МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ТЕХНОЛОГІЙ ТА ДИЗАЙНУ
Факультет ринкових, інформаційних та інноваційних технологій
Кафедра інформаційно-комп'ютерних технологій
та фундаментальних дисциплін

ДИПЛОМНА МАГІСТЕРСЬКА РОБОТА

на тему:

«СИСТЕМА УПРАВЛІННЯ ВЕРСТАТОМ НА ОСНОВІ ОПТИМАЛЬНОГО РІШЕННЯ ПІДБОРУ МІКРОКОНТРОЛЕРІВ»

Виконав: студент групи МгЧКІ-20
спеціальності 123
«Комп'ютерна інженерія»
В.В. Лисенко

Керівник: к.ф.-м.н., професор
М.В. Ярмоленко
Рецензент: С. А. Міценко
(прізвище та ініціали)

Черкаси 2021

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ТЕХНОЛОГІЙ ТА ДИЗАЙНУ

Факультет *ринкових, інформаційних та інноваційних технологій*

Кафедра *інформаційно-комп'ютерних технологій та фундаментальних дисциплін*

Спеціальність *123 Комп'ютерна інженерія*

Освітня програма *Комп'ютерна інженерія*

ЗАТВЕРДЖУЮ:

Зав. кафедри ІКТ та ФД

_____ Михайло ЯРМОЛЕНКО

«_____» _____ 2021 року

З А В Д А Н Н Я

НА ДИПЛОМНУ МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Лисенку В'ячеславу Васильовичу

1. Тема роботи: СИСТЕМА УПРАВЛІННЯ ВЕРСТАТОМ НА ОСНОВІ ОПТИМАЛЬНОГО РІШЕННЯ ПІДБОРУ МІКРОКОНТРОЛЕРІВ

Науковий керівник роботи Михайло ЯРМОЛЕНКО к.ф.-м.н., професор

затверджені наказом вищого навчального закладу від «4» жовтня 2021 р. № 286

Строк подання студентом роботи: 01 грудня 2021 р.

Вихідні дані роботи: Характеристики і тип мікроконтролерів, програмне забезпечення для верстата з числовим програмним управлінням

2. Зміст дипломної роботи (перелік питань, які потрібно розробити): Вступ. Розділ 1. Ознайомлення з мікроконтролерами та вибір оптимального варіанту. Розділ 2. Розробка верстату з числовим програмним управлінням. Розділ 3. Експериментальні дослідження. Висновки. Список використаних джерел.

3. Консультанти розділів дипломної магістерської роботи

Розділ	Ім’я, прізвище та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Вступ	Михайло ЯРМОЛЕНКО, к.ф.-м.н., професор		
Розділ 1	Михайло ЯРМОЛЕНКО, к.ф.-м.н., професор		
Розділ 2	Михайло ЯРМОЛЕНКО, к.ф.-м.н., професор		
Розділ 3	Михайло ЯРМОЛЕНКО, к.ф.-м.н., професор		
Висновки	Михайло ЯРМОЛЕНКО, к.ф.-м.н., професор		

6. Дата видачі завдання: _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного магістерської роботи	Терміни виконання етапів	Примітка про виконання
1	Вступ	Вересень, 2021 р.	Виконано
2	Розділ 1. Ознайомлення з мікроконтролерами та вибір оптимального варіанту	Вересень, 2021 р.	Виконано
3	Розділ 2. Розробка верстату з числовим програмним управлінням	Жовтень, 2021 р.	Виконано
4	Розділ 3. Експериментальні дослідження	Жовтень, 2021 р.	Виконано
5	Висновки	Листопад, 2021 р.	Виконано
6	Оформлення дипломної магістерської роботи (чистовий варіант)	Листопад, 2021 р.	Виконано
7	Здача дипломної магістерської роботи на кафедру для рецензування (за 14 днів до захисту)	Грудень, 2021 р.	Виконано
8	Перевірка дипломної магістерської роботи на наявність ознак плагіату (за 10 днів до захисту)	Грудень, 2021 р.	Виконано
9	Подання дипломної магістерської роботи на затвердження завідувачу кафедри (за 7 днів до захисту)	Грудень, 2021 р.	Виконано

Студент _____ В’ячеслав ЛИСЕНКО
(підпис)

Науковий керівник роботи _____ Михайло ЯРМОЛЕНКО
(підпис)

АНОТАЦІЯ

Дипломна магістерська робота присвячена створенню системи управління верстатом на основі оптимального рішення підбору мікроконтролерів та програмного забезпечення. Виходячи з поставленого завдання в роботі було розглянуто типи мікроконтролерів, їх характеристики, також було розглянуто варіанти програмного забезпечення для різних мікроконтролерів. Проведено детальний аналіз роботи верстата і натурні експерименти. Проведений аналіз програмно-апаратних засобів та описано їх переваги і недоліки.

Для досягнення поставленої мети магістерської роботи розв’язані такі задачі:

– Аналіз принципів роботи верстатів з ЧПУ.

– Ознайомлення з системою команд.

– Принципи позиціювання і траєкторії руху інструментів.

– Аналіз наявних мікроконтролерів, і придатність їх для створення верстатів.

– Пошук безкоштовного програмного забезпечення, з подальшим аналізом його придатності для використання.

– Підрахування приблизної вартості верстата.

– Побудувати дослідний зразок верстата, встановити і налаштувати програмне забезпечення.

– Дослідити на практиці теоретичний матеріал.

– Дослідити і спробувати побудувати автономну систему управління.

Обсяг магістерської роботи складає 113 сторінок. При написанні магістерської роботи була використана література та Інтернет-джерела.

Ключові слова: комп'ютерні мікросхеми, проектно-розрахункова автоматична система, мікроконтролери.

ABSTRACT

The master's thesis deals with the creation of a machine control system based on the optimal solution for the selection of microcontrollers and software. Based on the assigned task the types of microcontrollers, their characteristics, as well as software options for different microcontrollers were considered in the thesis. The detailed analysis of the machine operation and field experiments were performed. The analysis of software and hardware was carried out and their advantages and disadvantages were described.

To achieve the goal of the master's thesis the following tasks were solved:

– Analysis of the principles of CNC machines;

– Acquaintance with the command system;

– Principles of positioning and toolpaths;

– Analysis of available microcontrollers and their suitability for creating machine tools;

– Search for free software with subsequent analysis of its suitability for use;

– Calculating the approximate cost of the machine;

– To build a prototype machine, install and configure software;

– To investigate theoretical material in practice;

– To study and try to build an autonomous control system.

The master's thesis consists of an explanatory note on 113 pages.

Keywords: computer chips, project and calculation automatic system, microcontrollers.

СПИСОК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

CAD - комп’ютерна підтримка проектування

FDD - привід гнучких дисків

RAM - пам’ять з довільним доступом

RMS - середнє квадратичне значення змінного струму

ROM - пам’ять тільки для читання

SD - Secure Digital Memory Card

TTL - транзисторно-транзисторна логіка

VREF - опорна напруга

АЦП - аналого-цифрові перетворювачі

ВІС - велика інтегральна схема

ДЗЗ - датчиків зворотного зв'язку

ДУ - драйвер управління

ЕОМ - електронно-обчислювальна машина

ЕРС - електрорушійна сила

КГП - кульково-гвинтова пара

КД - кроковий двигун

НВІС - надвелика інтегральна схема

ОЗП - оперативно-запам'ятовуючий пристрій

ПЗ - програмне забезпечення

ПЗП - постійний запам'ятовуючий пристрій

ПК - персональний комп’ютер

РО - робочий орган

САМ - автоматизована система технологічної підготовки виробництва

САПР - система автоматизованого проектування і розрахунку

УП - потік, що управляє

ЦАП - цифро-аналогові перетворювачі

ЧПУ - числове програмне управління

ЗМІСТ

ВСТУП

На сьогоднішній день величезне кількість промислових підприємств використовує верстати з числовим програмним управлінням, їх аналоги з ручним або механічним управлінням. Різниця між ними лише в тому, що переміщення робочих органів верстатів з ЧПУ здійснюється за рахунок електроніки під управлінням спеціальної комп'ютерної програми [1].

Переваги використання верстатів з числовим програмним управлінням у тому щоб забезпечує більш якісний рівень автоматизації виробничого процесу. Виготовлення деталей ведеться у автоматичному режимі практично без участі людини оператора-верстату, роль якого полягає у виконанні операцій контролю за процесом і участі в підготовчому і заключному етапі: у первинній наладці і подальшому контролю за ходом виконання програми і дотриманням автоматикою всіх технологічних процесів. Автономна робота верстатів з ЧПУ може продовжуватися безперервно і достатньо довго, причому якість отримуваних виробів залишається високим. Таким чином одна людина може одночасно стежити і контролювати великий парк верстатів з ЧПУ.

Ще одна перевага – забезпечення виробничої універсальності: щоб верстат перейшов до виготовленню іншого виду продукції, необхідно лише замінити спеціально підготовлене програмне забезпечення. При цьому в будь-якій час можна повернутися до попередньої програмі вже перевіреною у справі. Замінювати програми можна незліченну кількість разів.

Третя важлива перевага – можливість повторювати одні і ті ж дії багато разів. - програма дозволяє проводити на необхідному рівні якості тисячі повністю однакових по відношенню один до одного деталей. До того ж використання ЧПУ дозволяє робочому механізму виконувати максимальну кількість дій для отримання необхідного результату. Як наслідок ми отримуємо максимальне завантаження устаткування і отримання максимальну продуктивності.

Нарешті, четверта перевага – можливість виготовляти деталі достатньо складної форми, такі, виготовлення яких з використанням звичайною верстатною технікою неможливе [2].

Автоматичне числове програмне управління верстатом здійснює спеціалізований комп'ютер – важлива складова верстата – за допомогою спеціального програмного забезпечення. У програму закладені всі параметри обробки і траєкторії робочого механізму що дозволяють чітко дотримуватись режимів різання і здійснювати повноцінну обробку. Винахід і подальше вдосконалення верстатів з ЧПУ, без перебільшень, стало наступною віхою нового етапу науково-технічної революції – адже раніше керувати верстатами доводилося виключно у ручних режимах.

Читаючи дані керуючого програмного забезпечення комп'ютерний модуль посилає на відповідні двигуни ті або інші вузли електронні сигнали. А значить, фактично керує всіма циклами обробки, оскільки під впливом двигунів відбувається переміщення, яке здійснює механічну обробку деталі робочими механізмами верстата [3].

Актуальність теми

Зі стрімким розвитком технологій, появою доступних мікроконтролерів, і масовою появою персональних комп’ютерів в населення, почали розвиватись хобійні верстати з числовим програмним управлінням. В них як правило стоїть простенькій мікроконтролер відносно проста механіка, проте не дивлячись на це все, верстати дають прийнятну точність, ними можна гравіювати, фрезерувати, наносити фрезою або лазером маркування на деталі чи заготовки, можна навіть робити друковані плати, зфрезеровуючи зайву фольгу – утворюючи провідники. Такі верстати стають незамінними супутниками для «хобійщиків» та малого бізнесу.

Проте верстати не така проста річ як здається на перший погляд, вибір мікроконтролера та програмного забезпечення, під управлінням якого буде працювати верстат – досить складний, і в більшості випадків ніхто цим не займається купуючи дорогі готові рішення. Не так багато ентузіастів, які займаються вивченням теми верстатів з числовим програмним керуванням. З відомих мені – Олександр Шенрок, він займається дослідженням і створенням лазерних граверів, і віднедавна як і почав цікавитись автономними системами.

Для людей які хочуть зробити щось власними руками, не бажають платити десятки тисяч лише за назву бренду – тема актуальна, вона також актуальна для людей, які хочуть почати власний бізнес на сувенірній продукції, при тому суттєво не вкладаючись в обладнання, і для людей які хочуть самоствердитись.

Мета дослідження полягає в з'ясуванні принципу роботи верстат з числовим програмним керуванням, які є контролери, які з них можна використати для побудови верстата, яке є програмне забезпечення, в чому переваги-недоліки, спробувати побудувати верстат з ЧПУ, і розробити до нього автономну систему керування.

Об'єкт дослідження: верстат з числовим програмним управлінням, мікроконтролери, програмне забезпечення для мікроконтролерів верстату, програмне забезпечення для пульту керування верстатом з ЧПУ.

Предметом дослідження є сукупність теоретичних, методичних і практичних положень, що визначають процеси дослідження роботи верстата, мікроконтролерів, та систем керування ними.

Методи дослідження базуються на принципах порівняння, обробки даних та вибору програмного забезпечення.

Наукова новизна. В ході виконання роботи отримані наступні результати, які мають наукову новизну: було написано програму для пульта керування верстатом з ЧПУ, використовуючи простий і дешевий мікроконтролер, що дало змогу зробити верстат автономним, було використано програму для 3D принтера, як керуючу для лазерного гравера, з подальшою її модифікацією.

Практична цінність результатів досліджень полягає в тому, що ці результати можуть бути використані при виборі мікроконтролерів, і програмного забезпечення для верстатів з ЧПУ, спираючись на результати моїх досліджень, можна робити висновки: чи варто робити верстат, власноруч.

Апробація результатів магістерської роботи.

Основні результати магістерської роботи доповідались під час II Всеукраїнської конференції здобувачів вищої освіти і молодих учених «Інноватика в освіті, науці та бізнесі: виклики та можливості» 18 листопада 2021

року. Київ : КНУТД, 2021.

Публікація.

Результати досліджень були опубліковані у фаховому виданні:

Lysenko V.V., Dmytryuk S.V. The Choice of Optimal Solution for the Microcontroller Selection to Create an Autonomous Machine Control System. *Сучасні електромеханічні та інформаційні системи*: монографія / за заг. ред. І.В.Панасюка. Київ : КНУТД, 2021. С. 23 – 28.

Дипломна магістерська робота складається зі вступу, 3 розділів, висновків, списку використаних джерел (50 найменувань на 6 сторінках). Загальний обсяг магістерської роботи 113 сторінок комп’ютерного тексту.

РОЗДІЛ 1
ОЗНАЙОМЛЕННЯ З МІКРОКОНТРОЛЕРАМИ ТА ВИБІР ОПТИМАЛЬНОГО ВАРІАНТУ

1.1 Обґрунтування і вибір типу управління верстатом з ЧПУ

Виходячи з інформаційної ознаки і по кількості потоків інформації для управління верстатом з числовим програмним управлінням використовується замкнутий і розімкнений тип. Рисунок 1.1

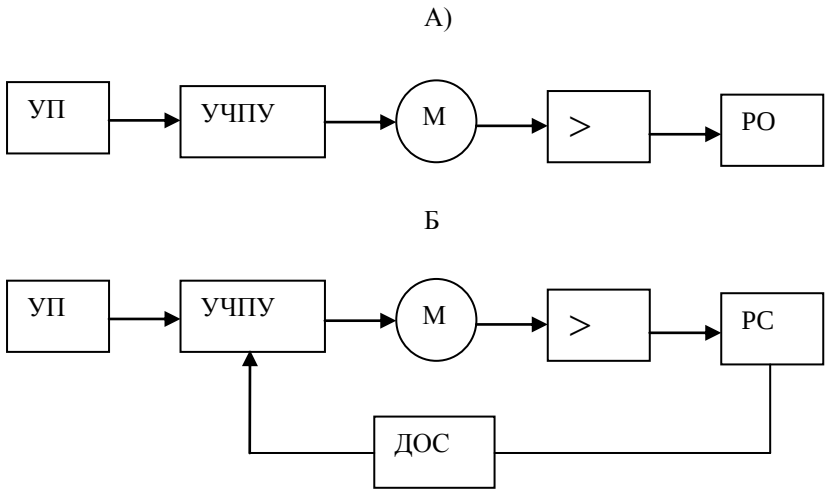

Рисунок 1.1 – типи управління верстатом ЧПУ а) розімкнене, б) замкнуте

У разі розімкненого управління існує тільки один інформаційний потік, який прямує від УП (потік, що управляє) на РО (робочий орган). Верстати, що мають таке управління, оснащені кроковими двигунами ШД, мотором М.

