

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ
ТА ЗАСОБІВ ПРОТОТИПУВАННЯ ТЕХНОЛОГІЧНОЇ СИСТЕМИ
RESEARCH AND SOFTWARE IMPLEMENTATION OF METHODS
AND TOOLS FOR TECHNOLOGICAL SYSTEM PROTOTYPING

Виконав: студент 2 курсу, групи КНТ-132м
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ПОЛЯНСЬКИЙ Е. Д.

(ПРИЗВИЩЕ та ініціали)

Керівник ПАРХОМЕНКО А. В.

(ПРИЗВИЩЕ та ініціали)

Рецензент ЗЕЛЕНЬОВА І. Я.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук та технологій
Кафедра програмних засобів
Ступінь вищої освіти магістр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
« » 2023 року

ЗАВДАННЯ

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА

ПОЛЯНСЬКОГО Едуарда Дмитровича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи): Дослідження та програмна реалізація методів та засобів прототипування технологічної системи. Research and Software Implementation of Methods and Tools for Technological System Prototyping
керівник проєкту (роботи): к.т.н., доц. ПАРХОМЕНКО Анжеліка Володимирівна

(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від «14» листопада 2023 р. № 437

2. Строк подання студентом проєкту (роботи): 1 грудня 2023 року

3. Вихідні дані до проєкту (роботи): завдання, література

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити): 1. Аналіз предметної області. 2. Дослідження сучасних технологій цифрової індустрії. 3. Проєктування програмного забезпечення. 4. Конструювання програмного забезпечення.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень):
Слайди презентації

6. Консультанти розділів проєкту (роботи):

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ПАРХОМЕНКО А. В., доцент		
Нормоконтроль	БЄЛОВА А. В., асистент		

7. Дата видачі завдання «08» вересня 2023 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання
2	Аналіз предметної області	2-3 тиждень	Розділ 1
3	Дослідження методів вирішення проблеми	4-5 тиждень	Розділ 2
4	Проєктування програмного забезпечення	6 тиждень	Розділ 3
5	Конструювання програмного забезпечення	7-9 тижні	Розділ 4
6	Тестування та експериментальне дослідження програмного забезпечення.	10 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	11 тиждень	ПЗ, Додатки
8	Захист роботи	12 тиждень	

Студент(ка)

_____ Едуард ПОЛЯНСЬКИЙ
(підпис) (Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

_____ Анжеліка ПАРХОМЕНКО
(підпис) (Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
104 с., 8 табл., 53 рис., 2 дод., 44 джерела.

ТЕХНОЛОГІЧНА СИСТЕМА ДЛЯ ЗГИНАННЯ ДРОТУ, ПРОТОТИП,
ЦИФРОВИЙ ДВІЙНИК, ВІРТУАЛЬНА ТА ДОПОВНЕНА РЕАЛЬНІСТЬ,
АЛГОРИТМ УПРАВЛІННЯ, ARDUINO, ГРАФІЧНИЙ ІНТЕРФЕЙС.

Об'єкт дослідження – процес прототипування технологічної системи для згинання дроту.

Предмет дослідження – методи та засоби цифрової індустрії для прототипування технологічної системи для згинання дроту (ТСЗД).

Мета роботи – дослідження технологій цифрової індустрії для прототипування технологічної системи та розробки програмного забезпечення для числового програмного управління.

Матеріали, методи та технічні засоби: плата Arduino Nano, плата розширення CNC шилд V 4, крокові двигуни NEMA-17-01, модулі драйвера крокового двигуна DRV8825, CAD система PTC Creo, середовища програмування Arduino IDE та Processing.

Результати. Розроблено методику прототипування на основі цифрового двійника ТСЗД з використанням технологій віртуальної, доповненої реальності та адитивного виробництва, а також програмне забезпечення для числового програмного управління прототипом ТСЗД на основі взаємодії Arduino та Processing.

Висновки. Розроблене програмне забезпечення для керування прототипом ТСЗД забезпечує інтуїтивно-зрозумілі функції управління та дозволяє підвищити ефективність технологічного процесу.

Галузь використання – прототипування ТСЗД для аерокосмічної галузі, автомобіле-, судно-, машинобудування та ін.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 104 pages, 8 tables, 53 figures, 2 appendixes, 44 sources.

TECHNOLOGICAL SYSTEM FOR WIRE BENDING, PROTOTYPE, DIGITAL TWIN, VIRTUAL AND AUGMENTED REALITY, CONTROL ALGORITHM, ARDUINO, GRAPHIC INTERFACE.

The object of studying is the process of prototyping a technological system for wire bending.

The subject of research are methods and tools of digital industry for prototyping a technological system for wire bending (TSWB).

The purpose of the work is research of digital industry technology for technological system prototyping and development of computer numerical control software.

Materials, methods and technical tools: Arduino Nano board, CNC shield V 4, stepper motors NEMA-17-01, stepper motor driver modules DRV8825, CAD system PTC Creo, programming environment Arduino IDE and Processing.

Results. A prototyping technique was developed based on a digital twin of the TSWB using virtual, augmented reality and additive manufacturing technologies, as well as software for computer numerical control of the TSWB prototype based on the interaction of Arduino and Processing.

Conclusions. The developed software for controlling the TSWB prototype provides intuitive and understandable control functions and allows to increase the efficiency of the technological process.

The area of usage is prototyping of the TSWB for the aerospace industry, automotive, shipbuilding, mechanical engineering, etc.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок.....	8
Вступ.....	9
1 Аналіз предметної області.....	10
1.1 Огляд особливостей технологічної системи для згинання дроту....	10
1.2 Дослідження існуючих прототипів ТСЗД.....	12
1.3 Аналіз структурних та функціональних особливостей досліджуваного прототипу ТСЗД.....	15
1.4 Математична модель для розрахунку параметрів механізмів прототипу.....	25
1.5 Постановка завдань магістерської роботи.....	30
2 Дослідження сучасних технологій цифрової індустрії.....	31
2.1 Аналіз концепції Індустрії 4.0.....	31
2.2 Цифрові двійники.....	35
2.3 Розробка методики прототипування на основі ЦД.....	39
2.4 Результат розробки віртуальної частини ЦД.....	45
2.5 Результат розробки фізичної частини ЦД.....	48
2.6 Висновки до розділу 2.....	50
3 Проєктування програмного забезпечення для керування прототипом ТСЗД.....	51
3.1 Призначення та вимоги до програмного забезпечення.....	51
3.2 Розробка архітектури системи.....	52
3.3 Вибір інструментарію програмної реалізації.....	54
3.4 Висновки до розділу 3.....	55
4 Конструювання програмного забезпечення для керування прототипом ТСЗД.....	56
4.1 Розробка програмного забезпечення для Arduino.....	56

4.2 Розробка програмного інтерфейсу.....	63
4.3 Інструкція з використання програмного забезпечення.....	68
4.4 Висновки до розділу 4.....	75
Висновки.....	76
Перелік джерел посилань.....	77
Додаток А Текст програми.....	81
Додаток Б Слайди презентації.....	97

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

AR	- Augmented Reality;
CAD	- Computer Aided Design;
CAE	- Computer Aided Engineering;
CAM	- Computer Aided Manufacturing;
CNC	- Computer Numerical Control;
EEPROM	- Electrically Erasable Programmable Read-Only Memory;
FEA	- Finite Element Analysis;
GUI	- Graphical User Interface;
HMI	- Human Machine Interface;
IDE	- Integrated Development Environment;
IoD	- Internet of (very Big) Data;
IoS	- Internet of Services;
IoT	- Internet of Things;
SRAM	- Static Random Access Memory;
USB	- Universal Serial Bus;
VR	- Virtual Reality;
AB	- адитивне виробництво;
ЖЦ	- життєвий цикл;
ККД	- коефіцієнт корисної дії;
ПЗ	- програмне забезпечення;
ПК	- персональний комп'ютер;
ТСЗД	- технологічна система для згинання дроту;
ЦД	- цифровий двійник;
ЧПУ	- числове програмне управління;
ШІМ	- широтно-імпульсна модуляція.

ВСТУП

Інструменти швидкого прототипування дозволяють дизайнерам та інженерам швидко створювати функціональні моделі або продукти. За останні десятиліття ринок цих інструментів надзвичайно розвинувся [1].

Створення прототипу є важливим кроком життєвого циклу (ЖЦ) продукту. Окрім виявлення проблем із поточним дизайном, прототипи надають інвесторам і потенційним клієнтам фізичний доказ здійсненності концепції. Після усвідомлення важливості створення прототипів для будь-якої компанії-початківця чи підприємця стає життєво важливим мати надійні, точні та універсальні інструменти швидкого прототипування. Впровадження таких інструментів скорочує загальний графік проєкту, мінімізує накладні витрати, забезпечує більшу кількість власних розробок і загалом дозволяє компаніям виводити продукти на ринок швидше та з більшою впевненістю у своїх проєктах [2].

Нові адитивні технології, такі як 3D-друк, а також інструменти субтрактивного виробництва, такі як лазерні різакі, фрезерні та токарні верстати з ЧПУ, дозволяють швидко й точно виготовляти багато різних прототипів компонентів. Ці технології надають можливість дизайнерам реалізувати свої ідеї набагато швидше, ніж було можливо раніше. Незважаючи на свої переваги, технології швидкого прототипування обмежені максимальними розмірами деталей, які можуть бути створені та інколи їх високою вартістю [3]-[5].

Згинання дроту є ще одним новим методом виробництва, який розширив можливості дизайнерів. Але, дослідження ринку показало, що апаратне та програмне забезпечення для технологічних систем згинання дроту або дуже дороге, або має певні обмеження. Тому, подальші дослідження в цій галузі є актуальними.

Метою роботи є дослідження технологій цифрової індустрії для прототипування технологічної системи та розробки програмного забезпечення для числового програмного управління.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд особливостей технологічної системи для згинання дроту

Технології згинання дроту широко використовуються в аерокосмічній галузі, автомобіле-, судно-, машинобудуванні та інших важливих сферах [6].

Існуючі технологічні системи для згинання дроту (ТСЗД) призначені для згинання дроту різного діаметру, виготовленого з широкого діапазону сплавів [7].

У загальному випадку ТСЗД з числовим програмним управлінням (ЧПУ) включає механізм накопичення, механізм вирівнювання, механізм подачі, механізм згинання дроту (рис.1.1) [6].

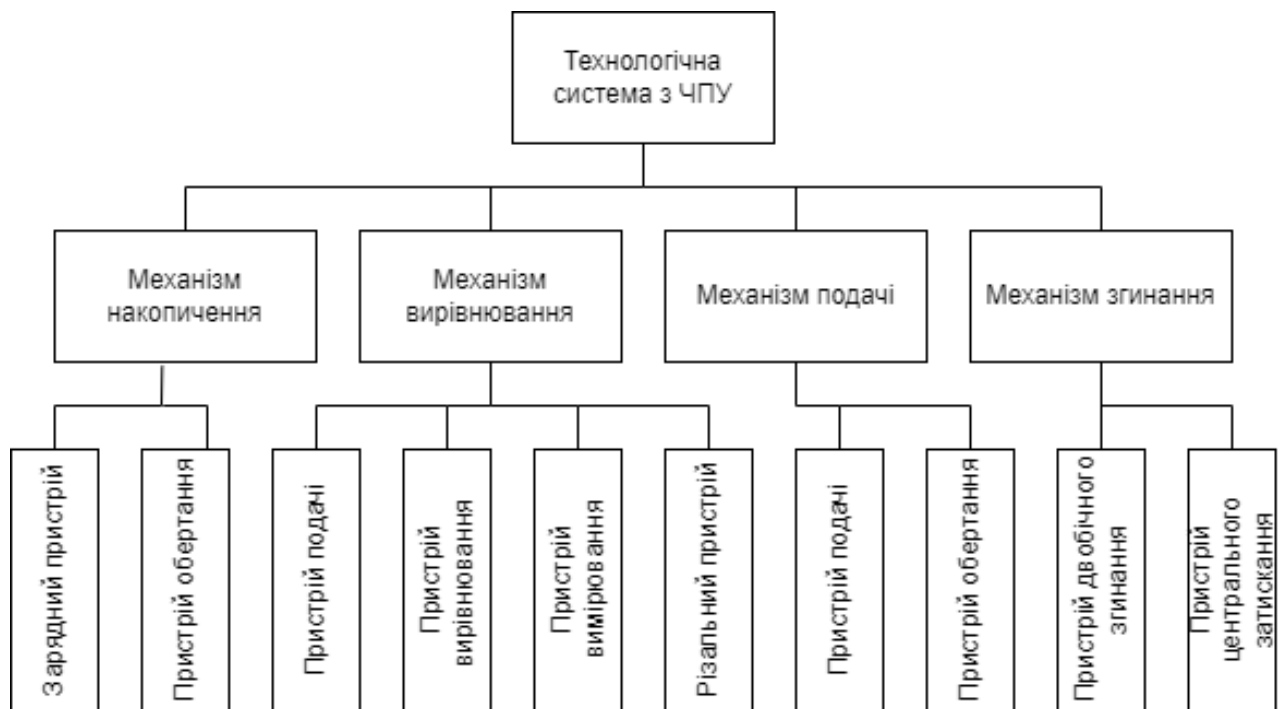


Рисунок 1.1 – Структурна схема ТСЗД з ЧПУ [6]

На великих виробництвах використовують автоматичні верстати для згинання дроту та прутків різного діаметру. Головна частина найпростішого верстата – диск, що обертається, на якому встановлені центральний і згинальний "пальці" (рис.1.2). За допомогою ЧПУ верстата можна задати вигин дроту у трьох вимірах, створюючи об'ємні фігури різної складності [7].



Рисунок 1.2 – Зовнішній вигляд верстата з ЧПУ для згинання дроту [7]

Автоматичні верстати для згинання дроту зазвичай є повністю автоматичними технологічними системами, що виготовляють деталі з високою точністю, швидкістю, повторюваністю та призначені для цілодобової роботи. Ці верстати оснащені трьома сервоприводами, що виконують подачу дроту, поворот дроту у просторі та його згинання, забезпечують високу продуктивність за максимально високої точності згинання та відсутність шумів. Для зручності керування, верстати оснащені пультом керування, що дозволяє керувати верстатом на відстані або безпосередньо поблизу необхідної частини машини [7].

Характеристики ЧПУ:

- промисловий комп'ютер зі спеціалізованим програмним забезпеченням;
- простий і зручний інтерфейс, інформація, що виводиться, відображається на кольоровому сенсорному екрані, що полегшує керування верстатом;
- програми згинання дроту зберігаються на жорсткому диску комп'ютера, а також передбачена можливість запису програм на USB-носії;

- дозволяє створити необмежену кількість конфігурацій з необмеженою кількістю кроків у програмі;
- контролює швидкості всіх трьох приводів (швидкості подачі, обертання та згинання);
- вбудована самодіагностика із виведенням помилок на екран;
- можливість відновлення системи з диска чи USB;
- можливість підключення до локальної мережі [7].

ЧПУ програма дозволяє працювати у кількох режимах:

- ручний режим - окремо працює кожен привід при натисканні кнопки;
- режим навчання – програма запам'ятовує всі дії оператора;
- напівавтоматичний режим – покрокове виконання операцій програми ЧПУ;
- автоматичний режим - безперервне виконання операцій програми ЧПУ [7].

Спрощено процес згинання металевого дроту можна описати так: спочатку дріт для обробки буде відправлено механізмом накопичення, потім він буде вирівняний механізмом для вирівнювання, його розміри буде виміряно і дріт буде розрізано різальним пристроєм. Після цього відріз дроту буде переданий до згинального механізму для згинання і сформований дріт буде виведено із зони згинання.

Існуючі проблеми в подібних технологічних системах пов'язані з низькою точністю, нестабільністю, низьким коефіцієнтом корисної дії (ККД), складним обслуговуванням та ін. Вони спричиняють зменшення ефективності виробництва та підвищення собівартість продукції, тож дослідження в цій галузі тривають.

1.2 Дослідження існуючих прототипів ТСЗД

Віртуальні та фізичні прототипи часто використовуються як у сфері проектування так і у сфері технологічної підготовки виробництва. Кілька

прикладів фізичних прототипів для дослідження основних принципів технологічного процесу згинання дроту та способів автоматизованого керування ним наведено в [2], [8]- [10].

В проєкті [2] пропонується автоматичний тривимірний механізм згинання дроту, який є гнучким і здатним виконувати будь-який тип згинання для промислових застосувань. Авторами описано проблеми сучасного виробництва, наведено процес модернізації верстата для згинання дроту та запропоновано принцип керування стабілізацією геометрії. Представлено технологічну систему, яку можна ефективно використовувати для створення будь-яких типів згинів дроту під керуванням мікроконтролера.

Інший приклад прототипу машини для згинання дроту у 3D фігури з керуванням на основі Arduino наведено на рис. 1.3 [8]. Це типова мехатронна система, оскільки вона включає механічну, електричну та програмну підсистеми. Мозком прототипу є плата Arduino, яка разом з іншими електронними компонентами встановлена на спеціально розробленій друкованій платі, що не є зручним та вимагає проєктування та виготовлення такої плати.

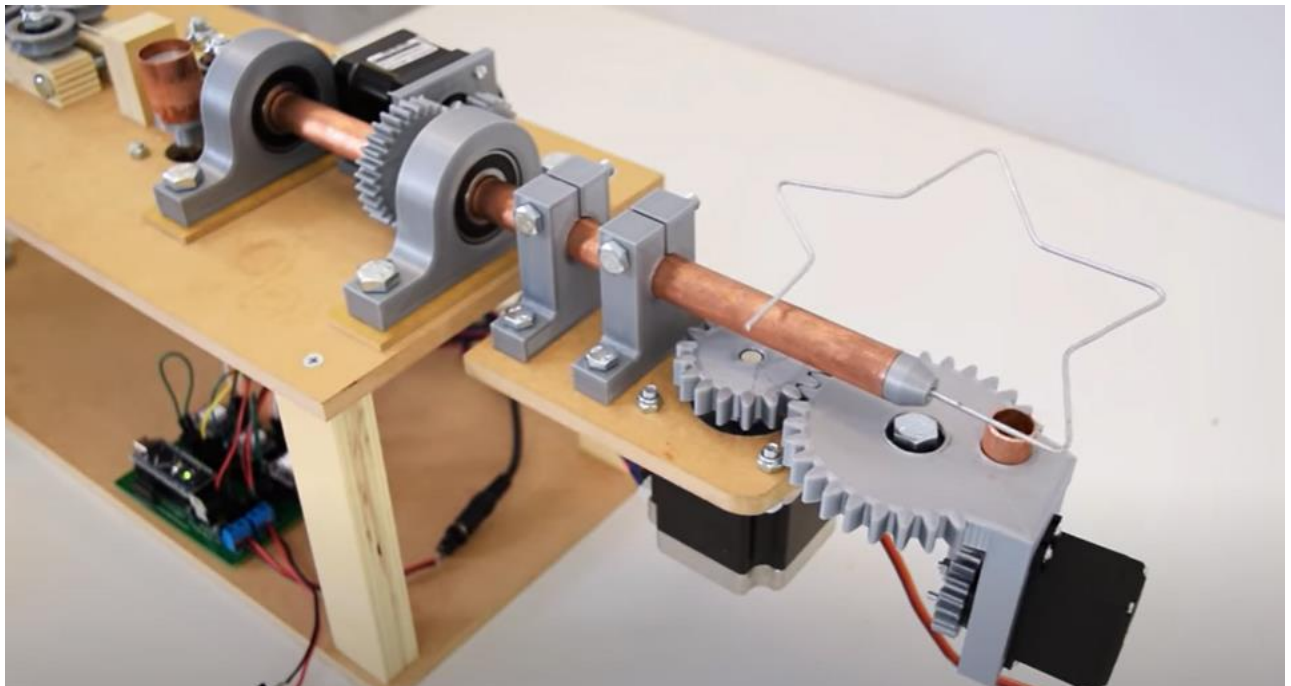


Рисунок 1.3 – Прототип ТСЗД [8]

Призначення цього верстату - формування та згинання дроту малого діаметра ($< \varnothing 2$ мм) у різні фігури та конструкції, такі як канцелярські скріпки, декоративні елементи, пружини. Перевагою цього верстата є можливість згинати одну деталь у двох площинах. Згин виготовляється за допомогою згинального інструменту під час лінійного або обертового руху.

Спочатку дріт проходить через ряд роликів, які відіграють роль випрямлячів. За допомогою крокового двигуна дріт точно подається механізмом подачі до механізму згинання дроту. Кроковий двигун через зубчасту передачу за допомогою шпильки згинає дріт у площині XZ. Сервопривід дозволяє змінювати напрямок обертання, переміщаючи штифт вгору та вниз, за допомогою зубчастої рейки. Кроковий двигун через зубчасту передачу обертає весь згинальний механізм для згинання в іншій площині.

Існують інші проєкти як з відкритим вихідним кодом, так і закриті. Більшість з них мають однакову базову кінематику, але менш складні за механікою, що накладає деякі обмеження на можливу реалізацію деталей.

Наприклад, в роботі [9] також представлено прототип 2D машини для згинання дроту з керуванням на базі Arduino через монітор послідовного порту. Представлена машина може формувати дріт у вигляді деяких 2D геометричних фігур: трикутник, квадрат, прямокутник, шестикутник і спіраль. Для цього проєкту використовувалась багатоцільова друкована плата, на яку було встановлено потрібні компоненти.

В роботі [10] представлено прототип машини для згинання дроту, що здатна згинати дріт 0,8/0,9/1 мм у будь-яку двовимірну форму, в тому числі пружину. Головною метою було зробити його досить точним. Друга мета полягала в тому, щоб зробити його із загальнодоступних частин і компонентів. Усі конструктивні частини надруковані на 3D-принтері, а всі металеві частини доступні в більшості будівельних магазинів.

Однак представлені в роботах [9]-[10] рішення не забезпечують реверсу та додаткового згинання площини, як машина [8], тому функціональність істотно відрізняється.

Основні недоліки прототипів:

а) випрямлячі часто не працюють належним чином, відповідно дріт ковзає, а не котиться, що призводить до того, що дріт залишається вигнутим після операції вирівнювання;

б) дріт у механізмі подачі також ковзає, що впливає на точність і швидкість отримання вигнутої деталі;

в) неконтрольоване обертання вже зігнутої деталі під власною вагою при згині дроту в багато осьовому напрямку;

г) центр маси головки, що згинає, знаходиться надто далеко від нейтральної осі, а сама головка досить важка, що може призвести до коливань і низької точності та швидкості;

д) ергономіка всієї системи недосконала.

Таким чином, різниця між прототипом і професійним обладнанням величезна. Основні цілі професійного обладнання досягти дуже важко, оскільки зазвичай таке обладнання використовується для масового виробництва, де кількість деталей за робочу зміну є вирішальним параметром. Тим не менш, принципи реалізації технологічного процесу та алгоритми керування ТСЗД можуть бути удосконалені шляхом дослідження та модернізації прототипу.

1.3 Аналіз структурних та функціональних особливостей досліджуваного прототипу ТСЗД

Досліджуваний прототип складається з електромеханічної та керуючої підсистем (рис. 1.4).

Основні компоненти прототипу представлені в таблиці 1.1.

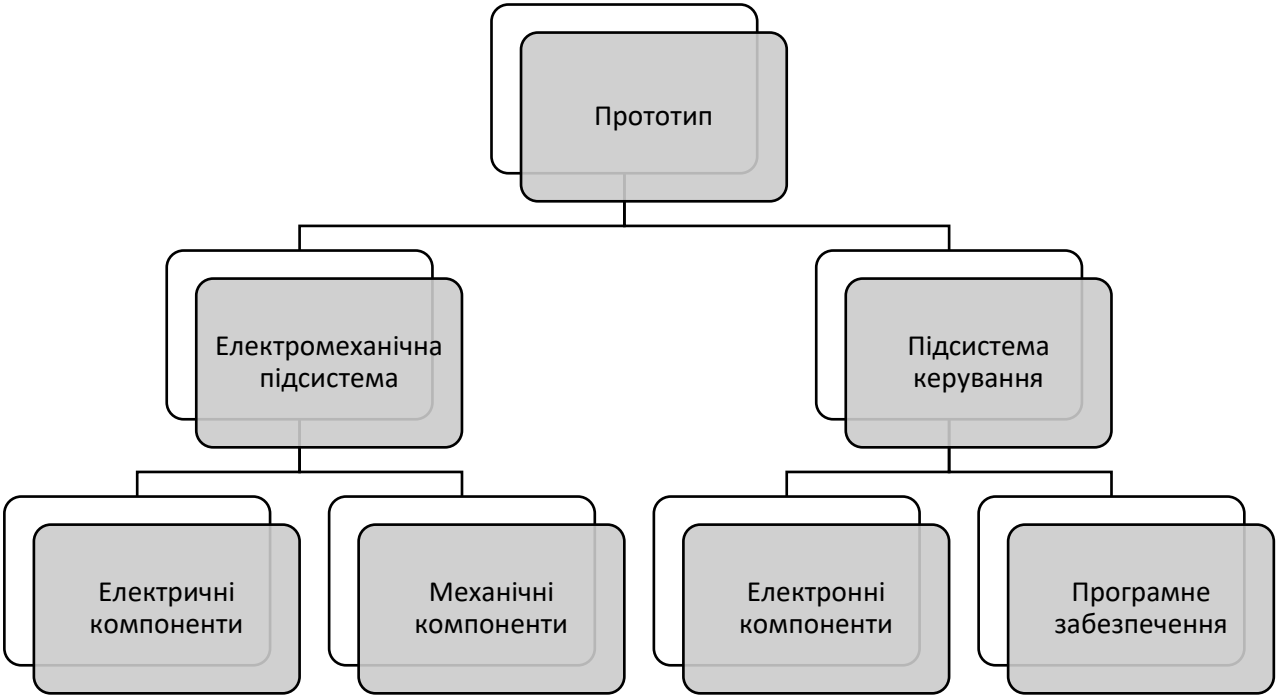


Рисунок 1.4 – Загальна структура прототипу

Таблиця 1.1 - Опис компонентів прототипу

№	Назва	Кількість
1	Joy-it NEMA-17-01 Модуль драйвера крокового двигуна DRV8825	3
2	Joy-it Arduino Nano V 3	1
3	Joy-it com-Motor02 сервомотор	2
4	TRU COMPONENTS Макетна плата	1
5	MAKERFACTORY MF-5911311 Дротові мостові з'єднувачі	1
6	Вугільний плівковий резистор 0.5 W 1.0 КОм	3
7	Електролітичний конденсатор 10 V 100 мкF	5*25
8	Електролітичний конденсатор 50 V 0.47 мкF	3*25
9	Elko аксіальний електролітичний конденсатор 25 V 100 мкF	5
10	Elko радикальний електролітичний конденсатор 16 V 470 мкF	3
11	Блок живлення HN Power HNP-LED24EU-120L6	1
12	Мікроперемикач VA2-16S1-03D0-Z	2
13	L7805 Регулятор напруги 5 V 1 A	2*5
14	Oukeda 42STH47-1684A кроковий двигун	3
15	CNC шилд V 4 for Arduino Nano	1

Електромеханічна підсистема складається з набору зібраних деталей, таких як рама, вал, шестерні, три крокові двигуни та один серводвигун (рис. 1.5).

