ПрАТ «ПРИВАТНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«ЗАПОРІЗЬКИЙ ІНСТИТУТ ЕКОНОМІКИ
ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ»

Кафедра комп'ютерної інженерії

ДО ЗАХИСТУ ДОПУЩЕНИЙ
Зав. кафедри
_____

д.т.н., проф. Переверзєв А.В.

ВИПУСКНА РОБОТА

Електронний сейф на базі контролера Arduino

Виконав
ст. гр. КІ-118к9 _____ Ю.Ю. Кійко

Керівник
доцент _____ К. С. Суха

Запоріжжя
2022

ПрАТ «ПРИВАТНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД «ЗАПОРІЗЬКИЙ ІНСТИТУТ ЕКОНОМІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ»

Кафедра комп'ютерної інженерії

ЗАТВЕРДЖУЮ
Зав. кафедрою
д.т.н., проф.
Переверзєв А.В.

13.03.2022 р.

З А В Д А Н Н Я

НА ВИПУСКНУ РОБОТУ МОЛОДШОГО СПЕЦІАЛІСТА

Студенту гр. *КІ-118к9*, спеціальності *123 «Комп'ютерна інженерія»*

Кійку Юрію Юрійовичу

Тема: *Електроний сейф на базі контролеру Arduino*

затверджена наказом по інституту 09.2-17 від 12 березня 2022 р.

2. Термін здачі студентом закінченої роботи: 19 червня 2022 р.

3. Перелік питань, що підлягають розробці:

1. *Вивчити літературу, присвячену темі розробки.*
2. *Ознайомитися з загальними питаннями розробки систем на мікроконтролерах.*
3. *Розглянути принципи роботи платформи Arduino, її характеристики та взаємодію з периферійними пристроями.*

4. *Проаналізувати схемотехнічні рішення, що дозволяють Arduino для моделювання електроного сейфу. Вивчити базові принципи роботи проекту.*

5. *Підібрати принципову електричну схему та вивчити принципи функціонування окремих компонентів та її функціонування в цілому*

6. *Освоїти методику роботи з інструментальними засобами розробки програмного забезпечення для мікроконтрелерів*

7. *Виконати монтаж компонентів схеми та налагодити оптимальні режими роботи пристрою*

8. *Запрограмувати пристрій та відлагодити його*

9. *Оформити результати роботи у вигляді звіту.*

Дата видачі завдання: 13 березня 2022 р.

Керівник випускної роботи _____ К. С. Суха

Завдання отримав до виконання _____ Ю. Ю. Кійко

РЕФЕРАТ

Пояснювальна записка складається з: 73 сторінки, 31 малюнок, 1 таблиця, 1 додаток, 34 першоджерела.

Об’єкт дослідження – «Електронний сейф на базі контролера Arduino».

Мета роботи – створення працюючої моделі сейфа на базі контролера.

Методи дослідження – проектування, моделювання та програмування мікроконтроллерної системи.

Для досягнення поставленої мети у цій дипломній роботі обрано апаратну частину проекту, що складається:

1. Контролер Arduino NANO;
2. Підвищуючий перетворювач XL6009;
3. Літій-іонний акумулятор 18500;
4. сервопривід MG995 Tower Pro 360;
5. Мембранна матрична клавіатура;
6. RTC DS1302;
7. LCD 1602;
8. I2C інтерфейсний модуль.

Створено програмний код для управління контроллером.

Створено робочий макет «Електронний сейф на базі контролера Arduino», що може бути використано для подальшого застосування на практиці.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

Слово / словосполучення	Скорочення
A	
Це опорна напруга, аналого-цифровий перетворювач (англ. Analog Reference)	AREF
D	
Це окремі факти, статистичні дані або елементи інформації (англ. Data)	DAT
E	
електрично стирається ПЗП, що перепрограмується (англ. Electrically Erasable Programmable Read-Only Memory)	EEPROM
G	
Точка нульового потенціалу мікросхеми (англ. Ground)	GND
I	
Послідовна асиметрична шина для зв'язку між інтегральними схемами усередині електронних приладів (англ. Inter-Integrated Circuit)	I2C
L	
Екран на основі рідких кристалів (англ. liquid crystal display)	LCD
M	
Вхід ведучого, вихід веденого (англ. Master In Slave Out)	MISO
Вихід ведучого, вхід веденого (англ. Master Out Slave In)	MOSI
R	
Електронний пристрій, який вимірює хід часу (англ. Real Time Clock)	RTC

Використовуються як файл для збереження параметрів програми та інших ресурсів програми	RESX
S	
послідовний синхронний стандарт передачі даних повного дуплексу (англ. Serial Peripheral Interface)	SPI
T	
одна з назв рідкокристалічного дисплея , в якому використовується активна матриця, керована тонкоплівковими транзисторами. (англ. thin film transistor)	TFT
U	
послідовний інтерфейс для підключення периферійних пристроїв до обчислювальної техніки (англ. Universal Serial Bus)	USB
Універсальний асинхронний приймач (англ. Universal asynchronous receiver/transmitter)	UART
V	
Напруга живлення схеми (англ. Voltage Collector-to-Collector)	VCC

ЗМІСТ

ВСТУП

РОЗДІЛ 1
РОЗГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ ПРОЕКТА «Електроний сейф на базі контролера Arduino»

1.1 Огляд функціювання сейфа

Усі сейфи діляться на кілька видів, виходячи зі своїх характеристик. Розглянемо основні критерії різновидів даних пристроїв.

За своїм призначенням сейфи поділяються на кілька типів, які наведені нижче.

Меблеві. Це найбільш поширений різновид компактного сейфа для дому або невеликого офісу. У даному сегменті можна зустріти і прості варіанти дорожчих та зламостійких рішеннях. Завдяки своїм компактним габаритам, вони легко можуть ховатися в меблях, але кріплення до стіни є обов'язковою умовою безпеки.

Рисунок 1.1.1 – Меблевий сейф

Збройові. Призначені для зберігання мисливської гладкоствольної та нарізної зброї. Залежно від розмірів, можуть вміщати у собі стволи різної довжини в різній кількості. Є обов'язковою умовою зберігання будь-якої вогнепальної зброї. Для дотримання норм закону достатньо купити

бюджетний варіант, який захистить від доступу дітей, гостей та сторонніх осіб. Для зберігання дорогої колекційної зброї можна взяти більш серйозний, стійкий до злому варіант.

Рисунок 1.1.2 – Збройовий сейф

Офісні. Сейфи для офісу зазвичай. більші за своїми габаритами, ніж меблеві. Вони являють собою шафи для зберігання документів, грошей та цінностей. Так само, як і інші варіанти, можуть мати клас зламостійкості. Також, оскільки вони кріпляться до підлоги, можуть бути і вогнетривкими.

Рисунок 1.1.3 – Офісний сейф

Депозитні. Це окремий різновид офісних. Відрізняються подібні рішення тим, що мають спеціальний проріз, подібний до поштової скриньки, через яку є можливість покласти в сейф гроші або папери не відкриваючи його. Це підвищує рівень безпеки у випадках, коли продавці або менеджери приймають готівку від клієнтів.

Рисунок 1.1.4 – Депозитний сейф

Готельні. Розроблено спеціально для зберігання цінностей постояльців готелю. Часто постачаються тільки кодовими замками, завдяки чому при утреті ключів не знадобиться їх механічне аварійне розтин. Як правило, не є злостійкими і створені більше для спокою за збереження цінних речей у номерах, а також для захисту від зазіхань на них з боку персоналу готелю.

Рисунок 1.1.5 – Офісний сейф

банківські. Серйозні протизламні рішення із спеціальними запірними механізмами.

Рисунок 1.1.6 – Банківський сейф

1.2 Огляд загальних характеристик сейфа

1.2.1 Характеристика за типом опору

За типом опору сейфи діляться на декілька типів які наведені нижче.

Зламостійкі. Здатні забезпечувати захист від розтину відмичками та силовими методами протягом певного проміжку часу. Також оснащені посиленими стінками, ригелями та антизрізами дверей. Кріплення може здійснюватися як до підлоги, так і до стіни, що несе. Згідно з європейською сертифікацією, кожному подібному сховищу призначається певний клас, який затверджується після проведення над ним випробувань, що включають спроби розтину брухтом, болгаркою, дрилем, кувалдою, газовим пальником та іншими спеціальними інструментами.

Вогнестійкі. Відрізняються здатністю протягом певного часу витримувати високі температури, перебуваючи в умовах відкритого вогню. Досягається це

завдяки наявності між стінками охолоджуючого розчину. Кріплення можливе лише до підлоги, тому що саме в цій точці опори температура є мінімальною. Тривалість часу, протягом якого такий сейф може перебувати у вогні, визначається класом його вогнестійкості відповідно до європейських стандартів.

Вогні та зломостійкі. Це симбіоз двох вищеописаних варіантів. Вони одночасно здатні протистояти як вогню, так і спробам несанкціонованого розтину.

1.2.2 Характеристика по типу встановлення

Наступна класифікація поділяє сейфи на кшталт їх монтажу. Відповідно до неї, сховища бувають.

Окремі. Чи не вбудовуються в стіни або меблі. Використовуються для зберігання цінностей та документів.

Вбудовані. Монтується у спеціально утворену нішу у стіні, що дозволяє приховати їх від сторонніх очей. Прихованість розташування такої схованки є додатковою перевагою.

Переносні. Використовуються для транспортування та інкасації. До цієї групи можна віднести схованки та кешбокси.

1.2.3 Характеристика по типу механізма замикання

Наступна класифікація поділяє сейфи на кшталт застосовуваного механізму замикання. Відповідно до неї існує 4 використовуваних типів замків які наведені нижче.

Ключові. Це зазвичай класичні сувальдні замки з ригелями.

Електронні. Ці замки відкриваються за допомогою введення ключа. У деяких варіантах використовується як єдиний замок, а в деяких – як додатковий.

Електронні замки із ключем для аварійного відкриття.

Комбіновані. Варіанти, у яких застосовується одразу кілька різних типів замків. Подібне рішення значно підвищує надійність.

1.3 Контролери Arduino

Мікроконтролер – виконаний у вигляді мікросхеми спеціалізований комп'ютер, що включає мікропроцесор, оперативну та постійну пам'ять для збереження виконуваного коду програм і даних, порти вводу-виводу і блоки зі спеціальними функціями.

Використовується для керування електронними пристроями. здатний виконувати прості завдання. Використання однієї мікросхеми значно знижує розміри, енергоспоживання і вартість пристроїв, побудованих на базі мікроконтролерів.

Рисунок 1.3.7 – Контролер

Arduino - це електронна платформа з відкритим вихідним кодом, яка дозволяє взаємодіяти з навколишнім світом. Завдяки їй можна створити все, що спаде на думку - від простих електронних приборів або автоматизації побуту.

Рисунок 1.3.8 – Логотип Arduino

Плати мають на борту все необхідне для комфортної роботи, але їхньої функціональності часто буває недостатньо. Щоб зробити свій проєкт більш інтерактивним, можна використовувати різні модулі та плати розширень, сумісні з платформою Arduino. Сюди входять датчики (температури, освітлення, вологи, газу/диму, атмосферного тиску), пристрої введення (клавіатури, джойстики, сенсорні панелі) та виведення (сегментні індикатори, LCD/TFT дисплеї, світлодіодні матриці).

На програмному рівні платформа Arduino є безкоштовним середовищем розробки Arduino IDE. Мікроконтролери треба програмувати мовою C++, з деякими відмінностями та полегшеннями, створеними для швидкої адаптації початківців. Компіляцію програмного коду та прошивку мікроконтролера середовище розробки бере на себе.

Найпопулярнішими контролерами Arduino є Arduino NANO та Arduino UNO.

Arduino Uno – Це, можна сказати, стандарт лінійки контролерів Arduino. Для нього написано багато скетчів, і в інтернеті можна знайти безліч проектів. Він є найпростішим і зрозумілішим для новачків, тому що багато навчальних наборів побудовані на застосуванні саме цього контролера. Контролер підтримує всі базові інтерфейси (UART, I2C, SPI). Плата легко підключається до комп'ютера за допомогою USB-кабелю - що також є додатковою зручністю роботи з

контролером. В основному пам'яті та потужності чіпа достатньо для виконання щодо простих та поширених завдань.

Рисунок 1.3.9 – Arduino UNO

Arduino Nano – Даний контролер дуже схожий по функціоналу на лінійку Uno, адже побудовані плати на тому самому контролері (ATmega328P). Основна відмінність плат Nano – в їх формфакторі – такі плати набагато менші за розміром, що робить зручним розміщення їх у малих корпусах, або в місцях, де є обмеження простору.

Рисунок 1.3.10 – Arduino NANO

1.4 Додаткові компоненти управління

Для управління електронним сейфом знадобляться додаткові компоненти, для замикання та взаємодії людини з приладом. Також у електроному сейфі буде реалізована система запису останнього відкриття сейфу яка буде виводитися на головний єкран.

Замикання в електронному сейфі буде здійснюватися за допомогою сервоприводу, для зразка який не має класу захисту сервопривід буде виконувати функцію замикання дуже добре, оскільки сейф буде дуже легким та компактним.

Сервоприводом є будь-який тип механічного приводу (пристрою, робочого органу), що має у складі датчик (положення, швидкості, зусилля тощо) і блок управління приводом, що автоматично підтримує необхідні параметри на датчику (і відповідно, на пристрої) відповідно до заданого зовнішнього значення (положення ручки управління або чисельного значення від інших систем). Простіше кажучи, сервопривід є "автоматичним точним виконавцем" - отримуючи на вхід значення керуючого параметра (в режимі реального часу), він "своїми силами" (на основі показань датчика) прагне створити і підтримувати це значення на виході виконавчого елемента.

Рисунок 1.4.11 – Сервопривід

Людина повинна взаємодіяти з сейфом, для налаштування захисту вмісту

сейфа. Для електронного типу замикання рішення про доступ осіб приймається на основі сигналів від різних датчиків : біометричних датчиків, набірної клавіатури, дистанційного керування і т.д. Часто є частиною складної контролю доступу, іноді невіддільний від неї. Як виконавчі механізми використовуються електромеханічні або електромагнітні запірні пристрої.