У випадку замкнутого управління існує два інформаційних потоки: один йде від УП, інший – від ДОС (датчиків зворотного зв'язку).

Конструктивно найбільш складними є замкнуті системи управління, проте вони і функціонують точніше останніх, оскільки в них фактичний відпрацювання пересувань порівнюється із даними, а також виконується корекція УЧПУ за інформацією датчиків зворотного зв'язку.

У системах управління замкнутого типу виділяються адаптивні або самоналагоджувальні системи. Такі системи містять додаткові інформаційні потоки, здатні коректувати процедуру обробки залежно від реальних умов

(коливання припусків, затуплення інструменту і інше).

У сучасних ЧПУ верстатах як правило в основному використовуються крокові двигуни. Завдяки практично повній безінерційності крокових двигунів і величезного обертального моменту, з'явилася можливість повністю відмовитися від датчиків зворотного зв'язку, що істотно спрощує конструкцію верстата і його вартість [4].

Управління кроковим двигуном

Кроковий електродвигун (ШД) - це синхронний безщітковий електродвигун з декількома обмотками, в якому струм, що подається в одну з обмоток статора, викликає фіксацію ротора. Для його роботи потрібний спеціальний електронний пристрій - драйвер управління (ДУ) ШД. Сучасні драйвери працюють по певному протоколу, з регульованими рівнями вхідних сигналів, необхідними логічними рівнями. Більшість ДУ працюють по протоколу STEP/DIR/ENABLE і сумісні з TTL рівнем сигналів 0..5 У [5].

Драйвера управління кроковим двигуном ЧПУ верстата вибирають по наступних параметрах:

Сила струму. Струм, яких може забезпечити ДУ, як правило, змінюється в достатньо великих межах, але ДУ потрібно підбирати такий, який зможе видавати струм, рівний струму фази даного ШД. Бажано, щоб максимальна сила струму ДУ була на 20 - 30% більше. Це дає запас на випадок, якщо необхідно отримати значно більший момент від ШД. Виробники періодично випускають пари ШД/ДУ з оптимально вибраними характеристиками. Приклад HY-TB3DV-M / NEMA23HS7430.

Напруга живлення. Її значення достатнє сильно впливає на момент сили і швидкість перемикання двигуна, а так само на вібрації, нагріваючи ШД. Як правило, максимальна напруга живлення ДУ приблизно рівна граничному струму I, помноженому на 7-10. Чим вище індуктивність ШД – тим вище напруга необхідна для драйвера. Є емпірична формула і $U = 32\sqrt{L}$, де L – індуктивність обмотки ШД. U-максимальне значення напруги живлення драйвера. Значення U,

отримувана по представленій формулі, приблизна, але це значення дозволяє від чогось відштовхуватися при виборі ДУ.

Опторозв’язка по входу. Практично у всіх ДУ, Опторозв’язка стоїть обов'язково оскільки пробій ключа може привести до сильного імпульсу, який виведе з ладу дорогий контролер ЧПУ. Опторозв’язка додатково зменшує наведення перешкод від силової частини ДУ в контролер, що управляє.

Функція пригнічення резонансу. Резонанс ШД – явище, яке з'являється постійно і відрізняється тільки резонансною частотою. Вона, в першу чергу, залежить від моменту інерції навантаження, що збільшує напругу ДУ і заданої сили струму фази ШД. При появі резонансу ШД починає вібрувати і знижувати обертовий момент, аж до зупинки валу. Для зменшення резонансу застосовують мікрокрок і - вбудовані алгоритми придушення резонансу. Вібруючі в резонансі ротор ШД створює мікроколивання ЕДС індукції в обмотках, і по їх параметру і амплітуді ДУ визначає, наявність резонансу і його величину. Залежно від набутих значень ДУ трохи зсуває кроки двигуна за часом один стосовно одного - така створена нерівномірність компенсує резонанс.

Протокол управління. Необхідно переконатися, що ДУ працює по необхідному вам протоколу, а рівні вхідних сигналів відповідають з потрібними нам логічними рівнями. Більшість ДУ спілкуються по протоколу STEP,DIR,ENABLE і сумісно з TTL рівнем сигналів 0..5 У.

Присутність захисних функцій. Серед яких захист від перевищення напруги живлення, струму обмоток, і т.д. від короткого замикання обмоток, від переполюсовки напруги живлення, від неправильного підключення фаз ШД.

1.2 Аналіз вирішень схемотехніки верстатів з ЧПУ

Загальна структура верстатів з ЧПУ:

Пульт оператора (або консоль введення-виведення), що дає можливість вводити в керуючу програму, встановлювати режими роботи; зробити операцію уручну. Зазвичай, усередині шафи пульта управління сучасного ЧПУ розташовані і решта її частин;

Дисплей (або операторська панель) – для зорового контролю режимів роботи і змінною керуючої програми/даних; може бути виконаний у вигляді окремого устаткування для дистанційного контролю устаткуванням;

Контролер – комп'ютеризоване устаткування, що виконує завдання формування траєкторії переміщення робочого інструменту, технологічних команд управління засобами автоматики устаткування, загальним управлінням, зміною керуючих програм, діагностики і додаткових розрахунків (траєкторії руху робочого інструменту, режимів роботи);

ПЗП – пам'ять, необхідна для довгострокового зберігання (роки і десятки років) робочих програм і констант; інформація з ПЗП доступна тільки для читання;

ОЗУ – пам'ять, призначена для короткострокового зберігання робочих і системних програм, використовуваних у нинішній момент [6].

В ролі контролера виступає промисловий контролер: мікропроцесор на якому побудована вбудовувана система; програмований логічний контролер або значно ускладнений пристрій управління, таке як промисловий комп'ютер.

Важливим параметром CNC-контролера є кількість керованих осей, які він здатний синхронізувати для одночасного переміщення – це вимагає висока продуктивність і необхідне ПО.

У якості вживаних механізмів використовуються сервоприводи, ШД.

Для передачі інформації між виконавчим устаткуванням і системою управління верстата, як правило, використовується мережа. У простіших верстатах шафа управління вмонтовується в безпосередній близькості від виконавчих пристроїв.

Керуюча система зчитує інструкції спеціалізованої мови програмування програми, який потім інтерпретатором системи ЧПУ переводиться з вхідної мови в команди управління головним приводом, приводами подач, контролерами управління вузлів верстата (наприклад, включити/виключити подачу емульсії, що охолоджує).

Розробка програм, що управляють, в сьогодення час виконується з

використанням спеціальних модулів для систем автоматизованого проектування (САПР) або окремих систем автоматизованого програмування (САМ), які по електронній моделі генерують програму обробки.

Найбільш поширена мова програмування ЧПУ для металоріжучого устаткування описаний документом ISO 6983 Міжнародні комітети із стандартів і називається «G-код». В окремих випадках – наприклад, системи управління гравіювальними верстатами – мова управління принципово відрізняється від стандарту. Для простих завдань, наприклад, розкрою плоских заготовок, система ЧПУ як вхідна інформація може використовувати текстовий файл у форматі обміну даними – наприклад, DXF або HPGL. **DXF** (англ. **Drawing** eXchange **Format**) – відкритий формат файлів для обміну графічною інформацією між додатками САПР. Був створений фірмою Autodesk для системи AUTOCAD. Підтримується практично всіма CAD-системами на платформі PC.

HPGL (іноді HP-GL) є основною мовою управління принтерами, використовуваним плотерами Hewlett-Packard. Його назва є абревіатурою **Hewlett-Packard Graphics Language.** В даний момент він є стандартним майже для всіх плотерів.

На сьогоднішній день на ринку існують пульти, які здатні тільки задавати настройки режимів обробки і перемішати робочими інструмент в ручному режимі, але немає пульта, який здатний зчитувати траєкторії із зовнішніх носіїв інформації і перетворювати цю інформацію в машинний код для виконання безпосередньою самою програмою виготовлення кінцевої продукції. Оскільки для цих завдань використовують промисловий комп'ютер, на якому можна у разі потреби відкоригувати програму, що управляє. Але з розвитком технологій програми, що управляють, вже на стадії написання можна випробувати на віртуальних верстатах і виявити всі недоліки і відхилення в роботі, так що даний функціонал коректування в процесі виробництва вже не потрібне. Від ЧПУ потрібне тільки зчитування і конвертація керуючої програми для управління верстатом.

1.3 Вибір мікроконтролера

На більшості, виробництвах для роботи ЧПУ верстатів використовується як система управління промисловий комп'ютер (ПК), який виконує роль зчитувача параметрів обробки і траєкторій руху робочого механізму написаної на спеціальній мові програмування і передачі їх верстату. Проте, даний ПК достатньо дорогі і займають багато місця, якщо виконувана програму не потрібна часта зміні і редагування параметрів траєкторій, то можна використовувати достатньо дешевий і компактний пульт управління на мікроконтролері, який зчитуватиме з пам'яті програму і перетворювати її в машинні команди для виготовлення кінцевої продукції. Для вибору потрібного мікроконтролера який відповідав би всім вимогам ознайомимося з причиною появи, історією створення і модельним рядом який представлений зараз на ринку [20].

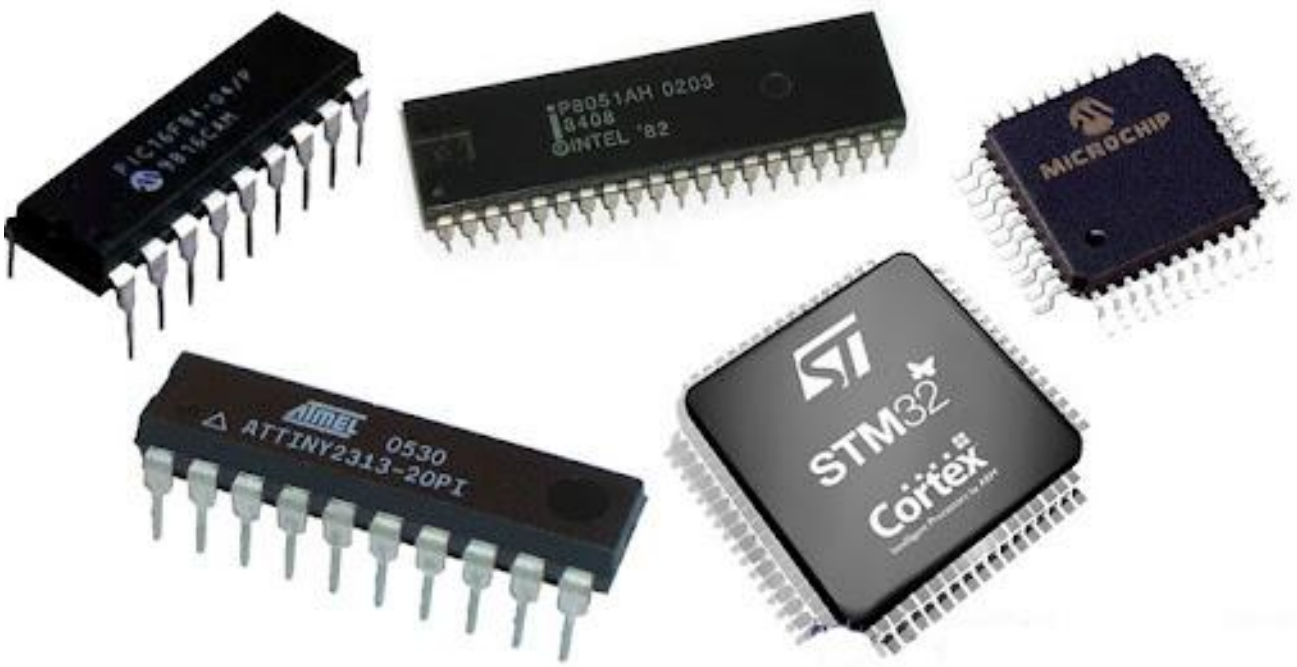

Рисунок 1.2 – Мікроконтролери

Одна з основна причин що стимулювала появу мікроконтролерів – впровадження засобів обчислювальної техніки у всі сфери людської діяльності, у тому числі і в промисловій. Це спричинило мініатюризацію і зниження вартості виробів. Прогрес в технології створення мікросхем в значній мірі сприяв цьому.

У 1967 р. фірма Texas Instruments випустила 1 калькулятор на інтегральній мікросхемі, який і запустив процес мініатюризації засобів обчислювальної техніки. Проте мікросхеми для калькуляторів створювалися стосовно вимог кожного конкретного замовника. Ці процесори для калькуляторів були не потужними і не відповідали вимогам, що пред'являються до обчислювальних потужностей [21].

Працюючи над замовленням однієї з японських фірм над розробкою програмованих калькуляторів, фахівець фірм Intel запропонував нову концепцію проектування мікропроцесорів, яка полягала в створенні мікропроцесорів загального призначення, здатних виконувати абсолютно будь-які арифметико-логічні операції. У 1969 р. фірмою Intel був представлений перший в світі спеціалізований інтегральний чіп Intel 4004 промисловий серійний випуск з 1971 р., який по суті, був 4-розрядним програмованим мікропроцесором. Мікропроцесор Intel 4004 містив приблизно 2250 транзисторів і міг виконувати приблизно 60 тис. операцій в секунду. Продуктивність Intel 4004 була дуже низькою, щоб він міг в повній об'ємі виконувати функції процесора. Проте він міг служити основним мікропроцесором для калькуляторів. Цікавий факт, що назва фірми Intel походить від слів "Integrated Electronics" (інтегральна електроніка). Можливо, смисловий зміст назви Intel грає головну роль в тому, що фірма з моменту свого створення (1968 р.) і по сьогодні (сьогодні у фірмі Intel працюють близько 65 тис. співробітників) займає лідируюче положення серед світових виробників мікросхем.

У 1976 р. створений перший 8-розрядний мікроконтролер, на кристалі якого були інтегровані основні елементи мікропроцесорної системи, що управляє : процесор, пам'ять типу ROM і RAM, порти введення/виведення і таймери.

У 1978 р. з'явився 16-розрядний мікропроцесор Intel 8086 (вітчизняний аналог К1810вм86) - перший мікропроцесор масового використання. З серійного випуску Intel 8086 почалося масштабне проникнення персональних комп'ютерів у всі сфери діяльності. Створенню мікропроцесора Intel 8086 передував випуск 8-розрядних мікропроцесорів Intel 8008 (1972 р.) і Intel 8080 (1974 р.) [22].

