє собою 8-розрядний RISC-мікроконтролер, що має швидке процесорне ядро, Flash-пам’ять програм, пам’ять даних SRAM, порти введення/виведення і інтерфейсні схеми. Мікроконтролер - це вдосконалена версія мікропроцесора. Він містить на кристалі центральний процесор (ЦП), пам'ять тільки для читання (ПЗП), оперативну пам'ять (ОЗП), блок вводу/виводу, контролер переривань тощо.
2. Були розглянуті різні категорії, типи та конфігурації AVR мікроконтролерів. Зазначені переваги й недоліки багатьох з них.
3. Проведено огляд лінійки пристроїв Arduino. Показані реальні розміри деяких пристроїв та розказано про їх архітектуру.
4. Детально показана робота з програмним забезпеченням Arduino IDE, пояснений процес завантаження програмного середовища, встановлення, налаштування додаткових бібліотек та конфігурація налаштувань.
5. Було змодельовано проєкт у вигляді термостату на платформі Arduino. Пояснено процес реалізації кожного етапу побудови приладу від схеми до програмної частини.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) Alf-Egil Bogen. The Story of AVR; 2008.
2) Elliot Williams. Make: AVR Programming. Maker Media. 2014.
3) J. M. Hughes. Practical Electronics: Components and Techniques. O'Reilly. 2015.
4) Timothy Margush. Some Assembly Required. CRC Press. 2011.
5) J. M. Hughes. Arduino: A Technical Reference. O'Reilly. 2016.
6) Simon Monk. Programming Arduino: Getting Started with Sketches. McGraw-Hill. 2011.
7) Grady Koch. Programming the Arduino. 2020.
8) Serhii Tsyrulnyk. SOFTWARE AND HARDWARE SYSTEM «ARDUINO LEARNER KIT». 2021.
9) Dimosthenis E. Bolanakis. Embedded Programming with Arduino. 2021.
10) Jonathan Oxer and Hugh Blemings. Practical Arduino. Apress. 2009.
11) Patrick Di Justo and Emily Gertz. Atmospheric Monitoring with Arduino. Maker Media. 2013.
12) Emily Gertz and Patrick Di Justo. Environmental Monitoring with Arduino. Maker Media. 2012.
13) Stanley Lippman. C++ Primer. Addison-Wesley. 2012.
14) Stephen Prata. C++ Primer Plus. Addison-Wesley. 2011.
15) Hamed Saghaei. Applied Projects using AVR microcontrollers (Series ATmega16, 32, and 128). 2021.
16) Gaute Myklebust. The AVR Microcontroller and C Compiler Co-Design. 2023.
17) S. Korbel, V. Janes .Interesting applications of Atmel AVR microcontrollers. 2004.
18) Ansgar Meroth, Petre Sora. Programming of AVR Microcontrollers. 2023.
19) Tomas Fryza. Basic C Code Implementations for AVR Microcontrollers. 2007.
20) Florian J. Knoll, Stephan Hussmann, Leif Ole Harder. Learn to program step-by-step: Advanced C Programming for AVR RISC Microcontroller (English Edition). 2021.