Кодовий механізм замикання – замок, для відкриття якого необхідно ввести з клавіатури кодову послідовність,

Рисунок 1.4.12 – Набірна клавіатура.

Біометричний механізм замикання - електронний замок , в основі роботи якого лежить біометрична автентифікація , тобто підтвердження прав доступу за біометричними параметрами, наприклад, відбитками пальців.

Рисунок 1.4.13 – Біометричний сканер

1.5 Висновки за розділом

Здійснивши огляд загальних характеристик та функціонування сейфа, було прийнято рішення про створення готельний сейф з кодовим замком, оскільки сейф буде макетним, класу захисту він не буде мати. також буде встановлено деякі модифікації, наприклад запис дати та часу відкриття сейфа.

Для керування розумним сейфом буде використаний мікроконтролер Arduino Nano, аналог версії UNO, для роботи з невеликою кількістю систем управління він ідеально підходить, також він більш компактний.

РОЗДІЛ 2
ВИБІР ТА ОБҐРУНТУВАННЯ ПРОГРАМНО-АПАРАТНОГО КОМПЛЕКСУ

2.1 Загальні характеристики Arduino NANO

Arduino Nano - це невелика, повнофункціональна налагоджувальна плата, адаптована для роботи з макетними платами, побудована на базі мікроконтролера ATmega328p

Таблиця 2.1.1

Характеристики Arduino Nano:

Характеристика	Заначення
Мікроконтролер	ATmega328P
Тип корпусу	TQFP-32
Робоча напруга	5В
Вхідна напруга (рекомендована)	7-12В
Цифрових входів / виходів	14 (з яких 6 можуть бути використані як PWM)
Аналогових входів	8
Сила струму на входах / виходах	40 мА
Сила струму для 3.3В виходу	50 мА
Пам'ять	32 кБ з яких 2кб використовується бутлоадер
SRAM	2 кБ
EEPROM	1 кБ
Частота	16 МГц

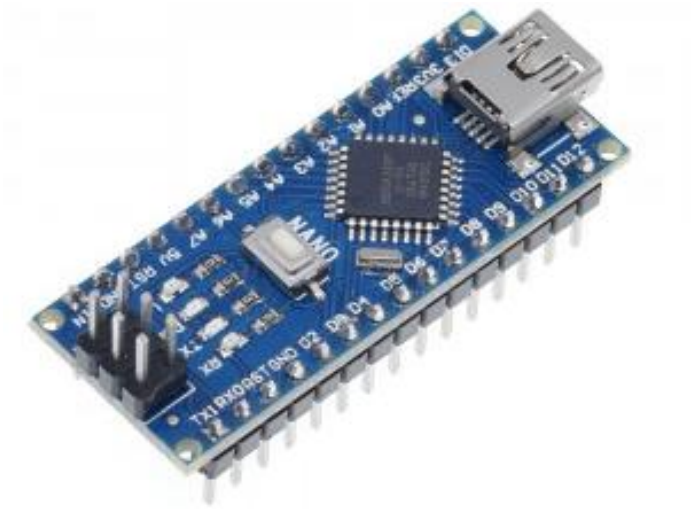

Рисунок 2.1.1 – Мікроконтролер Arduino NANO.

Таблиця 2.1.2

Характеристики ATmega328p

Характеристика	Заначення
Тактова частота	0 – 20 МГц
Об'єм Flash-пам'яті	32 кб
Об'єм SRAM-пам'яті	2 кб
Об'єм EEPROM-пам'яті	1 кб
Напруга живлення	1,8 - 5,5 В
Струм, що споживається в режимі роботи	0,2 мА (1 МГц, 1,8 В)
Споживаний струм у режимі сну	0,75 мкА (1 МГц, 1,8 В)
Кількість таймерів/лічильників	2 восьмибітні, 1 шістнадцятибітний
Загальна кількість портів	23
Кількість PWM виходів	6
Кількість каналів ADC (аналогові входи)	6
Кількість апаратних USART (Serial):	1
Кількість апаратних SPI:	1 Master/Slave

Продовження таблиці 2.1.2

Кількість апаратних I2C/SPI:	1
Дозвіл ADC:	10 біт

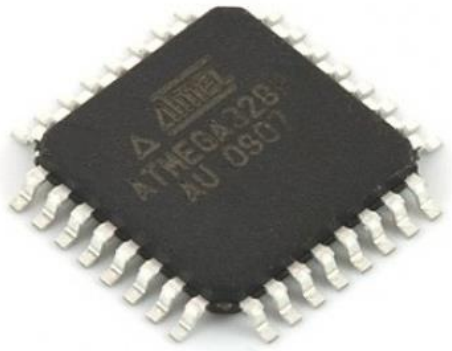

Рисунок 2.1.2 – ATmega328p

2.1.1 Принцип роботи.

Для підключення систем управління використовуються цифрові та аналогові порти, комунікація між пристроями та контролером виконується аналоговими та цифровими сигналами.

Цифровий сигнал — сигнал, який можна як послідовності дискретних значень. У наш час найпоширеніші двійкові цифрові сигнали (бітовий потік) у зв'язку з простотою кодування та використанням у двійковій електроніці. Для передачі цифрового сигналу аналоговими каналами використовуються різні види маніпуляції.

В цифровій електроніці цифровий сигнал це послідовність імпульсів (сигнал в імпульсно-амплітудній модуляції), тобто. послідовність електричних імпульсів квадратної форми із фіксованою довжиною, кожен з яких може займати один із рівнів амплітуди, яких можлива дискретна кількість. Особливим випадком є логічний сигнал або двійковий сигнал, який змінюється між високим і низьким рівнем сигналу.

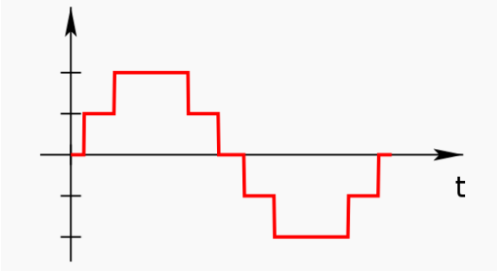

Рисунок 2.1.3 – Цифровий сигнал

Аналоговий сигнал — сигнал (напруга, струм тощо), неперервний на всьому проміжку часу. Аналоговий сигнал є або вираженим синусоїдальним коливанням, накладанням синусоїдальних коливань певної амплітуди і частоти. Протилежністю аналоговим сигналам є дискретний сигнал, який має обмежені часові рамки (дискрета, імпульс). Аналоговий сигнал є традиційним для використання у радіо-телекомунікаційних системах, системах автоматичного керування тощо. При передачі інформації аналоговим сигналом, його видозміна можлива шляхом зміни частоти чи амплітуди коливань.

Перевагою аналогового сигналу над дискретним є відсутність невизначеності між відліками, яку має дискретний сигнал.

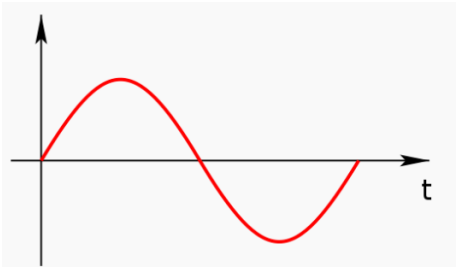

Рис 2.1.4 – Аналоговий сигнал

Аналоговий сигнал використовує певні властивості середовища для передачі інформації. В електричних сигналах для передачі інформації, може змінюватися напруга, струм, фаза або частота сигналу.

Таблиця 2.1.3

Світлодіодна індикація контролеру Arduino NANO

Ім'я світлодіода	Призначення
RX та TX	Блимають під час обміну даними між Arduino Nano та ПК.
L	Користувальницький світлодіод підключений до 13 пін мікроконтролера. При високому рівні світлодіод вмикається, за низького – вимикається.
ON, POW або PWR	Наявність живлення на Arduino Nano.

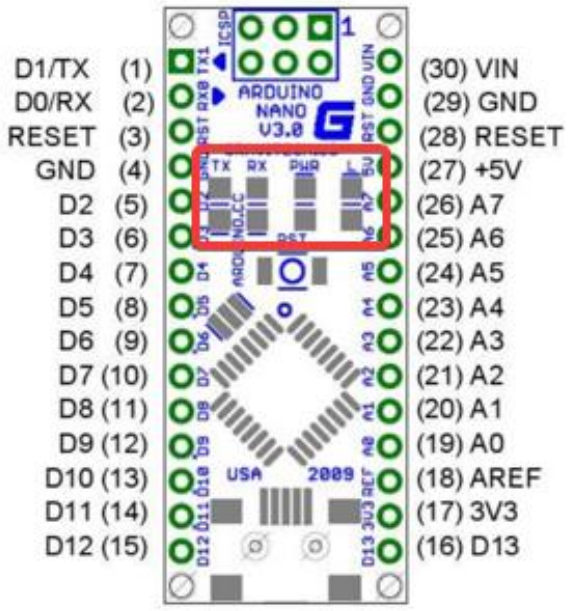

Рисунок 2.1.5 – Світлодіодні індикатори контролеру Arduino NANO

2.1.2 Роспіновка контролеру

Піни живлення:

VIN: Вхідний пін для підключення зовнішнього джерела живлення з напругою від 7 до 12 вольт.

5V: Вихідний пін від регулятора напруги на платі з виходом 5 вольт та максимальним струмом 800 мА. Живити пристрій через висновок 5Vне

рекомендується - ви ризикуєте спалити плату.

3.3V: Вихідний пін від стабілізатора мікросхеми FT232R з виходом 3,3 вольта та максимальним струмом 50 мА. Живити пристрій через пін 3V3не рекомендується - ви ризикуєте спалити плату.

GND: Вихід землі.

AREF: Пін для підключення зовнішньої опорної напруги АЦП щодо якого відбуваються аналогові виміри при використанні функції analogReference() з параметром EXTERNAL.

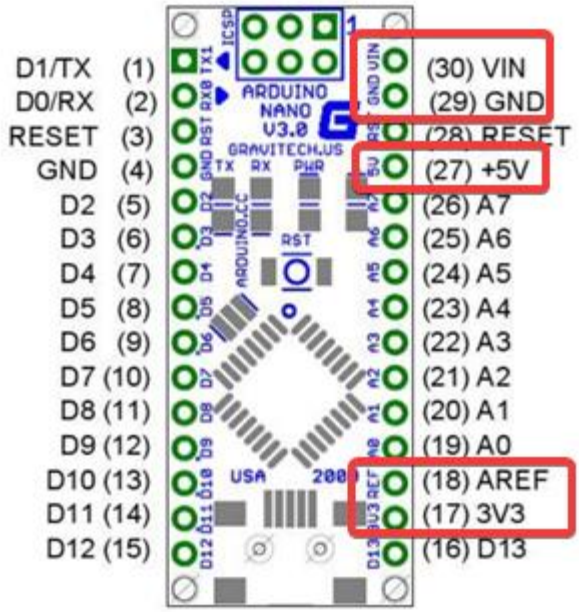

Рисунок 2.1.6 – Піни живлення контролеру

Порти введення та виводу дуже різнообразні, та мають багато характеристик і виконуюсь не тільку свою основну функцію, а і додаткову.

Цифрові входи/виходи: піни 0– 13 Логічний рівень одиниці – 5 В, нуля – 0 В. Максимальний струм виходу – 40 мА. До контактів підключені резистори, що підтягують, які за замовчуванням вимкнені, але можуть бути включені програмно.

PWM: піни 3, 5, 6, 9, 10і 11 дозволяє виводити аналогові значення у вигляді PWM-сигналу. Розрядність PWM не змінюється та встановлена в 8 біт.

ADC: піни A0– A7Дозволяє уявити аналогову напругу у цифровому

вигляді. Розрядність ADC не змінюється і встановлена на 10 біт. Діапазон вхідної напруги від 0 до 5 В. При подачі більшої напруги ви вб'єте мікроконтролер.

TWI/I2C: піни A4(SDA)і A5(SCL) Для спілкування з периферією за інтерфейсом I2C. Для роботи використовуйте бібліотеку Wire .

SPI: піни 11(MOSI), 12(MISO), 13(SCK)і 10(SS) спілкування з периферією за інтерфейсом SPI. Для роботи – використовуйте бібліотеку SPI .

UART: піни 0(RX)та 1(TX) Використовується для комунікації плати Arduino з комп'ютером або іншими пристроями за послідовним інтерфейсом. Виходт 0(RX)та 1(TX)з'єднані з відповідними USB-UART перетворювачами FT232R . Для роботи з послідовним інтерфейсом використовуйте методи бібліотеки Serial.

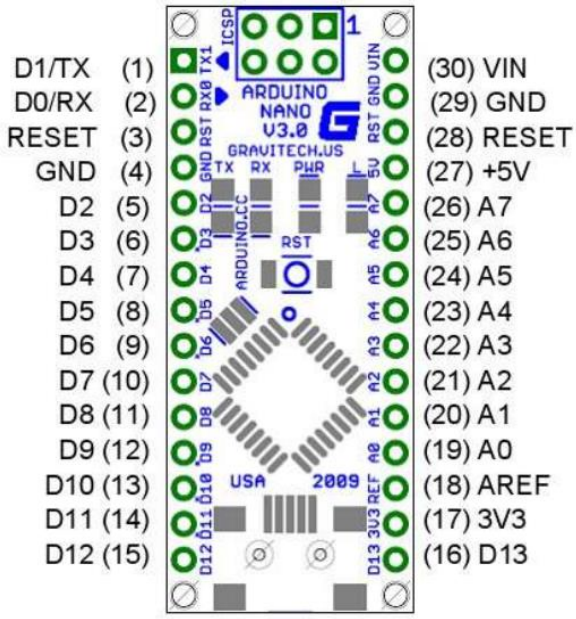

Рисунок 2.1.6 – Роспіновка контроллера

2.2 Вибір додаткових компонентів

Оскільки контролер був обраний, далі для конструювання повноцінного проекту треба обрати додаткові компоненти.