Створення фірмою Texas Instruments в 1982 р. першого однокристального програмованого цифрового сигнального процесора (DSP) TMS32010. У Раніше , із-за складності алгоритмів і специфіки математичних операцій, використовуваних при створенні інтелектуальних приводів електричних двигунів архітектура сигнальних процесорів використовується як основа для сучасних універсальних програмованих DSP-контролерів, широко вживаних в системах управління електродвигунами всіх типів.

Структурна схема будь-якої електронно-обчислювальної машини містить наступні блоки: процесор, що складається з арифметико-логічного пристрою (ALU), схем управління і регістрів; пам'ять; периферійні пристрої введення/виведення даних.

Спочатку блоки, що входять в ЕОМ, створювалися на базі стандартних дискретних логічних мікросхем, які виконували порівняно прості функції.

Тому самі електронно-обчислювальної машини мали великі габарити (наприклад, СУПЕРЕОМ Сгау-1 складалася з 300 тис. мікросхем і займали об'єм приблизно трьох кубічних метрів).

Успіхи інтегральної технології привели до появи великих і надвеликих інтегральних схем (БІС і СБІС) з розміщенням до 10 і 100 тисяч, а в даний час - 10 мільйонів транзисторів на одному кристалі. Високий ступінь інтеграції БІС і СБІС дозволила в одній мікросхемі реалізувати окремі блоки ЕОМ, наприклад, процесор.

У мікросхемах перших мікропроцесорів (наприклад, Intel 8080) були реалізовані тільки сам процесор і додаткові пристрої, що здійснюють управління обміном даними із зовнішньою пам'яттю і пристроями введення/виведення даних.

Проте, окрім пристроїв, що входять до складу процесора, на кристалі СБІС (БІС) можуть бути виконані пам'ять для зберігання програм (ROM), даних або проміжних результатів (RAM), периферійні пристрої введення/виведення. Такі СБІС (БІС) відносяться до класу однокристальних мікро ЕОМ. Однією з мікросхем, що серійно випускаються, однокристальна мікро ЕОМ стала Intel 8048 (18048). Однокристальні мікро ЕОМ почали активно використовуватися там, де

була потреба в нескладній цифровій обробці даних: побутових приладах, простих системах управління, контролю і т. д. Окрім однокристальних мікро ЕОМ для цифрової обробки в подібних системах можна використовувати і замовлені, спеціалізовані інтегральні мікросхеми (Application Specific Integrated Circuit - ASIC).

Ідея інтеграції на одному кристалі спільно з процесором і пам'яттю великої кількості стандартних пристроїв, загального призначення втілилася в появу мікроконтролерів. Одним з перших мікроконтролерів, що серійно випускаються, можна вважати Intel 8051. Незабаром мікроконтролер 8051 завоював популярність. В даний час мікроконтролери з набором команд 8051 випускаються 10 фірм-виробників Analog Devices, Atmel, Dallas, Semiconductor, Oki, Philips, Infineon Technologies, Silicon Storage Technologies, Temic і багатьма іншими.

На відміну від універсальних мікропроцесорів, призначених в основному для математичної обробки даних, мікроконтролери мають розширений набір вбудованих периферійних пристроїв. Це можуть бути додаткові блоки пам'яті типу RAM, ROM, EPROM, EEPROM або FLASH. Периферійні пристрої різного призначення: універсальні таймери і таймери спеціального призначення; "сторожові" таймери; контролери зовнішніх інтерфейсів (UART, USART, SPI, SCI, PC, j 1850, USB- або CAN-шини) і рідкокристалічних дисплеїв; монітор джерел живлення; аналогові і цифрові компаратора; схему перезапуску; аналого-цифрові (АЦП) і цифро-аналогові (ЦАП) перетворювачі та ін. Таким чином, мікроконтролери містять всі периферійні пристрої, потрібні для створення закінчених вбудованих систем управління, контролю і що важливо для подальшого розуміння, стандартні пристрої, які у разі використання мікропроцесора в системі виконувалися б на базі додаткових зовнішніх по відношенню до мікропроцесору мікросхем. У мікроконтролерах, велика частина периферійних пристроїв виконані на одному кристалі з процесором, що додає системам на базі мікроконтролерів значно більшу гнучкість і універсальність. Прикладом подібних високоінтегрованих мікроконтролерів можуть служити мікроконвертери ADcU812/814/816/824/834 (Analog Devices), H8/300L (Hitachi),

51XA-G49 і 51XA-G3 (Philips), MC68HC05/08/ 11/12/16, мікроконтролери сімейств Ат89/at 90 (AVR) /AT tiny/AT mega (Atmel), C16x (Infineon), H8/300 і (Motorola), PICmicro (Microchip), MSP430F (Texas instruments) і багато інших [23].

Що таке мікроконтролер

Перш за все, розберемося з самим поняттям «мікроконтролер». Мікроконтролер можна визначити як мініатюрний комп'ютер на базі одного-єдиного чіпа що включає, крім процесора ряд допоміжних елементів, таких, як ОЗУ, ППЗУ, таймер, і т.д. Мікроконтролер призначений для виконання будь-яких заздалегідь визначених завдань.

Найпростіше порівнювати мікроконтролер з ПК. Як і ПК, мікроконтролер має процесор, ОЗУ і постійну пам'ять. Проте, на відмінність від персонального комп'ютера, всі ці елементи розташовані на одному чіпі.

ПК створений для того, щоб виконувати великі завдання загального призначення. Наприклад, ви можете використовувати комп'ютер, для набору тексту, написання програм, зберігання і запуску відео файлів, серфінг по Інтернету, і т.д. Мікроконтролери призначені для виконання конкретних завдань наприклад комутації кондиціонера коли температура в приміщенні опускається нижче певного значення, або навпаки вище.

Існує декілька популярних сімейств мікроконтролерів, які використовуються для різних цілей. Найбільш поширеними з них є сімейства мікроконтролерів 8051, PIC і AVR. І останні в даний час набирають особливу популярність на рахунок проекту Arduino «Мікроконтролери це просто». В наслідок чого документації на них достатньо багато, існує велике співтовариство по підтримці розробників-початківців і як наслідок популяризації, ціни на них значно менше ніж у конкурентів.

Історія сімейства

Сімейство мікроконтролерів AVR було засноване в 1996 р. компанією Atmel, а розробниками архітектури є Alf-Egil Bogen і Vegard Wollan. Так і сформувалась назва сімейства – від перших букв імен розробників – A і V, і

першої букви абревіатури RISC – типу архітектури, на якій заснована архітектура мікроконтролерів. Також це скорочення часто розшифровують як Advanced Virtual RISC (модернізований ефективний RISC).

Одним з першим мікроконтролером в серії був AT90S8515, проте першим мікроконтролером, випущеним на ринок, став AT90S1200. Це відбулося в 1997 р.

На сьогоднішній день, на ринку доступні 3 лінійки мікроконтролерів: TINYAVR – з невеликим об'єм пам'яті, проте має невеликі розміри, і підходить для самих елементарних завдань.

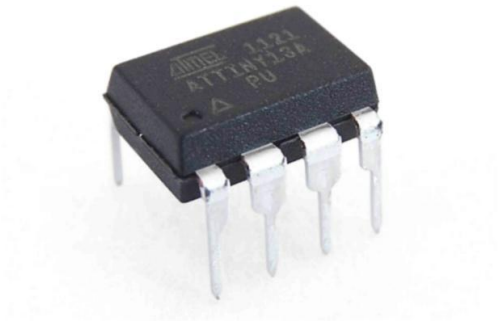

Рисунок 1.3 – Зовнішній вигляд мікроконтролера TINYAVR

MEGAAVR – найбільш популярна лінійка що має достатній об'єм вбудованої пам'яті (до 256 КБ) та додатковою периферією що служить для завдань середньої і високій складності.

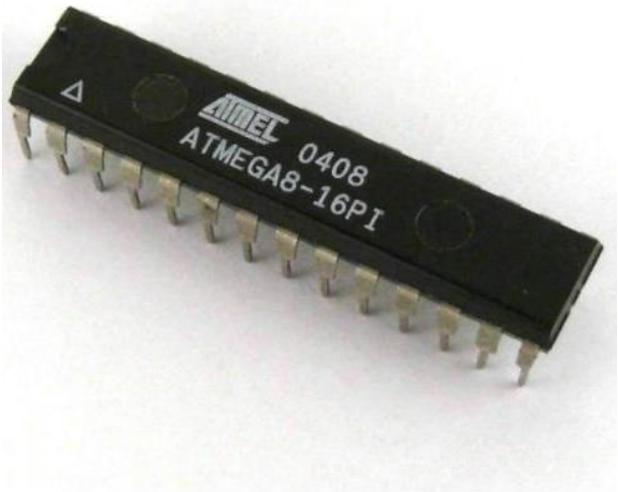

Рисунок 1.4 – Зовнішній вигляд мікроконтролера MEGAAVR

XMEGAAVR – використовується в просунутих комерційних завданнях, що вимагають великого об'єму пам'яті і високої продуктивності.

Рисунок 1.5 – Приклад мікроконтролера XMEGAAVR

Таблиця 1.1 – Порівняльні характеристики різних лінійок

Назва серії	Число контактів	Об'єм флеш-пам’яті	Особливість
TINYAVR	6-32	0,5 – 8 КБ	Невеликий розмір
MEGAAVR	28-100	4-256 КБ	Периферійні пристрої
XMEGAAVR	44-100	16-384 КБ	Система переривань, підтримка DMA

1.4 Особливості сімейства

Раніше всього мікроконтролери цій серії є високошвидкісними. Багато інструкцій процесор мікроконтролера виконує за один цикл. Мікроконтролери AVR приблизно в 4 рази швидші, ніж PIC. Крім того, вони споживають значно менше енергії і можуть працювати в декількох режимах економії енергії.

Багато контролерів AVR 8-розрядні, хоча є і 32-розрядний різновид контролерів AVR32. Крім того, як було зазначено вище, AVR належать до типу

RISC-мікроконтролерів. Архітектура RISC (Complex Instruction Set Computers) означає, що набір інструкцій, які може виконувати процесор пристрою, урізаний, але, в той же час, дана архітектура дає перевагу в швидкодії. Аналогом архітектури RISC є архітектура CISC (Complex Instruction Set Computers).

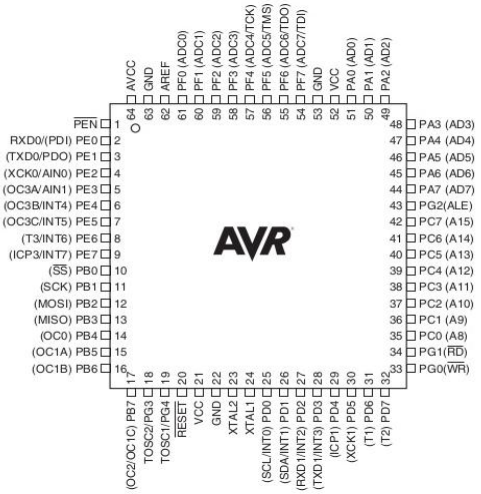

Рисунок 1.6 – 32-розрядний різновид контролерів AVR32

8-бітність контролера має на увазі, що він здатний передавати і приймати 8-бітові дані. Надані регістри введення/виведення також 8-бітоні.

Архітектура контролера базується на регістрах. Це означає, що для зберігання початкових значень операції і результату в контролері застосовуються регістри.

Процесор контролера отримує дані з двох вхідних регістрів, виконує логічну операцію і зберігає результат у вихідному регістрі. Все це займає 1 виконуваний цикл.

Архітектура контролера

Контролер AVR має всього лише 32, 8-бітових регістра загального призначення. Протягом циклу процесор бере дані з двох регістрів і відправляє їх в арифметико-логічний пристрій (АЛУ), який проводить обчислення над даними і поміщає їх у випадковий регістр. АЛУ може виконувати як арифметичні, так і логічні дії над регістрами. Також АЛУ може виконувати і дії з одним регістром. При цьому контролер не має операнда-акумулятора, на відміну від контролерів

сімейства 8051 – для операцій можуть використовуватися будь-які операнди, і результат операції також може бути поміщений в будь-який операнд.

Мікроконтролер відповідає вимогам Гарвардської обчислювальної архітектури, згідно якої комп'ютер має незалежну пам'ять для програм і даних. Тому у той час, поки виконується одна операція, відбувається попереднє витягання з пам'яті наступної операції.

Контролер здатний виконувати одну операцію за цикл. З цього виходить, що якщо тактова частота мікроконтролера складає 1 Мгц., то його продуктивність складе 1 млн. операція в секунду. Чим вище тактова частота контролера, тим вище буде його продуктивність. Проте при виборі тактової частоти контролера слід дотримуватись компромісу між його швидкістю і енергоспоживанням.

Крім флеш-пам’яті і процесора контролер має такі периферії, як порти введення-виведення, аналого-цифровий перетворювач, таймери, комунікаційні інтерфейси – I2C, SPI і послідовний порт UART. Вся ця периферія можуть контролюватися на програмному рівні.