Три двигуни прототипу є кроковими двигунами OUKEDA NEMA17-01 (рис. 1.6), які можуть бути оптимально використані для завдань у сферах автоматизації, ЧПУ (наприклад, гравірувальні лазери, 3D-принтери, фрезерні верстати тощо) або робототехніки [11]. Особливості та характеристики двигуна такого типу наведено в таблиці 1.2.

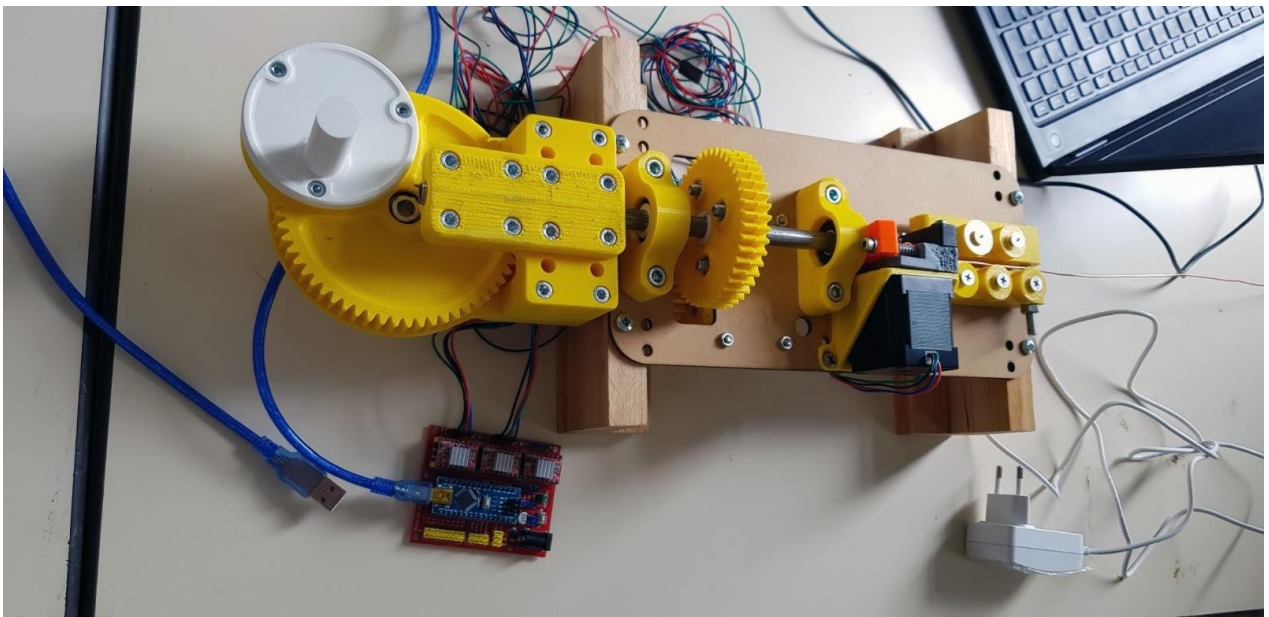


Рисунок 1.5 – Зовнішній вигляд прототипу та його компонентів



Рисунок 1.6 – Зовнішній вигляд крокового двигуна OUKEDA NEMA17-01 [11]

Таблиця 1.2 – Особливості та характеристики крокового двигуна OUKEDA NEMA17-01 [12]

Характеристика	Значення
1	2
Діаметр валу	5 мм (Одинарний вал)
Специфікація	42STH47-1684A
Підключення	4 з'єднувальні кабелі
Кроків на оборот	200
Розміри	42,3 x 42,3 x 47 мм
Утримуючий момент	43.1 Н см
Номінальна напруга	2.8 В
Номінальний струм	1.68 В
Кут кроку	1.8°
Кількість фаз	2
Температурний діапазон	-20°C — +50°C
Електропроводка	Котушка 1А: зелена Котушка 1В: чорна Котушка 2А: синя Котушка 2В: червона

Основними компонентами підсистеми керування є мікроконтролер Arduino Nano V 3 (рис. 1.7) та три модулі драйверів DRV8825 для крокових двигунів.

ARD Nano V 3 – це особливо маленький мікроконтролер, який завдяки штифтовому роз'єму, що виходить у нижній частині, спеціально розроблений для роботи зі змінними платами. Мікроконтролер має оригінальний чіп і тому повністю сумісний з Arduino. Незважаючи на компактний дизайн, Nano V 3 майже повністю відповідає функціональному діапазону більшого R3DIP.

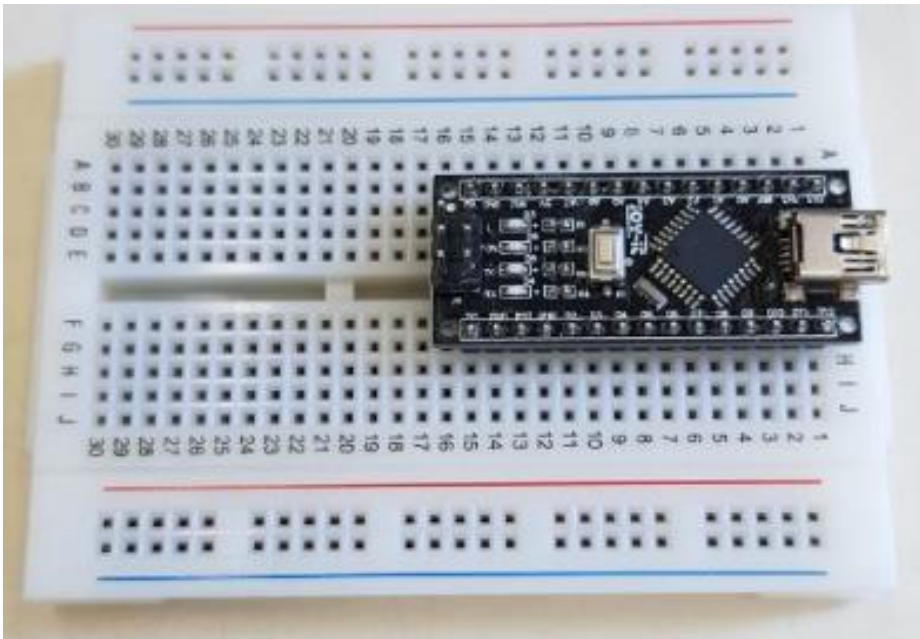


Рисунок 1.7 – Зовнішній вигляд Arduino Nano, підключеної до макетної плати

Інтегрований інтерфейс USB можна використовувати для живлення схеми та плати, а також для передачі програм на мікроконтролер. Розпіновку Arduino Nano можна побачити на рис. 1.8. Її додаткові характеристики представлені в таблиці 1.3 [13].

Arduino Nano V 3.0 GRBL Pinout				ATMega 328P	
				For Grbl v0.9 with variable spindle PWM ENABLED	
Pinout Ref					Pinout Ref
D13	Spindle Direction	D13	D12	Limit Z-Axis	D12
3V3	Not Used	3V3	D11	Variable spindle PWM	D11
VREF	Not Used	VREF	D10	Limit Y-Axis	D10
A0	Reset/ Abort	A0	D9	Limit X-Axis	D9
A1	Feed Hold	A1	D8	Stepper Enable/Disable	D8
A2	Cycle Star/ Resume	A2	D7	Direction Z Axis	D7
A3	Coolant Enable	A3	D6	Direction Y Axis	D6
A4	(Not Used/ Reserve)	A4	D5	Direction X Axis	D5
A5	Probe	A5	D4	Step Pulse Z Axis	D4
A6	Not Used	A6	D3	Step Pulse Y Axis	D3
A7	Not Used	A7	D2	Step Pulse X Axis	D2
		5V	GND		
		RST	RST		
		GND	RX1		
		VIN	TX1		

(NOTE: The Z-limit and the spindle enable pin are swapped, because we had to access the hardware PWM on D11 for variable spindle PWM output to work.) We are still updating this pin configuration at the moment by weighing future options. We'd like to only change the pins once. Stay tuned!

Рисунок 1.8 – Arduino Nano V 3 піни [13]

Таблиця 1.3 – Характеристики мікроконтролера Arduino Nano [13]

Характеристика	Значення
Тип мікроконтролера	ATmega328P-AU
Тактова частота	16 МГц
Робоча напруга	5 В
Аналогові виводи	8
Цифрові виводи	14 (6 з ШІМ)
Напруга постійного струму на контакт I/O	40 мА
Пам'ять	32 Кб (2 Кб для завантажувача)
SRAM	2 Кб
EEPROM	1 Кб
Вхідна напруга	7 – 12 В
Споживаний струм	19 мА
Розміри (W x D)	18 x 45 мм

Модуль DRV8825 (рис. 1.9) забезпечує інтегроване рішення драйвера двигуна для принтерів, сканерів та іншого автоматизованого обладнання. Пристрій має два драйвери Н-мостів і може керувати біполярним кроковим двигуном або двома двигунами постійного струму. Блок вихідного драйвера для кожного складається з N-канальних силових МОП-транзисторів, налаштованих як повні Н-подібні мости для керування обмотками двигуна. DRV8825 може забезпечувати вихідний струм до 2,5-А пік або 1,75-А RMS (з належним радіатором при 24 В і 25°C). Простий інтерфейс крок/напрямок дозволяє легко підключатися до схем контролера. Штифти дозволяють конфігурувати двигун в режимах від повного кроку до 1/32 кроку. Режим розпаду програмується. Функції внутрішнього відключення передбачені для захисту від перевантаження по струму, захисту від короткого замикання, блокування від зниженої напруги та перегріву [14]. Інформацію про розпіновку окремого драйвера можна побачити на рис. 1.10.

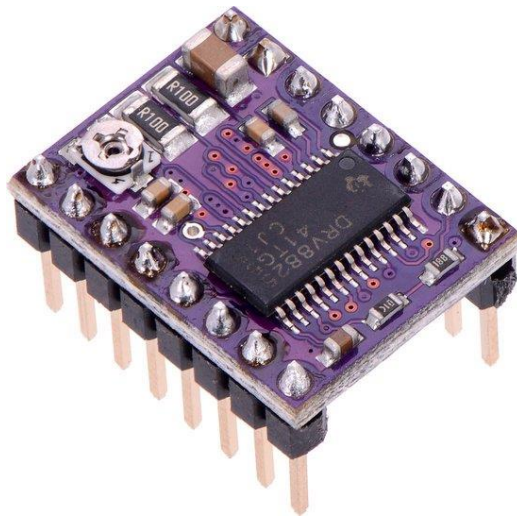


Рисунок 1.9 – DRV8825 модуль драйвера [14]

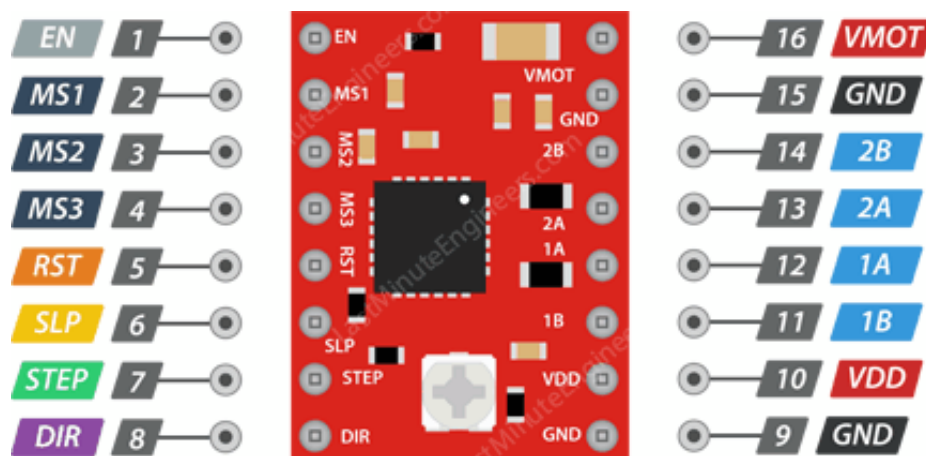


Рисунок 1.10 – DRV8825 піни модуля драйвера [14]

У попередній версії проекту пропонувалося змонтувати компоненти (плату Arduino, драйверні модулі, конденсатори, резистори та регулятори напруги) підсистеми керування на макетній платі та з'єднати їх дротами з кроковими двигунами та серводвигуном. Однак у такому випадку використання HN Power HNP-LED24EU-120L6, який забезпечує живлення, буде проблематичним. Крім того, використання компонентів поверхневого монтажу знижує надійність з'єднань, а велика кількість дротів призводить до їх сплутування.

Для вирішення зазначеної проблеми було вирішено використати CNC Shield V 4 (рис. 1.11). CNC Shield V 4 можна використовувати з Arduino NANO. Як плату розширення для верстатів з ЧПУ. На платі є три слоти для модулів драйвера крокового двигуна, а також можливості для підключення інших модулів, наприклад, кінцевих вимикачів. CNC Shield V 4 має інтерфейс I2C, вхідну напругу 7,5 В - 12 В постійного струму та вихідну напругу 5 В постійного струму через перетворювач напруги або безпосередньо 12 В [15]. Він служить заміною макетної плати та дозволяє легко підключати Arduino Nano, драйверні модулі, крокові двигуни та серводвигун, а також надає можливість використовувати згаданий блок живлення HN Power HNP-LED24EU-120L6. Крім того, його використання запобігає згаданій проблемі щодо низької надійності поверхневого монтажу компонентів на макетній платі та сплутування дротів.

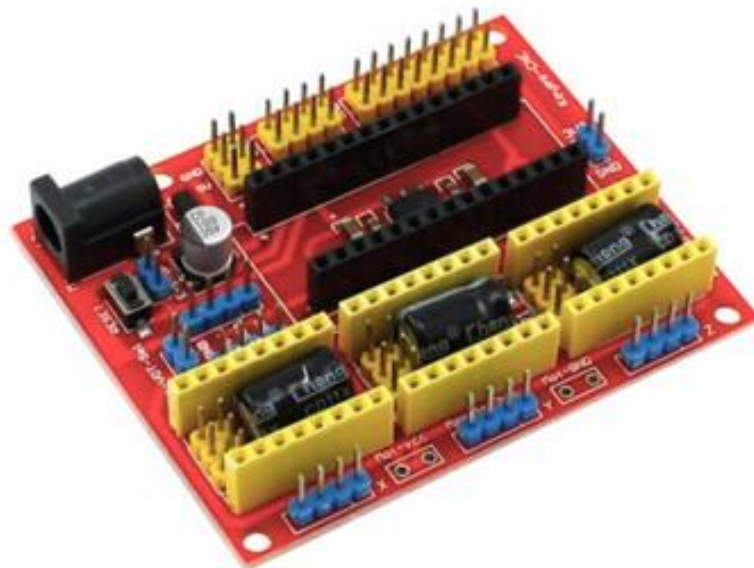


Рисунок 1.11 – Вигляд шилда CNC Shield V 4 [15]

Під час монтажу компонентів на шилді, радіатори охолодження були встановлені на кожному з драйверів DRV8825, як показано на рис. 1.12.

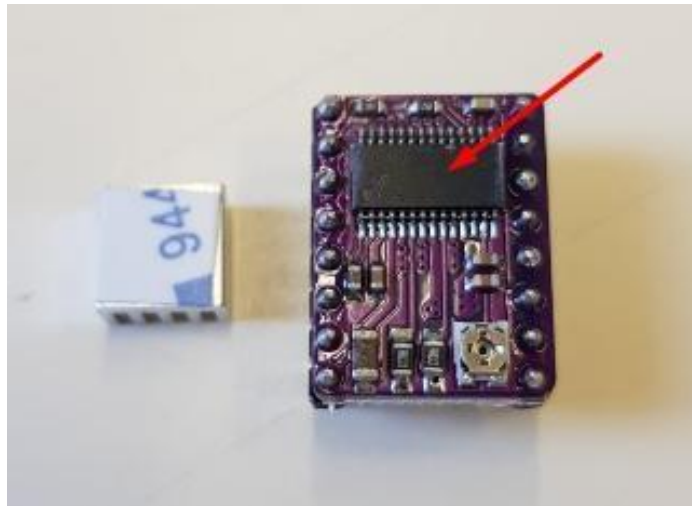


Рисунок 1.12 – Монтаж радіаторів на модуль драйвера DRV8825

Також, Arduino Nano і модулі драйвера були підключені до шилда відповідно до рис. 1.13.

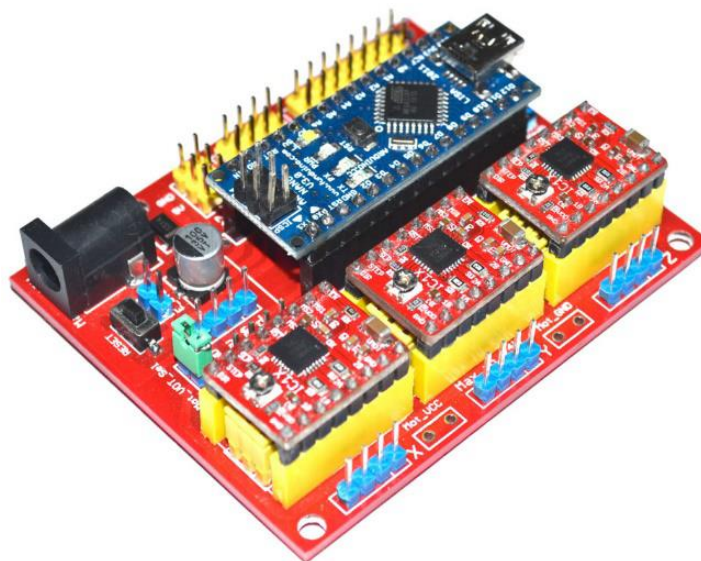


Рисунок 1.13 – Вигляд шилда CNC Shield V 4 з підключеними модулями драйвера DRV8825 та платою Arduino Nano [15]

Потім до роз'ємів X, Y, Z шилда, що складаються з чотирьох виводів (по два виводи на обмотку двигуна), було підключено крокові двигуни з урахуванням схеми узгодження обмоток з виводами (рис. 1.14).

WIRING DIAGRAM

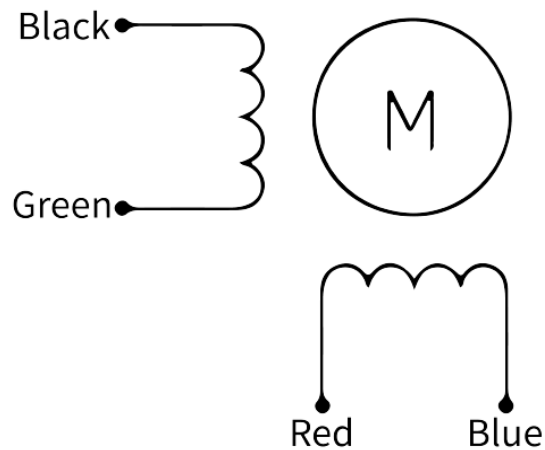


Рисунок 1.14 – Схема підключення крокового двигуна [16]

Приклад підключення дротів двигуна до шилда з встановленим драйвером можна побачити на рис. 1.15.

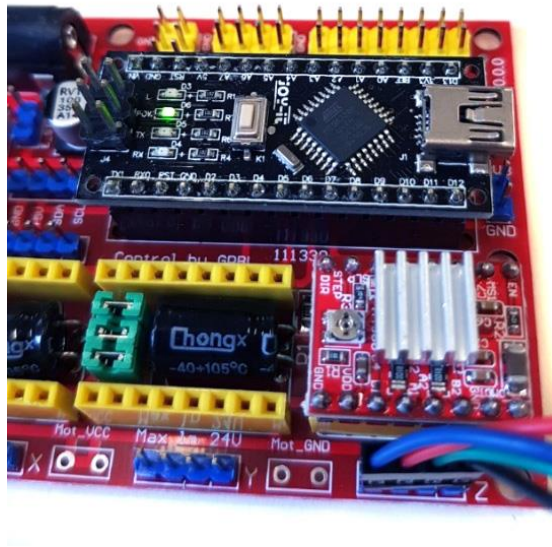


Рисунок 1.15 – Підключення крокового двигуна до шилда за допомогою DRV8825 і дротів

Після цього було підключено джерело живлення та перевірено значення напруги на контактах драйвера: VDD і GND на відповідність 5 В, VMOT і GND – на відповідність 12 В.

До початкової конструкції можливо додати механізм подачі, що має вигляд звичайного екструдера, схожого на ті, що використовуються в 3D-принтерах (рис. 1.16). Щоб надалі його можна було налаштувати для подачі дроту різного діаметру, потрібні додаткові математичні розрахунки.

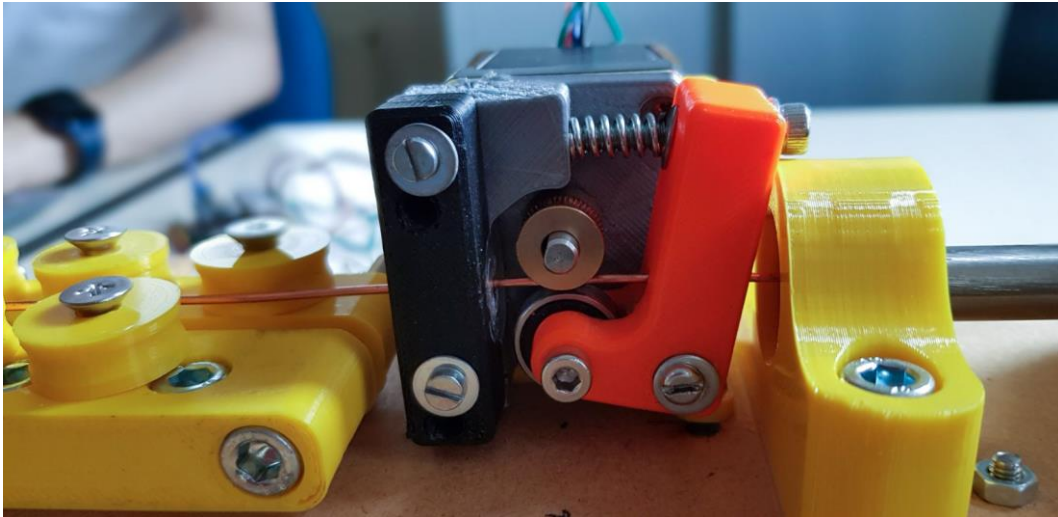


Рисунок 1.16 – Зовнішній вигляд механізму подачі, з'єднаного з двигуном подачі

1.4 Математична модель для розрахунку параметрів механізмів прототипу

Визначення сили, яку треба прикласти, щоб зігнути дріт, може бути виконано за формулою (1.1):

$$[\sigma_{steel}] = \frac{M_2}{W_x} = \frac{F \cdot l \cdot 32}{\pi \cdot d^3} \Rightarrow F = \frac{\pi \cdot d^3 \cdot [\sigma_{steel}]}{32 \cdot l}, \quad (1.1)$$

де $W_x = \pi \cdot \frac{d^3}{32}$ – моментальний опір для штанги;

$[\sigma_{steel}] = 160$ МПа – напруга вигину для сталі.

Результат розрахунку:

$$[\sigma_{steel}] = \frac{3,14 * 8 * 10^{-9} * 160 * 10^6}{32 * 0,027} = 4,65 N$$

Визначення моменту на шестерні ($z_2 = 30$) за формулою (1.2):

$$M_2 = F * l = 4,65 * 0,027 = 0,1256 N * m \quad (1.2)$$

Визначення моменту на передачі ($z_1=18$) за формулою (1.3):

$$\frac{z_2}{z_1} = \frac{M_2}{M_1} \Rightarrow M_1 = \frac{M_2 * z_1}{z_2} = \frac{0,1256 * 18}{30} = 0,0753 N * m, \quad (1.3)$$

де $M_1 = 0,0753 N * m$ – необхідний крутний момент для згинання сталевого дроту діаметром 2 мм.

Вибір крокового двигуна Nema 17, з доступним моментом за формулами (1.4), (1.5):

$$M_{1S} = 0,85 * M_{1Smax} = 0,85 * 0,59 = 0,5 Nm \quad (1.4)$$

$$M_{1S} > M_1 \quad (1.5)$$

Визначення крутного моменту на шестерні 2 ($z=30$) за формулою (1.6):

$$\frac{z_2}{z_1} = \frac{M_2}{M_1} \Rightarrow M_2 = \frac{M_1 * z_2}{z_1} = \frac{30 * 0,5}{18} = 0,78 Nm \quad (1.6)$$

Визначення прикладеного крутного момент за формулою (1.7):

$$M_{load} = l * mg = 0,065 * 15 = 0,975 Nm \quad (1.7)$$

Якщо вибране співвідношення та крутний момент крокового двигуна не відповідають вимозі $M_2 > M_{load}$, тоді інше передавальне число необхідно визначити.