2.2.1 Живлення сейфу

Підключати все це можна безпосередньо до пін або, у разі тестування системи, просто скористатися USB-портом. Система сама розрахує, скільки їй потрібно, і внутрішніми силами перетворює вхідну напругу до відповідних значень.

Живлення на Arduino може підключатися через три різні джерела які наведенні нижче.

Mini USB, коли ви тестуєте проект на ПК. Це дуже важливий і зручний момент, адже немає необхідності, під час програмування та тестування вашого продукту, підводити струм додатково, що заощаджує сили. А наявність систем, що дають змогу через таке джерело регулювати характеристики струму, спрощує деякі завдання.

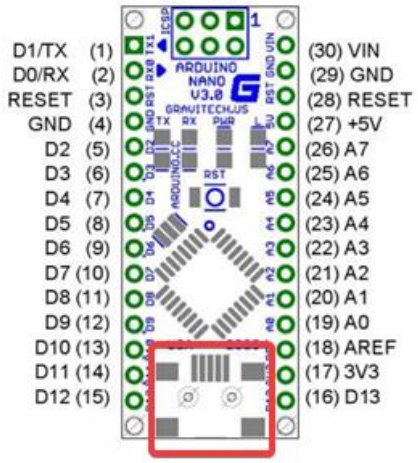

Рисунок 2.2.7 – Порт для живлення через Mini USB

Безпосередньо через нерегульовані джерела 6-20 вольт. Це відбувається через 30 пін, і подібно до виходу на цифровий сигнал, цей вхід сприймає весь діапазон

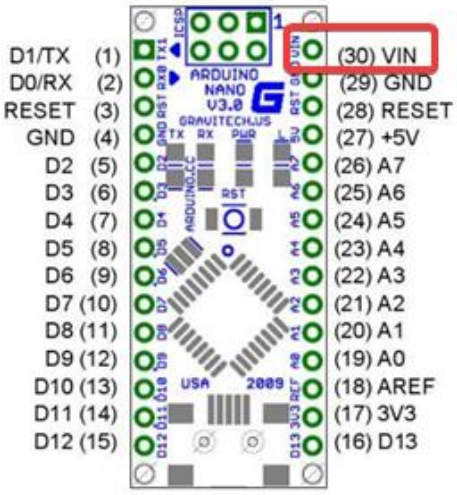

Рисунок 2.2.8 – Порт живлення через порт VIN

Через регульовані джерела 5 вольт. Це стандартний та часто використовуваний спосіб подавати харчування на Arduino uno. У ньому є невеликий недолік, який полягає в тому, що вам потрібно якось перетворити вхідну напругу до 5 вольт, але рішень даної задачі вже безліч, і всі їх ви можете знайти у відкритому доступі на нашому сайті. Цей вхід знаходиться на 27 піні.

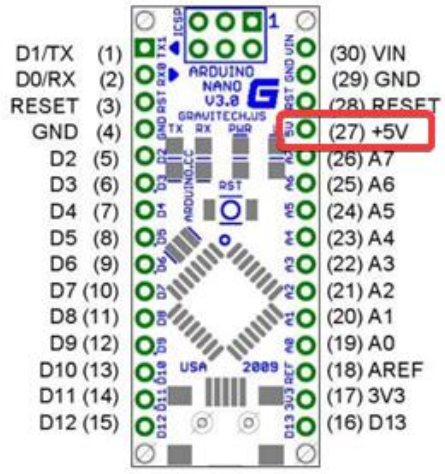

Рисунок 2.2.9 – Порт живлення через порт +5V

2.2.2 Інші компоненти

Літій-іонний акумулятор – це особливий електричний акумулятор, який включає іони літію. Такі батареї широко поширені в сучасній побутовій електронній техніці та використовуються у різних побутових та промислових приладах.

Рисунок 2.2.10 – Літій-іонний акумулятор 18500

Переваги акумулятора:

1. Має високу реальну ємність, що відповідає заявленій (2850 мАг).
2. Здатний працювати з високими струмами навантаження до 10 А.
3. Видає повну потужність навіть за мінімального рівня заряду акумулятора.

Оскільки літій-іонний акумулятор 18500 має напругу 3.7 буде використовуватися Підвищуючий перетворювач XL6009

Підвищуючий перетворювач XL6009 - це перетворювач напруги, що підвищує, створений на основі мікросхеми XL6009. Пристрій здатний видавати стабільну напругу на виході від 1.25 до 35 вольт при будь-якій вхідній напрузі від 5 до 32 вольт.

Також має захист від короткого замикання та перегріву, що забезпечує додаткову безпеку в роботі з пристроєм. Має компактні розміри, невисоку вартість і досить широкий діапазон як вхідної, так і вихідної напруги, що дозволяє

його використовувати в різних приладах.

Рисунок 2.2.11 – Підвищуючий перетворювач XL6009

Для управління електронним сейфом знадобляться компоненти для замикання та взаємодії людини з приладом.

Замикання в електронному сейфі буде здійснюватися за допомогою сервоприводу, було обрано сервопривід MG995 Tower Pro 360.

MG995 Tower Pro 360 – це цифрова модель сервоприводу з досить потужним моментом, що крутить, аж до 12 кг при живленні пристрою в 6 В.

Характеристики:

1. Напруга живлення: 4,2 - 7,2 В
2. Кут повороту: 360 °
3. Зусилля на валу при живленні 4,8 В: 9 кг/см
4. Зусилля на валу при живленні 6 В: 12 кг/см
5. Робоча швидкість при живленні 4,8 В: 0,17 с/60°
6. Робоча швидкість при живленні 6 В: 0,13 с/60°
7. Довжина дроту: 300 мм
8. Розміри: 40 х 19 х 42 мм
9. Вага: 55 г

Рисунок 2.2.12 – сервопривід MG995 Tower Pro 360

Для віддчинення сейфа методом кодового пароля потрібен пристрій введення. Було обранно мембранну матричну клавіатуру.

Мембранна матрична клавіатура для Arduino – це компактна недорога клавіатура для введення буквено-цифрових даних на Ваш пристрій, створений на основі Arduino. Клавіатура має кнопки в 4 ряди, з яких 10 цифрових, 4 буквені та 2 кнопки з символами. За допомогою такого модуля Ви можете значно розширити функціонал пристрою та вводити потрібні команди вручну, програмувати та займатися налагодженням робочих процесів.

Рисунок 2.2.13 – Мембранна матрична клавіатура

Клавіатура легка та зручна в підключенні. Для того, щоб підключити пристрій, потрібно лише з'єднати шлейф з Вашим приладом. Модуль виконаний із міцного інженерного пластику, а запланований час життя складає аж до 1 мільйона натискань. Розмір клавіатури складає всього 77х70 мм, а вага 10 грам, що дозволяє використовувати клавіатуру навіть у найменших роботизованих пристроях.

Для запам'ятовування пароля для подальшого використання, буде використовуватися енергонезалежна пам'ять EEPROM.

EEPROM – електрично стирається ПЗУ (ЕСППЗУ), що перепрограмується, один з видів енергонезалежної пам'яті (таких, як PROM і EPROM). Пам'ять такого типу може стиратися та заповнюватися даними до мільйона разів.

Принцип роботи EEPROM заснований на зміні та реєстрації електричного заряду в ізольованій області напівпровідникової структури.

Зміна заряду «запис» і «стирання» проводиться додатком між затвором та витоком великого потенціалу, щоб напруженість електричного поля в тонкому діелектрику між каналом транзистора та кишенею виявилася достатньою для виникнення тунельного ефекту. Для посилення ефекту тунелювання електронів у кишеню під час запису застосовується невелике прискорення електронів шляхом пропускання струму через канал польового транзистора явище інжекції гарячих носіїв.

Читання виконується польовим транзистором для якого кишеня виконує функцію затвора. Потенціал плаваючого затвора змінює порогові характеристики транзистора, що реєструється ланцюгами читання.

Основною особливістю класичного осередку EEPROM є наявність другого транзистора, який допомагає керувати режимами запису та стирання. Деякі реалізації виконувались у вигляді одного тризатворного польового транзистора, один затвор плаваючий і два звичайні.

Реалізація виведення основної інформації щодо сейфа, буде через LCD дисплей

LCD 1602 – екран типу LCD для виведення команд та тексту у візуальний вигляд. Такий дисплей чудово поєднується з різними платами розробки Raspberry Pi, та багатьма іншими, і стане чудовим доповненням Вашого роботизованого пристрою.

Дисплей має підсвічування по всій поверхні та забезпечує візуалізацію 16-ти символів у два рядки символами. Кожен із символів складається з точкової матриці 5x8, що забезпечує відмінну читаність.

Рисунок 2.2.15 – LCD 1602

LCD – електронний пристрій візуального відображення інформації, принцип дії якого ґрунтується на явищі електричного переходу в рідких кристалах. Дисплей складається з довільної кількості кольорових або монохромних точок, і джерела світла або відбивача. Кожна з кольорових точок рідкокристалічного дисплея складається з кількох комірок, попереду яких встановлюються світлові фільтри. Тобто колір певної точки і її яскравість визначається інтенсивностями світіння комірок, з яких вона складається.

Характеристики:

1. Тип дисплея: LCD;
2. Драйвер: PCF8574;
3. Інтерфейс: I2C (0x27);
4. Роздільна здатність: 16 стовпців, 2 рядки;
5. Колір блакитний;
6. Час відгуку: 250 мс;
7. Кут огляду: 35 градусів;
8. Енергоспоживання: ~4 mA екран, ~120 мА підсвічування;
9. Напруга живлення: 5 вольт;
10. Робочі температури: -20...+70 градусів;
11. Розміри: 27,7 х 42,6 мм.

LCD дисплей має 16 пінів для комфортного користування, для контролера Arduino NANO це половина всіх пінів, для скорочення кількості дисплеєм входів та виходів контролера, прийнято рішення використати інтерфейсний модуль I2C.

I2C інтерфейсний модуль на мікросхемі PCF8574T для розширення кількості портів вводу/виводу для контролерів. Може використовуватися як інтерфейсна плата для підключення LCD 1602 і 2004, так і як самостійний пристрій. Для регулювання контрастності дисплея встановлений змінний резистор.

При використанні модуля як розширювач портів вводу/виводу слід враховувати те, що вивід P3 має інверсний вихід з відкритим колектором.

У деяких партіях розширювачів встановлені мікросхеми PCF8574AT, які повністю ідентичні за функціональними можливостями але відрізняються діапазоном адрес шини I2C замість 0x20-0x27, вони відгукуються на адреси 0x38-0x3f.

Рисунок 2.2.16 – I2C інтерфейсний модуль

Характеристики:

1. Інтерфейсна мікросхема: PCF8574AT/T;
2. Інтерфейс: I2C;
3. Діапазон адрес I2C: PCF8574T - 0x20-0x27 PCF8574AT - 0x38-0x3f;
4. Максимальна кількість під'єднаних однотипних модулів: 8;
5. Напруга живлення: 5 В;
6. Розмір: 5.2 х 1.8 х 1.4 см;
7. Сумісність: РКІ 1602 и 2004.

2.3 Середовище розробки

Середовище розробки Arduino складається з вбудованого текстового редактора програмного коду, області повідомлень, вікна виведення тексту (консолі), панелі інструментів з кнопками команд, що часто використовуються, і декількох меню. Для завантаження програм та зв'язку середовище розробки підключається до апаратної частини Arduino.

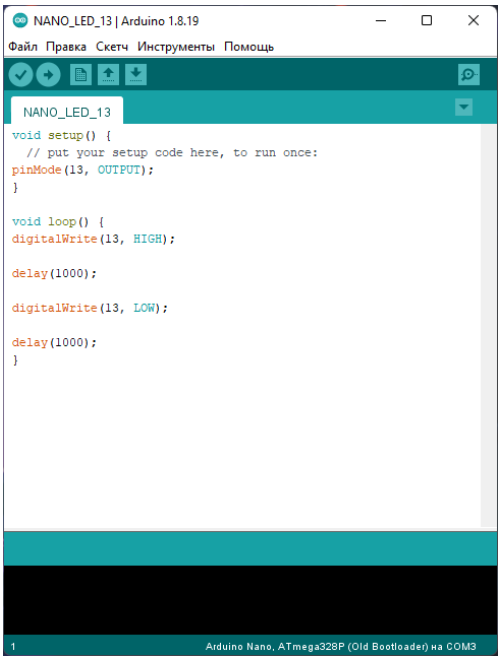

Рисунок 2.3.17 – Інтерфейс Arduino IDE

Програма, написана серед Arduino, називається скетч. Скетч пишеться в текстовому редакторі, який має інструменти вирізки/вставки, пошуку/заміни тексту. Під час збереження та експорту проекту в області повідомлень з'являються пояснення, також можуть відображатися помилки. Вікно виводу тексту (консоль) показує повідомлення Arduino, які містять повні звіти про помилки та іншу інформацію. Кнопки панелі інструментів дозволяють перевірити та записати програму, створити, відкрити та зберегти скетч, відкрити моніторинг послідовної шини.