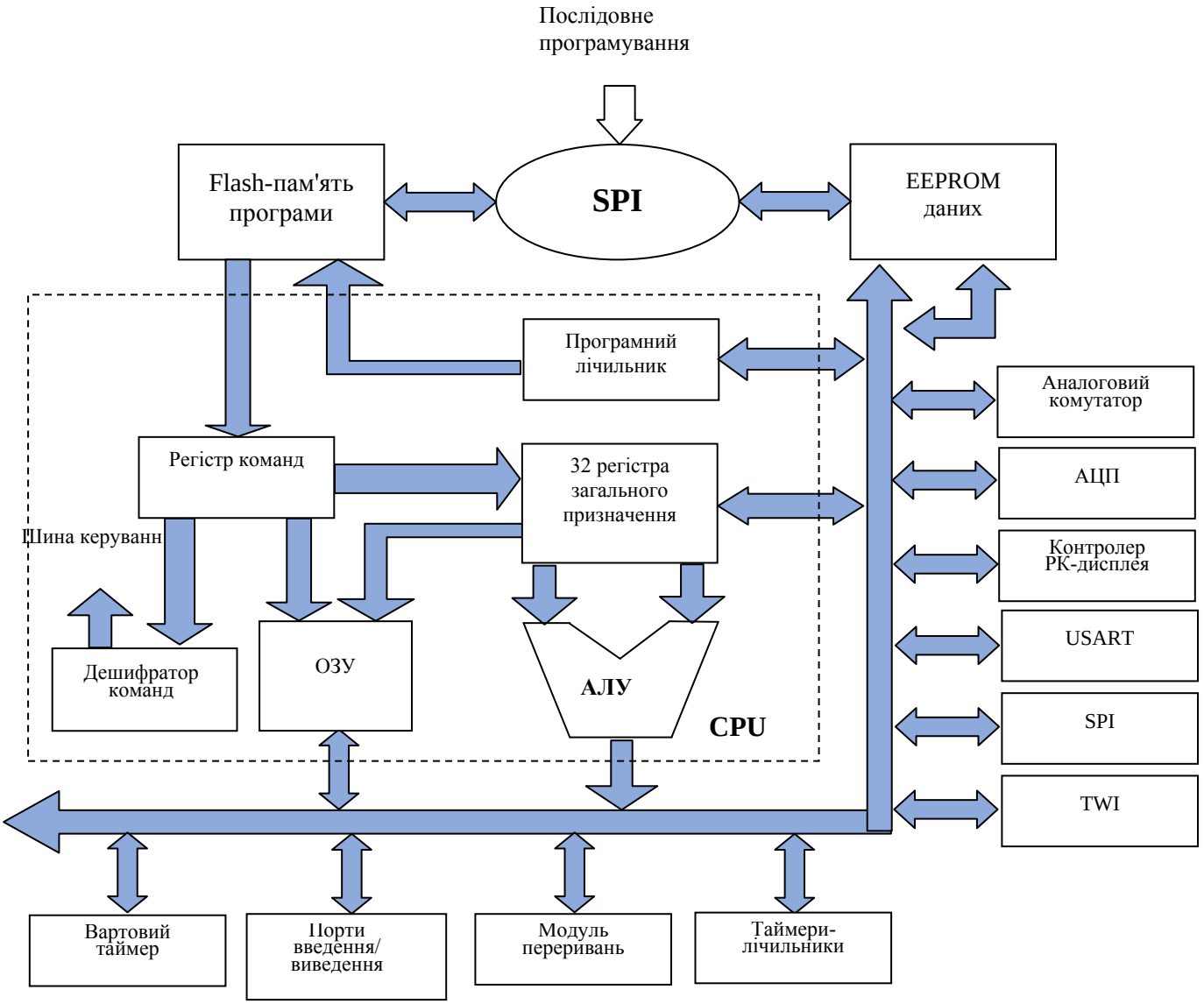

Рисунок 1.7 – Типова архітектура мікроконтролерів AVR

Програми для мікроконтролера

Як вже згадувалося вищим, мікроконтролер подібний до ПК, а з цього витікає, що, як і ПК, AVR також може виконувати яку-небудь програму хоча і всього одну в певний момент часу.

Програма мікроконтролера може зберігатися у внутрішній пам'яті контролера і складається з серії елементарних команд, які вибирають дані і здійснюють з ними якісь операції. У деяких випадків це означає зчитування вхідних даних, перевірка їх стану і виведення відповідних вихідних значення. Іноді потрібна зміна даних і здійснення з ними операцій, а також передача даних якому-небудь зовнішньому периферійному пристрою, наприклад, індикатору, або

послідовному порту.

Для таких простих завдань використовуються набори елементарних команд, кожна з яких має аналог на доступнішій людському сприйняттю мові. Тому найбільш поширеним способом написання програм для контролера є написання їх на мові машинних команд.

Перевагою машинних команд є дуже швидкий компактний і ефективний код, але створення таких програм одночасно вимагає і ґрунтовних знань роботи процесора мікроконтролера, ручного управління пам'яттю і контролю структури програми. Тому часто для написання програм використовуються інші мови високого рівня, такі, як Basic, C, і Java. В цьому випадку завдання по контролю структури програми і управлінню пам'яттю бере на себе компілятор який створює прошивки. Окрім цього, часто використовувані функції можуть бути при цьому поміщені в спеціальні бібліотеки і витягуватися з них в міру необхідності [24].

Мікроконтролери сімейства AVR на сьогоднішній день широко використовуються в комп'ютерах, для автоматизації управління електронною технікою, різними електроприладами і механізмами, що використовуються в промисловості, бізнесі, а також побутових потребах. Невисока вартість, великий асортимент і широкі можливості мікроконтролерів цієї серії сприяли їх великій популярності.

1.5 Вибір платформи мікроконтролера

Розглянемо як варіант доступний і дуже поширену платформу Arduino Nano побудовану на по досить доступному мікроконтролері ATMEGA328-PU/

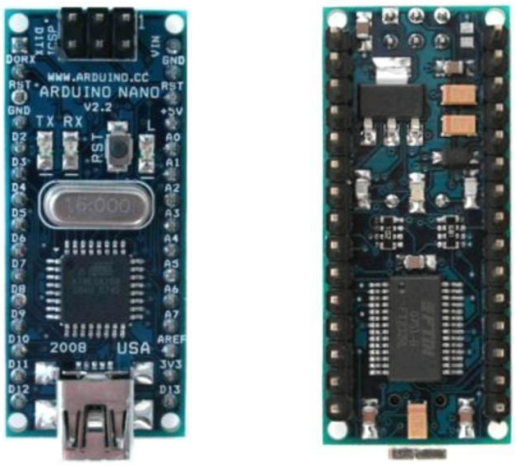

Рисунок 1.8 – Arduino Nano

1.5.1 Загальні відомості

Платформа Nano побудована на основі мікроконтролері ATmega328 має невеликі розміри і може застосовуватися в різних проектах. Вона має схожу з Arduino Duemilanove структуру, проте відрізняється зовнішнім виглядом. Відмінність полягає в браку основного роз'єму постійної напруги живлення і роботі через кабель MINIUSB.

Таблиця 1.2 – Основні параметри платформи

Мікроконтролер	Atmel ATmega328
Робоча напруга	5 В
Вхідна напруга	7-12 В
Вхідна напруга	6-20 В
Цифрові Входи/Виходи	14
Аналогові входи	8
Постійний струм через виводи	30 мА
Постійна пам'ять	32 Кб (2 Кб для завантажувача)
ОЗУ	2 Кб
EEPROM	1 Кб
Стандартна частота	16 Мгц
Розміри	1,9 см x 4,3 см
Живлення:	5 В

Живиться Arduino Nano через інформаційний кабель MINIUSB, або від нестабілізованого джерела живлення 6-20 В (вивід 30), або від стабілізованого 5 В (вивід 27), зовнішнього джерела постійного напруга. Автоматично використовується джерело живлення з найвищим напругою живлення.

Мікросхема FTDI FT232RL отримує живлячу напругу, тільки якщо платформа отримує напругу живлення від USB. Що дозволяє працювати від зовнішнього джерела, напруга 3.3В відсутня, оскільки вона формується мікросхемою FTDI, при цьому світлодіоди RX і TX спалахують тільки при появі сигналів на виводах 0 і 1.

1.5.2 Входи і Виходи мікроконтролера

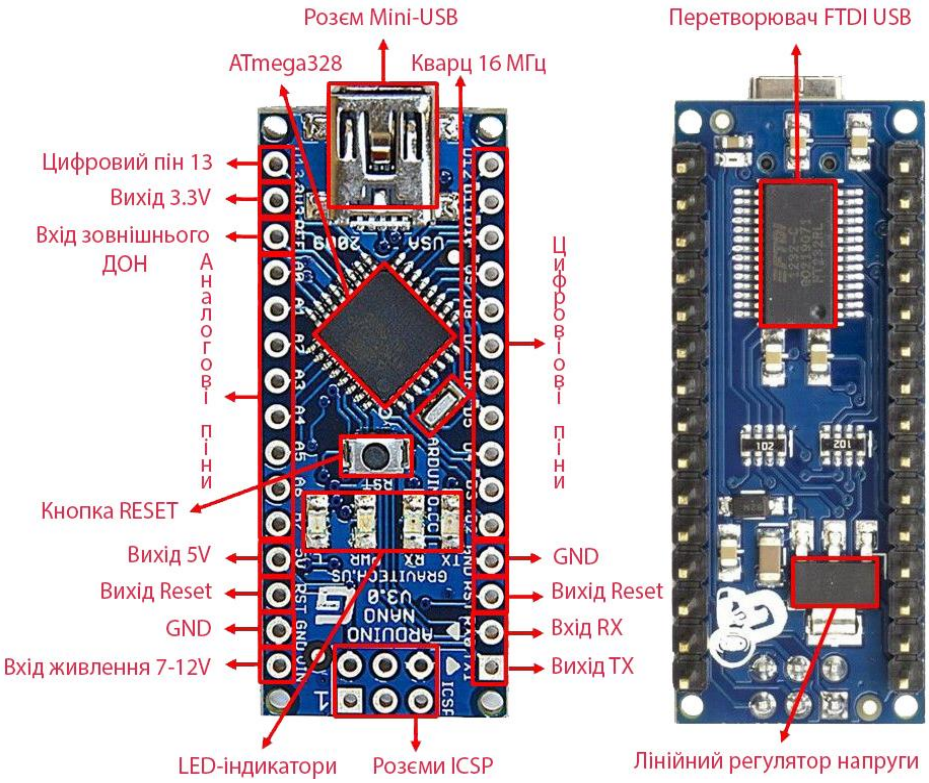

Рисунок 1.9 – Виводи мікроконтролера

Кожен з 14 цифрових виходів використовуючи функції pinMode(), digitalWrite(), і digitalRead(), може компілюватися в програмі, як вхід, так і вихід. Виводи працюють тільки за наявності живлячої напруги 5В. Кожен вивід має підтягуючий резистор опором приблизно 10-40 кОм і може видавати на виході до 30 мА. Деякі з наявних виводів, мають специфічні функції:

Послідовний порт Serial: 0 (RX) і 1 (TX). Використовуються для обміну, отримання і передачі даних.

Зовнішні переривники: 2 і 3. Дані виводи також можуть бути використані на переривання при низькому значенні або на граничному, задньому фронті, або при зміні значення.

ШИМ: 3, 5, 6, 9, 10, і 11. Кожен з виводів задає ШИМ із значенням до 8 біт за рахунок спеціальної функції analogWrite().

SPI: 10 (SS), 12 (MISO), 11 (MOSI), 13 (SCK). За допомогою даних контактів реалізований зв'язок SPI.

На даній платформі розміщено вісім аналогових контактів, кожен має дозвіл 10 бітний. Стандартно виводи володіють діапазоном вимірювання до 5 В відносно

нуля, не дивлячись на це є здатність змінити верхнє значення за допомогою спеціалізованого функціонала analogReference(). Так само є контакти із специфічними функціями:_

I2C: A4 (SDA) і A5 (SCL). За допомогою команд виводів можуть здійснюється зв'язок по інтерфейсу I2C (TWI).

Також на платі розведені пара виводів:

AREF. Необхідно для окремо живлення аналогових входів. Використовується з командою analogReference().

Reset. Для перезавантаження мікроконтролерного процесора. Зазвичай, як правило, застосовується для приєднання кнопки перезапуску на платах розширення, що надає доступ до кнопки безпосередньо на самій платі.

1.5.3 Зв'язок

На даній платформі встановлена безліч пристроїв, для реалізації зв'язку з комп'ютером, а також іншими пристроями Arduino або мікроконтролерами іншого сімейства. ATmega328 має послідовний інтерфейс UART TTL (5 В), що реалізований контактами RX і TX. Розташована на платі дана мікросхема направляє даний інтерфейс через USB, а драйвери надають емульований порт COM програмі на персональному комп'ютері. Монітор послідовної шини (Serial Monitor) програми Arduino дозволяє відправляти і приймати текстові значення при підключенні. Світлодіоди RX і TX платформи, указуватимуть на обмін інформації по TX і RX.

ATmega328 також підтримують поширені інтерфейси I2C (TWI) і SPI. У Arduino є бібліотека Wire що надає зручний доступ до шини I2C.

1.5.4 Програмування

Платформа може програмуватися на мові високого рівня C++, але також для зручності початківців програмується за допомогою спеціального спрощеного ПО Arduino 1.6.5. Для цього в меню Інструменти > Board вибираємо «Arduino Nano, Мікроконтролер ATmega328 поставляються зі всім необхідним, що спрощує запис програм без використання зовнішніх програматорів. Зв'язок реалізований по протоколу STK500.

Є також можливість не удаватися до завантажувача і програмувати платформу по внутрішньосхемному протоколу виводами блоку ICSP [25].

1.5.5 Автоматичне (програмне) перезавантаження

Arduino Nano спроектований таким чином, що перед записом подальшого програмного коду, перезавантаження реалізоване самою програмою, а не натисненням спеціальної кнопки на платі. Одна з ніжок FT232RL, здійснює управлінням протоколом даних (DTR), підключена до виводу перезавантаження ATmega328 через захисний конденсатор 100 нФ. Запуск даної лінії, перезавантажує процесор мікроконтролера. Програма Arduino, задіюючи даний функціонал, завантажує код натисненням кнопки Завантажити в самому середовищі розробки. Подача сигналу низького рівня по лініях DTR синхронізована з початком поточного запису програми, що скорочує час завантажувача.

Функція застосовує ще одне призначення. Перезавантаження Arduino відбувається постійно при підключенні до програми Arduino IDE 1.6.5 на персональному комп'ютері. Наступні декілька секунд після перезавантаження запускається завантажувач. Під час прошивки відбувається тайм-аут декількох байтів початкового коду щоб уникнути отримання помилкових даних. Якщо проводиться одноразова відлаогодження прошивки, записаної в плату, або введення яких-небудь інших даних при кожному запуску необхідно впевниться, що програма на комп'ютері чекає протягом першої секунди перед передачею даних [26].

Виводи платформи мікроконтролера

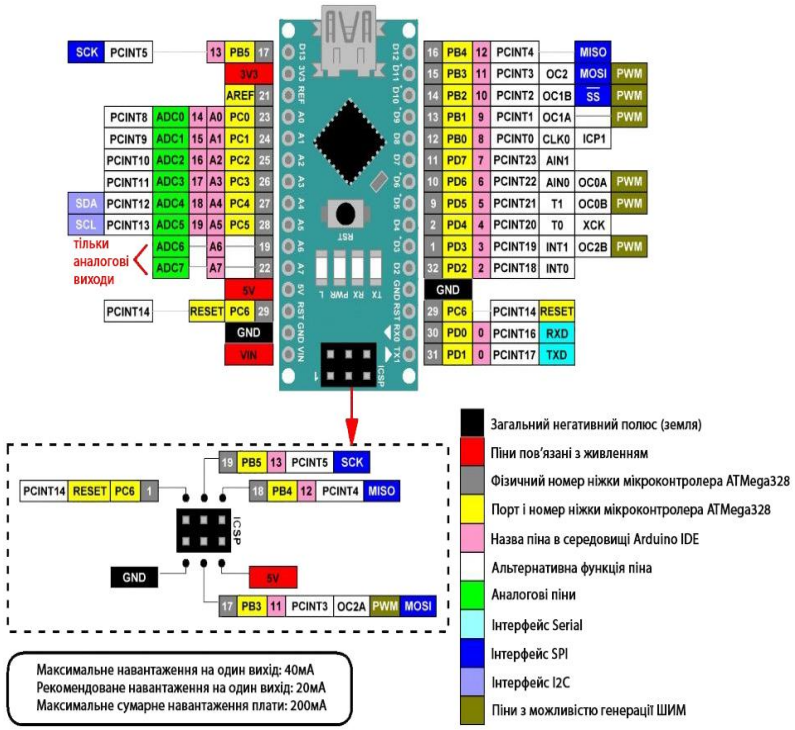

Рисунок 1.10 – Позначення виводів МК

Таблиця 1.3 – Виводи Arduino nano

№ виводу	Найменування	Тип	Опис
1 - 2, 5 - 16	D0 - D13	Введення - вивід	Цифрове уведення-виведення портів D0 - D13
3, 28	RESET	Введення	Скидання (активний рівень - низький)
4, 29	GND	Живлення	Загальний живлення
17	3V3	Вивід	+3,3 У з мікросхеми FT232
18	AREF	Введення	Опорна напруга АЦП
19 - 26	A0 - A7	Введення	Аналоговий вхід, канали 0 -7
27	+5V	Введення - вивід	+5 У на вивід від регулятора на платі, або +5 У на вхід від зовнішнього джерела живлення

1.6 Програма керування верстатом з ЧПУ

Автономний верстат з ЧПУ, що розробляється, працюватиме під управлінням спеціального програмного забезпечення, яке повинне уміти зчитувати параметри і траєкторії обробки розробленої заготівки. Для цього необхідно вибрати на якій мові писатимуться всі необхідні параметри.