Визначення необхідної жорсткості пружини для забезпечення руху роликів згідно з рівнянням (1.8):

$$\sum F_y = 0 \quad (1.8)$$

$$N_1 - N_2 + N_3 - N_4 + N_5 + F_{spr} = 0$$

$$N_1 = F_{spr} = k * x \text{ (припустимо, що } x=10 \text{ мм руху пружини)}$$

$$\sum F_x = 0$$

$$F_{Fr1} = F_{Fr2} = F_{Fr2} = F_{Fr3} = F_{Fr4} = F_{Fr5}$$

$$F_{step} - F_{Fr1} - F_{Fr2} - F_{Fr2} - F_{Fr3} - F_{Fr4} - F_{Fr5} = 0$$

$$F_{step} = 5 * F_{Fr1}$$

$$F_{step} = 5 * \mu * N$$

$$5 * \mu * k * x = F_{step}$$

$$k = \frac{F_{step}}{5 * \mu * x} = \frac{39,75}{5 * 0,2 * 0,01} = 3975 \text{ N/m}$$

де $\mu = 0,2$ коефіцієнт тертя сталі до пластмаси;

$F_{step} = 39,75 \text{ N}$ сила, необхідна для переміщення дроту.

Визначення необхідної сили для переміщення дроту може бути виконано для даних, що наведено на рис. 1.17 та рис.1.18.

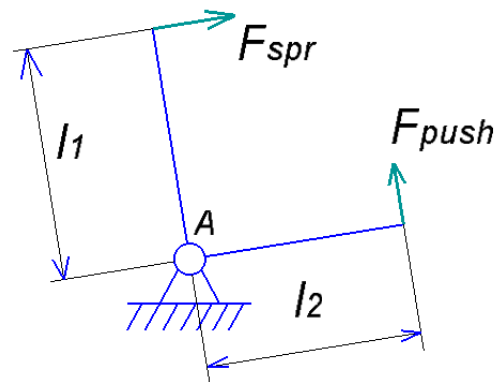
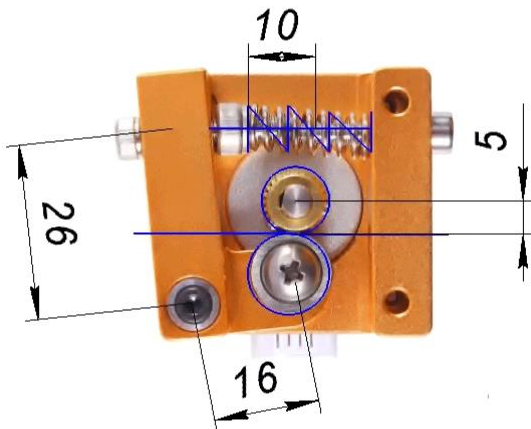


Рисунок 1.17 – Сили та відстані, прикладені до механізму подачі

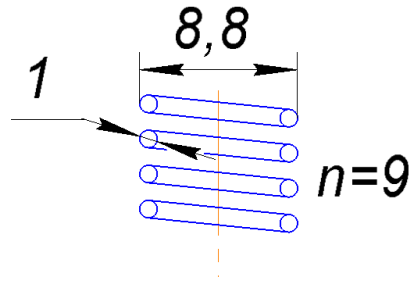


Рисунок 1.18 – Обрані параметри пружини

Визначення жорсткості пружини за формулою (1.9):

$$c = \frac{G \cdot d^4}{8 \cdot D^3 \cdot n} = \frac{80000 \cdot 10^6 \cdot 0,001^4}{8 \cdot 0,0088^3 \cdot 9} = 1630 \text{ N/m} \quad (1.9)$$

де $G = 80000 \text{ МПа}$ для сталі;

$n = 9$ – кількість пружинних кілець;

$D = 8,8$ – зовнішній діаметр;

$d = 1$ – діаметр пружинного дроту.

Визначення виштовхувальної сили F_{push} за формулою (1.10):

$$\sum M_A = 0 \quad F_{push} \cdot l_2 + F_{push} \cdot l_1 = 0 \quad (1.10)$$

$$F_{push} = \frac{-(F_{spr} \cdot l_1)}{l_2} = \frac{-(c \cdot x \cdot l_1)}{l_2} = \frac{-1630 \cdot 0,01 \cdot 0,026}{0,016} = 265 \text{ N}$$

Визначення необхідної сили для переміщення дроту може бути проведено згідно з параметрами на рис. 1.19 за формулою (1.11):

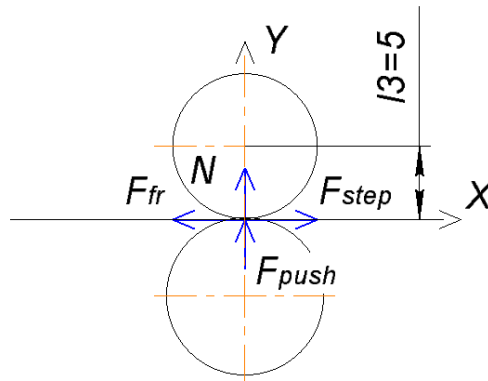


Рисунок 1.19 – Сили та відстані, що застосовуються в пристрої подачі дроту

$$\sum F_y = 0 \quad (1.11)$$

$$N + F_{push} = 0$$

$$N_1 = -F_{push} = 265 \text{ N}$$

$$\sum F_x = 0$$

$$F_{step} - F_{Fr} = 0$$

$$F_{step} = F_{Fr}$$

$$F_{step} = \mu * N = \mu * F_{push} = 0,15 * 265 = 39,75 \text{ N}$$

де $\mu = 0,15$ – коефіцієнт тертя сталі до сталі.

Визначення необхідного крутного моменту для переміщення дроту за формулою (1.12):

$$M = l_3 * F_{step} = 0,005 * 39,75 = 0,2 \text{ Nm} \quad (1.12)$$

Таким чином, Nema 17 кроковий двигун також підходить для переміщення дроту через стан $M_{St3} > M$.

Отримані результати розрахунків потрібно реалізувати при проектуванні конструкції механізму подачі методами віртуальної та доповненої реальності.

1.5 Постановка завдань магістерської роботи

В результат проведених досліджень було виконано огляд сфери застосування, конструктивних та функціональних особливостей реальних ТСЗД та їх прототипів.

Показано, що прототипи мають параметри та характеристики, що не відповідають реальним промисловим вимогам, але є корисними для пошуку шляхів модернізації технологічних систем та алгоритмів керування ними.

Тому, тема магістерської роботи є актуальною.

Метою роботи є дослідження технологій цифрової індустрії для прототипування технологічної системи та розробка програмного забезпечення для числового програмного управління.

Завдання роботи:

- дослідити методи та засоби цифрової індустрії для прототипування технологічних систем;
- розробити методику створення цифрового двійника ТСЗД з використанням технологій віртуальної, доповненої реальності та адитивного виробництва;
- виконати проєктування та конструювання програмного забезпечення ЧПУ прототипу ТСЗД;
- розробити інструкцію з використання програмного забезпечення.

2 ДОСЛІДЖЕННЯ СУЧАСНИХ ТЕХНОЛОГІЙ ЦИФРОВОЇ ІНДУСТРІЇ

2.1 Аналіз концепції Індустрії 4.0

На сьогоднішній день промислові вимоги в різних галузях машинобудування зросли завдяки експоненціальному технологічному прогресу, що відповідає останній промисловій революції.

Термін "Індустрія 4.0" широко використовується в останні роки для опису технологічного прогресу у виробничих процесах, однак його визначення охоплює більше сфер знань [17].

Концепція "Індустрія 4.0" – це нова структура, в якій виробничі та логістичні системи у формі кіберфізичних виробничих систем використовують доступну глобальну інформаційно-комунікаційну мережу для автоматизованого обміну інформацією, в якій узгоджуються виробничі та бізнес-процеси [18].

Після введення цього терміну з'явилося багато поглядів і визначень, які впорядковано показані в роботі [19]. На думку авторів роботи [20], Індустрія 4.0 - це наступний етап цифровізації виробничого сектору, зумовлений чотирма потрясіннями: вражаючим зростанням обсягів даних, обчислювальних потужностей та зв'язку, особливо новими малопотужними глобальними мережами; появою можливостей аналітики та бізнес-аналітики; новими формами людино-машинної взаємодії, такими як сенсорні інтерфейси та системи доповненої реальності; а також удосконаленнями у передачі цифрових інструкцій у фізичний світ, такими як передова робототехніка та 3-D друк.

В роботі [21] Індустрію 4.0 визначено як новий рівень організації та управління ланцюжком створення вартості протягом усього ЖЦ продукції. Усі визначення сходяться на тому, що Індустрія 4.0 - це нова тенденція, яка ґрунтується на різних технологічних досягненнях, що змінюють спосіб виробництва продукції. Ці технологічні досягнення показані на рисунку 2.1 у вигляді дев'яти стовпів, які формують основу Індустрії 4.0 [22].

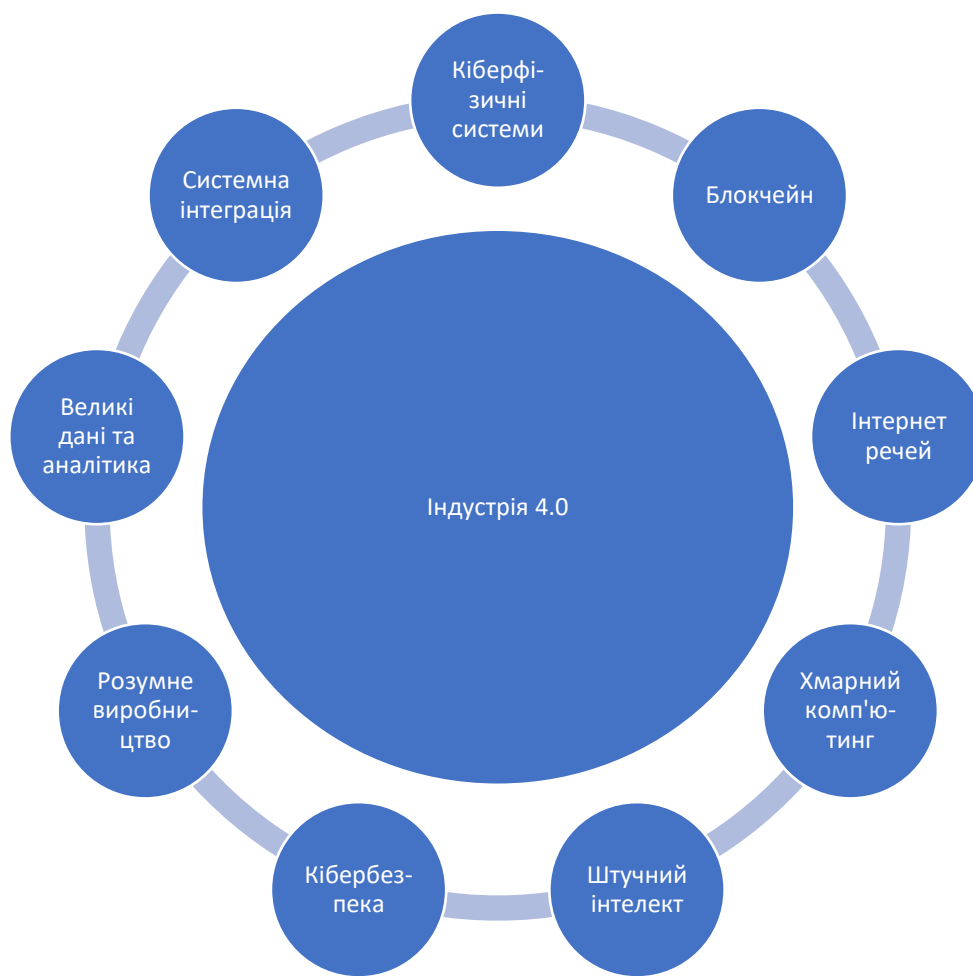


Рисунок 2.1 – Дев'ять стовпів Індустрії 4.0 [22]

На думку авторів дослідження [23], центральною метою Індустрії 4.0 є задоволення індивідуальних потреб клієнтів, що впливає на такі сфери, як управління замовленнями, дослідження та розробка, виробництво, введення в експлуатацію, доставка, утилізація та переробки продукції.

У роботі [24] зазначається, що метою Індустрії 4.0 є наскрізна цифровізація всіх фізичних активів та їх інтеграція в цифрові екосистеми з партнерами по ланцюжку створення вартості. Індустрія 4.0 має на меті працювати з вищим рівнем автоматизації, досягаючи вищого рівня операційної продуктивності та ефективності системи [25].

Для досягнення цієї мети автори роботи [26] запропонували чотири ключові компоненти Індустрії 4.0: кіберфізичні системи (зв'язок між реальним і віртуальним світом); Інтернет речей; Інтернет послуг; Розумне виробництво.

Інша точка зору наведена в роботі [27], де визначено чотири фундаментальні концептуальні підходи:

- кіберфізичні системи;
- Інтернет-технології (IoT (Інтернет речей), IoS (Інтернет послуг) та IoD (Інтернет даних));
- компоненти як носії інформації;
- цілісна безпека та захищеність, включаючи конфіденційність та захист знань.

Узагальнені цілі Індустрії 4.0 [28, 29]:

- забезпечити масову кастомізацію виготовленої продукції за допомогою ІТ-технологій;
- здійснювати автоматичну та гнучку адаптацію виробничого ланцюга;
- відстежувати деталі та продукти;
- сприяти комунікації між деталями, продуктами та машинами;
- застосовувати парадигми людино-машинної взаємодії (НМІ);
- досягти оптимізації виробництва на "розумних" заводах за допомогою Інтернету речей;
- надавати нові види послуг та бізнес-моделей взаємодії в ланцюжку створення вартості.

Індустрія 4.0 передбачає проведення досліджень і розробок у восьми ключових сферах для досягнення поставлених цілей в економічних секторах і підтримки прийняття її принципів у промисловості [30]:

- стандартизація та еталонна архітектура;
- управління складними системами;
- комплексна широкосмугова інфраструктура для промисловості;
- охорона та безпека;
- організація та проектування робіт;
- навчання та безперервний професійний розвиток;
- нормативно-правова база;

- ефективність використання ресурсів: шляхом підвищення продуктивності та ефективності використання ресурсів слід зменшити споживання сировини та енергії.

Ці наукові розробки в цих сферах безпосередньо вплинули на 10 основних технологічних категорій, пов'язаних з виробничим процесом у ланцюжку створення цінності Індустрії 4.0. Згідно з [31], основними технологіями є:

- інжиніринг, технології та організація виробництва;
- машини, робототехніка та співпраця людина-робот;
- інтелектуальні можливості;
- логістика;
- планування та контроль виробництва;
- організація праці, проєктування робочих місць та допомога;
- ресурсо- та енергоефективність.

Деякі наукові розробки є визначними в різних галузях інженерії, таких як цивільна, військово-морська, машинобудівна, автомобільна та аеронавігаційна. Наприклад, в автомобільній та аерокосмічній галузях проєкт Індустрії 4.0 "FORCAM" дозволяє компаніям вимірювати продуктивність своїх машин і установок в режимі реального часу, щоб швидко розпізнавати і виправляти помилки, а також безперервно оптимізувати їх продуктивність [33].

Технології адитивного виробництва (АВ) представлені як основні складові Індустрії 4.0 в роботі [34], автори якої зосередилися на трьох важливих аспектах: останні досягнення в галузі матеріалознавства, розробка технологічних процесів і вдосконалення дизайну. Таким чином, широке впровадження технологій 3D-друку, використання віртуальної і доповненої реальності – це важливі технології Індустрії 4.0.

Індустрія 4.0 зосереджена на створенні інтелектуальних продуктів, процесів і процедур. Сутність Індустрії 4.0 полягає в використанні Інтернету речей та Інтернету послуг, що означає повсюдне підключення людей, речей і машин. Це веде до з'єднання віртуального світу та фізичних об'єктів у реальному світі [35].

2.2 Цифрові двійники

Завдяки технології Інтернету речей (IoT) можливо створити точну цифрову копію сутності у фізичному світі, що дозволяє зрозуміти, як покращити проектування та виробництво, підвищити ефективність або виявити проблему, перш ніж вона виникне. Це дає можливість в режимі реального часу оновлювати статус продукту та процесів, а також проходити сценарії «Що, якщо?», не піддаючи ризику свої активи. Узагальнений опис цифрового двійника (ЦД) представлено на рис. 2.2.

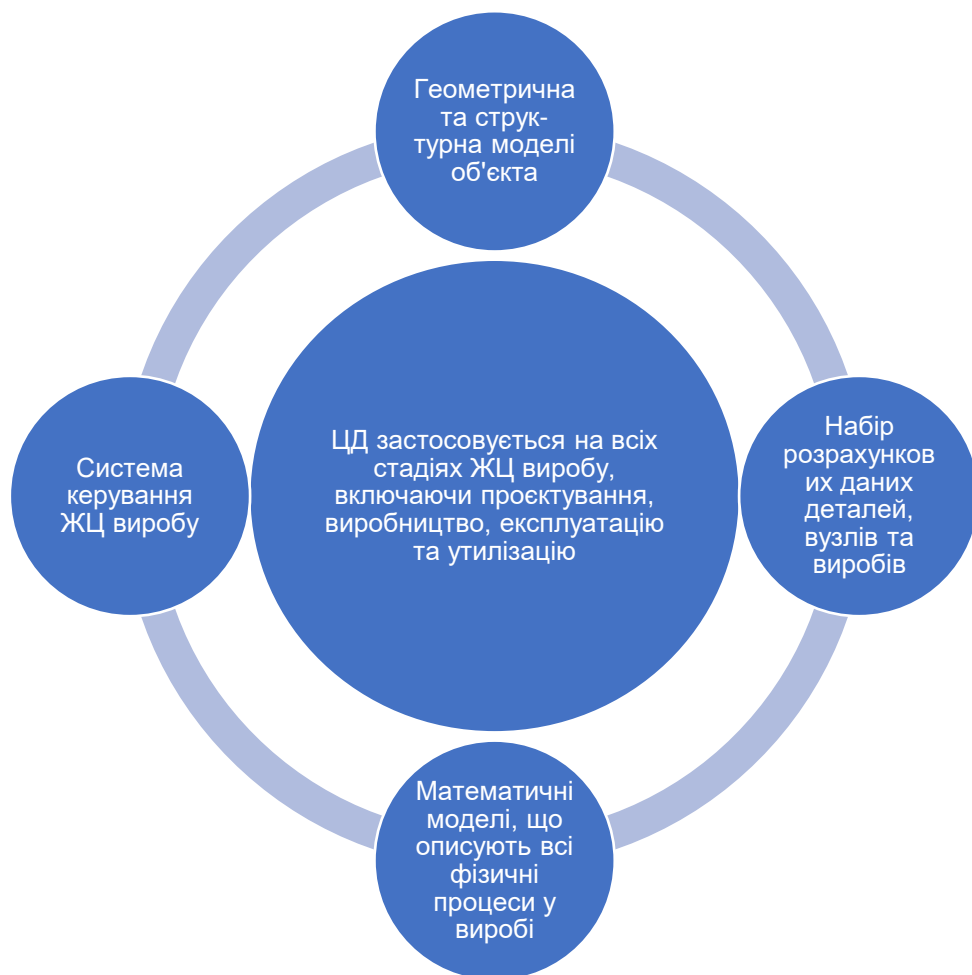


Рисунок 2.2 – Опис цифрового двійника [36]

Використання даних, зібраних датчиками, і створення комп'ютерної моделі продукту чи процесу дозволяє створити ЦД різних типів. Класифікацію ЦД можна умовно представити як показано на рис. 2.3 [36].

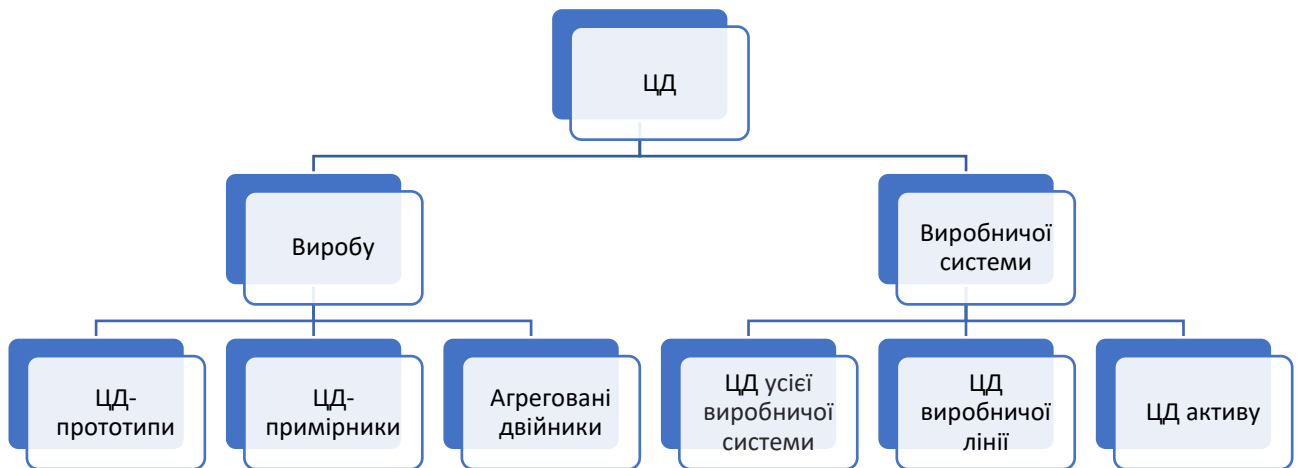


Рисунок 2.3 – Класифікація цифрових двійників [36]

Найбільш популярними типами ЦД є ЦД активу, ЦД процесу та мережевий ЦД [37].

ЦД використовується на всіх етапах створення проєктованих виробів (рис.2.4).

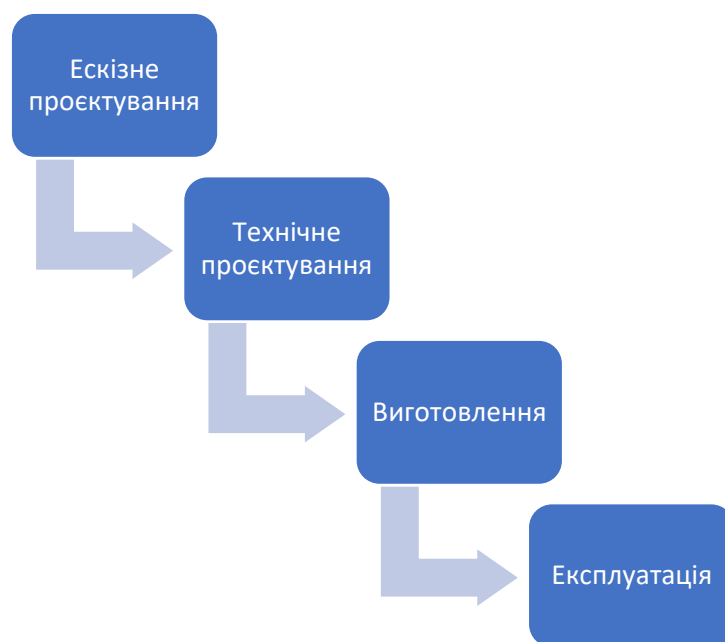


Рисунок 2.4 – Етапи створення ЦД [36]

Структура ЦД об'єднує цифрову модель, цифрову тінь та підключення (рис. 2.5).

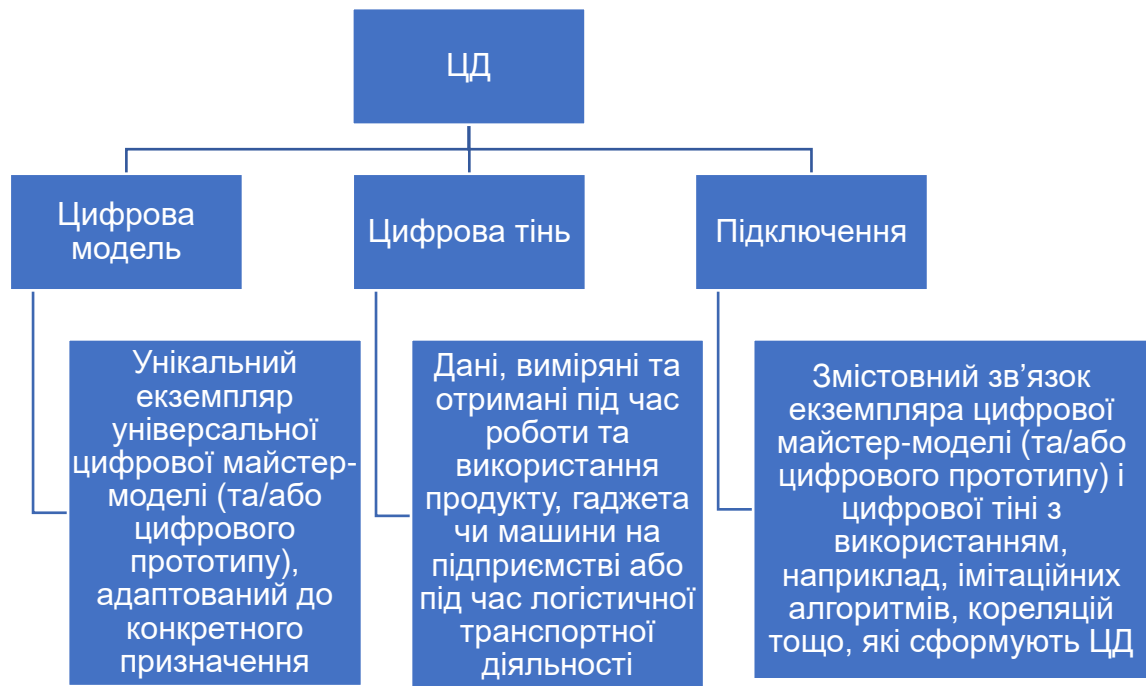


Рисунок 2.5 – Структура цифрового двійника

Таким чином, ЦД складається з трьох частин: фізичного продукту, віртуального продукту та зв'язку (підключення, дані, інформація) між фізичним і віртуальним продуктом (рис.2.6).