Головний інтерфейс Arduino IDE складається:

Verify – перевірка код на помилки;

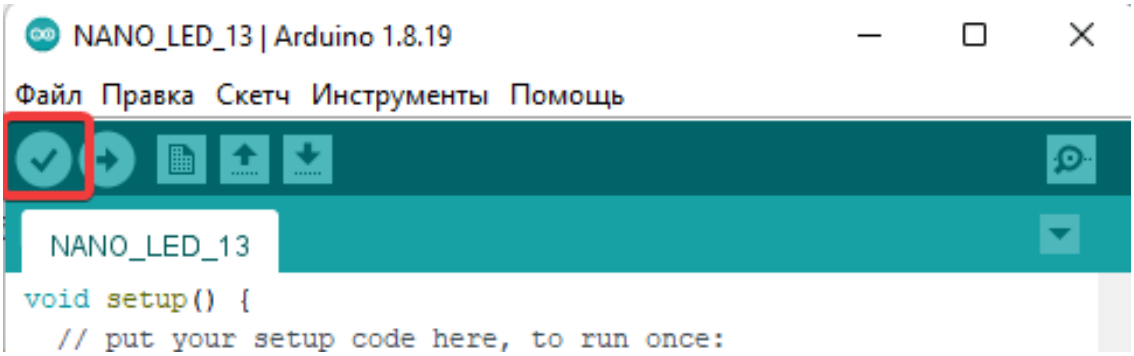

Рисунок 2.3.18 – Елемент інтерфейсу Verify

Upload – скомпілювати програму та "зашити" її в контролер;

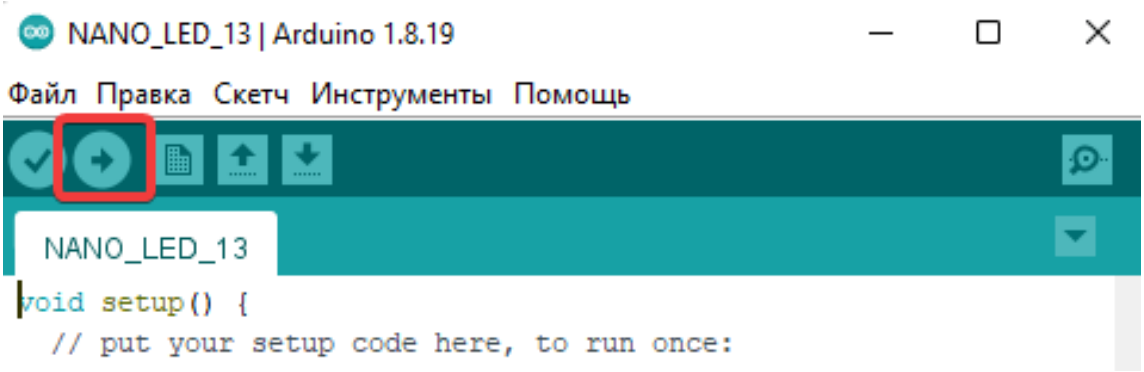

Рисунок 2.3.19 – Елемент інтерфейсу Upload

New – створити нову програму;

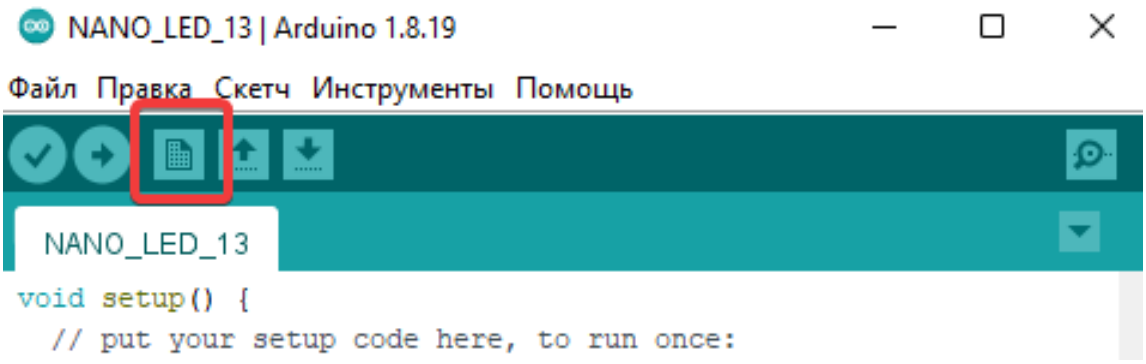

Рисунок 2.3.20 – Елемент інтерфейсу New

Open – команда відкриває меню зі списком усіх скетчів, доступних у вашій робочій папці. Після клацання файлу його вміст відкриється у поточному вікні.

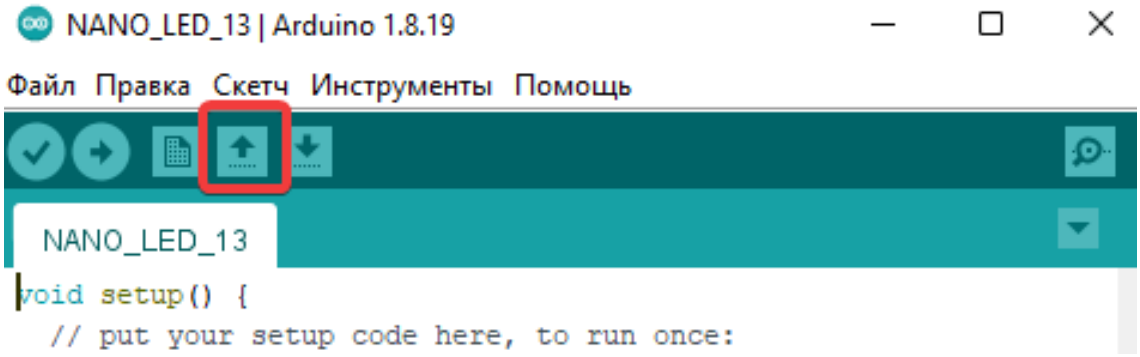

Рисунок 2.3.21 – Елемент інтерфейсу Open

Save – зберегти програму

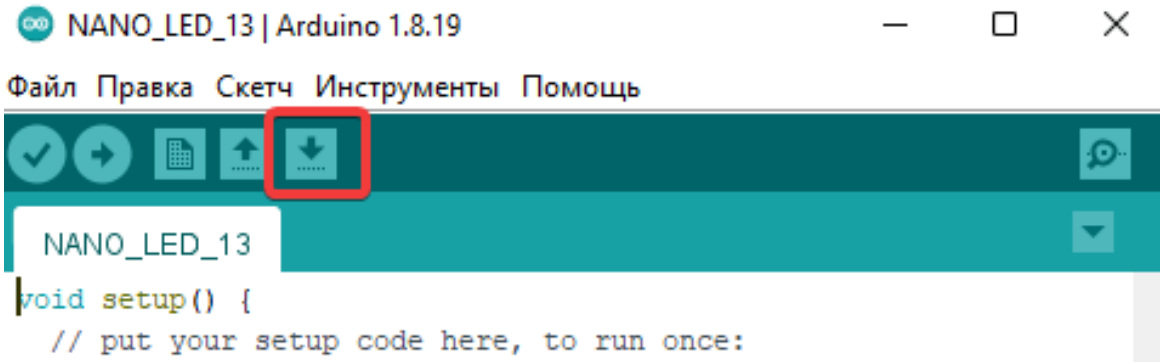

Рисунок 2.3.22 – Елемент інтерфейсу Save

Serial Monitor - відкрити програму " Serial Monitor " для роботи з послідовним інтерфейсом;

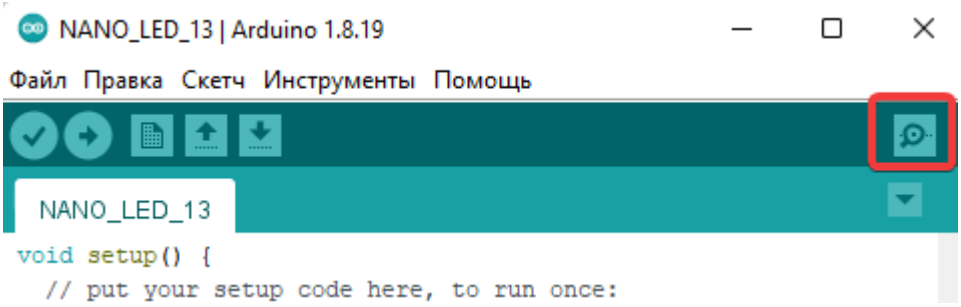

Рисунок 2.3.23 – Елемент інтерфейсу Serial Monitor

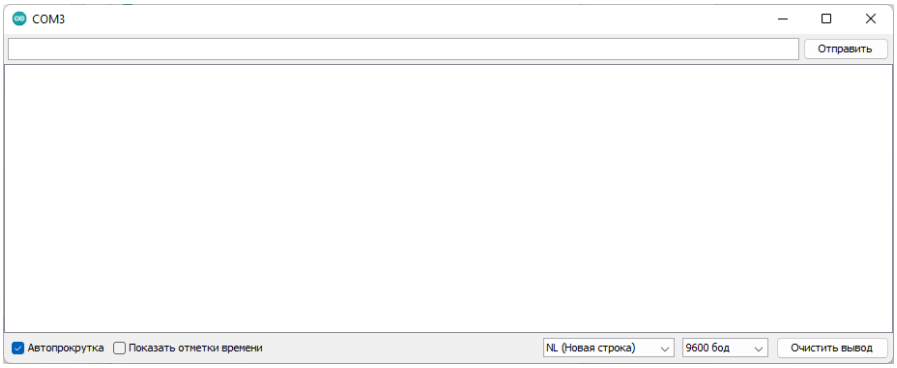

Рисунок 2.3.24 – програма "Serial Monitor"

Додаткові команди знаходяться в меню: File , Edit , Sketch , Tools та Help . У цих меню завжди активні ті пункти, які можна застосувати до поточного елемента або фрагмента коду.

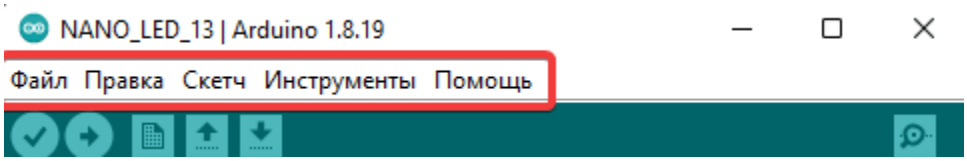

Рисунок 2.3.25 – Додаткові команди Arduino IDE

Меню Edit:

Copy for Forum (Скопіювати для форуму) Скопіювати код програми у буфер обміну у спеціальному форматі, зручному для постінгу на форум;

Copy as HTML (Скопіювати як HTML) Скопіювати код програми у буфер обміну у вигляді HTML-коду, зручного для вбудовування у веб-сторінки;

Меню Sketch:

Verify/Compile (Перевірити/Компілювати) Перевірити код програми на

помилки

Show Sketch Folder (Показати папку програми) Відкрити папку з файлом поточного скетчу.

Add File (Додати файл)Додає вихідний файл до поточної програми. Доданий файл з'явиться у новій вкладці головного вікна. Видалити файли можна за допомогою таб-меню.

Import Library (Імпорт бібліотеки) Додає бібліотеку до вашої програми шляхом вставки оператора #include спочатку коду. Докладніше про бібліотеки див. нижче. Крім того, у версіях IDE 1.0.5 та вище реалізована можливість імпортування бібліотек із .zip-архівів.

Меню Tools:

Auto Format (Автоформат) Ця команда наводить красу у вашому коді, а саме: робить однакові відступи відповідних відкриваючих та закривають фігурних дужок, додаткові відступи коду всередині логічних блоків.

Archive Sketch (Заархівувати скетч) Створити .zip-архів поточного скетчу. Результуючий архів міститься в папці з програмою.

Board (Плата) Вибрати модель Ардуїно. Опис різних плат Ардуїно див. нижче .

Serial Port (Послідовний порт) Це меню містить список усіх послідовних пристроїв, які є в системі (як фізичних, так і віртуальних). Їх список повинен оновлюватися автоматично, коли ви відкриваєте головне меню.

Programmer (Програматор) Дозволяє вибрати зовнішній програматор для прошивки мікроконтролера без використання з'єднання USB. Зазвичай ця функція потрібна рідко - наприклад для прошивки завантажувача в новий мікроконтролер.

Burn Bootloader (Прошити завантажувач) Це меню дозволяє прошити завантажувач у контролер Ардуїно. При нормальній роботі Ардуїно ця функція зазвичай не потрібна, але вона може бути корисною, якщо вам раптом потрібно

замінити мікроконтролер ATmega на новий (який з магазину йде без завантажувача). Перед прошивкою переконайтеся, що у меню Boards вибрано саме вашу плату.

2.4 Мова програмування

В Arduino IDE всі написані скетчі компілюються в програму мовою C/C++ із мінімальними змінами. Компілятор Arduino IDE значно спрощує написання програм для цієї платформи і створення пристроїв на Arduino.

Рисунок 2.4.26 – Логотип мови програмування C++

Мова програмування Arduino називається Arduino C і є мовою C++ з фреймворком Wiring, він має деякі відмінності в частині написання коду, який компілюється і збирається за допомогою avr-gcc, з особливостями, що полегшують написання працюючої програми – є набір бібліотек, що включає у собі функції та об'єкти. При компіляції IDE створює тимчасовий файл з розширенням *.cpp .

Програми, написані програмістом Arduino, називаються начерки або скетчі і зберігаються у файлах із розширенням *.ino. Ці файли перед компіляцією обробляються препроцесор Arduino. Також існує можливість створювати та підключати до проекту стандартні файли C++.

Програміст повинен написати дві обов'язкові для Arduino функції setup() та loop(). Перша викликається одноразово при старті, друга виконується у нескінченному циклі. У текст своєї програми (скетчу) програміст не повинен вставляти заголовні файли використовуваних стандартних бібліотек. Ці заголовні файли додасть препроцесор Arduino відповідно до конфігурації проекту. Однак користувацькі бібліотеки потрібно вказувати.

Менеджер проекту Arduino IDE має нестандартний механізм додавання бібліотек. Бібліотеки у вигляді вихідних текстів на стандартному C++ додаються до спеціальної папки в робочому каталозі IDE. Назва бібліотеки додається до списку бібліотек у меню IDE. Програміст зазначає необхідні бібліотеки, і вони вносяться до списку компіляції.

Arduino IDE не пропонує жодних налаштувань компілятора та мінімізує інші налаштування, що спрощує початок роботи для новачків та зменшує ризик виникнення проблем; але є директиви препроцесора, такі як #define, #include та багато інших.

Так виглядає повний текст найпростішої програми (скетчу) миготіння світлодіодом, підключеного до 13-го виведення (піну) Arduino, з періодом 2 секунди. Він доступний в середовищі розробки в Скетч>приклади>стандартні>Blink.