На сьогоднішній день існує більше 100 мов для написання програм

верстатам з ЧПУ, але до цих пір не існує єдиної мови, яка в достатній мірі задовольняла б всім необхідним вимогам. Мови відрізняються ступенем автоматизації і ступенем спеціалізації, найпоширеніша мова G-code.

G-код – умовне найменування мови для програмування пристроїв з числовим програмним управлінням. Ця мова була створена компанією Electronic I.A. в кінці 1950-х, початку 1960-х. Остаточне доопрацювання і схвалення відбулося в лютому 1980 року і було прийняте стандартом RS274D. Комітет стандартів ISO прийняв G-код стандартом ISO 6983-1:2009. Державний комітет із стандартів СРСР прийняв його як ГОСТ 20999-83. У радянській нормативній документації G-код іменується кодом ІСО 7-бит (ISO 7 bit). Створений для передачі інформації ЧПУ у вигляді коду написаного машинною мовою, аналогічно як і коди РС8С або AEG.

Створюючи систем з числовим програмним управлінням, застосовують програмне забезпечення управління верстатом, для якого написана програма обробки деталей як основна команда управління, G-код використовується у вигляді основної мови програмування, збільшуючи його на свій розсуд. Нижче представлений детальний опис G-коду з прикладами

G00 – переміщення інструменту при максимальній швидкості. Код G00 застосовується для переміщення робочого механізму від одного оброблюваної ділянки до іншого. Застосовується для максимально швидкого переміщення інструменту для різання до потрібного місця обробки деталі або до безпечної позиції. Для виконання обробки деталі ніколи не застосовується прискорене переміщення інструменту, тому що швидкість руху виконавчого механізму верстата досить висока і змінюється впродовж всієї роботи . Код G00 замінюється іншою командою при використанні наступного коду: G01, G02, G03.

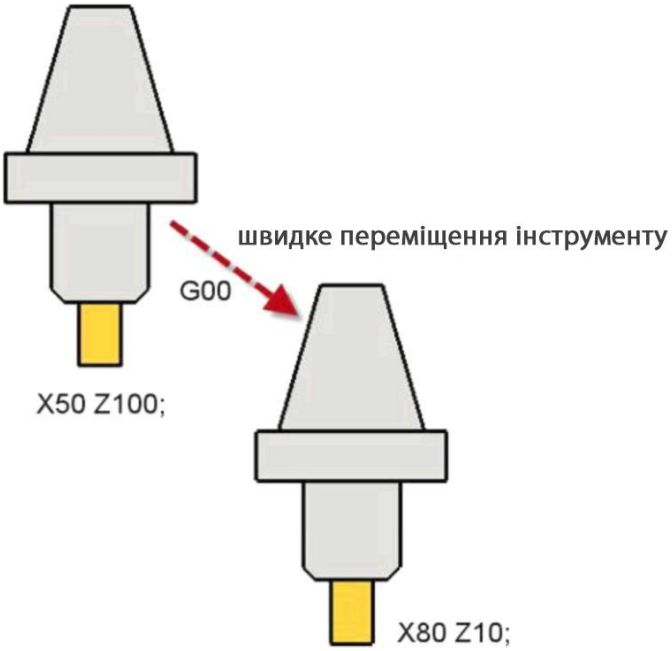

Рисунок 1.11 – Приклад прискореного переміщення інструменту.

G00 X80 Z10 – переміщення в точку з координатами (10; 80)

G01 – лінійна інтерполяція. Код G01 – виконавча команда що дозволяє здійснити переміщення виконавчого механізму прямій із заздалегідь встановленою швидкістю. Швидкість переміщення інструменту задається адресою - F. Код G01 відміняється за допомогою коду G00, G02HG03.

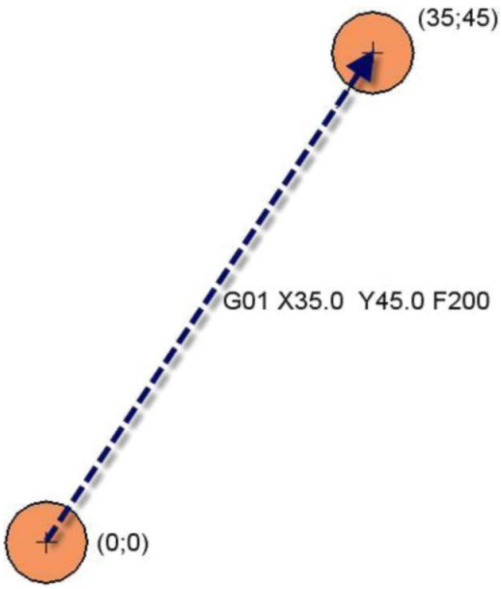

Рисунок 1.12 – Приклад забезпечення лінійної інтерполяції інструменту.

G01 X35 Y45 F200 – переміщення інструменту в точку з координатами (35; 45) по прямій із заданою швидкістю подачі 200 міліметрів в хвилину

G02 – кругова інтерполяція (дуга в напрямі за годинниковою стрілкою). Код G02 використовується для виконання інтерполяції з кругом, іншими словами для пересування інструменту по колу у напрямі годинникової стрілки із заздалегідь заданою швидкістю ходу. Швидкість переміщення заздалегідь приймається адресою - F. Код G02 відміняється кодами G03, G00, G01.

G03 – переміщення по колу (проти годинникової стрілки). Код G03 використовується для переміщення виконавчого інструменту по колу , іншими словами для позиціонування виконавчого інструменту по колу проти годинникової стрілки із заздалегідь встановленою швидкістю. Швидкість переміщення задається адресою - F. Код G03 знімається кодами G02, G01,G01.

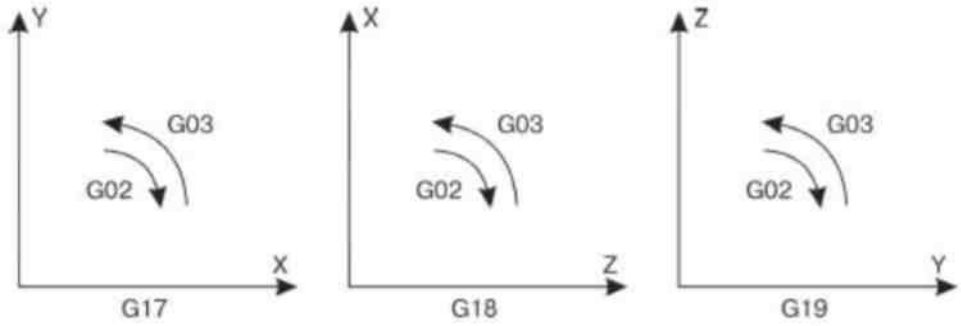

Рисунок 1.13 – Інтерполяція в колі та різних площинах

G04 – пауза. Виконуваний параметр задає витримку з інтервалом часу таким, що задається заздалегідь виконується кодом G 04 . Цей програмний код виконується разом з P - а так само адресою - X, він задає час паузи. Час знаходиться в інтервалі між 0.001 і 99999999 секундами. Код G04, P - або адреса - X виконуються в одному заході, він не повинен мати ніяких переміщень по осі координат.

Коли необхідно визначити час затримки, використовується команда P, в цьому випадку не можна програмно використовувати десяткову крапку. Адреса з індексом P призначає витримку за часом в мілісекундах, а індекс X призначає витримку в секундах. Коли командний код G04 встановлюється без часу затримки імпульсу, вона приймається верстатом як команда для точної зупинки інструменту.

Приклад:

G04 X1.6 – затримка 1.6 секунди;

G04 P2500 – затримка 2.5 секунд.

G09 – точна зупинка. Із-за збільшення швидкості і відповідно її уповільнення щодо осьових переходів виконавчих механізмів верстата , неможливо точно виконати зріз кутових кромок при переміщенні з різання від однієї до іншої точки. Цей недолік обробки виробу виглядає як в притупленні і закругленні кути.

Допустимо, коли ви обробляєте по периметру контур у вигляді прямокутника і намагаєтеся зробити кромку в кута дуже гострою (Рисунок 1.14). Працюючи в звичайному режимі, напевно, що при переміщенні від руху по осі X до переміщення по осі Y вийде скруглення кромок (Рисунок 1.15). Дуже сильно цей шкідливий ефект видно коли верстат працює при дуже високих швидкостях подачі і при роботі у величезних центрах обробки.

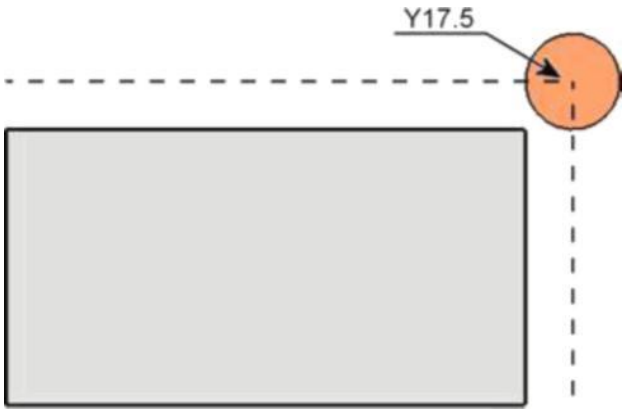

Рисунок 1.14 – Отримання гострої кромки в правому кутку

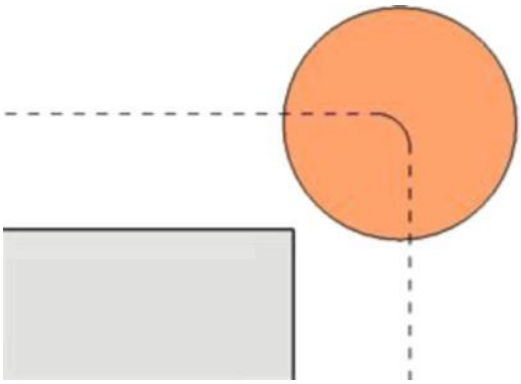

Рисунок 1.15 – Обрізання методом скруглення кромки

G09 це код немодальний, він використовується коли необхідне узгодження поточної траєкторії інструменту верстата із заданою заздалегідь траєкторією. При переміщенні з одного напрямку до іншого верстат зробить кінцеве переміщення в задану позицію.

G09 указується разом із заданою координатою, для якої буде виконано точне позиціонування. Що управляюча програма що гарантує отримання гострої кромки кута прямокутного контуру, виглядатиме так:

```
%
O0005
N100 G21
N102 G0 G17 G40 G49 G80 G90
N104 T1 M6
N106 G0 G90 G54 X30. Y-22.6 S1000 M3
N108 G43 H1 Z100.
N110 Z10.
N112 G1 Z-2. F90.
N114 Y-12
N116 G09 Y17.7
N118 X-25.
N120 X-35.
N122 Z8.
N124 G0 Z90.
N126 M5
N132 M30
%
```

Коли інструмент приходить в координату Y17.7, то верстат виконує точну зупинку. Час затримки в цій позиції визначається значенням параметра системи.

G10 – введення даних управління верстат. Команда G10 встановлює і змінює систему координат в роботі і вводить задані параметри для коректування інструменту верстата за допомогою програми управління.

По бажанню ввести інші параметри для коректування за допомогою програми управління, їх необхідно розмістити на початку коду. Таким чином, відбувається узгодження параметрів коректування і зміни в програмі управління.

Як правило для введення параметрів коректування використовується наступний вид коди:

G10 L11 P_ R _ ;

G10 – режим введення параметрів;

L11 – настройка коректування інструменту верстата ;

P – вибір коректування, який треба змінити;

R – значення, що вноситься.

Якщо команда G10 виконується разом з командою G90, то параметри в регістрах коректування будуть змінені. Якщо G10 працює спільно з командою G91, то параметри в коректорах складаються або віднімаються із значенням R. Наприклад, G10 G90 Lll P13 R91.14 замінює дійсне значення в коректуванні № 13 на нове значення 91.14.

При необхідності встановити або змістити робочу координатну систему, виконується наступна команда:

G10 L2 P _ X _ Y _ Z _ ;

де G10 – режим введення даних, його включення; L2 – установка робочої стандартної системи координат; P – ініціалізація системи координат; X, Y, Z – параметри, що вибирають задане розташування робочої системи координат.

Попереднє значення G10 буде модальним і буде ним до того часу, поки не надійде команда відміни G11. Перш ніж використовувати G10 необхідно уважно вивчити документацію до пристрою, тому що параметр G10 буває різноманітним за значенням.

G11 – означає, закінчення режиму введення даних у верстат. За допомогою даної команди G11 припиняє свою дію G10 для початку режиму введення параметрів у верстат.

G15 – відміняє режим введення даних полярних координат. За допомогою команди G15 ви деактивуєте установку в полярній системі координат і переходите

назад до введення даних з допомогою систему координат в прямокутній формі.

G16 – включає режим полярних координат. Дана команда G16 виконуватиметься в полярній системі координат. У зв'язку з чим застосовується позиціонування визначуване залежністю відстані і кута від точки нуля даної координатної системи або від поточної позиції інструменту.

Використання верстата в полярній системі координат вирішується в кожній з трьох осей системи. Кодом програми G17 ви здійснюєте переміщення по осях XY, з кодом програми G18 – по осях XZ, а за допомогою коди програми G19 – по осях YZ.

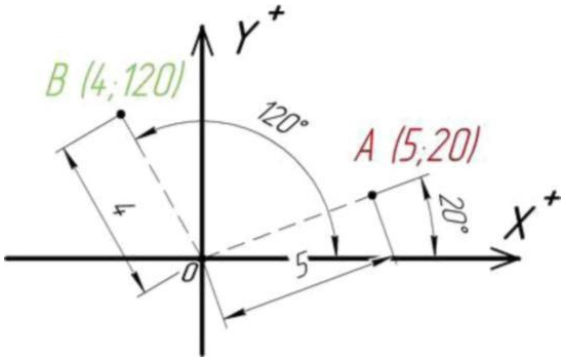

Рисунок 1.16 – Полярні координати: точка А (5;20) і точка В (4; 120)

Коли вибирається дана площина осей XY, то X - адреса указує на радіус, а Y вибирає щодо осі X його кут. Коли використовується вісь XZ, то адреса X позначає радіус, а адресу Z позначає відносний кут осі X. Коли задана вісь YZ, то адресне поле - Y задає радіус, а Z задає відносний кут осі Y. Кут відлічуваний проти годинникової стрілки, є позитивним.