Він служить мостом між фізичним і цифровим світом. На відміну від CAD (яка зосереджена виключно на цифровому світі) та IoT (яка значною мірою зосереджена на фізичному світі), ЦД характеризується двосторонньою взаємодією між цифровим і фізичним світами, що, може надати багато переваг. З одного боку, фізичний продукт можна зробити більш «інтелектуальним», щоб активно коригувати його поведінку в реальному часі відповідно до «рекомендацій», наданих віртуальним продуктом. З іншого боку, віртуальний продукт можна зробити більш «фактичним», щоб точно відображати реальний стан фізичного продукту.

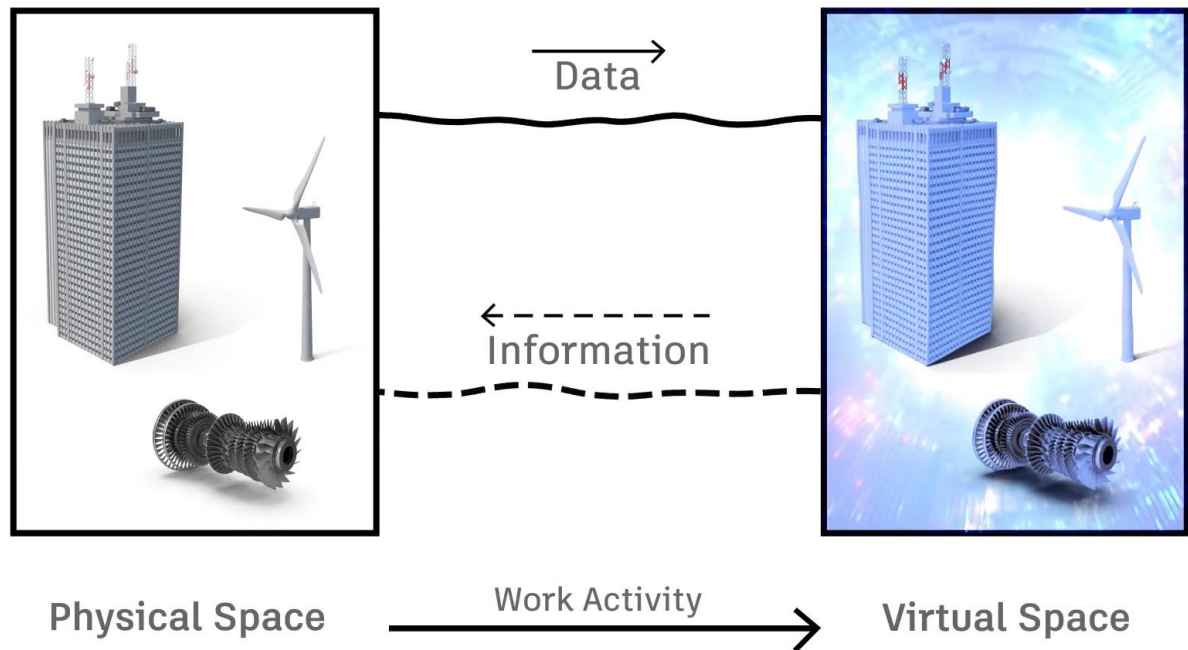


Рисунок 2.6 – Зв'язок частин ЦД [38]

Основні характеристики ЦД наступні:

а) відображення в реальному часі - у ЦД існують два простори: фізичний простір і віртуальний простір. Віртуальний простір є реальним відображенням фізичного простору, і він може забезпечувати надвисоку синхронізацію та відповідність фізичному простору;

б) взаємодія та конвергенція - цю характеристику можна пояснити з трьох аспектів:

- 1) взаємодія та конвергенція у фізичному просторі;
- 2) взаємодія та конвергенція між історичними даними та даними у реальному часі;
- 3) взаємодія та конвергенція між фізичним і віртуальним простором;

в) самоеволюція – ЦД може оновлювати дані в режимі реального часу, так що віртуальні моделі можуть постійно вдосконалюватися шляхом паралельного порівняння віртуального простору з фізичним.

2.3 Розробка методики прототипування на основі ЦД

У минулому ЦД здебільшого використовувався для діагностики несправностей, прогнозного обслуговування та аналізу продуктивності. Недостатньо зусиль було спрямовано на вивчення застосовності ЦД для дизайну продукту щодо того, як і яким чином зв'язок, синергія та спільна еволюція між фізичним продуктом і його цифровим представленням (віртуальний продукт) можуть призвести до більш інформованого, прискореного та інноваційного процесу проєктування [39].

Крім того, поки що ЦД використовувався лише для проєктування, експлуатації та обслуговування складних систем (наприклад, авіаційних двигунів, вітрових турбін, енергетичних заводів і так далі), тоді як мало зусиль було присвячено вивченню значення ЦД для дизайну споживчого продукту.

Сьогодні процес проєктування стає все більш оцифрованим, про що свідчить універсальне застосування пакетів цифрового проєктування (наприклад, CAD, CAM, CAE, FEA, тощо).

Нещодавно з'явилася нова парадигма розробки продукту на основі даних. Завдяки Інтернету речей (IoT) дані збираються безпосередньо з розумних продуктів, передаються в «хмару» в режимі реального часу та аналізуються за допомогою аналітики великих даних.

Сучасні дослідження зосереджені на синергії між парою віртуального та фізичного продукту. У віртуальному світі продукти створюються в лабораторіях для візуалізації структури продукту, імітації поведінки продукту та оптимізації продуктивності продукту. У фізичному світі, коли продуктами користуються кінцеві користувачі, їх продуктивність, поведінка та взаємодія з користувачами фіксуються датчиками та контролюються виконавчими механізмами. Традиційно віртуальні та фізичні продукти створюються, аналізуються та модернізуються окремо один від одного.

У світлі дизайну продукту, керованого даними, вимагається нова структура, яка може ефективно об'єднувати, інтегрувати та синхронізувати все

«більші» дані, пов'язані з віртуальним продуктом, фізичним продуктом і їх зворотним зв'язком [39].

Методика прототипування допоможе скеровувати виробників щодо створення ЦД та використання інформації, наданої ЦД для підтримки процесу проєктування продукту (рис. 2.7).

Блок 1 . Початок.

Блок 2. Розробка віртуального представлення фізичного продукту. Основними технологіями цього кроку є автоматизоване проєктування (CAD) і 3D-моделювання. Обидві технології широко використовуються в дизайні продукту. Віртуальний продукт включає три аспекти: елементи, поведінку та правила. На рівні елементів віртуальна модель продукту в основному включає геометричну модель і фізичну модель продукту, користувача та середовища тощо. На рівні поведінки не лише аналізують поведінку продуктів і користувачів, але й зосереджуються на аналіз продукту та взаємодії з користувачем, спричиненими поведінкою та моделюванням. На рівні правил віртуальна модель в основному включає моделі оцінки, оптимізації та прогнозування, створені відповідно до закону функціонування продукту.

Блок 3. Обробка даних для прийняття проєктних рішень. Дані, зібрані з різних джерел (тобто в основному з фізичного продукту, а також з Інтернету), аналізуються, інтегруються та візуалізуються. По-перше, аналітика даних необхідна для перетворення даних у більш конкретну інформацію, яка може бути напряму запитана дизайнерами для прийняття рішень. По-друге, оскільки дані про продукти збираються з різних джерел, інтеграція даних корисна для виявлення прихованих закономірностей, які неможливо розкрити на основі одного джерела даних. По-третє, технології візуалізації даних включені для представлення даних у більш явний спосіб. Нарешті, передові методи штучного інтелекту можуть бути включені для покращення когнітивних здібностей ЦД (наприклад, міркування, вирішення проблем і представлення знань), щоб певні відносно прості рекомендації можна було робити автоматично.

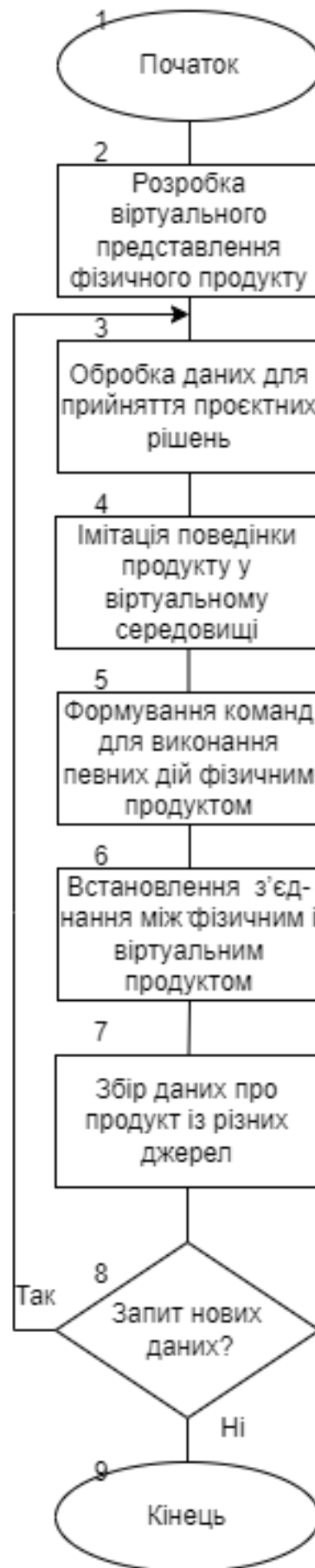


Рисунок 2.7 – Схема алгоритму методики прототипування на основі ЦД

Блок 4. Імітація поведінки продукту у віртуальному середовищі. Основні технології цього кроку включають моделювання та віртуальну реальність (VR). Моделювання ключових функцій і поведінки фізичного продукту у віртуальному світі широко використовується в дизайні продукту. З іншого боку, технології віртуальної реальності відіграють роль в залученні дизайнерів і навіть користувачів до «прямой» взаємодії з віртуальним продуктом у змодельованому середовищі. Останнім часом технології віртуальної реальності все частіше використовуються для підтримки створення віртуальних прототипів і дизайну продукту. Багато доступних апаратних пристроїв віртуальної реальності можна безпосередньо адаптувати для ЦД.

Блок 5. Формування команд для виконання певних дій фізичним продуктом. Відповідно до рекомендацій ЦД, фізичний продукт оснащений можливістю за допомогою різних приводів, адаптивно регулювати свою функцію, поведінку та структуру у фізичному світі. Сенсори та актуатори є двома технологічними основами ЦД. Перші відіграють роль у сприйнятті зовнішнього світу, тоді як другі відіграють роль у виконанні бажаних коригувань, які вимагає ЦД. На практиці найчастіше використовуються гідравлічні, пневматичні, електричні та механічні актуатори. Крім того, технології доповненої реальності (AR) можна використовувати для відображення деяких частин віртуального продукту у фізичному світі. AR дозволяє кінцевим користувачам переглядати стан своїх продуктів у реальному часі, тому останнім часом технології AR дедалі частіше застосовуються у сучасній індустрії.

Блок 6. Встановлення двостороннього та безпечного з'єднання в реальному часі між фізичним і віртуальним продуктом. Підключення можливе за допомогою низки технологій, таких як мережевий зв'язок, хмарні обчислення та безпека мережі.

По-перше, мережеві технології дозволяють продукту надсилати поточні дані в «хмару» для живлення віртуального продукту. Можливі мережеві технології для споживчих продуктів включають, наприклад, Bluetooth, QR-код, штрих-код, Wi-Fi, Z-Wave тощо.

По-друге, хмарні обчислення дозволяють розробляти, розгортати та підтримувати віртуальний продукт повністю в «хмарі», щоб до нього могли зручно отримати доступ як дизайнери, так і користувачі з будь-якого місця, де є доступ до Інтернету. Нарешті, оскільки дані продукту прямо чи опосередковано стосуються взаємодії між користувачем і продуктом, дуже важливо гарантувати безпеку з'єднань. У світі Інтернету речей багато зусиль було спрямовано на підключення фізичного та віртуального продукту, який можна адаптувати для дослідження ЦД.

Блок 7. Збір даних про продукт із різних джерел. Загалом, ЦД має обробляти різні типи даних, пов'язаних із продуктом. Для звичайних продуктів фізичні дані продукту зазвичай поділяються на:

- дані про продукт (коментарі клієнтів, записи про перегляд і завантаження);
- дані про навколишнє середовище;
- дані про клієнта;
- інтерактивні дані (дані про взаємодію між користувачем, продуктом та середовищем).

За допомогою сенсорної технології та технології Інтернету речей можна збирати деякі з вищевказаних даних у режимі реального часу та аналізувати інструкцію з використання продукту, записи перегляду вебсторінок клієнтами, завантажувати записи, оцінювати відгуки тощо.

Блок 8. Запит нових даних? Дозволяє перевірити потребу в додаткових даних, а також отримати решту даних. Зібрані дані передаються на етап 3, щоб закрити цикл створення більш функціонального віртуального продукту.

Після того, як ЦД успішно створений, він може полегшити дизайнерам виконання різних проєктних робіт. Відповідно до теорії та методології проєктування [40], ЦД відіграє роль у генеруванні нових рішень, збагаченні функціональної інформації проєктних рішень та представленні знань про дизайн. Нові рішення створюються на основі креативності, комбінування та модифікації.

ЦД буде найбільш корисним для дизайну на основі комбінацій. Найбільш часто використовуваною методологією проєктування на основі комбінацій є системний підхід до проєктування, який поділяє процес проєктування на чотири фази:

- планування та уточнення завдання;
- концептуальний дизайн;
- проєкт втілення;
- деталізований дизайн.

Кожен етап можна далі розділити на кілька під-етапів. Відповідні методики проєктування можуть бути включені в різні фази процесу проєктування системи. Наприклад, на етапі концептуального проєктування можна включити аксіоматичний дизайн, теорію вирішення винахідницьких задач (ТВВЗ), надійний дизайн і адаптований дизайн, щоб зменшити непотрібну складність дизайну, вирішити небажані суперечності дизайну, зробити продукт менш чутливим до невизначеності та підвищення адаптивності продукту до мінливих вимог, відповідно.

Оскільки проєктування на основі використання ЦД характеризується здатністю повністю використовувати дані для побудови віртуальної моделі, останньою фазою процесу проєктування пропонується спеціальний етап проєктування, який називається віртуальною верифікацією. Повний процес показано на рис. 2.8. Різні дії з проєктування та пов'язані з ними дані розділені на п'ять частин, які відповідають п'яти етапам проєктування.

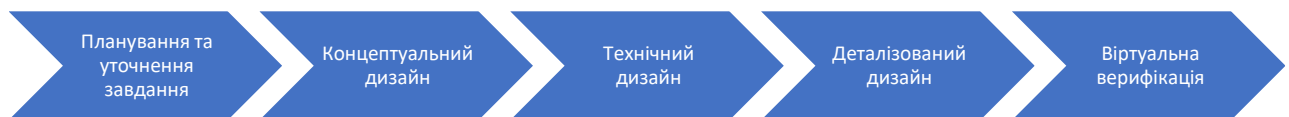


Рисунок 2.8 – Фреймворк дизайну на основі ЦД

2.4 Результат розробки віртуальної частини ЦД

Для створення віртуальної моделі було обрано застосунок Creo Parametric. Було розроблено окремі деталі механізму подачі дроту (extrude_base, extrude_bearing_arm, підшипник, вал, основа) (рис.2.9) та виконано їх складання (рис. 2.10).

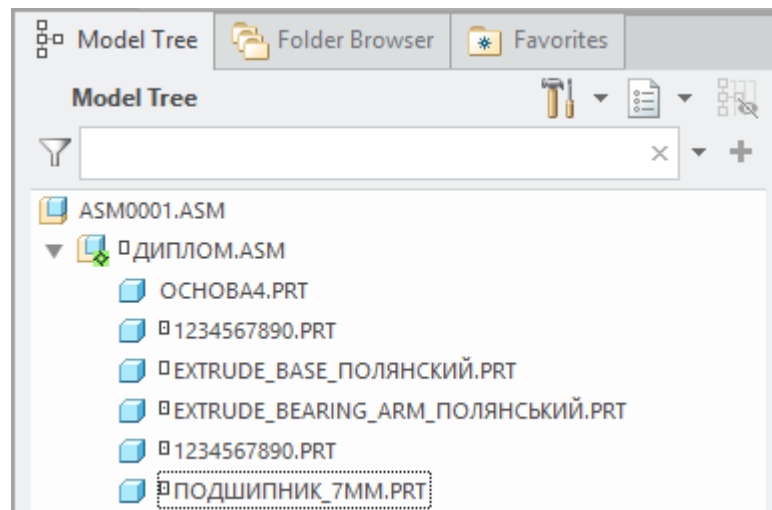


Рисунок 2.9 – Дерево побудови

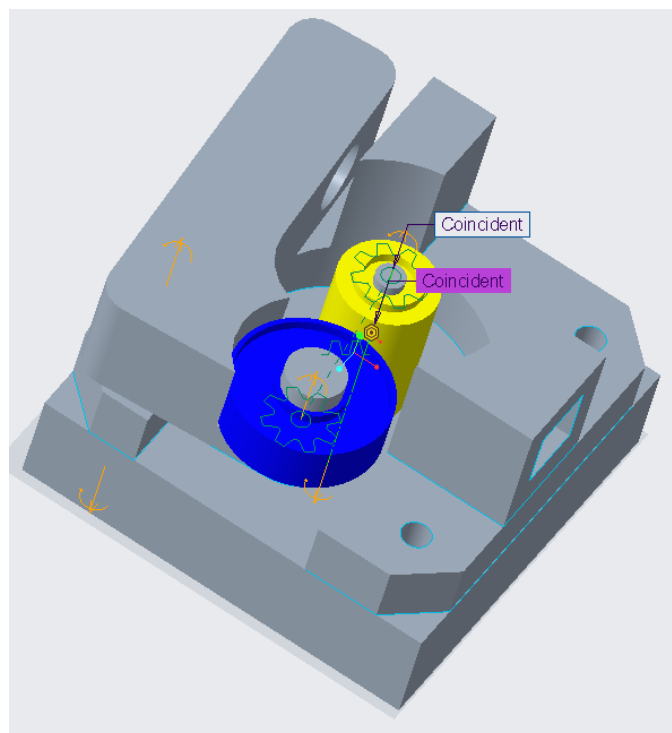


Рисунок 2.10 – Результат побудови віртуальної моделі механізму подачі

Наступним кроком створення було створення анімації для симуляції роботи механізму подачі дроту (рис. 2.11).

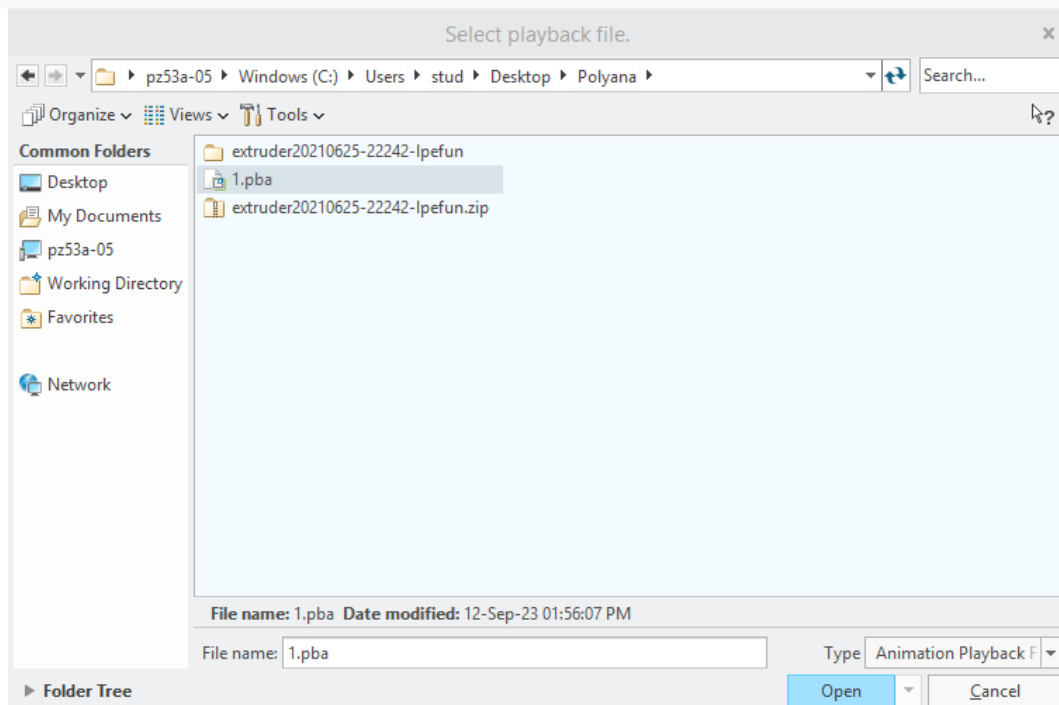


Рисунок 2.11 – Вікно завантаження анімації

Для створення доповненої реальності необхідно було в системі Creo з меню Model перейти до меню Tools, пункт Augmented Reality, за допомогою функції Publish Model завантажити модель до хмари.

На телефон або планшет необхідно було встановити застосунок під назвою Vuforia View - це програма для відтворення доповненої реальності.

Завантаживши файл у хмару використовуючи функцію Manage Model переходимо у хмару (рис. 2.12), де можна налаштувати доступ до цієї моделі.

Поділитися моделлю можна натиснувши кнопку у полі Share, після чого відкривається вікно (рис 2.13) у якому можна поділитись посиланням або вісканувати QR-код.

Далі через Vuforia View скануємо QR-код або через надіслане на пошту посилання завантажуюмо модель до програми, після чого необхідно навести телефон на поверхню, на якій необхідно відобразити модель та натиснути на екран, результат можна побачити на рисунку 2.14.

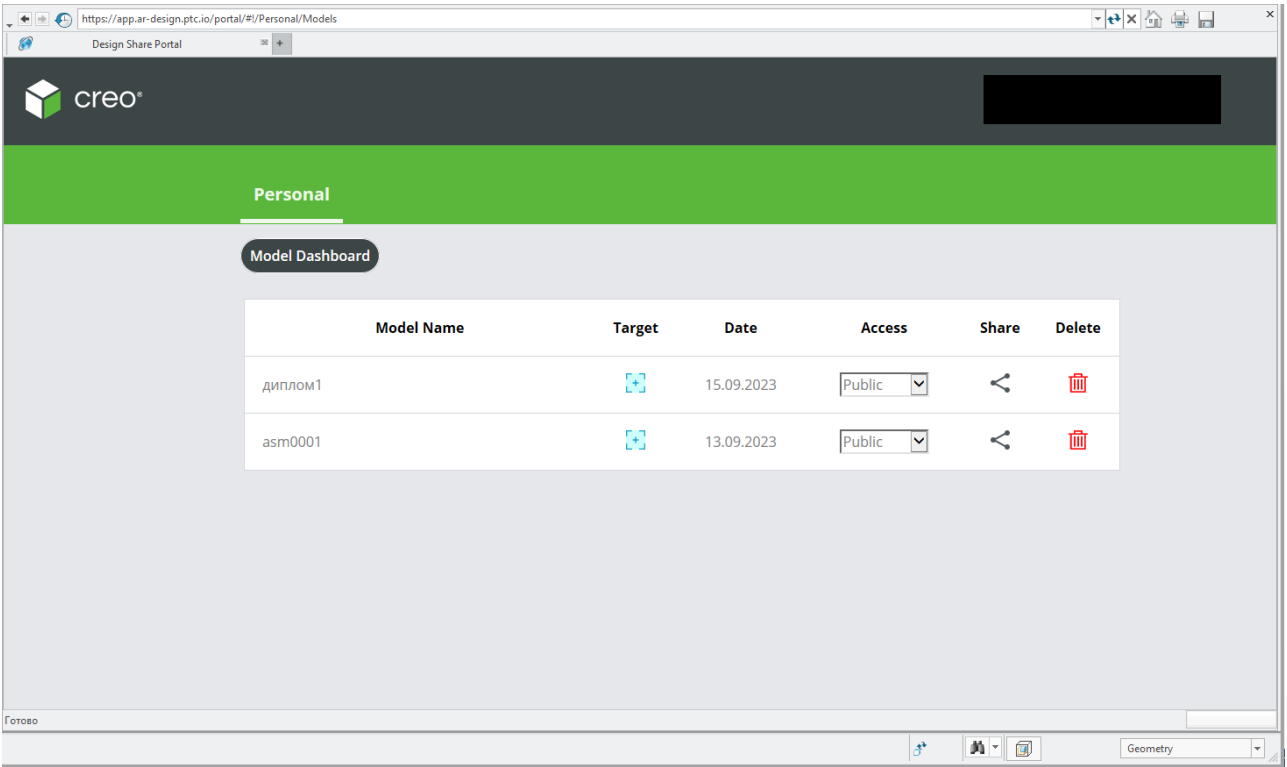


Рисунок 2.12 – Вікно керування моделлю в хмарі

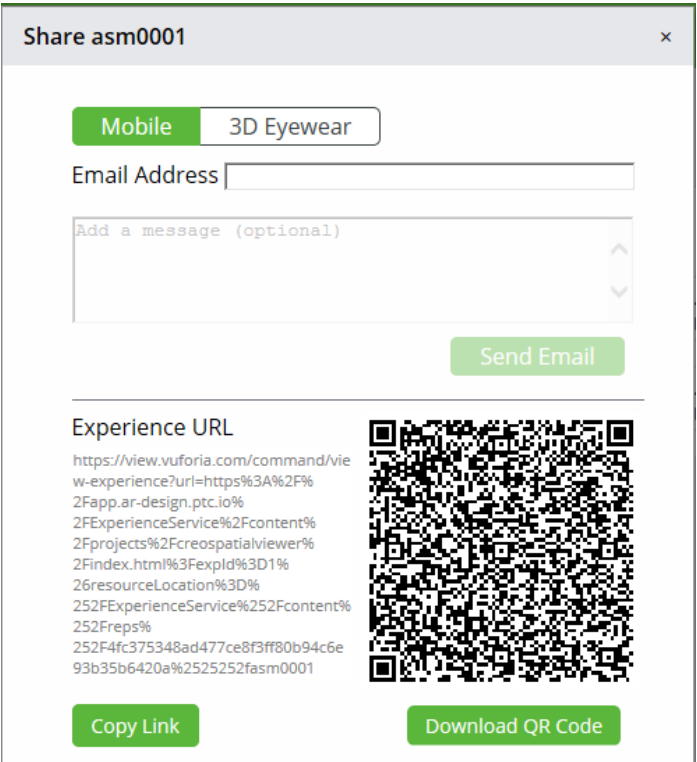


Рисунок 2.13 – Вікно «Поділитися»



Рисунок 2.14 – Відображення моделі засобами доповненої реальності

2.5 Результат розробки фізичної частини ЦД

Для створення фізичної моделі було використано апаратне забезпечення у вигляді 3D принтера Leapfrog CreatrHS. В якості матеріалу використовувався ABS-пластик. В якості програмного забезпечення використовувалася програма Simplify3D, в якій виконувалося розташування друкованих деталей (рис. 2.15) та налаштування параметрів друку (рис. 2.16).