Лістинг 2.3.1 – Приклад скетчу

```
void setup () {
      pinMode (13,OUTPUT); // Призначення порту 13 як вихідний порт
}
void loop () {
    digitalWrite(13,HIGH); // Установка порту 13 в стан "1", світлодіод загоряється
```

```
    delay(1000);   //   Затримка   на   1000   мілісекунд
digitalWrite ( 13 , LOW );
    digitalWrite(13,LOW); // Установка порту 13 стан "0",
світлодіод гасне
    delay ( 1000 ); // Затримка на 1000 мілісекунд
}
```

2.5 Висновки за розділом

Розглянувши варіанти для програмно-апаратно комплексу, було вирішено взяти такі компоненти:

1. Контролер Arduino NANO;
2. Підвищуючий перетворювач XL6009;
3. Літій-іонний акумулятор 18500;
4. сервопривід MG995 Tower Pro 360;
5. Мембранна матрична клавіатура;
6. LCD 1602;
7. I2C інтерфейсний модуль.

У якості середовища розробки було вирішено використовувати Arduino IDE, мова програмування яка використовується в Arduino IDE – Arduino C.

РОЗДІЛ 3

РЕАЛІЗАЦІЯ СИСТЕМИ «Електроний сейф на базі контролера Arduino»

3.1 Проєктування та збір проєкту

Продовжуючи тему живлення контролеру було обрано 2-й тип живлення через нерегульовані джерела, також було прийнято рішення про повністю автономну схему живлення, через акумулятор типу 18500 та перетворювач напруги.

Для працездатності контролера є необхідність у живленні контролеру, отже інших компонентів.

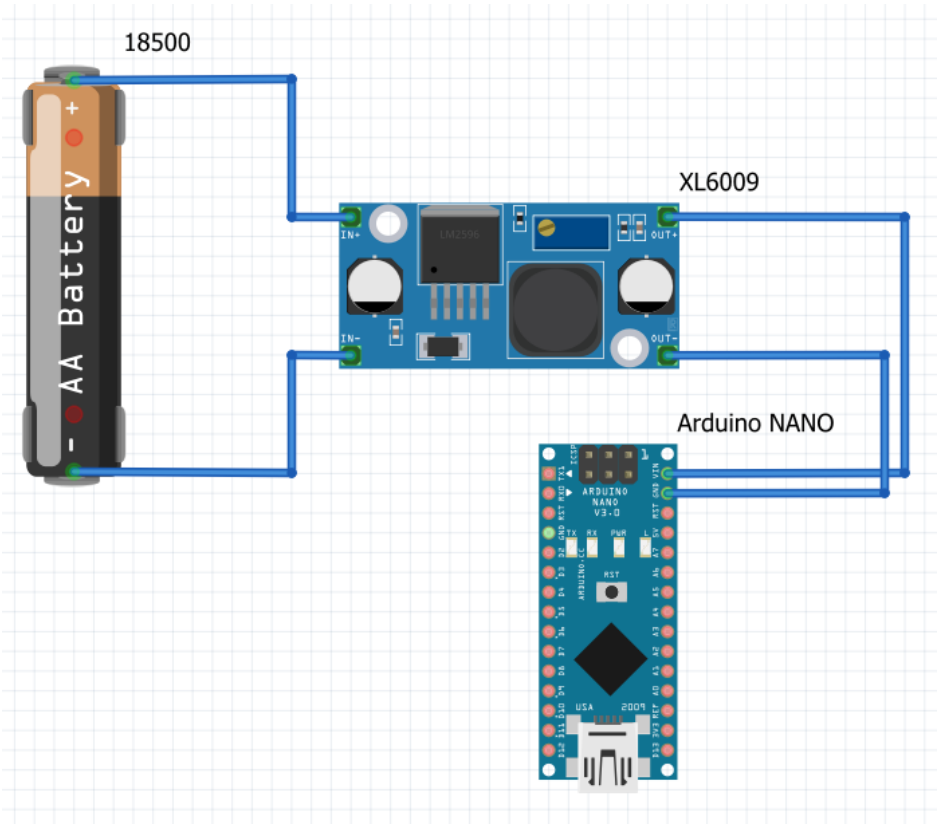

Рисунок 3.1.1 – Схема живлення контролеру

Достатньо під'єднати живлення до контролера, щоб сам контролер розподіляв напругу на інші елементи управління сейфом.

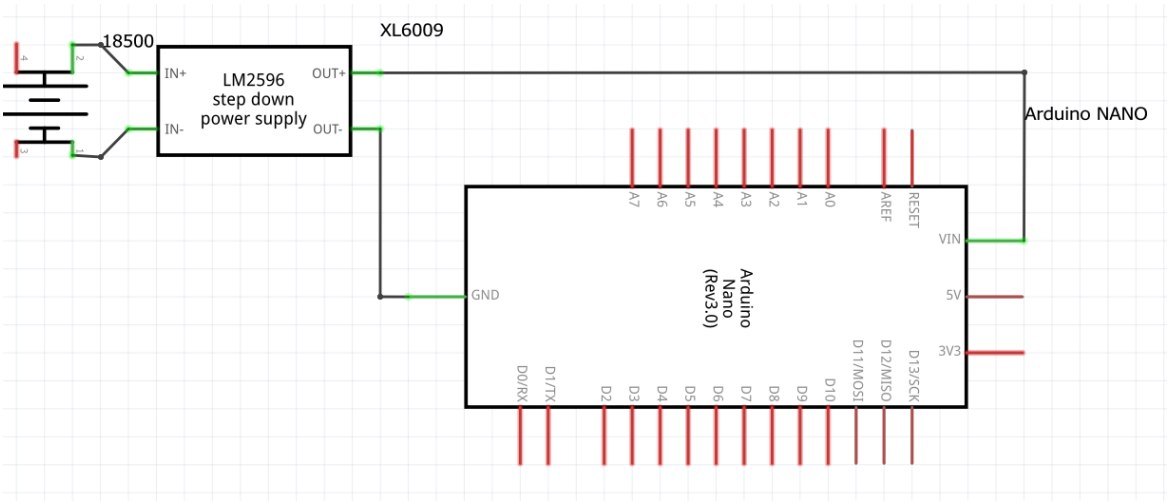

Рисунок 3.1.2 – Принципова схема живлення контролеру

Далі через перетворювач напруги підіймаємо напругу до ~12в, через перемінний резистор. Після підключення, налаштовується вихідна напруга струму через перемінний резистор, який знаходиться на самому перетворювачі.

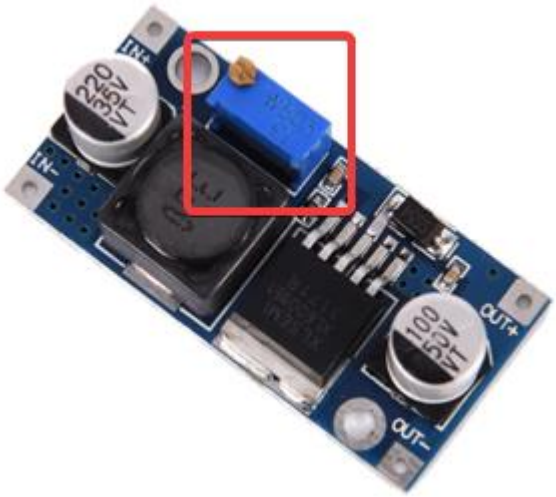

Рисунок 3.1.3 – Перемінний резистор для налаштування вихідної напруги

На входах(IN+/IN-) перетворювача, напруга має значення яке вказано на рисунку 3.1.4.

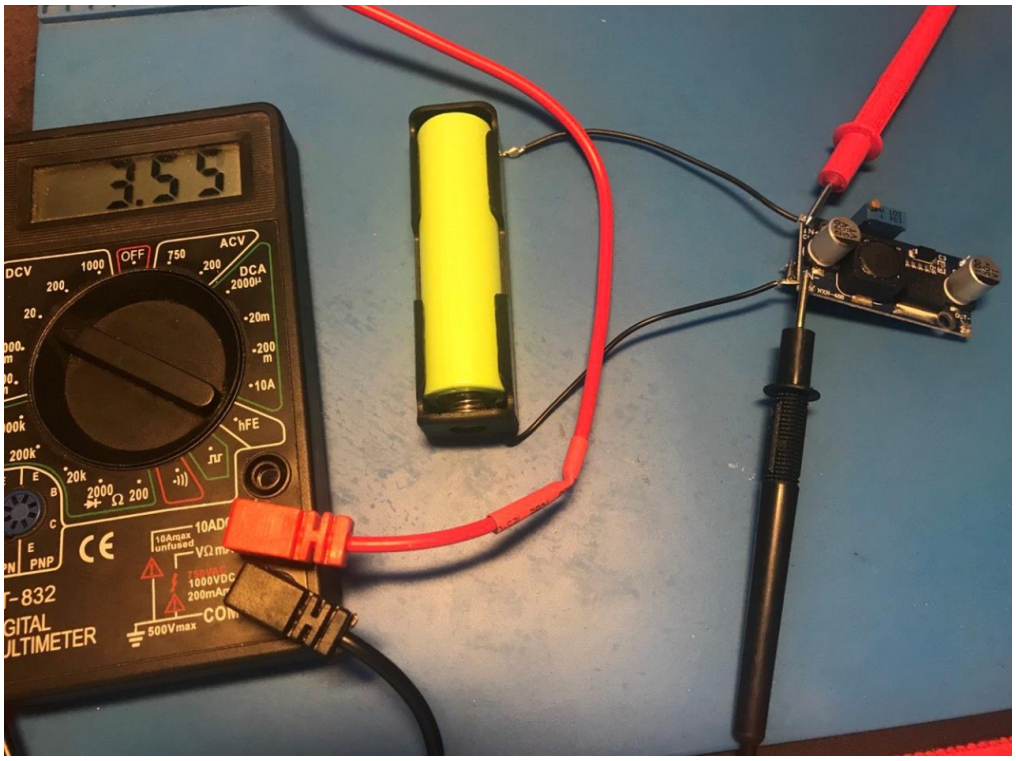

Рисунок 3.1.4 – Вхідна напруга в перетворювачі

Вхідна напруга в перетворювач, це є напруга літій-іонного акумулятора 18500. Зазвичай вона дорівнює ~3.7V. Тобто вихідна напруга акумулятора є правильною.

Після налаштування вихідної напруги як було вказано вище, на виходах(OUT+/OUT-) перетворювача напруга має значення вказане на рисунку 3.1.5.

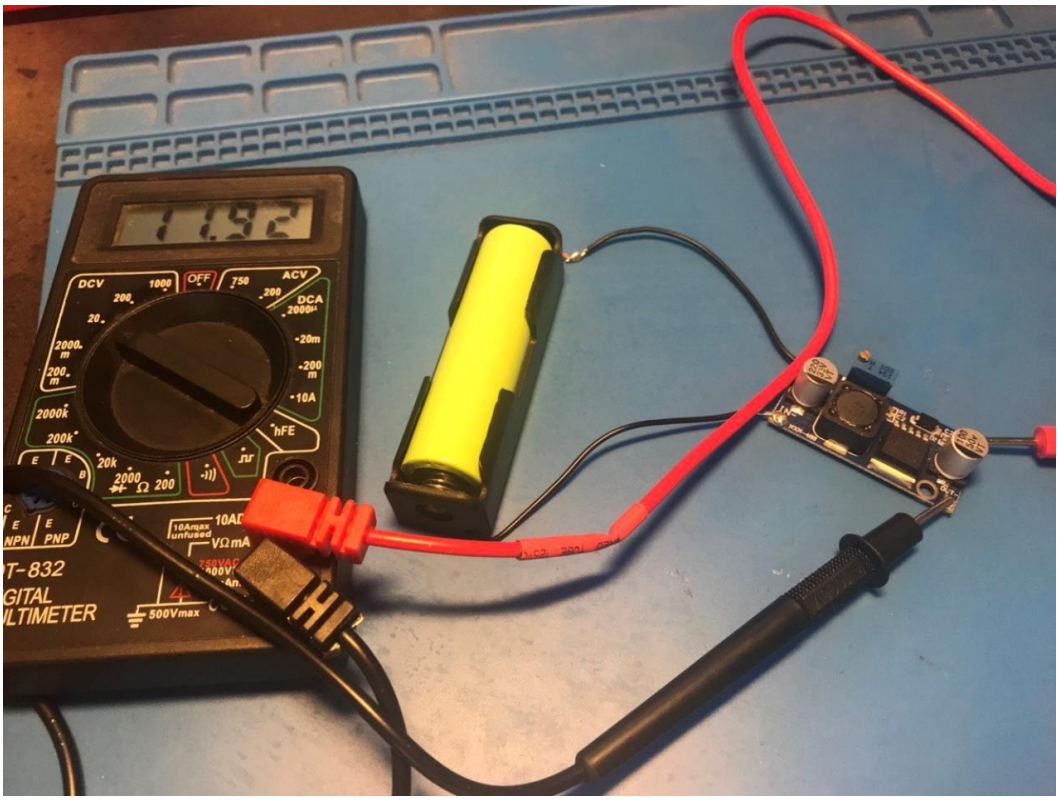

Рисунок 3.1.5 - Вихідна напруга в перетворювачі

Напруга зі значенням 11.92V підходить для живлення контролеру. Літій-іонний акумулятор має властивість видавати повну потужність навіть за мінімального рівня заряду акумулятора, це забезпечує стабільну роботу контролеру.

Далі треба під'єднати матричну клавіатуру та LCD1602 через інтерфейсний модуль.