Переміщення в полярній системі координат, на яку вказує команда G90, що діє, виконуються строго щодо нульової точки справжньої системи координат в роботі. При виконанні коду програми G91, переміщення в полярній системі координат виконуються з відношенням до поточного місцеположення. Як абсолютні або відносні змінні, можливе завдання параметрів кутів і радіусів. Полярне переміщення визначається кутом нульової точки системи в роботі і відстанню від реального розташування інструменту верстата.

Виключити перетворення полярних координат в прямокутну дозволяє підготовча функція

G16

G90 G17 G16

G81 G98 X4 Y30 Z-2 R0.7 F45

Y60

Y90

G15 G80

Код програми G16 є модальна функція, виходячи з цього вона діє до того часу, поки не вимкнеться кодом G15.

G17 – вибір осі XY. Попередній код G17 необхідний для осі XY як справжня вісь, що діє. Вісь XY стає робочою для виконання інтерполяції в круговому форматі, зміни координатної системи і циклів обробки деталі в круговій проекції.

G18 – команда вибору площини осі XZ. Попередній код G18 потрібний для осі XZ встановлюваним в параметрі поточної осі. Вісь XZ встановлюється основною при застосуванні кругової інтерполяції, круговому обертанню координатної системи і безперервних циклів обробки деталей.

G19 – команда вибору осей YZ. Попередня команда G19 необхідна для вісі YZ встановлюваною як поточна. Вісь YZ встановлюється основною при застосуванні кругової інтерполяції і шляхом постійних циклів обробки з обертанням системи координат.

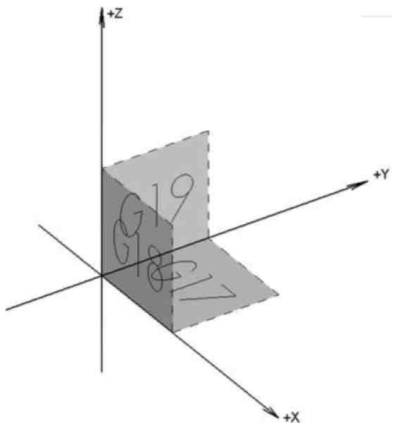

Рисунок.1.17. G19, G18, G17 застосовується для активації системи

координат у площині

G20 – код для введення параметрів в дюймах. Команда G20 використовує роботу верстата з дюймовими значеннями переміщень. Якщо використовується даний параметр, всі параметри, що вносяться, встановлюються як дюймові значення. Рекомендації у всіх циклах, використовуваних в дюймових системах, поставити виконувану команду G20 в початок коди програми, щоб у випадку, якщо в програмі, виконаній до цього, було включено метричне обчислення, необхідно задати вибір правильний формат вибору.

Приклад: N10 G20 G40 G49 G54 G80 G90 – код G20 в безпечному рядку. Код є модальною функцією і виконується до надходження коди G21.

G21 – введення параметрів в метричній системі координат. Код G21 включає режим роботи в метричних параметрах системи. Якщо включений даний режим, всі виконувані змінні розпізнаються метричними. У виконуваних програмних кодах, написаних в метричній системі, бажано використовувати команду G21 на початку програми, щоб у разі, коли в програмному коді, виконаному до цього, застосовувалося дюймове обчислення, використовувати правильний формат вибору.

Приклад: N10 G21 G40 G49 G54 G80 G90 – команда G21 в безпечному рядку. Даний код є модальна функція і виконується до команди G20.

G22 – використання умови переміщень до межі. Команда G22 включає межу значень для установки. В даному випадку інструмент верстата не йде за область обмежену межами. Дані межі встановлюються значеннями введеними в ЧПУ.

G23 – дезактивація режиму граничного позиціонування верстата при переміщенні. Використовуючи команду G23 задані параметри переміщення не враховуються. Команда G23 вимикає дію команди G22 і здійснює переміщення інструменту в різні робочі зони координатної сітки.

G27 – команда перевірки повернення до стартових позицій. Команда G27 виконується також як команда G28. Відмінність лише в тому, що позиція, в якій перемістився робочий інструмент, не є заданою раніше позицією, тоді випадку

команди G27 електроніка верстата, що управляє , сигналізує про аварію і видає аварійний сигнал.

Команди G27 і G28 використовуються в циклічних програмних макросах зміни інструменту в автоматичному режимі для роботи. Заздалегідь перш ніж виконати дані G - команди як правило вимикають калібрування інструменту верстата.

G28 – повернення позиції при старті в автоматичному режимі. Команда G28 служить для повернення в стартову позиції верстата. Тобто швидке позиціонування робочого інструменту в позицію умовного нуля верстата. Повернення до умовної нульової позиції використовується для умови параметрів перевірки і якості обробки матеріалу в програмі посередині обробки. Досить часто команду G28 використовують в кінцевому коді програми управління, для того, щоб після закінчення програми, робочий стіл повернувся в позицію, для зручного витягання обробленого матеріалу.

Для переміщення в початкове положення використовується умовний код наступного вигляду:

G90 G28 X0.0 Y0.0 Z0.0

У кадрі з G28 задаються осі з 0.0 параметрами, переміщення в позицію при старті здійснюється по трьом осям. Але не завжди обов'язково виконувати дану операцію. Іноді потрібне переміщення тільки по одній з них. Наприклад, для переміщення по осях Z в програмному коді для обробки необхідно використовувати даний крок:

G90 G28 Z0.0

Пильну увагу необхідно звернути на встановлений в кадрі код G90. Дана команда включає роботу верстата у відносних координатах. Інакше працює команда G28 вона проводить програмування якоїсь проміжної точки, куди буде переміщений інструмент, потім портал повернеться в початкову позицію. Дійсні координати верстата, введені в кадрі програми, є не що інше як координати проміжної точки. У даних прикладах ми вводили в координати проміжної точки з нульовими значеннями. У програмному кадрі вказана команда G90 що є

відносною координатою, верстат переміщатиметься щодо справжнього положення по кожній вісі на нуль міліметрів. Іншими словами нікуди не переміщатиметься. Ось тому за наявності коду кадру G90 G28 X0.0 Y0.0 Z0.0, керує, портал повернеться в початкову позицію без переміщення в проміжну крапку.

За наявності в програмі обробки кадру G90 G28 X10.0 Z20.0, портал верстата спочатку поїде вгору і управо, а потім переміститися в точку нуль. Команда G28 дозволяє здійснити переміщення з прискоренням, як і G00, в даному моменті воно можливо буде непрямолінійним. Іншими словами портал може щось «зачепити» . Програміст з великим досвідом спочатку піднімає шпиндель вгору, потім переміщає портал в початкову позицію:

G90 G28 X0.0 Y0.0 Z30.0

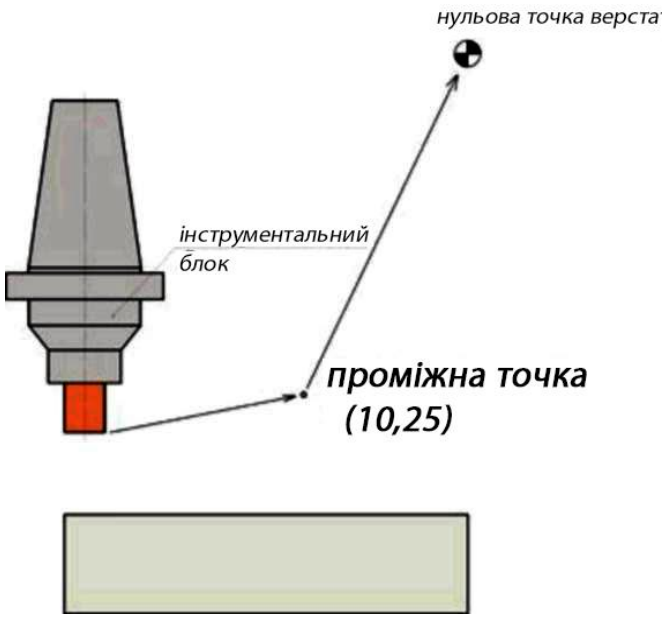

Рисунок 1.18 – Траєкторія руху порталу верстату

Коли в програмі є команда G91 G28 X10 Y25, то портал в першу чергу піде в проміжну крапку, потім повернеться в початкову позицію.

Не варто в кадрі з G28 використовувати команду з абсолютними координатами G91. Якщо в програмі є кадр G91 G28 X0 Y0 Z0, тоді є можливість "наїзду" робочого порталу на складові частини верстата або деталі.

G30 – команда для повернення в положення зміни інструменту. За

допомогою команди G30 здійснюється переміщення по осі Z до положення для заміни інструменту в роботі і вимикається використовується інструментальне калібрування. Для зміни інструменту використовується наступна команда підпрограми:

G30 G90 Z0

Варто відмітити що, якщо в кадрі програми замість G91 присутня команда G90, то шпиндель переміщатиметься до поверхні порталу.

G31 – команда для пропуску з реакцією на імпульс ззовні. У верстатах іноді використовують команду для пропуску з реакцією на імпульс ззовні. З допомогою немодальної команди G31, оператор виконує інтерполяцію по лінії схожу з G01, змішану з реалізацією відгуку імпульс поступає ззовні. Сигнал поступає від порталу подається від натиснення на шукану кнопку панелі управління верстата, як приклад клавіша Старт програми верстата.

За відсутності імпульсу пропуску, виконувана програма працює так як ніби був би заданий код G01. Якщо верстат прийняв команду, що поступила, то програма виконується і переходить до подальшої операції.

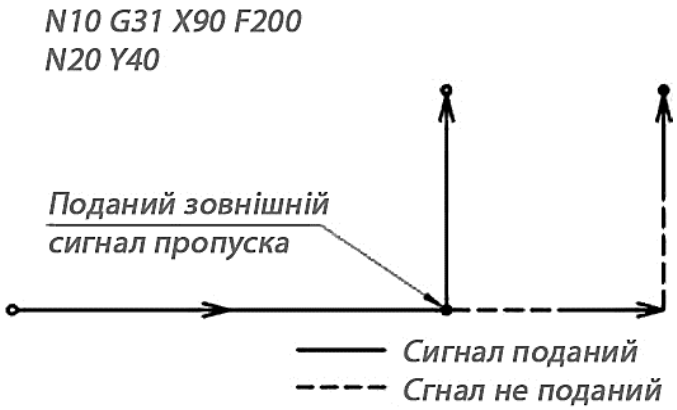

Рисунок 1.19 – Пропуск функцією з реакцією на зовнішній сигнал

G40 – команда відміняє корекцію інструменту по радіусу.

Корекція інструменту автоматом по радіусу інструменту вимикається шляхом подачі код D00 і G40. Команда G40 стоїть в рядку з кодом прямолінійного переміщенням з прискоренням по контуру деталі.

G1 G40 X90

G41 – коректування радіусу, інструмент розташовується зліва від деталі. Команда G41 застосовується для активації корекції радіусу інструменту в автоматичному режимі, що знаходиться від деталі зліва. Напрям переміщення зверху вниз, при спостереженні порталу зверху, від сторони «+Z» в положення «– Z».

G42 – корекція радіусу, інструмент розташовується з правого боку від деталі. Команда G42 застосовується для активації автоматичного коректування радіусу робочого інструменту, розташованого праворуч від оброблюваного об'єкту. Напрям зсуву задається, якщо дивитися зверху вниз, з боку «+Z» у напрямі «– Z».

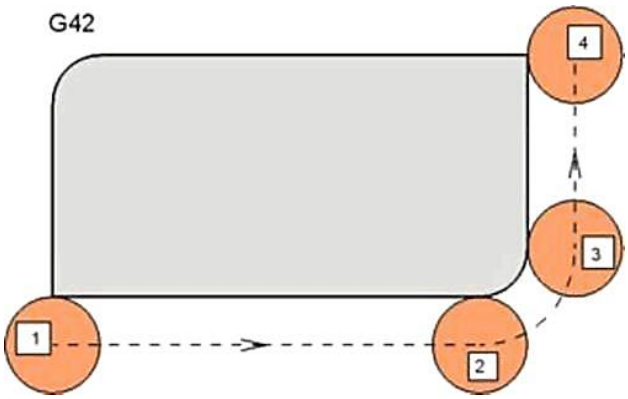

Рисунок 1.20 – Корекція з правого боку

G43 – коректування довжини інструменту. При виконанні програми, початкова позиція робочого інструменту визначається заданими координатами. Проблема полягає в тому, що в початковій позиції робочого механізму обробка не відбувається. Обробка відбувається кромкою робочого механізму, яка розташована на певному віддаленні від початкової точки робочого механізму. Для того, щоб в задану координату приходив робочий механізм, необхідно «показати» ЧПУ, на яку відстань по осі Z потрібно змістити цю стартову крапку.

Зміна довжини інструменту відбувається за рахунок подачі команди G43 і H-слова. Як правило, коректування відстань визначається одночасно з прискоренням пересування по осі Z.

Приклад: G43 H01 Z100

G49 – дезактивація компенсації відстані робочого механізму. Компенсація довжини робочого механізму відміняється шляхом завдання команди G49 або H00.

G50 – дезактивація режиму масштабування. Код G50 призначений для дезактивації масштабування G51.

G51 – активація режиму масштабування. У цьому режимі програміст компенсує коефіцієнт масштабування для координатних осей механізму. Режим включається за рахунок постійного коду G51 і відключається G50.

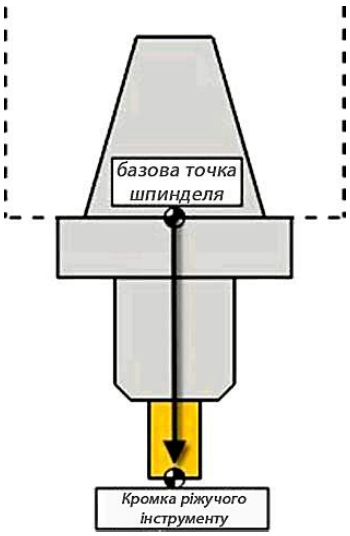

Рисунок 1.21 – Команда G43 H змінює базову точку робочого механізму до кромки ріжучого інструменту

Допускається вказівка-коефіцієнт масштабування для всіх разом або для певної осі. Якщо коефіцієнт масштабування більше 1, то система координат збільшується. Якщо цей коефіцієнт масштабування менше 1, то координатна система зменшується.

Для загальної зміни масштабу, як правило, застосовується наступна форма запису:

G51 X* Y* Z* P* де G51 – активує режиму масштабування; X – положення по осі X для проміжної точки масштабування; Y – положення по осі Y для проміжної точки масштабування; Z – положення по осі Z для проміжної точки масштабування; P – коефіцієнт масштабування для всіх осей разом.

Можливо також симетричне відображення з допомогою негативного значення масштабування. Для окремої зміни масштабу, як правило, застосовується наступний формат:

G51 X* Y* Z* I* J* K*,

де G51 – активує режиму масштабування; X – положення по осі X для проміжної точки масштабування; Y – положення по осі Y для проміжної точки масштабування; Z – положення по осі Z для проміжної точки масштабування; I – коефіцієнт масштабування для осей X; J – коефіцієнт масштабування для осей Y; До – коефіцієнт зміни для осей Z.