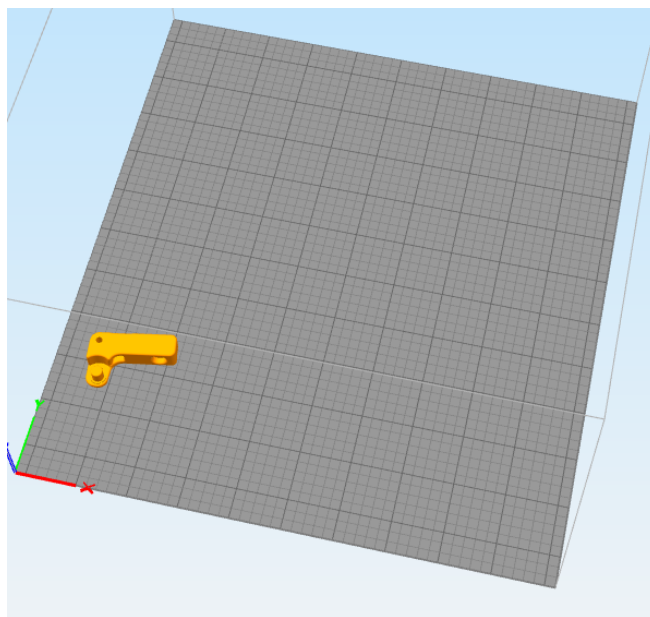


Рисунок 2.15 – Розташування деталі у робочій зоні

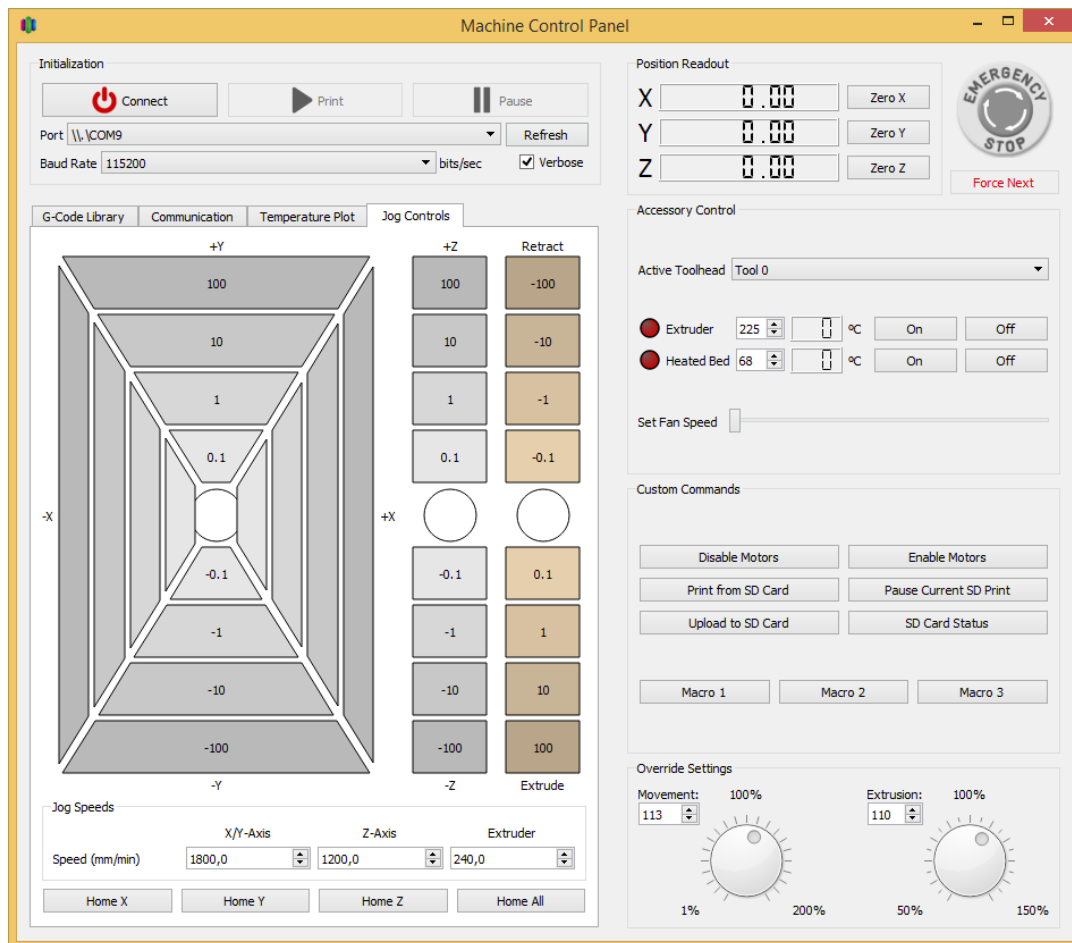


Рисунок 2.16 – Налаштування параметрів друку

На рис. 2.17 наведено результат виготовлення частин фізичного прототипу механізму подачі дроту.



Рисунок 2.17 – Роздруковані деталі механізму подачі дроту

2.6 Висновки до розділу 2

В даному розділі відповідно до мети та завдань роботи, виконано дослідження сучасних технологій цифрової індустрії для прототипування технологічних систем. Розроблено методику прототипування на основі ЦД.

На практиці розроблено віртуальну та фізичну частини ЦД вузла ТСЗД з використанням технологій віртуальної, доповненої реальності та адитивного виробництва. Схематичне представлення взаємодії віртуальної та фізичної складових ЦД механізму подачі дроту наведено на рис. 2.18.

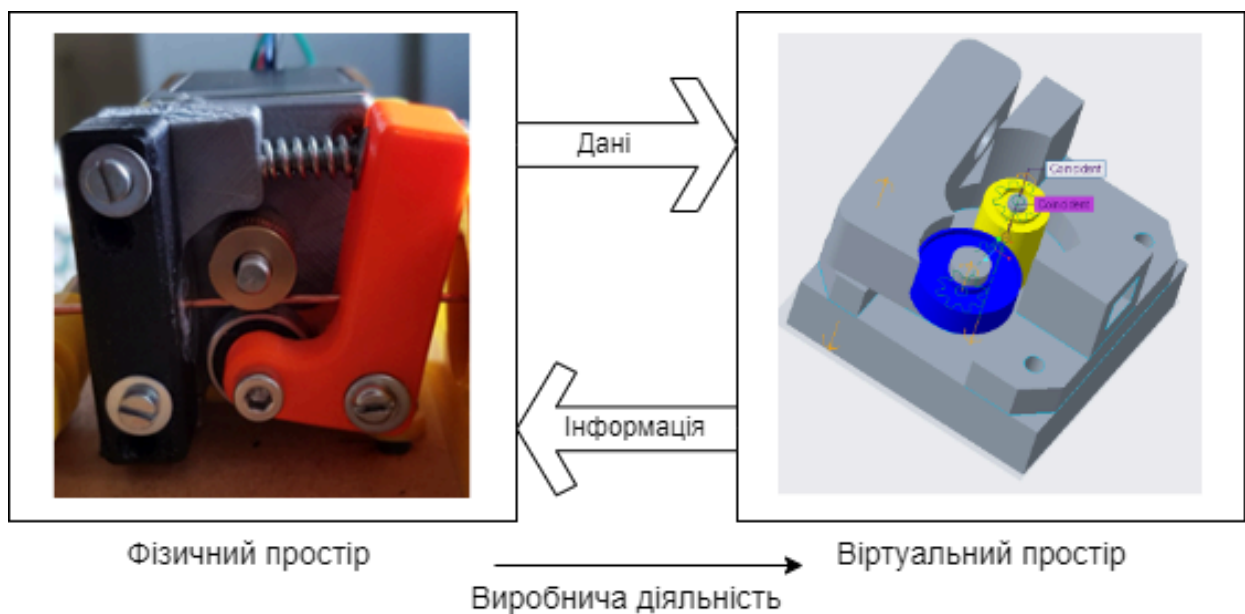


Рисунок 2.18 – Взаємодія складових ЦД механізму подачі дроту

Отже, ЦД – це набір віртуальних інформаційних конструкцій, які повністю описують потенційний або фактичний фізично виготовлений продукт. В оптимальному випадку будь-яка інформація, яку можна отримати під час перевірки виготовленого фізичного продукту, може бути створена до фізичного продукту.

Завдяки створенню ЦД можна зрозуміти, як покращити дизайн і роботу, підвищити ефективність або виявити проблему, перш ніж вона станеться.

3 ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ КЕРУВАННЯ ПРОТОТИПОМ ТСЗД

3.1 Призначення та вимоги до програмного забезпечення

Проєктоване програмне забезпечення (ПЗ) призначене для управління прототипом ТСЗД, який надає можливість згинати з дроту різні геометричні форми та представляє собою керуючий код для крокових двигунів.

Числове програмне управління здійснюється шляхом взаємодії користувача з контролером прототипу через графічний інтерфейс для забезпечення вибору геометричної форми та її параметрів.

Можна виділити дві основні групи користувачів розроблюваного ПЗ:

- користувач, що планує використовувати прототип ТСЗД з метою вивчення особливостей технологічного процесу згинання дроту;
- програміст, який займається розробками в галузі ЧПУ та може використовувати даний прототипу для дослідження та удосконалення алгоритмів управління.

Вимоги до розроблюваного ПЗ наступні:

Функціональні вимоги:

- взаємодіяти з контролером Arduino через COM порт передачі даних;
- формувати керуючі впливи на двигуни прототипу: двигун подачі подає дріт в машину, двигун згинання згинає дріт у 2D-форми, двигун осі Z дозволяє створювати тривимірні форми;
- розпізнавання та виконання команд користувача щодо вибору геометричної форми та введення її параметрів.

Нефункціональні вимоги:

- графічний інтерфейс має бути наочним і зрозумілим;
- ПЗ має бути представлене у вигляді програмного застосунку;
- програмне забезпечення має дозволяти резервну взаємодію з прототипом через монітор послідовного порту.

Діаграму прецедентів, що ілюструє процес взаємодії користувача та програміста з розроблюваним ПЗ представлено на рисунку 3.1.

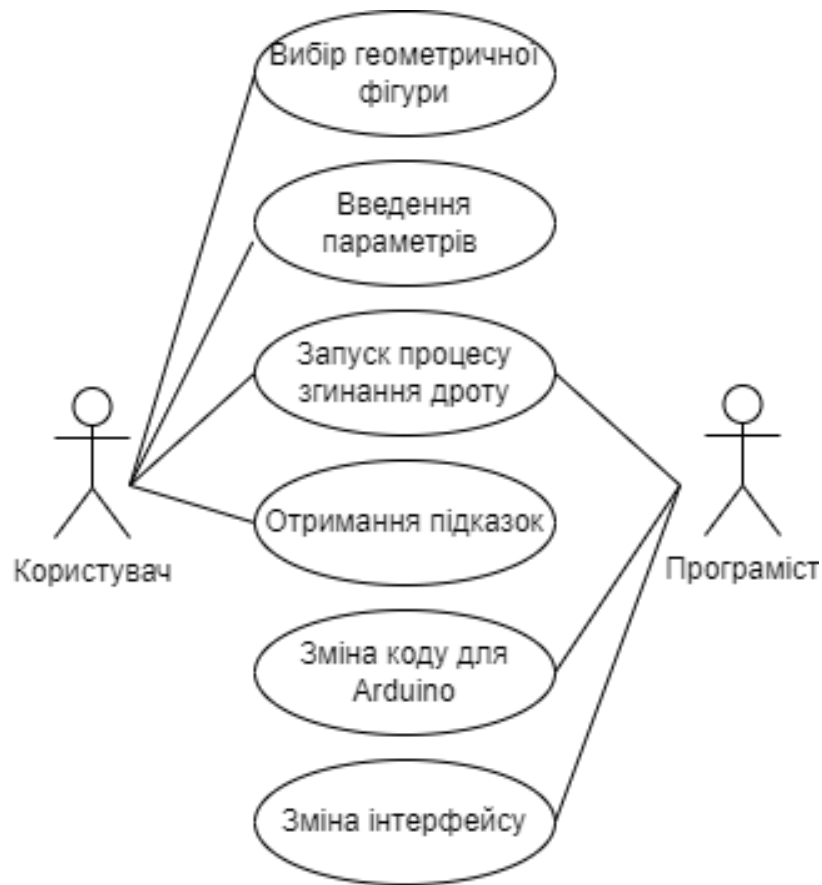


Рисунок 3.1 – Діаграма прецедентів

3.2 Розробка архітектури системи

Узагальнену архітектуру системи наведено на рис.3.2

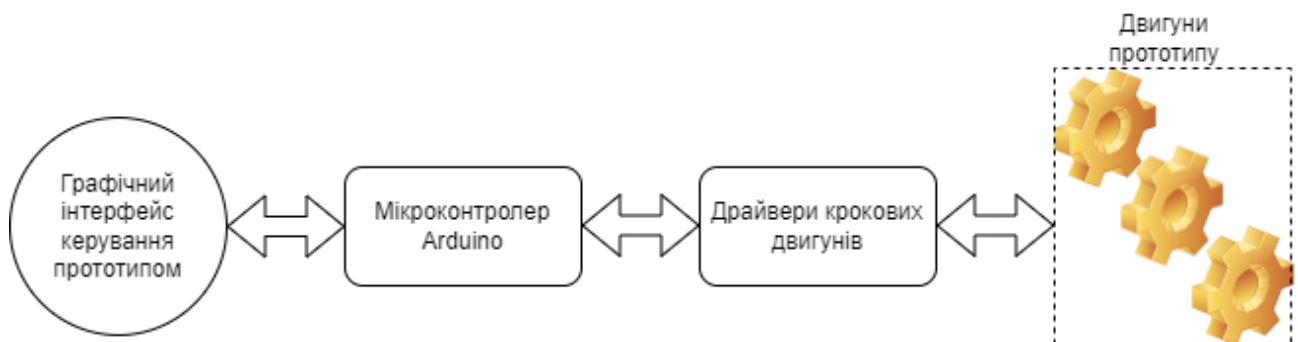


Рисунок 3.2 – Узагальнена архітектура системи

Графічний інтерфейс керування прототипом ТСЗД надає можливість вибору геометричної форми (квадрат, трикутник тощо), введення її параметрів (довжини сторони) та запуску технологічного процесу згинання дроту.

Відправка інформації в мікроконтролер Arduino відбувається через послідовний інтерфейс стандарту RS-232 комп'ютера.

Управління кроковим двигуном через Arduino здійснюється шляхом подачі імпульсів на обмотки двигуна у певній послідовності. Для полегшення керування кроковим двигуном існують спеціальні бібліотеки `stepper.h` та `accelstepper.h`, але можна обертати вал двигуна без стандартних бібліотек.

Драйвер крокового двигуна DRV8825 отримує на вході логічні сигнали STEP/DIR, які, як правило, представлені високим і низьким рівнем опорної напруги 5 В, та відповідно до отриманих сигналів змінює струм в обмотках двигуна, змушуючи вал повертатися у відповідному напрямку на заданий кут.

Модуль драйвера крокового двигуна має вбудований транслятор (перетворювач), що дозволяє йому керувати швидкістю і напрямом обертання крокових двигунів.

Крокові двигуни – це різновид двигунів постійного струму, в яких рух ротора відбувається невеликими дискретними кроками. В даний час крокові двигуни знаходять дуже широке застосування – починаючи від камер спостереження та закінчуючи складними роботами та механізмами.

Кроковий двигун (`stepper motor`) призначений для точного позиціонування чи переміщення об'єкта на задану кількість кроків валу.

На відміну від сервоприводу кроковий двигун не має зворотного зв'язку по куту повороту (не знає, де зараз знаходиться вал), проте може обертатися без обмежень.

Двигун NEMA17 сконструйований таким чином, що робить 200 кроків за оборот у стандартному повнокроковому режимі. Кожен крок повертає ротор на 1,8 градуси.

Крокові двигуни прототипу ТСЗД забезпечують роботу механізмів, що відповідають за подачу та згинання дроту у задану геометричну форму.

3.3 Вибір інструментарію програмної реалізації

Вибір інструментарію програмної розробки має бути виконано для двох задач – розробки графічного інтерфейсу для керування прототипом та розробки програми для мікроконтролера Arduino.

Processing – це безкоштовне середовище, що працює на платформах GNU/Linux, Mac OS X та Windows. Дозволяє читати передані через послідовний порт дані, а також створити простий графічний інтерфейс користувача (Graphical User Interface, GUI) для відправки команд на плату Arduino [41].

Processing - це відкрита мова програмування, заснована на Java з простим і зрозумілим сі-подібним синтаксисом. Є легким і швидким інструментарієм для програмування зображення, анімацію та інтерфейси. Дозволяє створювати 3D-аплікації (у тому числі і ігри), адже Processing має засоби підтримки OpenGL. Надає велику кількість функцій, логічний синтаксис та інструменти для побудови графічних примітивів, 3D-об'єктів, робота зі світлом, текстом, інструментами трансформації. Також дозволяє імпортувати та експортувати файли аудіо/відео/звукових форматів, обробляти події миші/клавіатури, працювати зі сторонніми бібліотеками (OpenGL, PDF, DXF), працювати в мережі [41].

Мова програмування Processing подібна до мови Arduino, але мова Arduino (на основі Wiring) реалізована на C/C++ та тому відрізняється від мови Processing, яка заснована на Java [42].

Альтернативами Processing є Vvvv, OpenFrameworks, Nodebox, Processing.js і Vuo [43].

Для розробки програм для Arduino традиційно використовується Arduino IDE – це програмне забезпечення з відкритим кодом, яке використовується для програмування плат Arduino, і є інтегрованим середовищем розробки, розробленим arduino.cc. Дозволяє писати та завантажувати код на плати Arduino та складається з багатьох бібліотек і набору прикладів міні-проектів. Програмне

забезпечення Arduino (IDE) сумісне з різними операційними системами (Windows, Linux, Mac OS X) і підтримує мови програмування (C/C++).

Альтернативами Arduino IDE є VScode, PlatformIO, React Studio, Ktechlab, B4R, плагін Arduino Eclipse та Atmel Studio. Це переважно інтегровані середовища розробки, але також можуть бути редактори коду або інструменти автоматизації електронного проєктування [44].

Середовище програмування Processing схоже із середовищем програмування Arduino IDE, це надає додаткові переваги при використанні та обґрунтовує їх вибір при розробці.

Порівняльну характеристику двох середовищ наведено в таблиці 3.1.

Таблиця 3.1 – Порівняльна характеристика середовищ Arduino IDE та Processing

Характеристики	Середовище розробки	
	Arduino IDE	Processing
Мова програмування	C / Wiring	Processing, /Java
Підтримка	+	+
Простота	+	+
Доступність	Open-source	Open-source

3.4 Висновки до розділу 3

В цьому розділі було проведено аналіз вимог до розроблюваного ПЗ, розроблено та описано архітектуру програмної системи. Розроблено діаграму прецедентів, що демонструє варіанти взаємодії користувача та програміста з розроблюваною програмною системою.

На основі порівняльного аналізу альтернативних варіантів середовищ розробки керуючої програми для та графічного інтерфейсу обґрунтовано вибір програмних засобів розробки.

4 КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ КЕРУВАННЯ ПРОТОТИПОМ ТСЗД

Розроблюване ПЗ складається з двох частин: програми для Arduino, що виконує керування кроковими двигунами та програми для ПК, що забезпечує інтерфейс між користувачем та прототипом ТСЗД.

4.1 Розробка програмного забезпечення для Arduino

Перед початком розробки ПЗ для усіх крокових двигунів, що були підключені до шилда CNC Shield V 4 було визначено контакти dir і step. Вивід dir визначає напрямок руху двигуна (НИЗЬКИЙ для прямого напрямку, ВИСОКИЙ для зворотного напрямку). Вивід step змушує двигун рухатися, коли значення потужності ВИСОКЕ, і зупиняє його, коли значення НИЗЬКЕ. Відповідно контакти dir і step (напрямок і крок) встановлюються так: 3, 6 - для двигуна згинання; 5, 7 - для двигуна подачі; 2, 4 — для крокового двигуна по осі Z.

Тестова версія V1 (Додаток А.1) керуючого коду дала можливість перевірити та дослідити поведінку кожного двигуна (основна ідея коду була взята з [16]). Програма просто змусила двигуни обертатися по-різному з різною швидкістю під різними кутами. Кроковий двигун повинен зробити 200 кроків, щоб завершити обертання на 360 градусів, тому за допомогою пропорцій можна визначити користувальницькі кути обертання та перевести їх у кроки. Кожна операція обертання реалізується просто за допомогою циклу з кількістю ітерацій, що дорівнює вказаному номеру кроку. Тестування програмного коду версії V1 показало, що двигун осі Z потребує занадто багато енергії для належної роботи, що погіршує та порушує роботу системи в цілому. Тому було вирішено більше не використовувати кроковий двигун вісі Z і натомість зосередитися на використанні двигунів подачі та згинання для створення простих двовимірних форм.

Розроблені дві остаточні версії коду V2 і V3 (Додаток А.2, А.3) забезпечують роботу двох основних двигунів —подачі та згинання, без використання крокового двигуна осі Z і серводвигуна (оскільки виникли проблеми, які були описані раніше).

Програмний код V2 призначений для безпосереднього прошивання в контролер Arduino та використання без застосунку, що обмежує можливості машини, але може бути зручним в певних ситуаціях. Код реалізує згинання дроту у формі квадрата.

Версія коду V3 дозволяє користувачеві встановлювати довжину сторони фігури, яка згинається, в версії V2 це значення визначається як константа, що відповідає 600 крокам. Це значення в кінцевому підсумку визначає роботу двигуна подачі, який під час процесу згинання дроту має обертатися щоразу перед фактичним згинанням плюс один раз наприкінці процесу, щоб завершити форму.

Що стосується двигуна згинання, то кількість кроків, які він робить, визначається типом форми, яку користувач хоче зігнути (було вирішено програмно реалізувати дві форми — квадрат і трикутник). Крім того, оскільки в системі задіяно кілька передач, кут повороту двигуна не збігається з кутом повороту кінцевої шестерні зі штифтом, відповідальним за згинання дроту. Таким чином, кути кожного повороту для згинання дроту в форми (90 градусів для квадрата і 120 градусів для трикутника) були перераховані з урахуванням коефіцієнта передачі кута між шестернями.

Оскільки кожна операція згинання запускається після попередньої подачі дроту, логіка полягає в тому, що двигун згинального пристрою повинен зробити певну кількість кроків в одному напрямку, а потім повернутися на таку ж кількість кроків у зворотному напрямку. Потім повторюється подача і згинання дроту. Кількість таких ітерацій дорівнює 2 для трикутника і 3 для квадрата.

Узагальнену схему алгоритму функціонування програми наведено на рисунку 4.1. Опис блоків алгоритму наступний.

Блок 1. Початок.

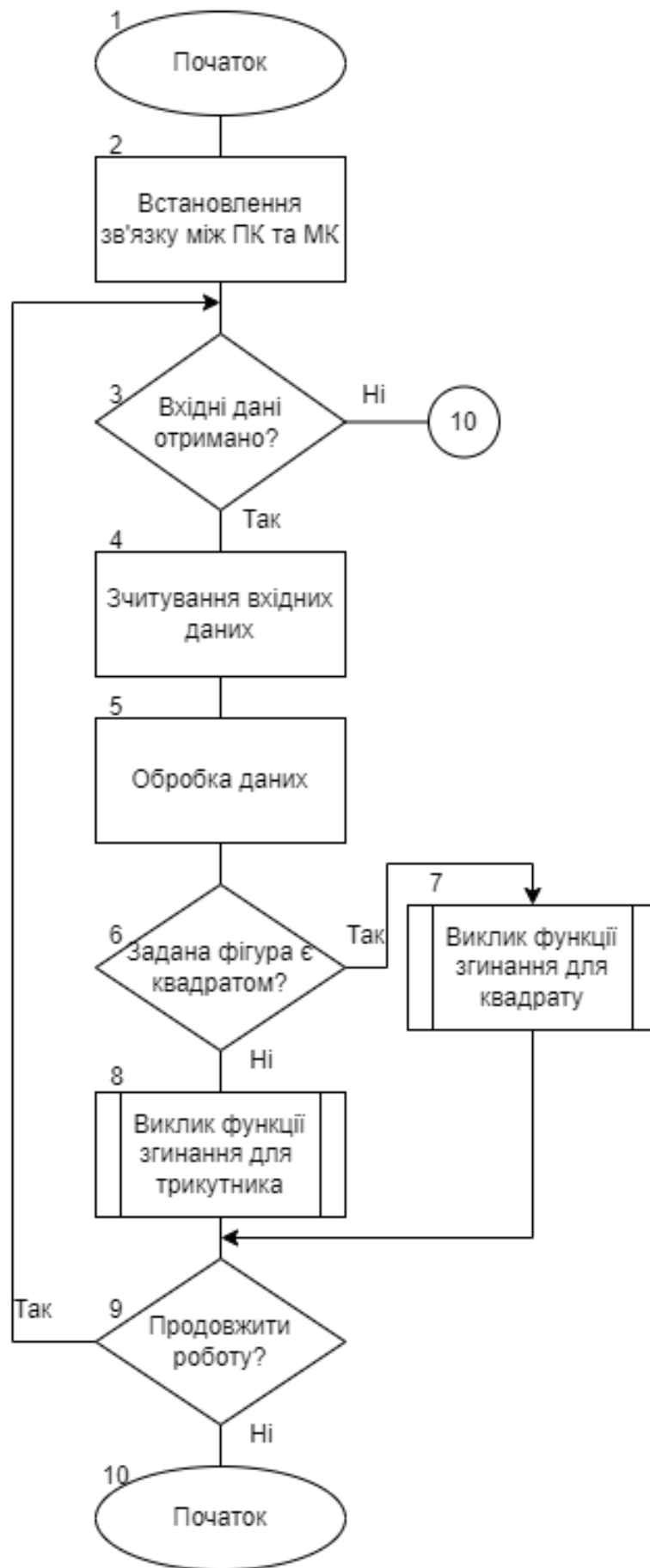


Рисунок 4.1 – Узагальнена схема алгоритму роботи програми для Arduino

Блок 2. Встановлення зв'язку між ПК та МК. Відкриття послідовного порту (СОМ порту ПК).