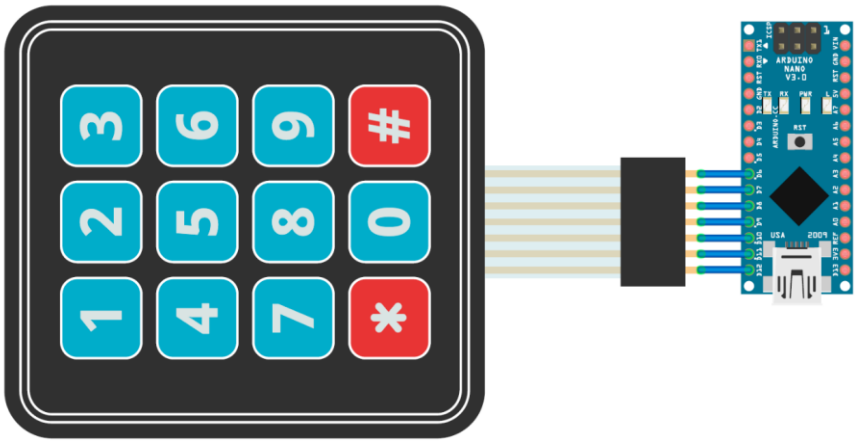

Рисунок 3.1.6 – Схема підключення мембранної клавіатури

Для економії ресурсу автономності сейфа, було створенно систему пробудження сейфа по кнопці яка знажодиться на мембранній клавіатурі.

При натисканні кнопки «*» пароль записується в пам'ять і система виходить із зміни пароля

Рисунок 3.1.7 – Кнопка «*»

При натисканні кнопки «#» скидається пароль (можна вводити заново)

Рисунок 3.1.8 – Кнопка «#»

Якщо нічого не натискати 10 секунд, автоматично вийдемо з режиму зміни пароля, пароль залишиться старий

Коли система не спить, натиснути «*» для входу в режим введення пароля.

Режим введення пароля:

Обробка пароля зроблена таким чином, що правильний пароль зараховується тільки при наборі правильної послідовності цифр, тобто якщо пароль 345, то можна вводити будь-які числа до тих пір, поки не з'явиться послідовність 345, тобто. 0345 відкриє замок, оскільки закінчується на 345.

Якщо пароль введено правильно, двері відчиняться

Якщо нічого не натискати, через 10 секунд система повернеться у звичайний (черговий) режим.

Якщо натиснути #, одразу вийдемо з режиму введення пароля.

Якщо натиснути секретну кнопку зміни пароля в режимі введення пароля, теж з нього вийдемо.

Для зручності і більш зрозумілої роботи с сейфом, додається два світродіода (Червоний та зелений).

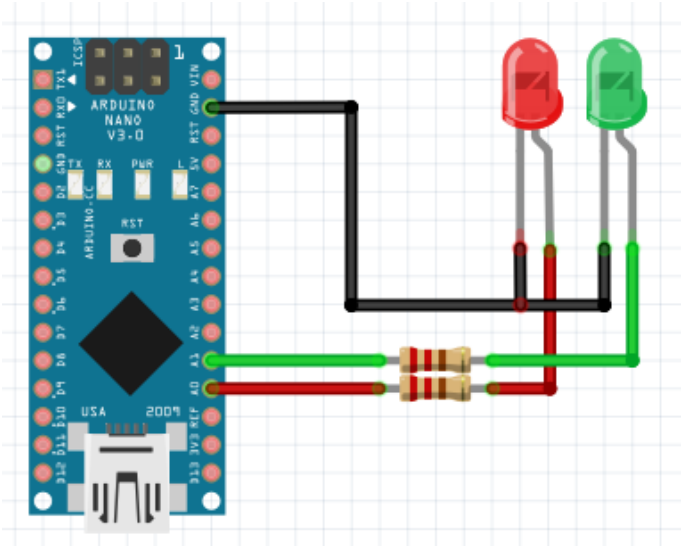

Рисунок 3.1.9 – Схема підключення світлодіодів

Механізм замикання, найнеобхідніша частина сейфа, замикання буде у вигляді сервоприводу.

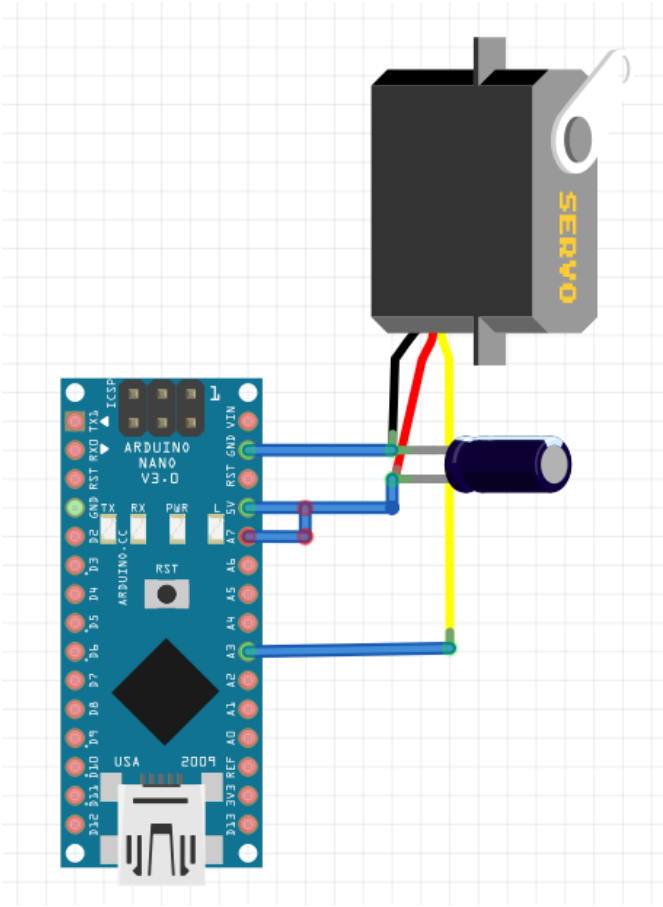

Рисунок 3.1.10 – Схема підключення механізму замикання

При закриванні сейфа потрібно щоб сейф закривався автоматично, було прийнято рішення встановити кнопку у середину. Після того як двірця при закриванні натисне на кнопку, сервопривід стане в положення "Зачинено".

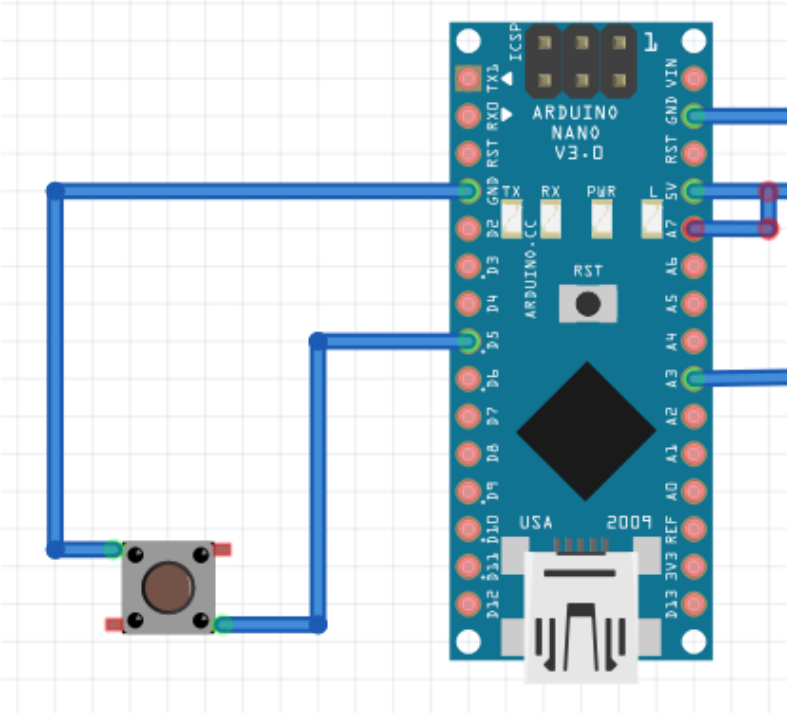

Рисунок 3.1.11 – Схема підключення автоматичного закриття

Секретна кнопка скидання доступу, служить для скидання пароля, введення нового пароля. Буде захована десь у корпусі.

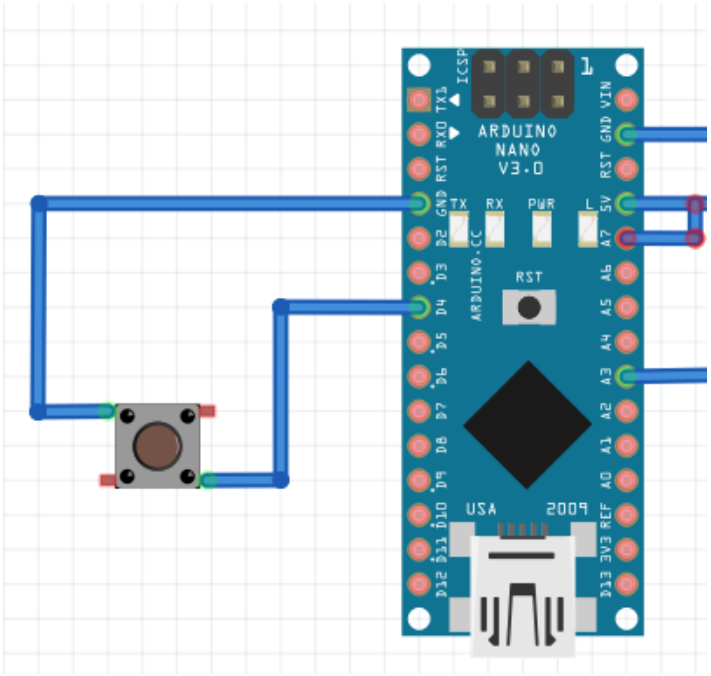

Рисунок 3.1.12 – Кнопка для скидання пароля та введення нового пароля

Оскільки сейф буде повністю автономний. Для енергоощадження буде реалізована система сну, сейф у час сну не буде реагувати поки не пробудиться. Щоб пробудити сейф, буде реалізована зовнішня кнопка.

Кнопка зовні. Служить для закриття дверей, а також для пробудження з енергозбереження. Буде розміщена на корпусі.

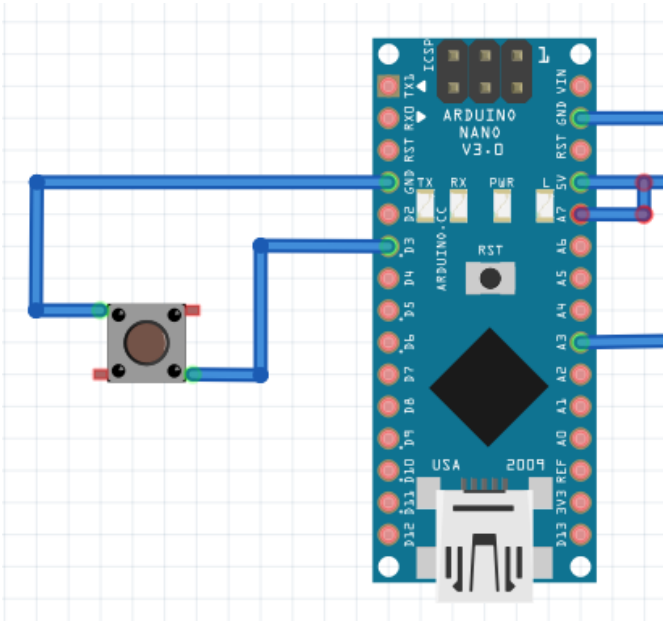

Рисунок 3.1.13 – Зовнішня кнопка

Кнопка всередині. Служить для відкриття та закриття дверей зсередини. Може бути розміщена на ручці дверей (з боку долоні або пальців), на самих дверях.

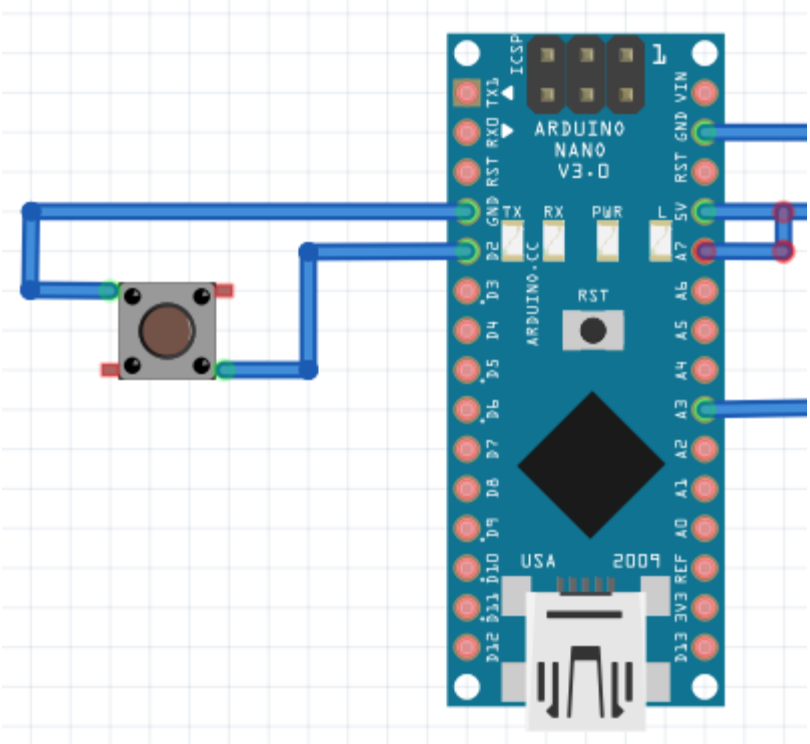

Рисунок 3.1.14 – Кнопка всередині

Рідкокристалічний дисплей 1602 з I2C модулем підключається до плати Arduino всього 4 проводами - 2 проводи даних та 2 проводи живлення.

Під'єднання LCD 1602 до Arduino проводиться стандартно для шини I2C: виведення SDA підключається до порту A4, висновок SCL – до порту A5. Живлення LCD дисплея здійснюється від порту +5V. Дивіться схему підключення LCD 1602 на фото нижче.