У функції симетричного відображення, застосовуються окрема зміна масштабування і можливість симетричного відображення заданих координат по одних або декілька осям.

Основна програма

G90 G01 F90

M98 P101

G51 X5 Y5 I-1 J1 K1

M98 P101 G51 X5 Y5 I-1 J-1 K1

M98 P101 G51 X5 Y5 I1 J-1 K1

M98 P101

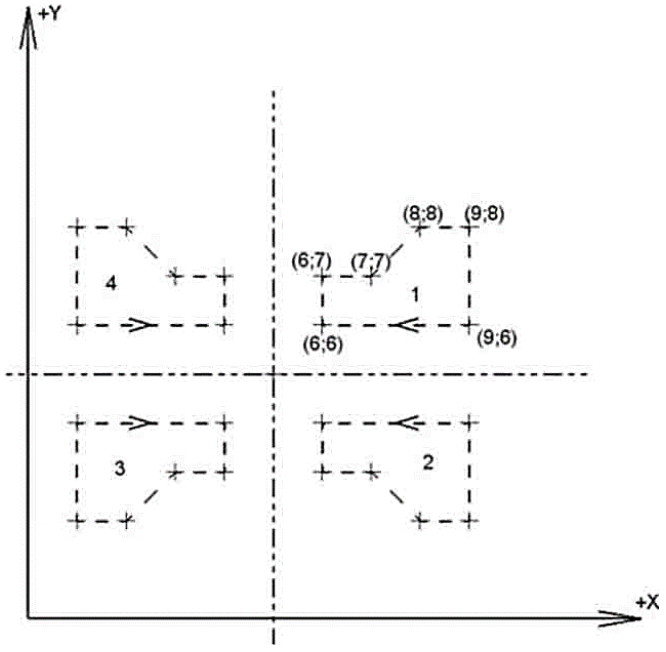

Рисунок 1.22 – Дзеркальне відображення траєкторії

Підпрограма

O0101

G90 X6 Y6

Y7

X7

X8 Y8

X9

Y6

X6

M99

G52 – локальна система координат. ЧПУ дозволяє задавати, окрім звичайних робочих систем позиціонування, ще і локальні системи позиціонування. Код G52 застосовується для знаходження підпорядкованої системи позиціонування в діапазоні поточної робочої координатної системи (G54–G59).

Коли ЧПУ верстат застосовує команду G52, то початок поточної робочої системи позиціонування зміщується на якесь значення, задане за допомогою команд визначених X, Y і Z:_G52 X*Y*Z*

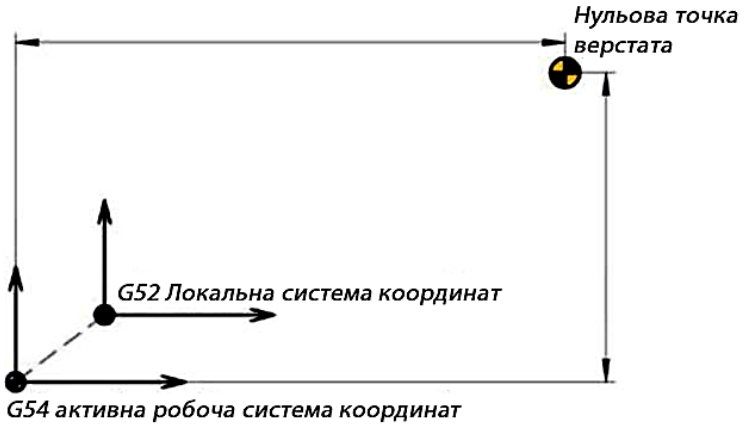

Рисунок 1.23 – Локальна система позиціонування

Команда G52 деактивується, якщо задається інша робоча система координат G54-G59 або за допомогою команди G52 X0 Y0 Z0.

G54-G59 – стандартні системи координат, що діють . За допомогою команд G54, G55, G56, G57, G58 і G59 задається, в якій системі координат, що діє, виконуватиметься обробка заготівки. Шляхом вибору різних координатних систем оператор має можливість за допомогою одних і тих же програм обробляти різні заготовки. Якщо була задіяна одна з координатних систем G54-G59, то вона застосовується до того моменту, поки не активується інша координатна система.

G60 – позиціонування в заданому напрямі. За допомогою цих команд, до всіх заданих позицій по яких осі можна пересувати з деякого напряму. Завдяки цьому виходить можливість прибрати помилки координування які можуть з'являтися із-за невикористаного ходу в системах. Як правило, напрям і величина позиціонування указуються параметрами ЧПУ.

G61 – режим дуже точного позиціонування зупинки. Команда G61 використовується для включення режиму зупинки. Команда точної зупинки детально описана в характеристиці команди G09. Єдина відмінність між кодом G61 і G09 є те, що G09 немодальна команда. Модальна команда G61 залишається включеною до тих пір, поки не буде дана команда на зміну поточного режиму, за допомогою іншої команди, такої як G63.

G64 – режим нарізування. Стандартний режим нарізування активується кодом G64.

G65 – немодальний виклик мікропрограми. Код G65 дозволяє виконати мікропрограми, розташовані в пам'яті ЧПУ. Формат для немодального виклику мікропрограми виглядає таким чином:

G65 P_L_ де G65 – команда для виклику мікропрограми; P – номер мікропрограми; L – кількість виконань мікропрограми. Якщо L не указується, то ЧПУ приймає, що L = l.

G66 – модальний виклик мікропрограми. Код G66 призначений для запуску мікропрограми, як і код G65. Єдина відмінність між цими кодами є в тому, що G66 є модальним командою і мікропрограми виконуються при кожній зміні позиції, поки не застосовуватиметься команда G67. Формат для модальної активації мікропрограми:

G66 P_L_ де G66 – команда для циклів мікропрограми; P – номер мікропрограми; L – кількість циклів мікропрограми.

Якщо L не указується, то ЧПУ вважає, що L = 1.

G67 – дезактивація модального виклику мікропрограми. За допомогою вказівки коду G67 дезактивувався режим модального виклику макропрограми G66.

G68 – обертання координатної системи. Модальна команда G68 дозволяє виконати поворот координатної сітки на заданий кут. Для реалізації такого обертання потрібно вказати площину обертання, центр повороту і кут обертання. Площина повороту встановлюється за допомогою коду G17 (площина XY), G18 (площина XZ) і G19 (площина YZ). Якщо бажана площина повороту вже активована, то завдання команд G17, G18 і G19 в кроці з G68 немає необхідності.

При програми G90 , що діє , центр обертання указується абсолютними координатами щодо стартової точки верстата, якщо не активована одна із стандартних робочих систем позиціонування. Якщо вибрана одна з робочих систем позиціонування G54–G59, то центр обертання задається щодо стартової точки робочої системи позиціонування, що діє . У разі активної команди G91 центр повороту указується щодо поточного позиціонування. Якщо ж позиції центру обертання не будуть задані, то як позиція центру повороту буде прийнята поточна позиція.

Кут повороту указується за допомогою R-слова даних. Формат запису для команди розвороту координат зазвичай записують таким чином:

G17 G68 X_Y_R_

G69 – відміна розвороту координат. За допомогою коду G68 відміняється режим повороту позиції.

G73–G89 – постійні цикли

G80 – Відміна постійного циклу

G81 – Стандартний цикл свердління

G82 – Свердлення з витримкою

G83 – Цикл переривчастого свердління

G73 – Високошвидкісний цикл переривчастого свердління

G84 – Цикл нарізування різьби

G74 – Цикл нарізування лівої різьби

G85 – Стандартний цикл обробки

G90 – активація режиму абсолютного позиціонування.

У цьому режимі, абсолютного місця положення G90 зміни робочих механізмів робиться щодо нульової точки механізму або щодо нульової точки робочої системи позиціонування G54-G59. Команда G90 є модальною і дезактивувався за рахунок команди відносного позиціонування G91.

G91 – активація режиму відносного місцеположення. За рахунок команди G91 включається режим відносного місцеположення. При інкрементному методі відліку за стартове місцеположення кожного разу використовується положення робочого механізму, в якому він знаходився перед початком руху до наступної програмованої точки. Команда G91 по суті є модальною і дезактивувався за допомогою команди абсолютного позиціонування G90.

G92 – переміщення абсолютної системи координат. Бувають ситуації, коли у оператора ЧПУ виникає необхідність задати певні змінні в регістрах поточної системи позиціонування для переміщення початкових точок в нове положення.

Код G92 використовують для переміщення поточного положення початкової точки за рахунок редагування значень в регістрах поточних зсувів. Коли ЧПУ виконає програму G92, то змінні в регістрах зсувів змінюються і стають рівними змінним, які задані X-, Y- і Z-параметрам.

G92X Y Z

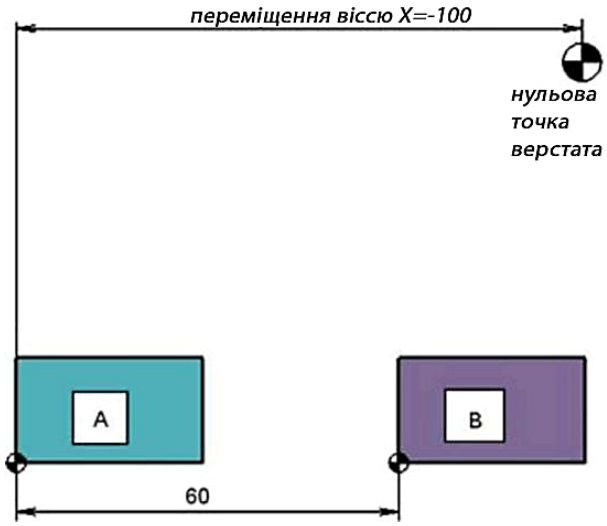

Рисунок 1.24 – За допомогою G92 ми указуємо абсолютної координати верстата і зміщуємо початкову точку позиціонування

Спочатку перемістимо робочий механізм у відому нам стартову початкову точку, а потім застосуємо G92:

…

.G00 X0 Y0

G92 X-70 Y0

…

Кадр G92 X-70 Y0 показує, що нинішнє поточне положення інструменту визначено координатами (–70; 0). Завдяки цьому, шукана початкова точка позиціонуватиметься на 70 мм правіше за нинішнє положення робочого інструменту.

Існує інший метод для досягнення цього ж результату. Можна спочатку перемістити робочий інструмент в позицію, яку треба прийняти новою нульовою крапкою, і потім виконати команду G92 X0 Y0.

…

G00 X60 Y0

G92 X0 Y0

…

Команда G92 сама по собі не виконує осьових переміщень. Задане за допомогою G92 переміщення координатної системи на багатьох верстатів може

бути задано поверненням в стартову крапку або знеструмленням верстата.

G94 – швидкість руху. За допомогою програми G94 задана швидкість переміщення задається в дюймах на хвилину або в міліметрах хвилину.

Якщо дюймовий режим G20, що діє, то швидкість переміщення F задається як переміщення в дюймах за 60 секунд. Якщо ж в даний момент активний метричний режим G21, то швидкість переміщення F приймається як переміщення в міліметрах за 60 секунд.

G20 F10 – швидкість переміщення 10 дюймів в хвилину; G21 F10 – швидкість переміщення 10 міліметрів в хвилину.

Модальний код G94 залишається включеним до тих пір, поки не буде виставлена команда G95.

G95 – швидкість переміщення механізму в дюймах/міліметрах на обертання. За рахунок команди G95 задає швидкість переміщення, приймається в дюймах на оборот робочого механізму або в міліметрах на оборот робочого механізму. Іншими словами швидкість переміщення F синхронізується із швидкістю переміщення робочого механізму S. При рівних значеннях F швидкість переміщення зростатиме при збільшенні числа подач.

G20 F0.1 – швидкість переміщення рівна 0.1 дюйма на 1 обертання; G21 F0.1 – швидкість переміщення рівна 0.1 міліметра на обертання. Модальна програма G95 залишається включеною до тих пір, поки не буде виставлена команда G94.

G98 – повернення до початкової площини в програмі. Початкова площина – це положення по осі Z (рівень), в якій знаходиться робочий механізм перед активацією поточного циклу. Код G98 дезактивувався за допомогою команди G99.

G99 – повернення до площини відведення в циклі. Якщо цикл працює спільно з кодом G99, то робочий інструмент переміщається до площини повернення посеред всіх поточних поверхонь. Площина повернення – це позиція по положенню Z (рівень), з якою стартує на поточній подачі, і в яку переміщається робочий інструмент після того, як дійшов до дна оброблюваної

заготовки. Площина повернення зазвичай задається в циклу при допомогою R-адреса. Команда G99 відміняється з допомоги програми G98.

Адреси/слова даних

X - програма осьового переміщення. Як правило, за X задаються вісь, яка можливо найбільше переміщує робочий орган пристрою. При цьому вісь X перпендикулярна до площини Z і паралельна робочій поверхні.

Позитивне або негативне значення, входить до складу цього слова, визначає кінцеву координату виконавчого органу верстата уздовж осі X. У кадрі можна задати X тільки 1 раз. Якщо в 1 кадрі буде декілька програм X, то ЧПУ працюватиме з останньою з них, яка з них ближче.

Приклад:

G01 G90 X100 F30 – лінійне переміщення по координаті X = 100 із швидкістю 30 мм/мін.

Коли X знаходиться в одній позиції з програмою витримки G04, то вона задає час цієї витримки в секундах.

Приклад:

G04 X11.0 – виконати затримку тривалістю 11 секунд.

Y є програмою осьового переміщення. Вісь Y перпендикулярна для 2х осей X і Z. Позитивне або негативне число, що входить до складу цього циклу даних, задає кінцеву позицію робочого органу верстата уздовж осі Y. У кадрі можна запрограмувати Y тільки 1 раз. Якщо в рядку будуть вказано декілька команд Y, то ЧПУ працює з останньою з них, яка значно ближче до знаку кінця кадру.

Приклад: G01 G90 Y20 F220 – лінійне переміщення в координату Y = 20 із швидкістю 220 мм/мін.

Z служить командою осьового пересування. Щоб задати позитивного напряму Z задають вертикальне направлення виводів робочого інструменту із заготівки. Іншими словами вісь Z завжди пов'язана з шпинделем верстата. Позитивна або негативна змінна, що входить до складу цього слова даних, задає кінцеву позицію виконуваного органу верстата уздовж осі Z. У кадрі можна задати Z тільки один раз. Якщо в кадрі будуть вказано декілька команд Z, то ЧПУ

працюватиме з останньою з них, яка значно ближче до кінця кадру.

Приклад:

G01 G90 Z7 F400 – лінійне переміщення в позицію Z = 7 із швидкістю 400 мм/мін.

A, B, З є програмами іншого переміщення. Під круговим пересуванням приймають кутове пересування, поворот осі верстата або кутове пересування керованого обертанням площини. Поворотні пересування робочого інструменту задають латинськими буквами – A, B (навколо Y) і З (навколо Z). Позитивні обертання навколо цих осей визначаються досить-таки просто.