Блок 3. Вхідні дані отримано? Перевірка наявності в буфері послідовного порту вхідних даних, введених користувачем (форма фігури та довжина її сторони).

Блок 4. Зчитування вхідних даних. Зчитування величин, що вказують тип обраної фігури та довжину її сторони.

Блок 5. Обробка вхідних даних. Переведення довжини сторони фігури у кількість кроків двигуна.

Блок 6. Задана фігура є квадратом? Якщо Так, то перехід на Блок 7. Інакше, на Блок 8.

Блок 7. Виклик функції для згинання квадрату. Функція реалізує формування керуючого впливу на механізми прототипу ТСЗД для згинання дроту у формі квадрату.

Блок 8. Виклик функції для згинання трикутника. Функція реалізує формування керуючого впливу на механізми прототипу ТСЗД для згинання дроту у формі трикутника.

Блок 9. Продовжити роботу? Перевірка необхідності продовжити роботу прототипу. Якщо Так, то перехід на Блок 3. Якщо Ні, то перехід на Блок 10.

Блок 10. Кінець. Вимкнення живлення.

Схему алгоритму функції формування фігури наведено на рисунку 4.2.

Опис блоків алгоритму наступний.

Блок 1. Початок. Початок функції, що реалізує згинання дроту у формі, що була обраною користувачем.

Блок 2. Цикл з $n = 1$ по iterations, де iterations – номер ітерацій згинання для реалізації фігури заданої користувачем форми.

Блок 3. Формування керуючого впливу на двигун подачі. Подача дроту на відстань, що відповідає заданій користувачем довжині сторони фігури.

Блок 4. Формування керуючого впливу на двигун згинання. Обертання мотора згинання в прямому напрямку на величину, що відповідає значенню кута

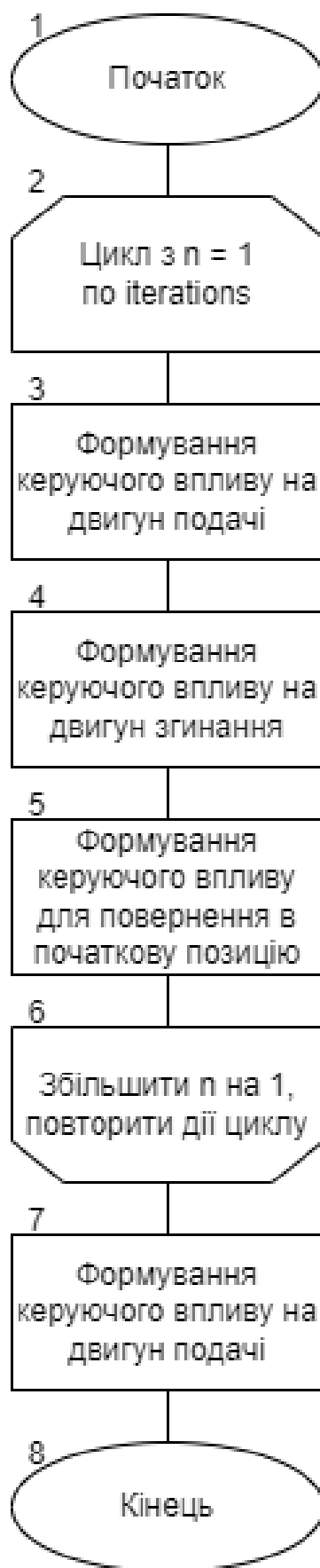


Рисунок 4.2 – Схема алгоритму функції формування фігури

90°, якщо обрана фігура квадрат, або 120°, якщо обрана фігура трикутник (власне згинання дроту).

Блок 5. Формування керуючого впливу для повернення двигуна обертання в початкову позицію. Обертання мотора згинання на таку саму величину в іншому напрямку.

Блок 6. Збільшити n на 1, повторити дії циклу.

Блок 7. Формування керуючого впливу на двигун подачі. Подача дроту на відстань, що відповідає заданій користувачем довжині сторони фігури.

Блок 8. Кінець.

У таблиці 4.1 наведено опис функцій програм, а у таблиці 4.2 – опис глобальних змінних.

Таблиця 4.1 – Функції програми для Arduino

Назва функції Arduino	Опис функції
1	2
void setup()	Одна з основних функцій, що виконується лише один раз на початку роботи програми
Serial.begin(9600)	Відкриває послідовний порт, встановлює швидкість 9600 біт/с
pinMode(dirPin_feeder, OUTPUT)	Встановлює режим роботи – вихід для вказаного піна
void loop()	Одна з основних функцій. Виконується циклічно до закриття вікна
Serial.available()	Повертає кількість байт (символів), доступних для зчитування з буфера послідовного порту
Serial.read()	Зчитує черговий доступний байт з буфера послідовного порту
void bending(int iterations, int steps, int size)	Користувацька функція, що виконує операцію згинання дрота. Приймає у якості аргументів кількість ітерацій згинання, кроків двигуна, розмір сторони фігури

Продовження таблиці 4.1

1	2
digitalWrite(dirPin_bender, HIGH)	Подає напругу (значення HIGH) на вказаний пін
digitalWrite(stepPin_feeder, LOW)	Припиняє подачу напруги (значення LOW) на вказаний пін
delayMicroseconds(2000)	Задає затримку у мікросекундах
delay(500)	Задає затримку у мілісекундах

Таблиця 4.2 – Опис змінних програми для Arduino

Назва змінної	Опис змінної
int dirPin_bender	Константа на позначення піна, що визначає напрям мотора згинання
int stepPin_bender	Константа на позначення піна, що відповідає за кроковий рух мотора згинання
int dirPin_feeder	Константа на позначення піна, що визначає напрям мотора подачі
int stepPin_feeder	Константа на позначення піна, що відповідає за кроковий рух мотора подачі
int stepsPerRevolution	Константа на позначення потрібної кількості кроків для одного повного обороту мотора
int commandValueSize	Зберігає значення обраної довжини сторони фігури
int commandValueShape	Зберігає дані про тип обраної фігури
int iterations	Позначає кількість ітерацій виконання операції згинання дроту. Також служить позначенням типу обраної фігури
int steps	Містить кількість кроків, які необхідно виконати двигуну (відповідно до значення кута повороту, що визначається типом обраної фігури)
int size	Зберігає значення обраної довжини сторони фігури
int n	Змінна – лічильник
int i	Змінна – лічильник

4.2 Розробка програмного інтерфейсу

Інтерфейс програми було створено в середовищі Processing та організовано двома способами - за допомогою монітору послідовного порту (Додаток А.4) та у вигляді програмного застосунку (Додаток А.5).

При використанні монітору послідовного порту система надає можливість ввести код фігури (3 – трикутник, 4 – квадрат), але довжина сторони залишається константою, що задана в програмі.

При організації інтерфейсу у вигляді програмного застосунку, після вибору форми та встановлення довжини її бічної сторони користувач може запустити виконання програми, натиснувши кнопку «BEND».

Схему алгоритму функціонування програми для інтерфейсу наведено на рис. 4.3.

Опис блоків алгоритму наступний.

Блок 1. Початок.

Блок 2. Формування вікна інтерфейсу Processing. Створення графічних елементів інтерфейсу (шкала, кнопки) та текстових полів.

Блок 3. Аналіз типу події щодо вибору фігури. Можливі два типи подій - курсор миші був натиснутий у межах кнопки, що позначає квадрат, або курсор миші був натиснутий у межах кнопки, що позначає трикутник.

Блок 4. Привласнення змінній *shape* значення, що дорівнює 4 (для квадрата) або 3 (для трикутника).

Блок 5. Візуалізація вибору користувача. Відбувається у вигляді прапорця.

Блок 6. Аналіз події щодо вибору на шкалі. Користувач може обрати розмір сторони фігури в сантиметрах.

Блок 7. Привласнення значення змінній *size*. Відбувається в залежності від обраного користувачем на шкалі значення.

Блок 8. Аналіз події щодо запуску програми. Відбувається після натискання користувачем в полі кнопки «Bend».

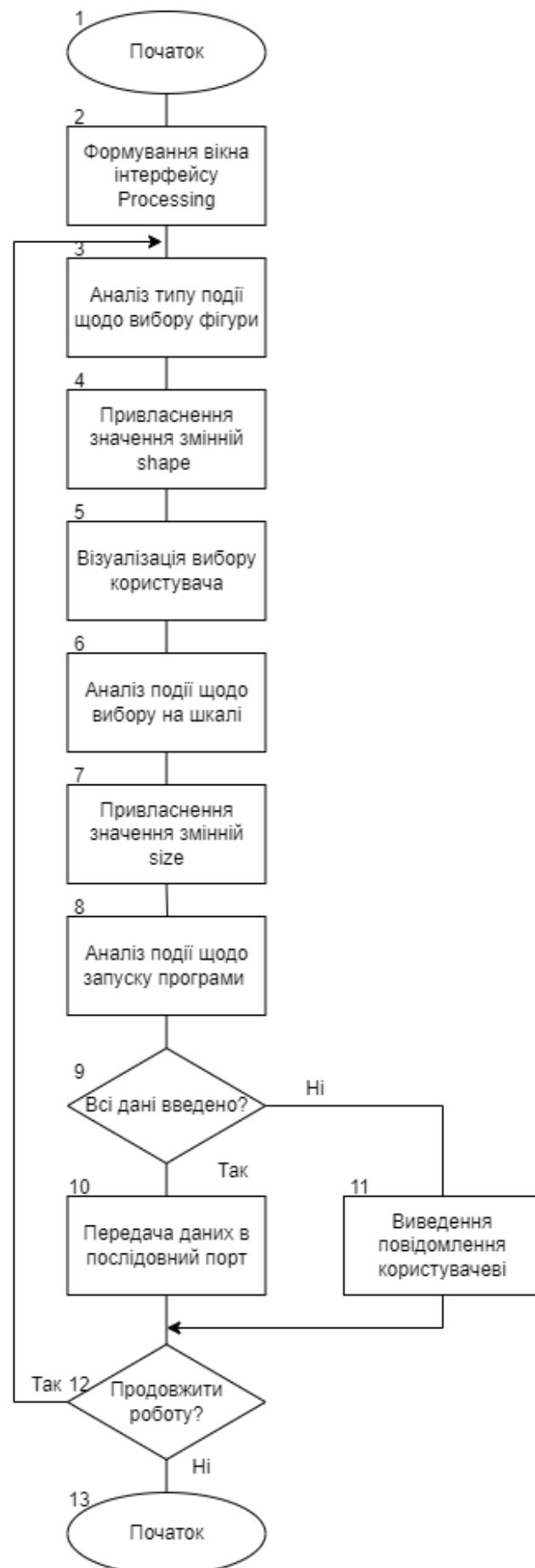


Рисунок 4.3 – Схема алгоритму функціонування програми для інтерфейсу

Блок 9. Всі дані введено? Перевірка наявності даних двох типів – форми фігури та величини її сторони. Якщо всі дані введено, то перехід на Блок 10. Інакше, на Блок 11.

Блок 10. Передача даних в послідовний порт. Дані передаються на виконання в контролер.

Блок 11. Виведення повідомлення користувачеві. Повідомлення можуть бути наступні: Оберіть довжину сторони фігури, Оберіть форму фігури, Оберіть довжину сторони та форму фігури.

Блок 12. Продовжити роботу? Якщо користувач бажає продовжити роботу, то перехід на Блок 3, інакше – на Блок 13.

Блок 13. Кінець.

При розробці була використана бібліотека `processing.serial.*` для роботи з послідовним портом.

У таблиці 4.3 наведено опис функцій програми.

Таблиця 4.3 – Функції програми для інтерфейсу

Назва функції Processing	Опис функції
1	2
<code>public void settings()</code>	Користувацька функція для задання початкових значень вікна Processing
<code>size(1380, 630)</code>	Функція задання розміру вікна Processing
<code>void setup ()</code>	Одна з основних функцій, що виконується лише один раз на початку роботи програми
<code>new Serial(this, Serial.list()[0], 9600)</code>	Відкриває послідовний порт, встановлює швидкість 9600 біт/с
<code>createFont("Georgia", 60)</code>	Створює текстовий шрифт із заданими характеристиками
<code>background(240,255,240)</code>	Задає колір, який буде використовуватися як фон у вікні Processing

Продовження таблиці 4.3

1	2
<code>square(60,180,240)</code>	Малює квадрат з заданими координатами верхнього лівого кута та довжиною окремої сторони
<code>triangle(380,180,640,180,510,420)</code>	Малює трикутник з заданими координатами усіх вершин
<code>void draw ()</code>	Одна з основних функцій. Виконується циклічно до закриття вікна
<code>serial.write(shape)</code>	Виводить значення до послідовного порту
<code>textSize(35)</code>	Задає розмір тексту, що виводиться
<code>text("Please choose the size\n"+"and the shape!",110,540)</code>	Виводить заданий набір символів у межах визначеної прямокутної області
<code>println(" ")</code>	Виводить до консолі Processing значення
<code>noStroke()</code>	Вимикає контур для елементів, що малюються
<code>rect(787,240,467,70)</code>	Малює прямокутник із заданими координатами верхнього лівого кута та довжинами сторін
<code>fill(1,190,32);</code>	Встановлює колір заповнювача
<code>stroke(1,190,32);</code>	Встановлює колір контура
<code>strokeWeight(5);</code>	Встановлює товщину контура
<code>line(mouseX,250,mouseX,298);</code>	Малює лінію з вказаними координатами початкової та кінцевої точки
<code>boolean isMouseInTriangle(float x1, float y1, float x2, float y2, float x3, float y3, float mouseX, float mouseY);</code>	Користувачка функція для перевірки знаходження курсора в межах намальованого трикутника (приймає у якості аргументів координати вершин трикутника та координати знаходження курсора)
<code>float triangleArea(float x1, float y1, float x2, float y2, float x3, float y3);</code>	Функція обчислення площі окремого трикутника (приймає у якості аргументів координати вершин)

У таблиці 4.4 наведено опис глобальних змінних.

Таблиця 4.4 – Опис змінних програми для Processing

Назва змінної коду Processing	Опис змінної
1	2
Serial serial	Змінна для роботи з монітором послідовного порту
PFont f	Змінна для опису шрифту тексту
int shape	Змінна зберігає дані про тип обраної фігури
int size	Змінна зберігає значення обраної довжини сторони фігури
mousePressed	Вбудована змінна. Зберігає інформацію про те, чи була натиснута кнопка миші
mouseY	Вбудована змінна. Містить поточну вертикальну координату миші
mouseX	Вбудована змінна. Містить поточну горизонтальну координату миші
int size_prev	Містить попереднє значення довжини сторони фігури
int i	Змінна – лічильник
float x1	Містить значення горизонтальної координати першої вершини трикутника
float y1	Містить значення вертикальної координати першої вершини трикутника
float x2	Містить значення горизонтальної координати другої вершини трикутника
float y2	Містить значення вертикальної координати другої вершини трикутника
float x3	Містить значення горизонтальної координати третьої вершини трикутника
float y3	Містить значення вертикальної координати третьої вершини трикутника
float fullArea	Містить значення площі трикутника-кнопки, що позначає форму фігури для вибору

Продовження таблиці 4.4

1	2
float area1	Містить значення площі першої утвореної трикутної області між двома вершинами трикутника-кнопки та курсором миші
float area2	Містить значення площі другої утвореної трикутної області між двома вершинами трикутника-кнопки та курсором миші
float area3	Містить значення площі третьої утвореної трикутної області між двома вершинами трикутника-кнопки та курсором миші

4.3 Інструкція з використання програмного забезпечення

Основну папку, яка містить усі важливі файли проєкту можна побачити відкритою на рис. 4.4.

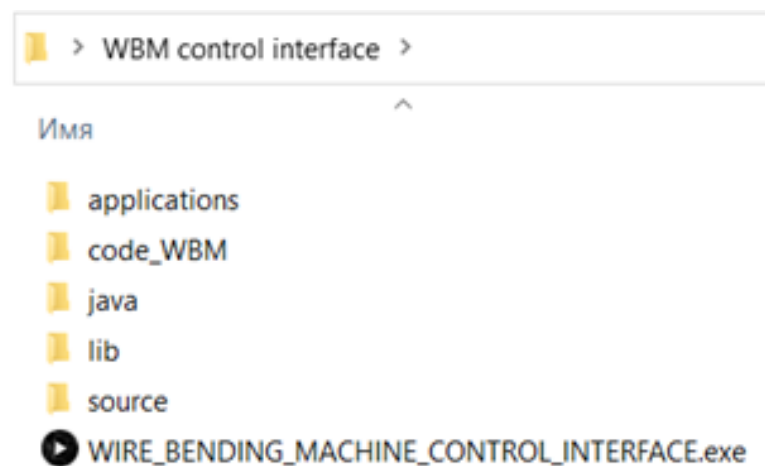


Рисунок 4.4 – Вигляд папки інтерфейсу керування WBM

Запуск файлу WIRE_BENDING_MACHINE_CONTROL_INTERFACE.exe відкриє програму інтерфейсу, де можна встановити параметри згинання та запустити процес згинання (рис. 4.5).

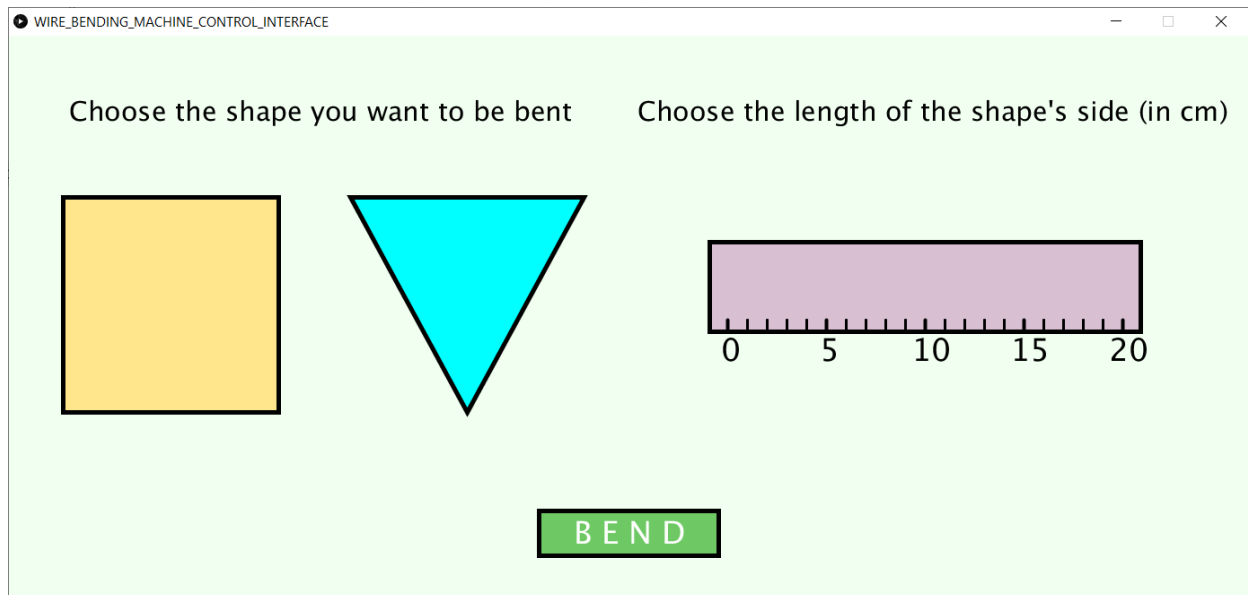


Рисунок 4.5 – Застосунок для інтерфейсу прототипу машини для згинання дроту

Зауважте, що програму слід запускати лише після підключення комп'ютера через COM-порт до Arduino Nano, інакше запуск призведе до того, що програма не буде працювати або працюватиме неправильно.

Ще одна деталь, яку слід враховувати, встановлюючи розмір сторони фігури в додатку, - це відстань між з'єднувачем трубки, з якої виходить дріт, і штифтом для згинання. Розташування ковпачка зі шпилькою таке, що при виборі розмірів сторін від 1 до 4 сантиметрів шпилька може не дістати до дроту або погано її зігнути, тому рекомендується вибирати значення більше зазначених. Після встановлення всіх параметрів натисніть кнопку «BEND». Після сеансу згинання ви завжди можете вибрати нові параметри або повторити той самий процес згинання.

Якщо відкрити папку code_WBM, ви перейдете до однойменного файлу, який містить основний код, попередньо завантажений на Arduino Nano (рис. 4.6).

Якщо ви хочете застосувати будь-які зміни та змінити роботу WBM, це слід зробити через середовище Arduino IDE (рис. 4.7), яке є програмою з відкритим кодом, яку можна встановити з Інтернету. Після встановлення Arduino IDE, відкрийте code_WBM через це середовище.

WBM control interface > code_WBM			
Имя	Дата изменения	Тип	Размер
 code_WBM.ino	14.07.2023 13:18	Документ Adobe Acrobat	2 КБ

Рисунок 4.6 – Основний файл керуючого коду машини для згинання дроту



Рисунок 4.7 – Arduino IDE середовище для редагування та завантаження програмного коду в контролер Arduino

Завершивши редагування коду, підключіть комп’ютер до Arduino та правильно внесіть налаштування в програмі: натисніть «Tools» (рис. 4.8); потім виберіть правильний СОМ-порт (який відповідає тому, до якого ви збираєтеся підключити Arduino) у меню «Port»; після цього виберіть плату Arduino Nano в

меню «Board» (рис. 4.9); нарешті натисніть кнопку  , щоб перевірити код, і натисніть кнопку  , щоб завантажити його в контролер Arduino.