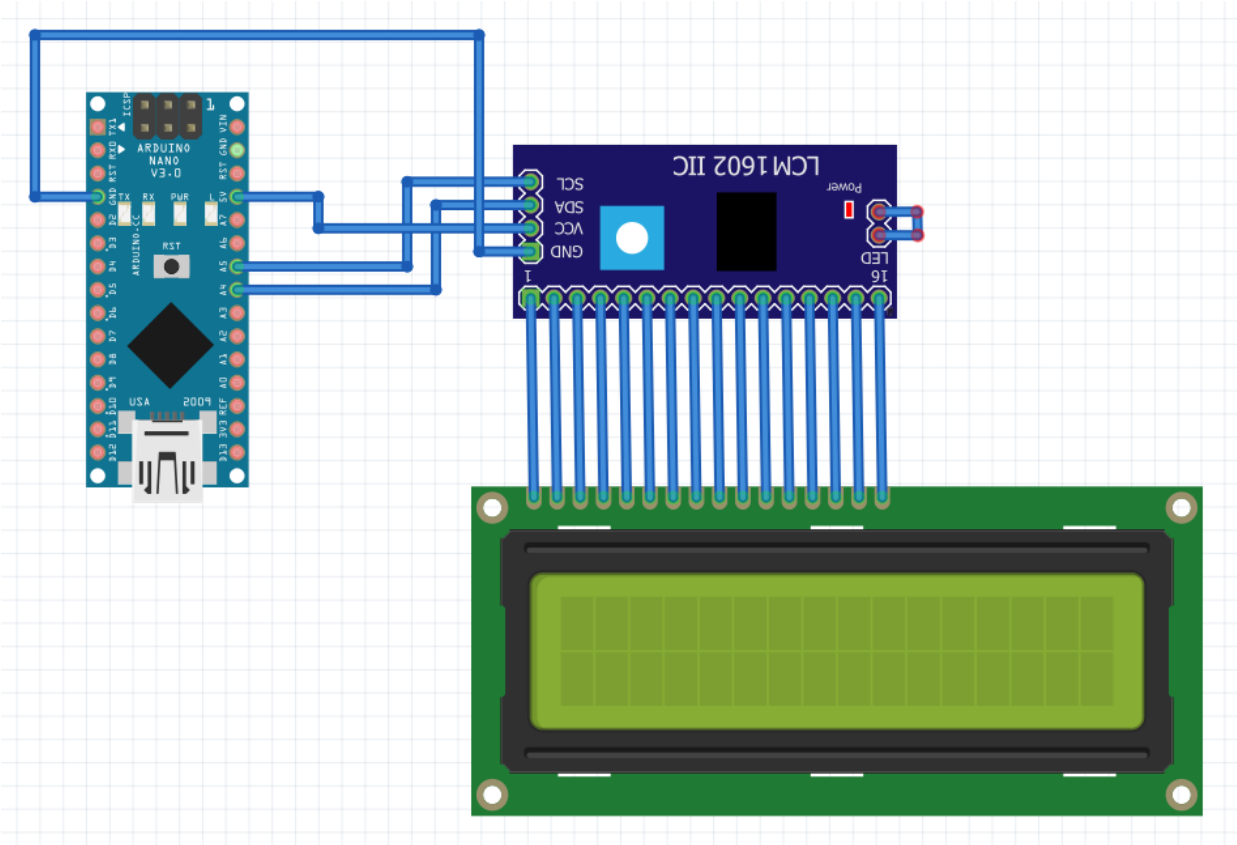

Рисунок 3.1.15 – Схема підключення LCD1602 та I2C

3.2 Тестування проєкту

Оскільки плата Arduino не буде підключена до комп'ютера через USB, необхідно передбачити зовнішнє живлення.

Спираючись на схему, була зібрана модель живлення контролеру.

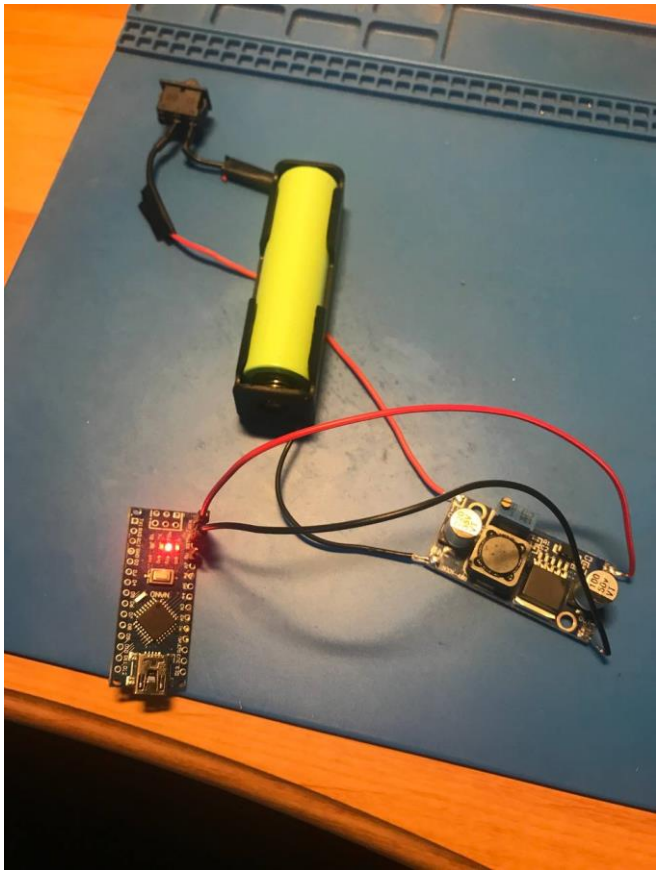

Рисунок 3.2.16 – Зібрана схема живлення контролеру

Після збирання схеми живлення, був зібраний корпус. Та вмонтовано LCD дисплей та мембрану матричну клавіатуру.

Рисунок 3.2.17 – Вмонтований дисплей на клавіатура

Для спілкування LCD дисплею і контролером, використовується інтерфейсний модуль I2C. Після вмонтування в корпус, модуль залишився на звороті дисплея, тобто в середині корпуса.

Рисунок 3.2.18 – Модуль I2C в середині корпуса

Далі була релізована система автозакриття через тактову кнопку, яка показана на рисунку нижче.

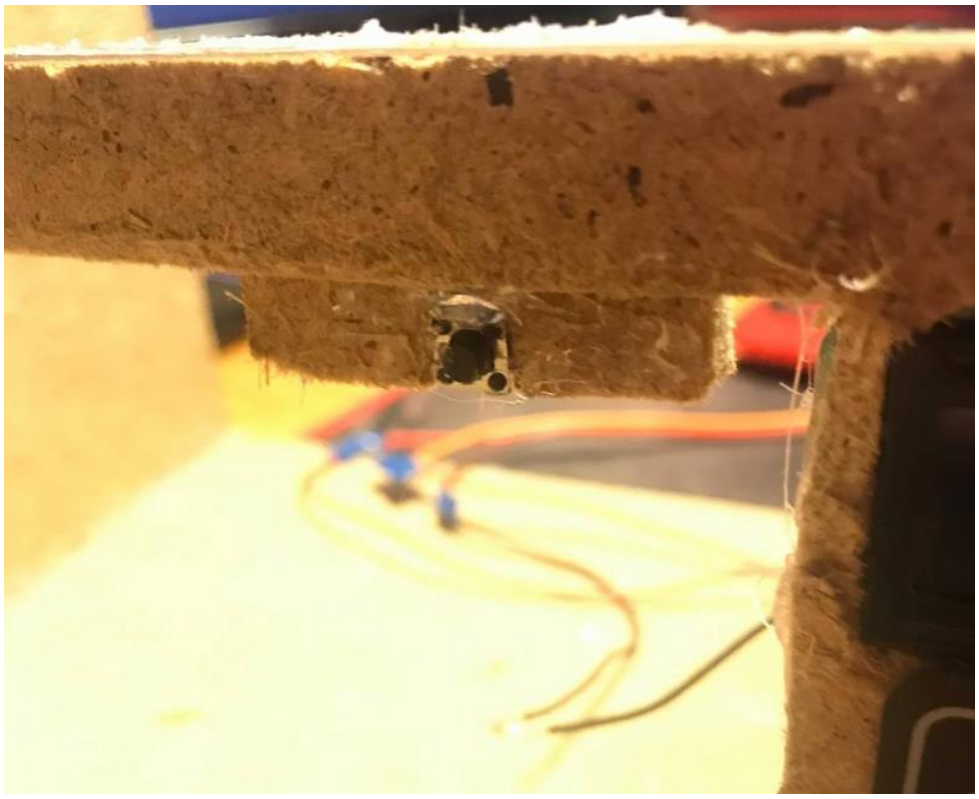

Рисунок 3.2.19 – Вмонтований дисплей на клавіатура

Після підключення всіх компонентів, та налагоджено роботу живлення та інших компонентів, та вмонтування у корпус, проєкт має такий вигляд:

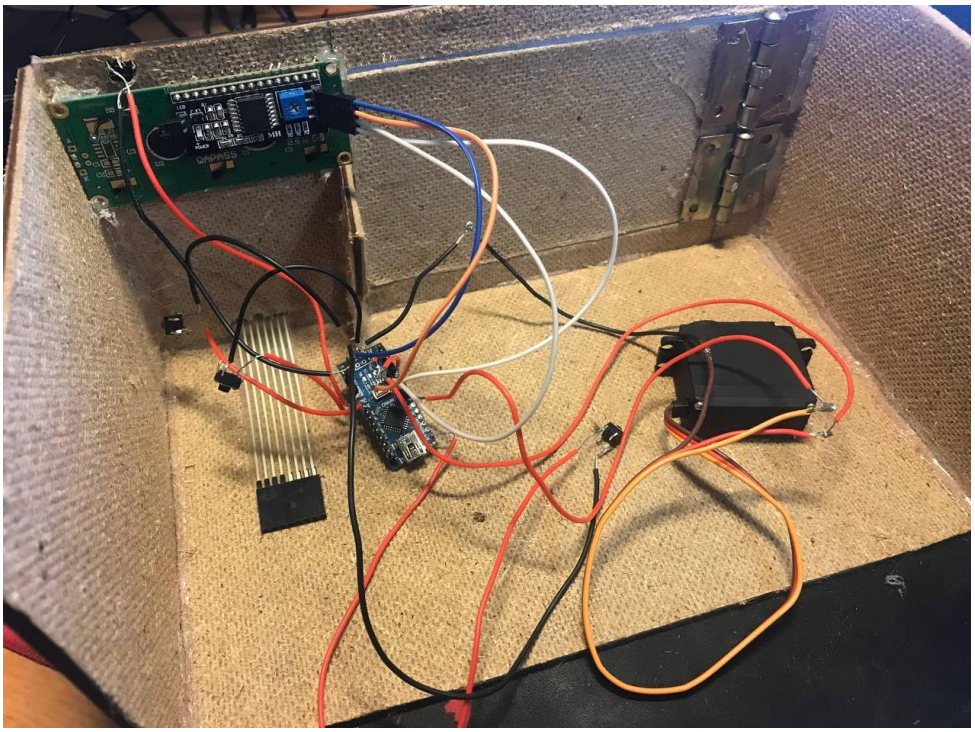

Рисунок 3.2.20 – Зібраний проєкт

У дії він має такий вигляд:

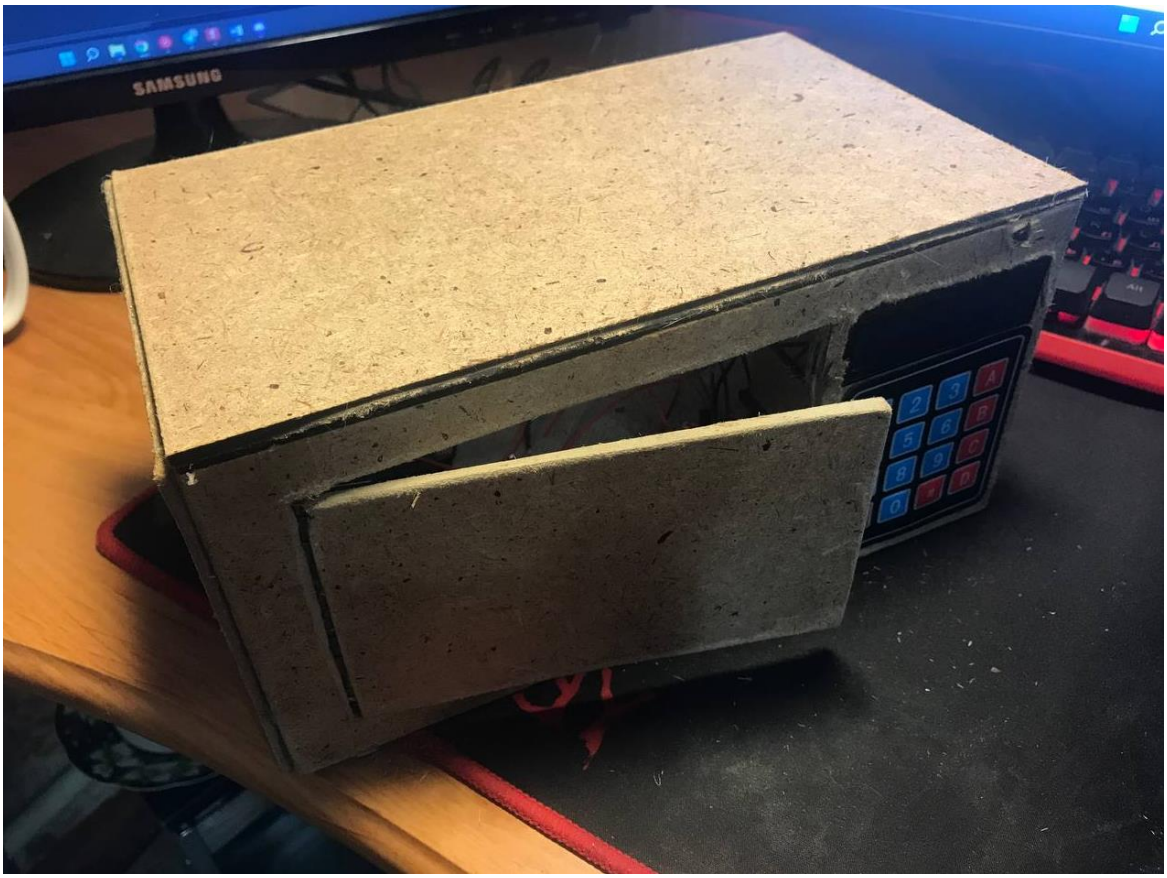

Рисунок 3.2.21 – Зібраний проєкт у дії

3.3 Висновок за розділом

Використовуючи обрані у другому розділі компоненти, був створений електроний сейф на базі контролеру Arduino. Який виконує функції:

1. Захист персональних цінних речей;
2. Запис останнього відкиття сейфа;

ВИСНОВКИ

Здійснивши огляд предметної області створення електронного сейфа на базі контролера Arduino було встановленно. Електронний сейф на базі контролера Arduino чудово підходить для зберігання цінних речей (гокументи, гроші) від

постороннії очей. Для реалізації цього проєкту були обрані та використані такі компоненти:

8. Контролер Arduino NANO;
9. Підвищуючий перетворювач XL6009;
10. Літій-іонний акумулятор 18500;
11. сервопривід MG995 Tower Pro 360;
12. Мембранна матрична клавіатура;
13. LCD 1602;
14. I2C інтерфейсний модуль.