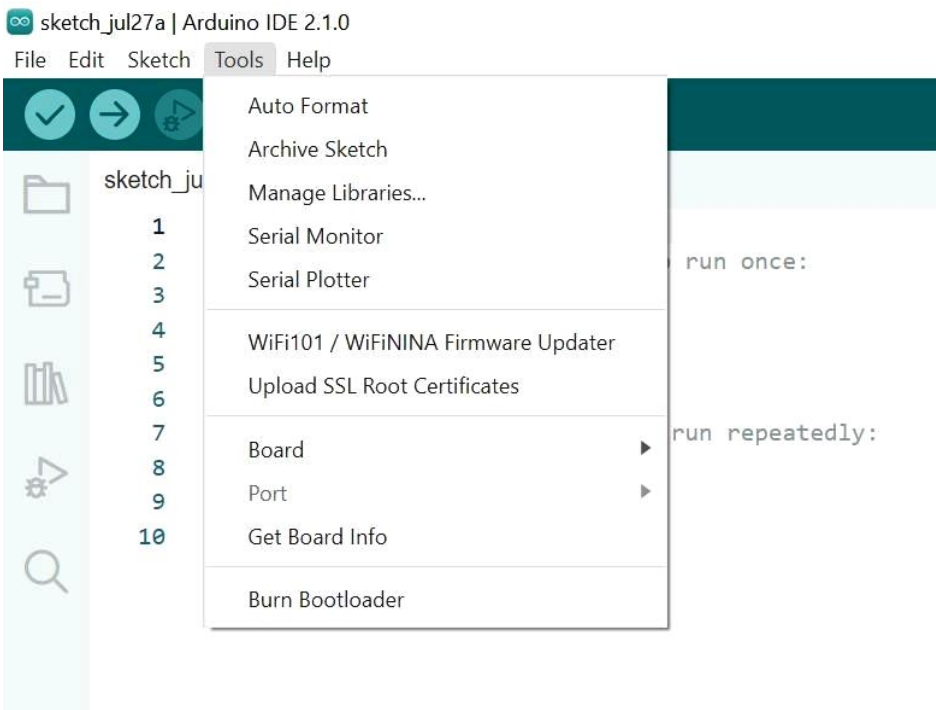


Рисунок 4.8 – Встановлення інформації про порт і плату (крок 1)

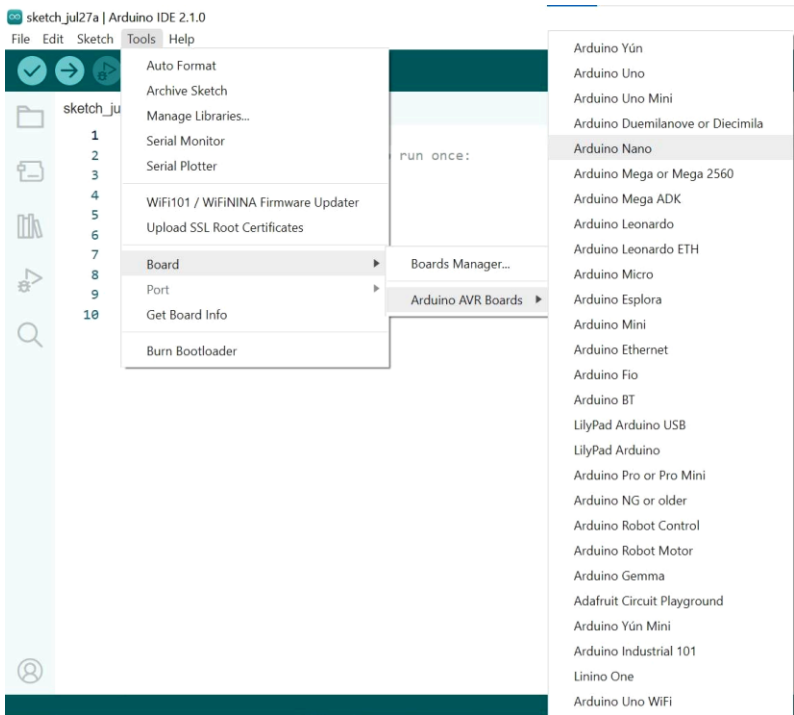
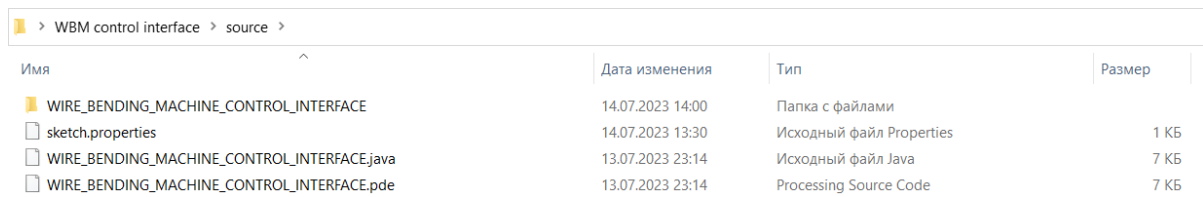


Рисунок 4.9 – Встановлення інформації про порт і плату (крок 2)

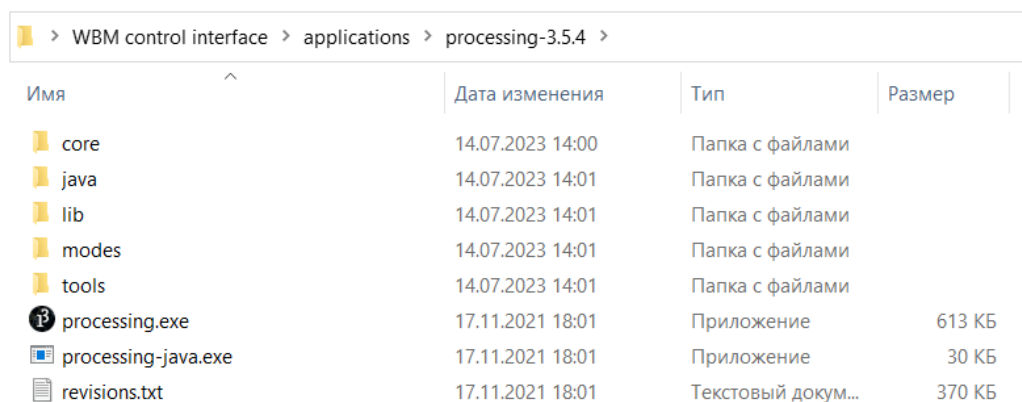
Код потрібно завантажити лише один раз після того, як до нього були застосовані зміни, і потім він зберігається на Arduino до завантаження наступної версії коду. Після цього застосунок Arduino IDE слід закрити, щоб не конфліктувати з іншими програмами. Відкриваючи вихідну папку, ви перейдете до файлів інтерфейсу керування. Файл `WIRE_BENDING_MACHINE_CONTROL_INTERFACE.pde` містить код інтерфейсу керування, створеного в Processing (рис. 4.10).



Имя	Дата изменения	Тип	Размер
WIRE_BENDING_MACHINE_CONTROL_INTERFACE	14.07.2023 14:00	Папка с файлами	
sketch.properties	14.07.2023 13:30	Исходный файл Properties	1 КБ
WIRE_BENDING_MACHINE_CONTROL_INTERFACE.java	13.07.2023 23:14	Исходный файл Java	7 КБ
WIRE_BENDING_MACHINE_CONTROL_INTERFACE.pde	13.07.2023 23:14	Processing Source Code	7 КБ

Рисунок 4.10 – Вихідні файли інтерфейсу керування прототипом

Якщо ви хочете змінити деякі деталі в ньому (що вимагається в деяких випадках після глобальної зміни основного коду Arduino), вам слід скористатися програмою з відкритим вихідним кодом Processing, яку можна знайти, дотримуючись шляху: applications → processing-3.5.4 (рис. 4.11).



Имя	Дата изменения	Тип	Размер
core	14.07.2023 14:00	Папка с файлами	
java	14.07.2023 14:01	Папка с файлами	
lib	14.07.2023 14:01	Папка с файлами	
modes	14.07.2023 14:01	Папка с файлами	
tools	14.07.2023 14:01	Папка с файлами	
processing.exe	17.11.2021 18:01	Приложение	613 КБ
processing-java.exe	17.11.2021 18:01	Приложение	30 КБ
revisions.txt	17.11.2021 18:01	Текстовый докум...	370 КБ

Рисунок 4.11 – Файли програми Processing для редагування коду інтерфейсу керування

Можливо запустити програму Processing, натиснувши на файл `processing.exe` (рис. 4.12).

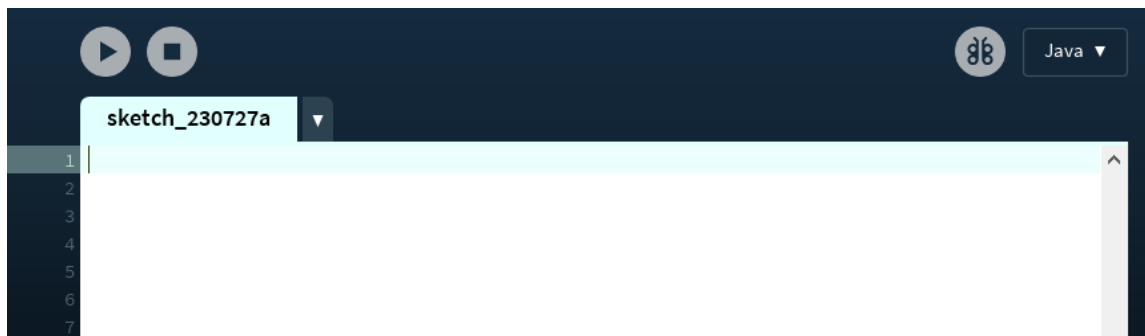


Рисунок 4.12 – Програмний застосунок Processing

Відкрийте `WIRE_BENDING_MACHINE_CONTROL_INTERFACE.pde` через Processing та застосуйте зміни.

Зауважте, що вам не потрібно нічого завантажувати в Arduino Nano після застосування змін до інтерфейсу керування обробкою, оскільки лише основний програмний код (файл `code_WBM.ino`), створений у Arduino IDE, завантажується в Arduino.

Що стосується інтерфейсу керування (файл `WIRE_BENDING_MACHINE_CONTROL_INTERFACE.pde`), то він має бути розташований на вашому комп'ютері, щоб він міг взаємодіяти з Arduino через COM-порт.

Просто запустіть новий інтерфейс керування, натиснувши кнопку «run» (рис. 4.12).



Ви також можете створити програму для інтерфейсу керування (подібну до тієї, що була продемонстрована на рис. 4.5), експортувавши проєкт (рис. 4.13) з вказаними налаштуваннями (рис. 4.14).

Зауважте, що це може знадобитися, якщо ви хочете надати користувачеві готовий програмний застосунок, який можна використовувати без Processing. В іншому випадку ви можете запустити виконуваний файл безпосередньо з середовища Processing.

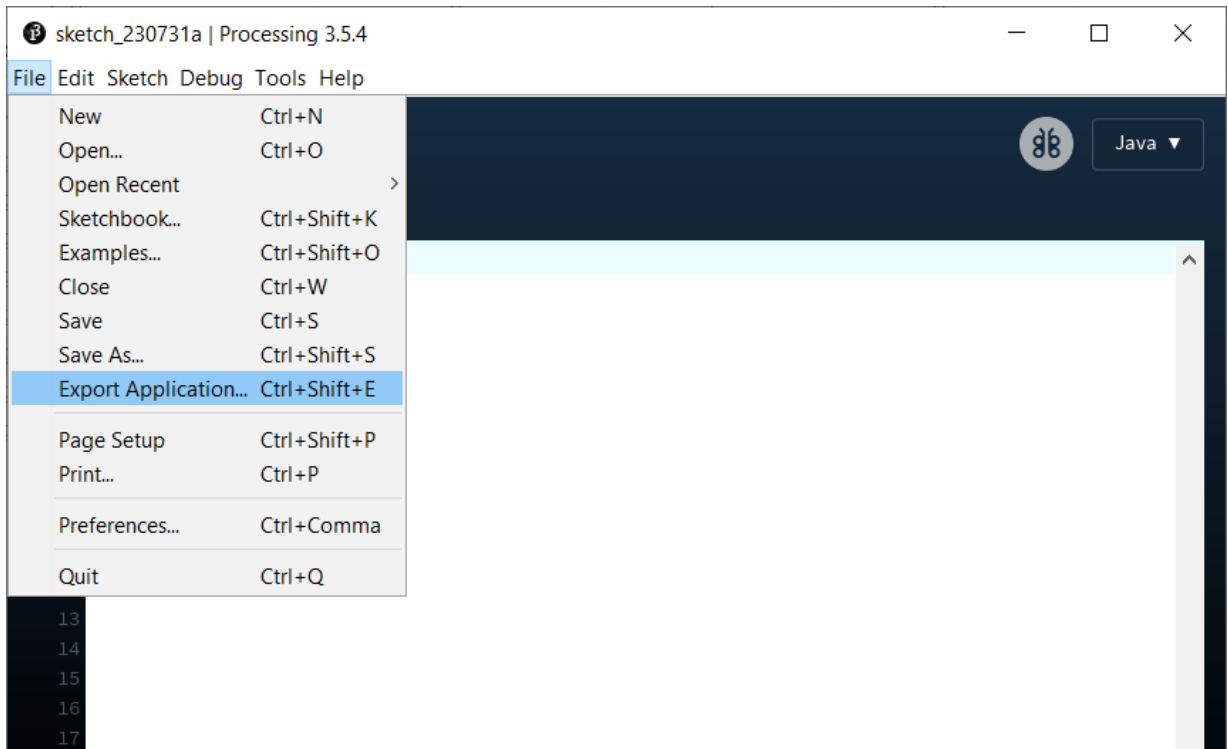


Рисунок 4.13 – Экспорт застосунку в Processing

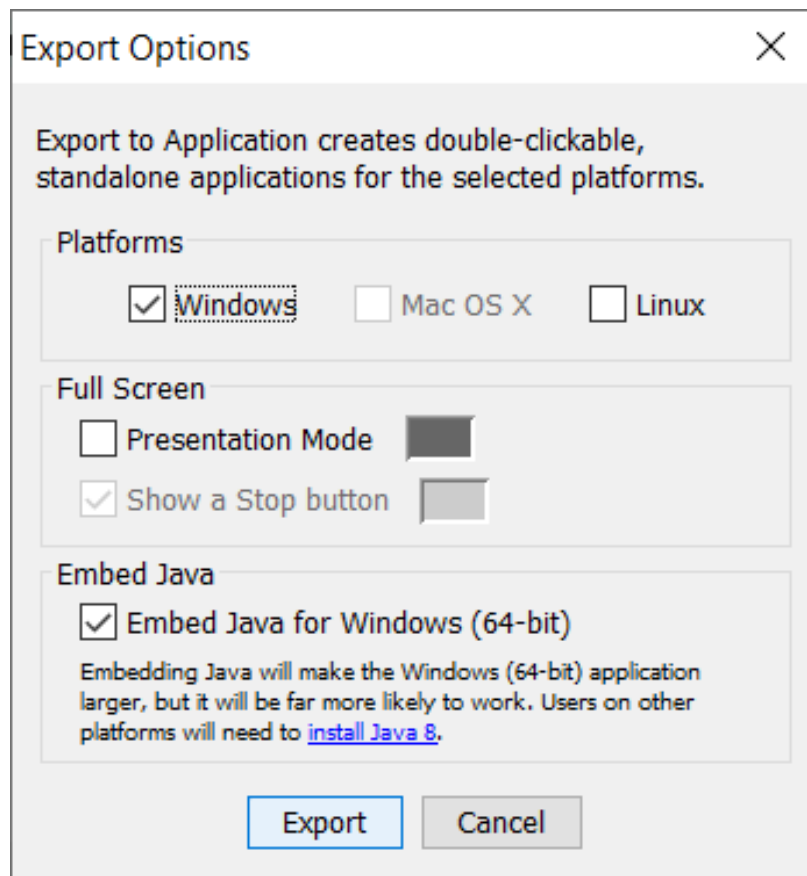


Рисунок 4.14 – Опції експорту в Processing

4.4 Висновки до розділу 4

В даному розділі відповідно до мети та завдань роботи виконано розробку програмного забезпечення для числового програмного управління прототипом ТСЗД.

Тестування ПЗ підтвердило його працездатність.

Створений програмний застосунок забезпечує інтуїтивно зрозумілу роботу користувача з програмою та контроль вхідних даних.

Програма є відкритою для внесення змін програмістом.

ВИСНОВКИ

В ході виконання магістерської роботи було проаналізовано структурні та функціональні особливості ТСЗД, а також проблематика ЧПУ таких систем. Показано, що актуальною є задача прототипування ТСЗД для пошуку шляхів їх модернізації, а також удосконалення алгоритмів управління.

Дослідження смарт-технологій цифрової індустрії дозволило виявити сучасну тенденцію розробки ЦД з використанням технологій доповненої та віртуальної реальності, адитивного виробництва та Інтернету речей. В роботі показано, що системи доповненої реальності є новою формою людино-машинної взаємодії, а 3-D друк - вдосконаленою технологією передачі цифрових інструкцій у фізичний світ.

Наукова новизна роботи полягає в тому, що розроблено методику прототипування технологічної системи та фреймворк дизайну на основі ЦД, що відрізняються від існуючих синергією пов'язаних віртуальних та фізичних складових, що дозволяє активно коригувати поведінку фізичного прототипу в реальному часі відповідно до рекомендацій, наданих віртуальною моделлю, а з іншого боку - зробити віртуальну модель більш відповідною реальному стану фізичного прототипу.

Практична цінність роботи полягає в тому, що використання розробленого апаратного та програмного забезпечення при прототипуванні ТСЗД дозволить виявити шляхи її модернізації, а також удосконалити алгоритми управління.

Тестування роботи прототипу ТСЗД з використанням реалізованого програмного коду та інтерфейсу керування, показало, що система функціонує правильно та дозволяє користувачеві згинати прості 2D форми.

У майбутньому планується встановити зв'язок між віртуальною та фізичною складовими ЦД для урахування реального функціонування при модернізації прототипу системи, щоб він став більш надійним і професійним, а також міг згинати більш складні 3D-форми.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. McEwen, A. Designing the Internet of Things / A. McEwen, H. Cassimally. – Wiley, 2014. – 338 p.
2. Arduino 3D wire bending machine / [S. Yadav, K. Karpe, G. Shinde et al.] // International journal of advance scientific research and engineering trends. – 2020. – Vol. 5, № 6. – P. 70 – 74.
3. CAD/CAM&Rapid prototyping application evaluation. [Electronic resource] / A. Boboulos. – Access mode: bookboon.com/en.
4. Gibson, I. Additive manufacturing technologies. 3D printing, rapid prototyping, and direct digital manufacturing / I. Gibson, D. W. Rosen, B. Stucker. – Springer, 2015. – 475 p.
5. Kumar, J. 3D printing and additive manufacturing technologies /J. Kumar, P. M. Pandey, D. I. Wimpenny.– Springer, 2019. – 325 p.
6. Research on key technology of wire-bending and equipment development [W.Tang, H. Zhu, M. Zhu et al.] // Management science and engineering. – 2016. – Vol. 10, № 1. – P.14-20.
7. Taiwan metiz alliance [Electronic resource]. – Access mode: https://www.metiz.com.tw/bending_wire_rods.html.
8. Arduino-3d-wire-bending-machine [Electronic resource]. – Access mode: <https://howtomechatronics.com/projects/arduino-3d-wire-bending-machine/>.
9. Diy-arduino-based-2d-wire-bending-machine [Electronic resource]. – Access mode: <https://electricdiylab.com/diy-arduino-based-2d-wire-bending-machine/>.
10. Spring-making-wire-bending-machine [Electronic resource]. – Access mode: <https://www.instructables.com/Spring-Making-Wire-Bending-Machine/>.
11. NEMA17-01 42STH47-1684A Stepper Motor [Electronic resource]. – Access mode: <https://asset.conrad.com/media10/add/160267/c1/-/en/001597325DS00/datablad-1597325-joy-it-stegmotor-nema-17-01-nema-17-01-04-nm-168-a-axel-diameter-5-mm.pdf>.

12. COM-MOTOR02-DATASHEET [Electronic resource]. – Access mode: <https://cdn-reichelt.de/documents/datenblatt/A300/COM-MOTOR02-DATASHEET.pdf>.
13. ARD Nano V 3 [Electronic resource]. – Access mode: https://joy-it.net/en/products/ARD_NanoV3.
14. DRV8825 Datasheet [Electronic resource]. – Access mode: https://www.alldatasheet.com/view.jsp?Searchword=Drv8825%20Datasheet&gad=1&gclid=CjwKCAjwkeqkBhAnEiwA5U-uM6rTmYNBuFtPqhuU8BYRDe8Cn2jGfei4nOK6w-o22rjEj-HeKKi84BoCvaQQA_vD_BwE.
15. CNC shield V 4 user manual [Electronic resource]. – Access mode: <http://osoyoo.com/2017/04/07/arduino-nano-cnc-shield-v4-0a4988/>.
16. How to control a stepper motor with DRV8825 driver and Arduino [Electronic resource]. – Access mode: <https://www.makerguides.com/drv8825-stepper-motor-driver-arduino-tutorial/>.
17. A contemporary approach to the MSE paradigm powered by artificial intelligence from a review focused on polymer matrix composites / [C. Gomez, A. Guardia, J. L. Mantari et al.] // Mechanics of advanced materials and Structures. – 2022. – Vol. 29(21). – P. 3076-3096.
18. Industry 4.0: a review on industrial automation and robotic / [M. Aiman, K. Bahrin, F. Othman et al.] // J. Teknol. – 2016. – Vol. 78. – P. 2180 –3722.
19. Lu, Y. Industry 4.0: a survey on technologies, applications and open research issues / Y. Lu // J. Ind. Inf. Integr. . – 2017. – Vol. 6. – P. 1–10.
20. Manufacturing's next act [Electronic resource] / C. Baur, D. Wee. – Access mode: <https://www.mckinsey.com/businessfunctions/operations/our-insights/manufacturings-next-act>.
21. Recommendations for implementing the strategic initiative Industrie 4.0. Final report of Industrie 4.0 working group [Electronic resource] / H. Kagermann, J. Wahlster, W. Helbig. – Access mode: <https://en.acatech.de/publication/recommendations-for-implementing-the-strategic->

initiative-industrie-4-0-final-report-of-the-industrie-4-0-working-group/.

22. Industry 4.0: The future of productivity and growth in manufacturing [Electronic resource] / [M. Rußmann, M. Lorenz, P. Gerbert et al.]. – Access mode: [https://web-assets.bcg.com/img-](https://web-assets.bcg.com/img-src/Industry_40_Future_of_Productivity_April_2015_tcm9-61694.pdf)

[src/Industry_40_Future_of_Productivity_April_2015_tcm9-61694.pdf](https://web-assets.bcg.com/img-src/Industry_40_Future_of_Productivity_April_2015_tcm9-61694.pdf).

23. Vaidya, S. Industry 4.0 – a glimpse / S. Vaidya, P. Ambad, S. Bhosle // *Procedia Manuf.* – 2018. – Vol. 20. – P. 233–238.

24. Lele, A. Disruptive technologies for the militaries and security / A. Lele // *Smart innovation, systems and technologies.* – 2019. – Vol. 132. – P. 205–215.

25. Peruzzini, M. Benchmarking of tools for user experience analysis in Industry 4.0 / M. Peruzzini, F. Grandi, M. Pellicciari // *Procedia Manuf.* – 2017. – Vol. 11. – P. 806–813.

26. Roblek, V. A complex view of Industry 4.0 / V. Roblek, M. Mesko, A. Krapez // *SAGE Open.* – 2016. – Vol. 6, № 2. – P. 1–11.

27. Reiner, A. Industrie 4. 0 – advanced engineering of smart products and smart production / A. Reiner // *High technology technological innovations in the product development: 19th International seminar, Piracicaba, Brasil, 9 October 2014: proceedings.* – 2014. – P. 1–14.

28. Virtual engineering object (VEO): toward experience-based design and manufacturing for Industry 4.0 / [S. I. Shafiq, C. Sanin, C. Toro et al.] // *Cybern. Syst.* – 2015. – Vol. 46, № 1–2. – P. 35–50.

29. Virtual engineering factory: creating experience base for Industry 4.0/ [S. I. Shafiq, C. Sanin, E. Szczerbicki et al.] // *Cybern Syst.* – 2016. – Vol. 47, № 1–2. – P. 32–47.

30. Thoben, K.-D. Industrie 4.0 and smart manufacturing – a review of research issues and application examples / K.-D. Thoben, S. Wiesner, T. Wuest // *Int. J. Autom. Technol.* – 2017. – Vol. 11, № 1. – P. 4–16.

31. Industrie 4.0 – from the perspective of applied research / [R. Neugebauer, S. Hippmann, M. Leis et al.] // *Procedia CIRP.* – 2016. – Vol. 57. – P. 2–7.

32. Maintenance in aeronautics in an Industry 4.0 context: the role of

augmented reality and additive manufacturing./ [A. Ceruti, P. Marzocca, A. Liverani et al.] // J. Comput. Des. Eng. – 2019. – Vol. 6. – P. 516–526.

33. Gruber, F. E. Industry 4.0: A best practice project of the automotive industry / F. E. Gruber // IFIP Adv. Inf. Commun. Technol.–2013.–Vol. 411.–P.36–40.

34. The role of additive manufacturing in the era of Industry 4.0 / [U. M. Dilberoglu, B. Gharehpapagh, U. Yaman et al.] // Procedia Manuf. – 2017. – Vol. 11. – P. 545–554.

35. Crnjac, M. From concept to the introduction of Industry 4.0 / M. Crnjac, I. Veza, N. Banduka // International journal of industrial engineering and management. – 2017. – Vol. 8, №. 1. – P. 21–30.

36. Digital twins [Electronic source]. – Access mode: <https://www.it.ua/knowledge-base/technology-innovation/cifrovoj-dvojn timer-digital-twin>.

37. Цифровий двійник [Електрон. ресурс]. – Режим доступу: <https://digitap.com.ua/company-products/czyfrovyj-dvijnyk/>.

38. What is a digital twin and why is it so valuable? [Electronic source]. – Access mode: <https://intandem.autodesk.com/about-digital-twins/>.

39. Digital twin-driven product design framework / [F. Tao, F. Sui, A. Liu et al.] // International journal of production research. –2019. –Vol. 57(12).–P. 3935–3953.

40. Nee, A. Y. C. Virtual and augmented reality applications in manufacturing / A. Y. C. Nee, S. K. Ong // IFAC Proceedings. – 2013. –Vol. 46 (9). – P. 15–26.

41. Processing и Arduino [Электрон. ресурс]. – Режим доступа: <https://robocraft.ru/arduino/336>.

42. Arduino/Processing language comparison [Electronic source]. – Access mode: <https://www.arduino.cc/en/Reference/Comparison>.

43. Processing Alternatives [Electronic source]. – Access mode: <https://alternativeto.net/software/processing/>.

44. Forget the Arduino IDE: 5 awesome alternatives [Electronic source]. – Access mode: <https://www.learnrobotics.org/blog/arduino-ide-alternatives/>.

Додаток А
Текст програми

A.1 Текст програми для крокових двигунів V1

```
#define dirPinFeeder 5
#define stepPinFeeder 7
#define dirPinBender 3
#define stepPinBender 6
#define dirPinZ 2
#define stepPinZ 4
#define stepsPerRevolution 200

void setup() {
  pinMode(stepPinFeeder, OUTPUT);
  pinMode(dirPinFeeder, OUTPUT);
  pinMode(stepPinBender, OUTPUT);
  pinMode(dirPinBender, OUTPUT);
  pinMode(stepPinZ, OUTPUT);
  pinMode(dirPinZ, OUTPUT);
}

void loop() {
  digitalWrite(dirPinFeeder, HIGH);
  digitalWrite(dirPinBender, HIGH);
  digitalWrite(dirPinZ, HIGH);

  for (int i = 0; i < stepsPerRevolution; i++) {
    digitalWrite(stepPinFeeder, HIGH);
    digitalWrite(stepPinBender, HIGH);
    digitalWrite(stepPinZ, HIGH);
    delayMicroseconds(2000);
    digitalWrite(stepPinFeeder, LOW);
    digitalWrite(stepPinBender, LOW);
    digitalWrite(stepPinZ, LOW);
    delayMicroseconds(2000);
  }
}
```

```

delay(1000);

digitalWrite(dirPinFeeder, LOW);
digitalWrite(dirPinBender, LOW);
digitalWrite(dirPinZ, LOW);

for (int i = 0; i < stepsPerRevolution; i++) {
  digitalWrite(stepPinFeeder, HIGH);
  digitalWrite(stepPinBender, HIGH);
  digitalWrite(stepPinZ, HIGH);
  delayMicroseconds(1000);
  digitalWrite(stepPinFeeder, LOW);
  digitalWrite(stepPinBender, LOW);
  digitalWrite(stepPinZ, LOW);
  delayMicroseconds(1000);
}

delay(1000);

digitalWrite(dirPinFeeder, HIGH);
digitalWrite(dirPinBender, HIGH);
digitalWrite(dirPinZ, HIGH);

for (int i = 0; i < 5 * stepsPerRevolution; i++) {
  digitalWrite(stepPinFeeder, HIGH);
  digitalWrite(stepPinBender, HIGH);
  digitalWrite(stepPinZ, HIGH);
  delayMicroseconds(500);
  digitalWrite(stepPinFeeder, LOW);
  digitalWrite(stepPinBender, LOW);
  digitalWrite(stepPinZ, LOW);
  delayMicroseconds(500);
}

delay(1000);

```

```

digitalWrite(dirPinFeeder, LOW);
digitalWrite(dirPinBender, LOW);
digitalWrite(dirPinZ, LOW);

for (int i = 0; i < 5 * stepsPerRevolution; i++) {
digitalWrite(stepPinFeeder, HIGH);
digitalWrite(stepPinBender, HIGH);
digitalWrite(stepPinZ, HIGH);
delayMicroseconds(500);
digitalWrite(stepPinFeeder, LOW);
digitalWrite(stepPinBender, LOW);
digitalWrite(stepPinZ, LOW);
delayMicroseconds(500);
}

delay(1000);
}

```

A.2 Текст програми для крокових двигунів V2

```

#define dirPin_bender 3
#define stepPin_bender 6
#define dirPin_feeder 5
#define stepPin_feeder 7
#define stepsPerRevolution 200
int val=0;
int count = 0;

void setup() {
pinMode(dirPin_bender, OUTPUT);
pinMode(stepPin_bender, OUTPUT);
pinMode(dirPin_feeder, OUTPUT);
pinMode(stepPin_feeder, OUTPUT);
digitalWrite(dirPin_feeder, LOW);
}

```

```

void loop() {
while(count!=1){
val=4;
bending (val,200);
count++;
}
}

void bending(int iterations, int steps){

for (int n = 1; n<iterations; n++){

for (int i = 0; i < 3*stepsPerRevolution; i++) {
digitalWrite(stepPin_feeder, HIGH);
delayMicroseconds(2000);
digitalWrite(stepPin_feeder, LOW);
delayMicroseconds(2000);
}
delay(1000);

digitalWrite(dirPin_bender, HIGH);
for (int i = 0; i < steps; i++) {
digitalWrite(stepPin_bender, HIGH);
delayMicroseconds(2000);
digitalWrite(stepPin_bender, LOW);
delayMicroseconds(2000);
}
delay(300);
digitalWrite(dirPin_bender, LOW);
for (int i = 0; i < steps; i++) {
digitalWrite(stepPin_bender, HIGH);
delayMicroseconds(2000);
digitalWrite(stepPin_bender, LOW);
delayMicroseconds(2000);
}
}
}

```

```

delay(1000);

}

for (int i = 0; i < 3*stepsPerRevolution; i++) {
digitalWrite(stepPin_feeder, HIGH);
delayMicroseconds(2000);
digitalWrite(stepPin_feeder, LOW);
delayMicroseconds(2000);
}

delay(500);

}

```

Удосконалена версія коду V3 дозволяє створити обрану користувачем фігуру (трикутник або квадрат), яка буде зігнута з дроту з вказаною користувачем довжиною сторони.

A.3 Текст програми для крокових двигунів V3

```

#define dirPin_bender 3
#define stepPin_bender 6
#define dirPin_feeder 5
#define stepPin_feeder 7
#define stepsPerRevolution 200
int commandValueSize;
int commandValueShape;

void setup() {
Serial.begin(9600);
pinMode(dirPin_bender, OUTPUT);
pinMode(stepPin_bender, OUTPUT);
pinMode(dirPin_feeder, OUTPUT);
pinMode(stepPin_feeder, OUTPUT);
digitalWrite(dirPin_feeder, LOW);
}

```

```

void loop() {
  if(Serial.available()>=2){
    commandValueShape=Serial.read();
    commandValueSize=Serial.read();
    commandValueSize=200*commandValueSize/3.77;
    if(commandValueShape==4){
      bending (commandValueShape,200,commandValueSize);
    }
    if(commandValueShape==3){
      bending (commandValueShape,230, commandValueSize);
    }
  }
}

void bending(int iterations, int steps, int size){

  for (int n = 1; n<iterations; n++){

    for (int i = 0; i < size; i++) {
      digitalWrite(stepPin_feeder, HIGH);
      delayMicroseconds(2000);
      digitalWrite(stepPin_feeder, LOW);
      delayMicroseconds(2000);
    }
    delay(1000);

    digitalWrite(dirPin_bender, HIGH);
    for (int i = 0; i < steps; i++) {
      digitalWrite(stepPin_bender, HIGH);
      delayMicroseconds(2000);
      digitalWrite(stepPin_bender, LOW);
      delayMicroseconds(2000);
    }
    delay(300);
  }
}

```

```

digitalWrite(dirPin_bender, LOW);
for (int i = 0; i < steps; i++) {
digitalWrite(stepPin_bender, HIGH);
delayMicroseconds(2000);
digitalWrite(stepPin_bender, LOW);
delayMicroseconds(2000);
}
delay(1000);

}

for (int i = 0; i < size; i++) {
digitalWrite(stepPin_feeder, HIGH);
delayMicroseconds(2000);
digitalWrite(stepPin_feeder, LOW);
delayMicroseconds(2000);
}
delay(500);

}

```

А. 4 Текст програми для інтерфейсу при використанні монітору послідовного порту

```

#define dirPin_bender 3
#define stepPin_bender 6
#define dirPin_feeder 5
#define stepPin_feeder 7
#define stepsPerRevolution 200
int commandValueSize=600;
int commandValueShape;

void setup() {
Serial.begin(9600);
pinMode(dirPin_bender, OUTPUT);
pinMode(stepPin_bender, OUTPUT);

```

```

pinMode(dirPin_feeder, OUTPUT);
pinMode(stepPin_feeder, OUTPUT);
digitalWrite(dirPin_feeder, LOW);
Serial.println();
    Serial.println("Choose the shape you want to be bent (type 3
for triangle or 4 for square)");
}

```

```

void loop() {
if(Serial.available()){
commandValueShape=Serial.read();
if(commandValueShape==4){
Serial.println("Bending square");
bending (commandValueShape,200,commandValueSize);
}
if(commandValueShape==3){
Serial.println("Bending triangle");
bending (commandValueShape,230, commandValueSize);
}
}
}
}

```

```

void bending(int iterations, int steps, int size){

for (int n = 1; n<iterations; n++){

for (int i = 0; i < size; i++) {
digitalWrite(stepPin_feeder, HIGH);
delayMicroseconds(2000);
digitalWrite(stepPin_feeder, LOW);
delayMicroseconds(2000);
}
delay(1000);
}
}

```

```
digitalWrite(dirPin_bender, HIGH);
for (int i = 0; i < steps; i++) {
digitalWrite(stepPin_bender, HIGH);
delayMicroseconds(2000);
digitalWrite(stepPin_bender, LOW);
delayMicroseconds(2000);
}
delay(300);
```

```
digitalWrite(dirPin_bender, LOW);
for (int i = 0; i < steps; i++) {
digitalWrite(stepPin_bender, HIGH);
delayMicroseconds(2000);
digitalWrite(stepPin_bender, LOW);
delayMicroseconds(2000);
}
delay(1000);
```

```
}
```

```
for (int i = 0; i < size; i++) {
digitalWrite(stepPin_feeder, HIGH);
delayMicroseconds(2000);
digitalWrite(stepPin_feeder, LOW);
delayMicroseconds(2000);
}
delay(500);
```

```
}
```

A.5 Текст програми для інтерфейсного застосунку

```
import processing.serial.*;
Serial serial;
```

```
PFont f;
```

```

int shape = 0;
int size_prev = 0;
int size = 0;
int i = 0;

public void settings() {
  size(1380, 630);
}

void setup () {
  serial = new Serial(this, Serial.list()[0], 9600);
  f = createFont("Georgia", 60);
  background(240,255,240);
  strokeWeight(5);
  stroke(0);
  fill(216,191,210);
  rect(780,230,480,100);
  fill(255,230,140);
  square(60,180,240);
  fill(0,255,255);
  triangle(380,180,640,180,510,420);
  fill(110,200,100);
  rect(590,530,200,50);
  fill(0);
  textSize(30);
  text("Choose the shape you want to be bent",67,95);
  text("Choose the length of the shape's side (in cm)",700,95);
  fill(255);
  textSize(35);
  text("B E N D",629,567);
  fill(0);
  for(int i=0;i<=20;i++){
    if (i==5 || i==0){
      text(i,793+i*22,363);
      strokeWeight(4);
      line(800+i*22,317,800+i*22,330);
    }
  }
}

```

```

    }
    else {
        if(i!=5 && i!=0 && i%5==0){
            text(i,785+i*22,363);
            strokeWeight(4);
            line(800+i*22,317,800+i*22,330);
        }
        else{
            strokeWeight(3);
            line(800+i*22,317,800+i*22,331);
        }
    }
    println("current size = " + 0 + " cm");
}

void draw () {
    if(!mousePressed){
        i=0;
    }
    if(mousePressed){
        if(mouseX>=60&&mouseX<=300&&mouseY>=180&&mouseY<=420){
            fill(1,190,32);
            stroke(1,190,32);
            strokeWeight(6);
            line(147,287,167,320);
            line(167,320,217,280);
            noStroke();
            fill(0,255,255);
            rect(465,225,90,90);
            shape = 4;
        }
        else {
            if(isMouseInTriangle(380, 180, 640, 180, 510, 420, mouseX,
mouseY)){

```

```

fill(1,190,32);
stroke(1,190,32);
strokeWeight(6);
line(475,265,495,298);
line(495,298,545,258);
noStroke();
fill(255,230,140);
rect(143,275,80,50);
shape = 3;
}
else{
  if(mouseX>=590&&mouseX<=790&&mouseY>=530&&mouseY<=580){
    if(shape==4&&size!=0){
      noStroke();
      fill(240,255,240);
      textSize(35);
      rect(860,470,500,300);
      rect(110,470,450,300);
      fill(1,190,32);
      text("You are bending a square\n"+"with a side of
"+size+"cm!",870,545);
      while(i<1){
        serial.write(shape);
        serial.write(size);
        i++;
      }
    }
    else{
      if(shape==3&&size!=0){
        noStroke();
        fill(240,255,240);
        textSize(35);
        rect(860,470,500,300);
        rect(110,470,450,300);
        fill(1,190,32);

```

```

    text("You are bending a triangle\n"+"with a side of
"+size+"cm!",870,545);
    while(i<1){
        serial.write(shape);
        serial.write(size);
        i++;
    }
}
else{
    if(shape==0&&size!=0){
        noStroke();
        fill(240,255,240);
        textSize(35);
        rect(860,470,500,300);
        rect(110,470,450,300);
        fill(250,100,32);
        text("Please choose the shape!",110,563);
    }
    else{
        if(shape!=0&&size==0){
            noStroke();
            fill(240,255,240);
            textSize(35);
            rect(860,470,500,300);
            rect(110,470,450,300);
            fill(250,100,32);
            text("Please choose the size!",110,563);
        }
        else if(size==0&&shape==0){
            noStroke();
            fill(240,255,240);
            textSize(35);
            rect(860,470,500,300);
            rect(110,470,450,300);
            fill(250,100,32);

```

```
text("Please choose the size\n"+"and the shape!",110,540);
}
}
}
}
}
else if
(mouseX>=780&&mouseX<=1260&&mouseY>=230&&mouseY<=330) {
    size=(mouseX-800)/22;
    if(size!=size_prev){
        println("size = "+size+" cm");
        size_prev=size;
    }
    if (mouseX<=800){
        noStroke();
        fill(216,191,210);
        rect(787,240,467,70);
        fill(1,190,32);
        stroke(1,190,32);
        strokeWidth(5);
        line(800,250,800,298);
        line(790,288,800,298);
        line(810,288,800,298);
    }
    else{
        if (mouseX>=1240){
            noStroke();
            fill(216,191,210);
            rect(787,240,467,70);
            fill(1,190,32);
            stroke(1,190,32);
            strokeWidth(5);
            line(1240,250,1240,298);
            line(1230,288,1240,298);
            line(1250,288,1240,298);
```

```

    }
    else{
        noStroke();
        fill(216,191,210);
        rect(787,240,467,70);
        fill(1,190,32);
        stroke(1,190,32);
        strokeWeight(5);
        line(mouseX,250,mouseX,298);
        line(mouseX-10,288,mouseX,298);
        line(mouseX+10,288,mouseX,298);
    }
}
}
}
}
}
}
}

boolean isMouseInTriangle(float x1, float y1, float x2, float
y2, float x3, float y3, float mouseX, float mouseY) {
    float fullArea = triangleArea(x1, y1, x2, y2, x3, y3);
    float area1 = triangleArea(mouseX, mouseY, x2, y2, x3, y3);
    float area2 = triangleArea(x1, y1, mouseX, mouseY, x3, y3);
    float area3 = triangleArea(x1, y1, x2, y2, mouseX, mouseY);
    return (area1 + area2 + area3) == fullArea;
}

float triangleArea(float x1, float y1, float x2, float y2,
float x3, float y3) {
    return abs((x1 * (y2 - y3) + x2 * (y3 - y1) + x3 * (y1 - y2))
/ 2.0);
}

```

Додаток Б
Слайди презентації

Національний університет «Запорізька Політехніка»
Кафедра програмних засобів

Дипломна кваліфікаційна робота магістра на тему:

ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ ТА ЗАСОБІВ ПРОТОТИПУВАННЯ ТЕХНОЛОГІЧНОЇ СИСТЕМИ

Виконав
ст. гр. КНТ-132м

Едуард ПОЛЯНСЬКИЙ

Керівник
к.т.н., доц.

Анжеліка ПАРХОМЕНКО

Запоріжжя
2023

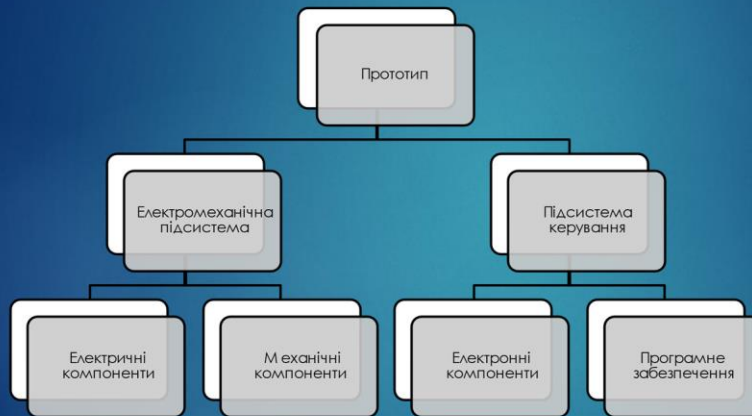
Рисунок Б.1 – Слайд 1



Рисунок Б.2 – Слайд 2

Актуальність роботи

3



Структура та зовнішній вигляд прототипу ТСЗД

Рисунок Б.3 – Слайд 3

Мета та завдання роботи

4

Мета роботи – дослідження технологій цифрової індустрії для прототипування технологічної системи та розробка програмного забезпечення для числового програмного управління.

Завдання роботи:

- ▶ дослідити методи та засоби цифрової індустрії для прототипування технологічних систем;
- ▶ розробити методику створення цифрового двійника ТСЗД з використанням технологій віртуальної, доповненої реальності та адитивного виробництва;
- ▶ виконати проектування та конструювання програмного забезпечення ЧПУ прототипу ТСЗД;
- ▶ розробити інструкцію з використання програмного забезпечення.

Рисунок Б.4 – Слайд 4

Дослідження особливостей цифрових двійників (ЦД)

5

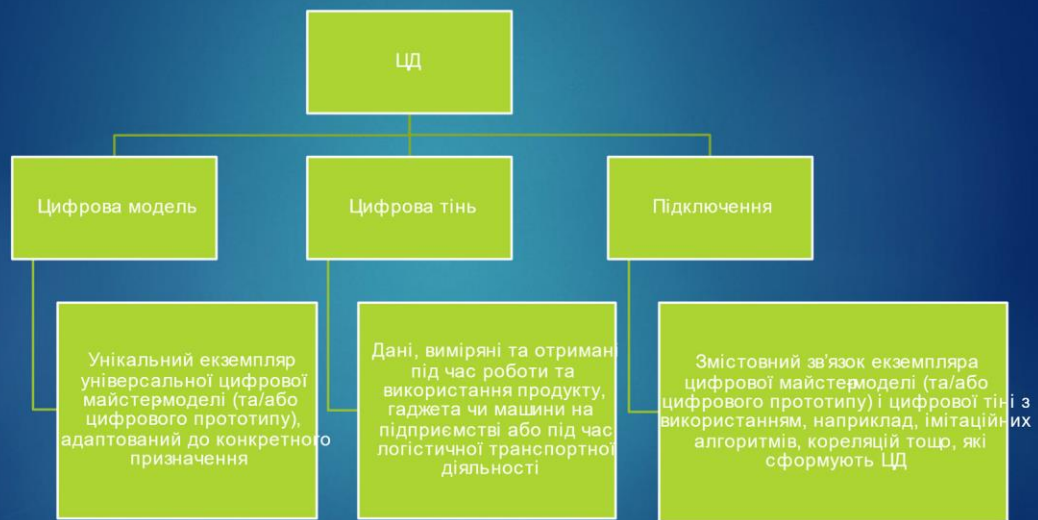
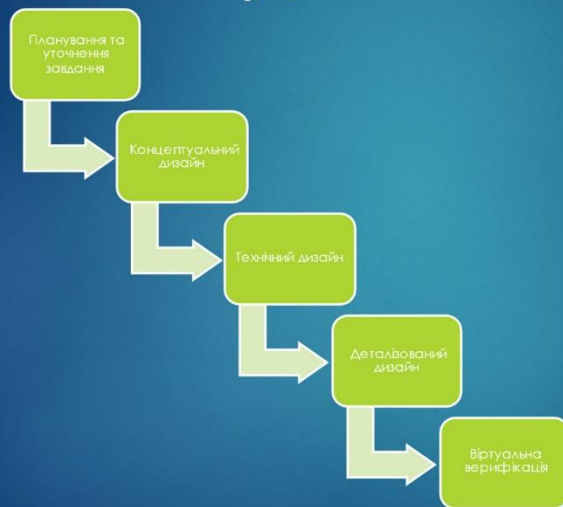


Рисунок Б.5 – Слайд 5

Розробка методу прототипування на основі ЦД

6



Фреймворк дизайну та схема алгоритму методики прототипування на основі ЦД

Рисунок Б.6 – Слайд 6

Проектування ПЗ

7



Діаграма прецедентів та узагальнена архітектура системи

Рисунок Б.7 – Слайд 7

Проектування ПЗ

8

Характеристики	Середовище розробки	
	Arduino IDE	Processing
Мова програмування	C / Wiring	Processing, /Java
Підтримка	+	+
Простота	+	+
Доступність	Open-source	Open-source

Альтернативи Arduino IDE:
 VScode,
 PlatformIO,
 React Studio,
 Ktechlab,
 B4R,
 плагін Arduino Eclipse
 Atmel Studio

Альтернативи Processing:
 Vvvv,
 OpenFrameworks,
 Nodebox,
 Processing.js
 Vuo

Порівняння та вибір середовищ розробки

Рисунок Б.8 – Слайд 8



Рисунок Б.9 – Слайд 9

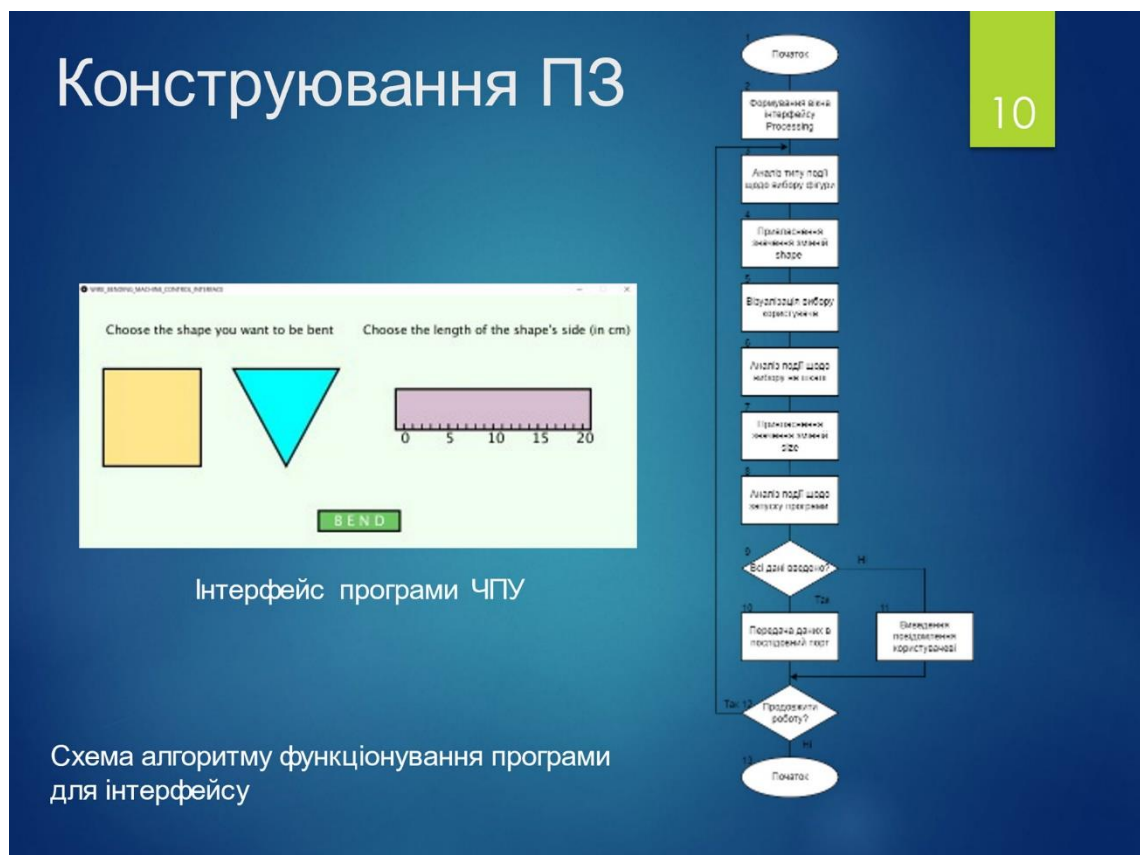
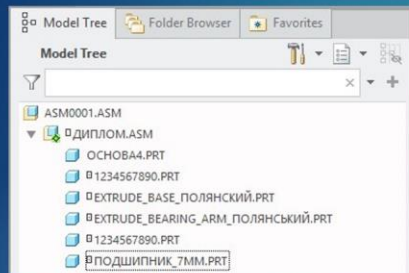


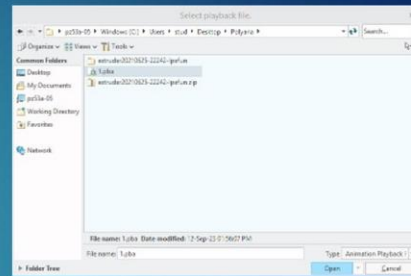
Рисунок Б.10 – Слайд 10

Результати розробки ЦД

11



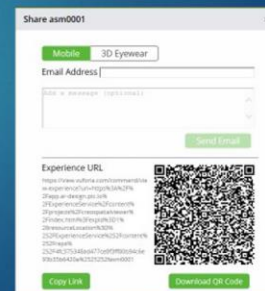
Дерево побудови



Вікно завантаження анімації



Вікно керування моделлю в хмарі

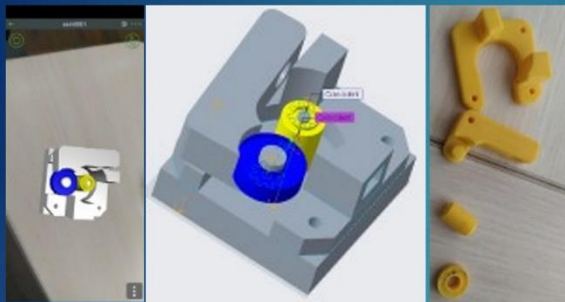


Вікно «Поділитися»

Рисунок Б.11 – Слайд 11

Результати розробки ЦД

12



Результати застосування технологій доповненої, віртуальної реальності та 3D друку

Структура цифрового двійника механізму подачі прототипу

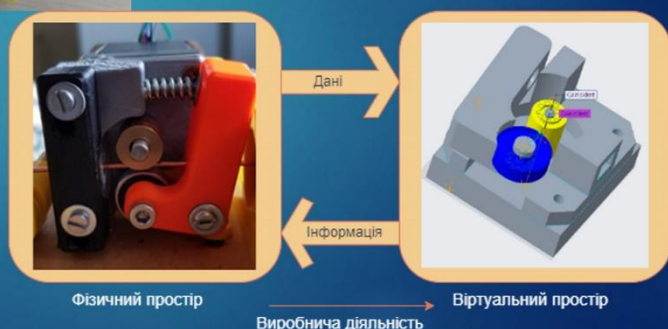


Рисунок Б.12 – Слайд 12

Висновки

13

Проведені дослідження смарт-технологій цифрової індустрії дозволили виявити сучасну тенденцію розробки ЦД з використанням технологій доповненої та віртуальної реальності, адитивного виробництва та Інтернету речей.

В роботі показано, що системи доповненої реальності є новою формою людино-машинної взаємодії, а 3-D друк - вдосконаленою технологією передачі цифрових інструкцій у фізичний світ.

Наукова новизна роботи полягає в тому, що розроблено методику прототипування технологічної системи та фреймворк дизайну на основі ЦД, що відрізняються від існуючих синергією пов'язаних віртуальних та фізичних складових, що дозволяє активно коригувати поведінку фізичного прототипу в реальному часі відповідно до рекомендацій, наданих віртуальною моделлю, а з іншого боку - зробити віртуальну модель більш відповідною реальному стану фізичного прототипу.

Практична цінність роботи полягає в тому, що використання розробленого апаратного та програмного забезпечення при прототипуванні ТСЗД дозволить виявити шляхи її модернізації, а також удосконалити алгоритми управління.

Рисунок Б.13 – Слайд 13

14

Дякую за увагу!

Рисунок Б.14 – Слайд 14