Також були підвищенні навички програмування та моделювання проєктів на базі контролеру. Для реалізації програмної частини, була використана утиліта Aerduino IDE з використання мови програмування C++ (спрощена).

ПЕЛЕЛІК ПОСИЛАНЬ

Характеристика джерела	Режим доступу
Електронний ресурс	Поняття та основні характеристики сейфу. – Режим доступу www. URL:

	https://wikipedia.org/wiki/%D0%A1%D0%B5%D0%B9%D1%84
Електронний ресурс	Класи захисту сейфу. – Режим доступу www. URL: https://locker.com.ua/r_information-news-news/news_id=70#:~:text=%D0%98%D1%82%D0%B0%D0%BA%2C%20%D1%81%D0%B0%D0%BC%D1%8B%D0%B9%20%D0%BF%D1%80%D0%BE%D1%81%D1%82%D0%BE%D0%B9%20%D0%BA%D0%BB%D0%B0%D1%81%D1%81%20%D0%B7%D0%B0%D1%89%D0%B8%D1%82%D1%8B,%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B0%20%D0%BF%D1%80%D0%B5%D0%B8%D0%BC%D1%83%D1%89%D0%B5%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D0%BE%20%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D1%83%D1%8E%D1%82%D1%81%D1%8F%20%D0%B2%20%D0%B1%D1%8B%D1%82%D1%83
Стандарти, техніко-економічні та технічні документи	Типи вмонтування сейфу. – Режим доступу www. URL: https://dnaop.com/html/67301/doc-%D0%94%D0%A1%D0%A2%D0%A33892-99
Електронний ресурс	Опис Arduino NANO. – Режим доступу www. URL: https://arduino.ua/prod166-arduino-nano-v3-0-avr-atmega328p-s-raspayannimi-razemami
Електронні ресурси віддаленого доступу	Arduino Nano V3 Manual. – Режим доступу www. URL: https://www.arduino.cc/en/uploads/Main/ArduinoNanoManual23.pdf
Електронний ресурс	Опис та характеристика пінів Arduino NANO. – Режим доступу www. URL: https://docs.arduino.cc/hardware/nano
Електронний ресурс	Офіційний сайт Arduino. – Режим доступу www. URL: https://www.arduino.cc/
Електронний ресурс	Сайт Arduino IDE. – Режим доступу www. URL: https://downloads.arduino.cc/arduino-1.8.19-windows.exe?_gl=1*11f8z1*_ga*NjIwMDAwMDUzLjE2NTUyNDM1NjM.*_ga_NEXN8H46L5*MTY1NTI0MzU2My4xLjEuMTY1NTI0NDQzMC41Nw
Електронний ресурс	Основні поняття та характеристик про LCD1602. – Режим доступу www. URL: https://www.robostore.com.ua/moduli-i-datchiki/monohromnye-displei/lcd-displej-1602-i2c-s-sinej-podsvetkoj/
Електронний ресурс	Характеристика та способи використання . – Режим доступу www. URL: http://easyelectronics.ru/interface-bus-iic-i2c.html

ДОДАТКИ

Додаток А. Вихідний код програми для Arduino

```
#define lock_type
#define tail_button 1
```

```
#define sleep_time 10000

#define sleep_enable 0
#define wake_button 0
boolean battery_monitor = 1;
#define bat_low 3000  ;
boolean open_bat_low = 1;
#define vol_calibration 0

#define servoMin 40
#define servoMax 170

#define gear_inv 0

#if lock_type == 1
#define close_button 0
#else
#define close_button 1
#endif

#if lock_type == 0
#include <Servo.h>
#define servo_pin A3
Servo servo;
#elif lock_type == 1
#define latch_pin A3
#elif lock_type == 2
#define relay1 A2
#define relay2 A3
```

```
#define gear_delay 400
#endif

#include <EEPROMex.h>
#include <LowPower.h>

#define LEDgrn A0
#define LEDred A1
#define set_pass_btn 4
#define tail_pin 5
float my_vcc_const = 1.1;

boolean set_pass_btn_flag;
boolean batteryOK = true;
boolean set_access_flag;
volatile unsigned long awake_timer, auto_awake_timer,
last_press;
volatile boolean inside_btn_flag, close_flag;
volatile byte sleep_count;
boolean wake_event = false, wait_for_event = true,
wakeUP_procedure_flag = false;

void setup() {
  Serial.begin(9600);
  if (vol_calibration) calibration();
  my_vcc_const = EEPROM.readFloat(1000);

  pinMode(2, INPUT_PULLUP);
  pinMode(3, INPUT_PULLUP);
```

```
  pinMode(set_pass_btn, INPUT_PULLUP);
  pinMode(tail_pin, INPUT_PULLUP);

  pinMode(LEDred, OUTPUT);
  pinMode(LEDgrn, OUTPUT);
  digitalWrite(LEDred, 0);
  digitalWrite(LEDgrn, 0);

  attachInterrupt(0, inside_btn, FALLING);

  attachInterrupt(1, outside_btn, FALLING);

#elif lock_type == 2
  pinMode(relay1, OUTPUT);
  pinMode(relay2, OUTPUT);
  digitalWrite(relay1, 1);
  digitalWrite(relay2, 1);
#endif
}

void open_door() {
  if (batteryOK) {
    digitalWrite(LEDgrn, 1);
    digitalWrite(LEDred, 0);
    last_press = millis();

#if lock_type == 0
    servo.attach(servo_pin);
    servo.write(servoMin);
```

```
    delay(500);
    servo.detach();
#elif lock_type == 1
    digitalWrite(latch_pin, !latch_inverse);
    delay(latch_time);
    digitalWrite(latch_pin, latch_inverse);
#elif lock_type == 2
    if (gear_inv) {
      digitalWrite(relay1, 0);
      delay(gear_delay);
      digitalWrite(relay1, 1);
    } else {
      digitalWrite(relay2, 0);
      delay(gear_delay);
      digitalWrite(relay2, 1);
    }
#endif
    Serial.println("Open door");
    if (!close_button) {
      door_state = 0;
      digitalWrite(LEDred, 0);
      digitalWrite(LEDgrn, 0);
    }
    else door_state = 1;
  } else {
    for (int i = 0; i < 3; i++) {
      digitalWrite(LEDred, 1);
      delay(500);
      digitalWrite(LEDred, 0);
```

```
      delay(500);
    }
  }
  if (sleep_enable) sleep_mode();
}

void close_door() {
  if (batteryOK) {
    digitalWrite(LEDred, 1);
    last_press = millis();
#if lock_type == 0
    servo.attach(servo_pin);
    servo.write(servoMax);
    delay(500);
    servo.detach();
#elif lock_type == 1
#elif lock_type == 2
    if (gear_inv) {
      digitalWrite(relay2, 0);
      delay(gear_delay);
      digitalWrite(relay2, 1);
    } else {
      digitalWrite(relay1, 0);
      delay(gear_delay);
      digitalWrite(relay1, 1);
    }
#endif
    Serial.println("Close door");
    door_state = 0;
```

```
    delay(500);
    digitalWrite(LEDred, 0);
    digitalWrite(LEDgrn, 0);
  } else {
    for (int i = 0; i < 3; i++) {
      digitalWrite(LEDred, 1);
      delay(500);
      digitalWrite(LEDred, 0);
      delay(500);
    }
  }
  if (sleep_enable) sleep_mode();
}

void wakeUP_procedure() {

}
void sleep_procedure() {

}

void loop() {
  if (wait_for_event && !wake_button) {
  }
  if (wakeUP_procedure_flag) {
    wakeUP_procedure();
    wakeUP_procedure_flag = false;
  }
```

```
  if (set_access_flag) {
    Serial.println("Pass change");
    set_access_flag = 0;
  }

  if (close_flag) {
    close_flag = 0;
    close_door();     // закрыть дверь
  }

  if (inside_btn_flag && !door_state) {
    inside_btn_flag = 0;
    open_door(); // команда для открытия двери
  }

  if (!digitalRead(set_pass_btn) && !set_pass_btn_flag) {
    awake_timer = millis();
    set_pass_btn_flag = 1;
  }
  if (digitalRead(set_pass_btn) && set_pass_btn_flag) {
    set_pass_btn_flag = 0;
    set_access_flag = 1;
  }
  if (millis() - awake_timer > sleep_time &&
!set_access_flag) {
    if (!door_state) {                digitalWrite(LEDred,
0);
      digitalWrite(LEDgrn, 0);
    }
```

```
    if (sleep_enable) {
      Serial.println("Sleep (idle)");
      sleep_mode();
    }
  }
  if (!digitalRead(tail_pin) && door_state) {
    Serial.println("Tail!");
    close_door();
  }
}

void inside_btn() {
  if (millis() - last_press > 500) {
    auto_awake_timer = millis();
    if (!door_state)
      inside_btn_flag = 1;
    else
      close_flag = 1;
    sleep_count = 0;
  }
}

// отработка прерывания нажатия снаружи
void outside_btn() {
  if (millis() - last_press > 500) {   // таймер
повторного открытия (побеждает глюк с приводом!)
    auto_awake_timer = millis();
    if (door_state) {              // если дверь ОТКРЫТА
      close_flag = 1;
```

```
    } else {
      Serial.println("Wake up");
      digitalWrite(LEDred, 1);
      digitalWrite(LEDgrn, 1);
      wakeUP_procedure_flag = true;
      awake_timer = millis();
    }
    sleep_count = 0;
  }
}
void sleep_mode() {
  sleep_procedure();
  if (!batteryOK) digitalWrite(LEDgrn, 0);
  Serial.println("Sleep");
  delay(50);
  if (tail_button && door_state && close_button) {
    LowPower.powerDown(SLEEP_500MS, ADC_OFF, BOD_OFF);
    awake_timer = millis() + sleep_time;
    if (!digitalRead(tail_pin)) {
      Serial.println("Tail!");
      close_door();
    }
  } else {
    if (sleep_enable && wake_button && !battery_monitor)
      LowPower.powerDown(SLEEP_FOREVER, ADC_OFF, BOD_OFF);
    if (sleep_enable && wake_button && battery_monitor) {
      LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
      awake_timer = millis() + sleep_time;
      battery_m();
```

```
    }
    if (sleep_enable && !wake_button) {
      LowPower.powerDown(SLEEP_4S, ADC_OFF, BOD_OFF);
      awake_timer = millis() + sleep_time;
      event_m();
      if (battery_monitor) battery_m();
    }
  }
}
void battery_m() {
  sleep_count++;
  if (sleep_count > 75) {
    sleep_count = 0;
    if (readVcc() < bat_low) {
      Serial.println("Low battery");
      wait_for_event = false;
      battery_monitor = false;
      if (open_bat_low) open_door();
      batteryOK = false;
    }
  }
}
void event_m() {
  Serial.println("Event awake");
  if (wake_event) {
    Serial.println("Wake up");
    digitalWrite(LEDred, 1);
    digitalWrite(LEDgrn, 1);
    wake_event = false;
```

```
    wakeUP_procedure_flag = true;
    awake_timer = millis();
    sleep_count = 0;
  }
}

void calibration() {
  my_vcc_const = 1.1;
  Serial.print("Real VCC is: ");
Serial.println(readVcc());
  Serial.println("Write your VCC (in millivolts)");
  while (Serial.available() == 0); int Vcc =
Serial.parseInt();
  float real_const = (float)1.1 * Vcc / readVcc();
  Serial.print("New voltage constant: ");
Serial.println(real_const, 3);
  EEPROM.writeFloat(1000, real_const);
  while (1);
}

long readVcc() {
#if defined(__AVR_ATmega32U4__) ||
defined(__AVR_ATmega1280__) || defined(__AVR_ATmega2560__)
  ADMUX = _BV(REFS0) | _BV(MUX4) | _BV(MUX3) | _BV(MUX2) |
_BV(MUX1);
#elif defined (__AVR_ATtiny24__) ||
defined(__AVR_ATtiny44__) || defined(__AVR_ATtiny84__)
  ADMUX = _BV(MUX5) | _BV(MUX0);
#elif defined (__AVR_ATtiny25__) ||
```

```
defined(__AVR_ATtiny45__) || defined(__AVR_ATtiny85__)
  ADMUX = _BV(MUX3) | _BV(MUX2);
#else
  ADMUX = _BV(REFS0) | _BV(MUX3) | _BV(MUX2) | _BV(MUX1);
#endif
  delay(2); // Wait for Vref to settle
  ADCSRA |= _BV(ADSC); // Start conversion
  while (bit_is_set(ADCSRA, ADSC)); // measuring
  uint8_t low  = ADCL; // must read ADCL first - it then
locks ADCH
  uint8_t high = ADCH; // unlocks both
  long result = (high << 8) | low;
  result = my_vcc_const * 1023 * 1000 / result;
}
```