Приклад:

G01 G90 C60 F350 – поворот столу на 60° із швидкістю 350 мм/мін.

I, J, До задаються під час кругової інтерполяції і використовуються для вказівки відносних положень від стартової точки дуги до її центру. Слово значень з I відноситься до X, слово даних з J – до Y, а слово даних до Z. При цьому залежно від місцеположення дуги значення можуть бути як позитивними, так і негативними.

R. При поточній круговій інтерполяції (G02/G03) R показує радіус, який сполучає початкову і кінцеву точки дуги.

Для більшості ЧПУ адреса R може бути командою на виконання скруглення при поточній лінійній інтерполяції. Числові змінні, що входять до складу R-слова даних, визначає радіус округлення.

У постійних циклах R визначає місцеположення площини відведення. При використанні з командою повороту координат R задається кут обертання координатної площини.

P як правило застосовується в постійних циклах і визначає значення часу витримки на дні виробу. Числове значення, що входить до складу P-слова даних, як правило, визначає час витримки в 1 мілісекунду.

Коли P з'являється в одному кадрі з програмою виклику підпрограми M98, то воно указує номер підпрограми, що запускається. В деяких випадках це ж слово значень також може показувати на частоту виклику програми.

Приклад:

M98 P1001 – виклик програми O1001.

Q часто застосовується в циклах, визначає глибину кожного робочого ходу інструменту.

У циклі Q визначає положення зрушення робочого інструменту від стінки виготовленого виробу для забезпечення безпечного виведення інструменту з робочої зони.

За допомогою D задається значення корекції на радіус робочого інструменту. Корекція радіусу робочого інструменту активується командами G41 і G42. За допомогою програми D00 можна відмінити команду корекції, що діє.

За допомогою H вибирається значення компенсації довжини робочого інструменту. Компенсація довжини робочого інструменту, як правило активується командою G43. За допомогою команди H00 можна дезактивувати компенсацію довжини інструменту, що діє.

Для завдання швидкості подачі служить F-адрес. Якщо в 1 кадрі будуть запрограмовані множина швидкостей подач то ЧПУ буде використовуватися з останнім з них. У разі використання F з командою G94 швидкість переміщення задаватиметься в дюймах (G20) або міліметрах (G21) в 1 хвилину. А у разі використання з G95 швидкість подачі буде встановлена в дюймах (G20) або міліметрах (G21), навпаки. F-адрес іменується модальною, по іншому, встановлена швидкість переміщення залишається постійною до тих пір, поки не змінна спільно з F або не змінений режим пересувань за допомогою команди G00.

За допомогою S задається число обертів. S-адрес є модальною, тобто встановлене число оборотів залишається постійним до тих пір, поки не заданий інші числові параметри S.

За допомогою T указується управління магазином інструментів. В числовому значенні T вказує номер інструменту, який необхідно перемістити в позицію заміни шляхом обертання інструментального магазину. Як правило T програмують в одному кадрі з командою зміни інструменту M06. В цьому випадку кількість значень T визначатиме номер інструменту, який необхідно

вибрати з магазина і встановити в руку.

Приклад:

Т3 М06 – задати інструмент № 3.

За допомогою N виконується нумерація кадрів УП. При виконанні номера кадру може бути вибраний в кадрі в будь-яку позицію, але зазвичай вказують на початок. Номер кадру не впливає на роботу, а допомагає програмістові орієнтуватися в змісті програми.

М-коди

М00 – запрограмована зупинка. Коли ЧПУ застосовує команду М00, то виконується так званий запрограмована зупинка. Всі осьові рухи зупиняються і стартують після того, як оператор натисне кнопку Старт циклу. При цьому робочий механізм продовжує працювати і інші функції як і раніше залишались активними. Якщо оператор натискає кнопку Старт, то робота циклу буде продовжена з позиції, наступного за командою М00.

М01 – зупинка по вибору. Код М01 служить для зупинка по вибору. Працює аналогічно команді М00, проте залишає вибір операторові – чи потрібно йому або не потрібно зупиняти роботу. Якщо кнопку натиснути, то при читанні команди з М01 відбувається зупинка. Якщо кнопку не натиснутий, то команда М01 ігнорується і виконання не переривається.

М02 – кінець циклу. Код М02 інформує про завершення програми.

М03 – пряме обертання. За допомогою команди М03 запускається пряме (за годинниковою стрілкою) обертання з вказаним числом обертів. Команда М03 як і раніше залишається виконуваною до тих пір, поки не буде зупинена командами М04 або М05.

М04 – зворотне обертання. За допомогою команди М04 запускається зворотне (проти годинникової стрілки) обертання з вказаним числом обертів. Код М04 як і раніше залишається виконуваним до тих пір, поки не буде дезактивована за допомогою команд М03 або М05.

М05 – зупинка. Код М05 припиняє обертання, але не зупиняє осьові пересування.

M06 – зміна інструменту. Завдяки коду M06, закріплений інструмент змінюється на наступний, який знаходиться в положенні готовності.

M19 – юстирування. За допомогою команди M19 здійснюється радіальне юстирування (поворот в певне положення), щоб виставити привід на позицію зміни інструменту.

M20 – відміна юстирування. За допомогою цієї команди відміняється команда юстирування шпинделя M19.

M30 – кінець програми. Код M30 інформує про завершення циклу.

M98 – виклик сторонньої підпрограми. Команда M98 призначена для виклику сторонньої підпрограми. Разом з командою використовується P-слово даних, воно вказує на номер підпрограми, що запускається.

Приклад:

M98 P1001 – викликати підпрограму O1001.

M99 – кінець програми. За допомогою коду M99 по завершенні підпрограми реалізується повернення до початкової програми, з якої була викликана дана підпрограма [4].

1.7 Розробка програми управління ЧПУ верстата

Спираючись на принцип роботи ЧПУ верстатів і особливості роботи G-коду, була розроблена програма для ARDUINO. Мова програмування пристроїв Arduino базується на C/C++ і скомпільований з бібліотекою AVR Libc і дозволяє використовувати різні її функції [28]. Разом з тим він простий в розумінні, і на даний момент Arduino – це один з найзручніших способів програмування пристроїв на мікроконтролерах AVR. Була написана прошивка для мікроконтролера яка зчитувала траєкторії із зовнішнього носія і передавала команди управління на ДУ ШД, текст цієї програми приведений нижче:

```
int motorPins[3][2]= {8,9},{10,11},{12, 13};
int count;
int count2[3]= {0,0,0};
int val = 0;
int rot=0;
int incomingByte = 0;
int sign=1;
long delayTime;
```

```
//////////////////ручне керування//////////////////
#define VR_X 2 // Вісь X підключена до Analog 2
#define VR_Y 1 // Вісь Y підключена до Analog 1
int SW1=A0; // кнопка ручний режим до Analog 0
int LED=A3; // Індикатор до Analog 3
int TOG=0; // Змінна для зберігання режиму роботи пульта
byte value_1, value_2=0;
int value_X, value_Y; // змінний для зберігання осей
//////////////////////////////////////////////////////
//Процедура настройки прошивки void setup() {
//////////////////ручне управління//////////////////
pinMode(SW1,INPUT);
digitalWrite(SW1,HIGH);
pinMode(LED,OUTPUT);
//////////////////////////////////////////////////////
int i;
Serial.begin(9600); //ця швидкість повинна співпадати з швидкістю, встановленою в програмі
for (i=0; i<3; i++) {
for (count = 0; count < 2; count++) {
pinMode(motorPins[i][count], OUTPUT); //установка режиму роботи цифрових пінів Arduino
}
}
delayTime=2000;
}
//Поворот двигуна з номером sm на один крок вперед
void moveForward(int sm){
digitalWrite(motorPins[sm][1], HIGH); //Задаєм напрямок
digitalWrite(motorPins[sm][0], HIGH);
digitalWrite(motorPins[sm][0], LOW);
}
//Поворот двигуна з номером sm на один крок назад
void moveBackward(int sm){
digitalWrite(motorPins[sm][1], LOW);
digitalWrite(motorPins[sm][0], HIGH);
digitalWrite(motorPins[sm][0], LOW);
}
//Затримка в мікросекундах
void delayMicros(long wt){
unsigned long mls;
unsigned int mks;
mls=(unsigned long)(wt / 1000);
mks=(unsigned int)(wt % 1000);
if (mls>0) delay(mls);
if (mks>0) delayMicroseconds(mks);
}
//Одночасний поворот двигунів 0, 1, 2 на x, y, z кроків відповідно
void MOVESM(long x, long y, long z){
long з[3], c2[3];
double c1[3], d[3];
long m, i;
boolean flg;
з[0]= x; з[1]= y; з[2]= z;
m = 1;
for (i=0; i<3; i++) {
if (m < abs(з[i])) m = abs(з[i]);
}
```

```
for (i=0; i<3; i++) {
c1[i]= 0;
d[i]= 1.0 * з[i]/ m;
c2[i]= 0; }
flg = false;
for (i=0; i<3; i++) {
if (abs(c1[i])< abs(з[i])) flg=true;
}
while (flg) {
flg=false;
for (i=0; i<3; i++) {
if (abs(c1[i])< abs(з[i]))
c1[i]+= d[i];
if (abs(c1[i]) - abs(c2[i]) >= 0.5) {
if (з[i]>0) {
c2[i]++;
moveForward(i);
} else {
c2[i]--;
moveBackward(i);
}
}
if (abs(c1[i])< abs(з[i])) flg=true;
}
delayMicros(delayTime);
}
}
//Основний цикл
void loop() {
//////////////////ручне управління//////////////////
value_1=digitalRead(SW1);
if(!value_1)
{
delay(50);
value_2=digitalRead(SW1);
if(!value_2)
{
if(TOG!=0) TOG=0;
else TOG=1;
digitalWrite(LED,TOG); //індикація режиму роботи
do{
}while(!digitalRead(SW1));
Serial.println("0");
}
}
//////////////Управління з пульта//////////////////
if (TOG==1)
{
value_X = analogRead(VR_X); // Зчитуємо аналогове значення осі Y
if(value_X >= 0 && value_X < 480)
{
MOVESM(1,0,0);
}
if(value_X > 540)
{
MOVESM(-1,0,0);
```

```
    }
    value_Y = analogRead(VR_Y); // Зчитуємо аналогове значення осі Y
    if(value_Y >= 0 && value_Y < 480)
    {
    MOVESM(0,1,0);
    }
    if(value_Y > 540)
    {
    MOVESM(0-1,0);
    }
    }
    ////////////////////////////////////////////////////
    if (Serial.available() > 0) { //Надійшла команда
    long з[4]={0,0,0,0};
    int i;
    sign=1;
    i=0;
    incomingByte = Serial.read();
    while (incomingByte!=';') { //Зчитуємо вхідний рядок, ознака кінця рядка знак "крапка з комою"
    if (з[i]==0) {
    if (incomingByte=='-')
    sign=-1;
    }
    if (incomingByte==") {
    з[i]*=sign;
    sign=1;
    i++;
    } else if (incomingByte>='0' && incomingByte<='9') {
    з[i]=c[i]*10+incomingByte-'0';
    }
    while (Serial.available() == 0) {
    delayMicroseconds(1); //Чекаємо черговий символ, якщо не прийшов
    }
    incomingByte = Serial.read();
    }
    з[i]*=sign;
    if (з[3]>0) delayTime=c[3];
    MOVESM(з[0],c[1],c[2]); //Обертаємо двигуни на задане число кроків
    Serial.println("OK"); //Відправляємо комп'ютеру повідомлення "OK", означає можна висилати нову
команду
    }
    else
    delayMicroseconds (1); //Якщо нічого не прийшло, чекаємо 1 мікросекунду.
    }
```

Принцип роботи даної прошивка полягає в тому, що траєкторії треба писати з вказівкою кількості кроків по осях швидкості подачі і необхідності включати або виключати інструмент.

X;Y;Z;S;T.

X,Y,Z – кількість кроків; S – швидкість; T – включення/виключення інструменту.

Але після проведення серії випробуванні на реальному устаткуванні було виявлені деякі недоліки цієї програми. Що укладає полягає в тому, що всі траєкторії майбутньої програми необхідно писати повністю вручну і якщо об'єкт має просту форму, то в цьому немає ніяких складнощів, але для складніших об'єктів написання траєкторій займає значний час і достатньо легко допустити помилку. Для усунення цього моменту необхідно писати спеціальну програму для комп'ютера, яка полегшувала і автоматизувала б цей процес. Але оскільки вже є величезна кількість інформації за принципом роботи G-code, в мережі Інтернету була знайдена вже готова прошивка для Atmega328, в якій вже був реалізований частковий функціонал G-коду.

Grbl master для роботи цієї прошивки необхідний зв'язок з комп'ютером який передаватиме інструкції і тексти траєкторій на Atmega328 по серіал порту. Grbl призначений для трьох осьових ЧПУ X, Y, і Z.

G-кодовий перекладач здійснює підмножину стандарту NIST rs274/ngc. Лінійний, круглий і гвинтовий рух все повністю підтримується.

При роботі всі параметри спочатку записуються в EEPROM, а вже потім обробляються контролером для побудови траєкторії руху робочого механізму, що значно підвищує якість готового виробу так - як мінімізується вірогідність пропуску кроку із за неякісного інформаційного дроту по якому передаються інструкції.

Grbl підтримує всі загальні операції, з якими стикаються у виробництві, але деякі функції не реалізовані такі як: змінні, бази даних інструменту, контролю циклу, цикли, арифметика і структурний контроль. Реалізовані тільки основні машинні операції і можливості.

Але так-як мета роботи зробити повноцінний пульт управління CNC верстата то необхідно позбавиться від використання в процесі роботи комп'ютера і для вирішення цього завдання було використано ще 1 плату Arduino Nano мета якої замінити комп'ютер. Для цієї плати була розроблена прошивка в якій реалізовані функції ручного управління 3 осями, вихід в стартову позицію, зчитування інструкції і траєкторій з SD карти на яку ті заздалегідь були

завантажені з комп'ютера. Текст прошивки представлений нижчим.

```
#include <SD.h>
#include <SPI.h>
const int chipSelect = 4;
const int butPinX = 5;
const int butPin_X = 6;
const int butPinY = 7;
const int butPin_Y = 8;
const int butPin_0 = 9;
int valueX, value_X, valueY, value_Y, value_0;
int TOG=0;
int t;
int pos;
int value;
int incomingByte=0;
void setup()
{
Serial.begin(115200);
pinMode(butPinX, INPUT);
digitalWrite(butPinX, HIGH);
pinMode(butPin_X, INPUT);
digitalWrite(butPin_X, HIGH);
pinMode(butPinY, INPUT);
digitalWrite(butPinY, HIGH);
pinMode(butPin_Y, INPUT);
digitalWrite(butPin_Y, HIGH);
pinMode(butPin_0, INPUT);
digitalWrite(butPin_0, HIGH);
!SD.begin(chipSelect);
}
void loop()
{
//////////////////ручне управління X//////////////////
if(!digitalRead(butPinX))
{
delay(50);
if(!digitalRead(butPinX))
{
Serial.println("G91 G0 X1");
do{
}while(!digitalRead(butPinX));
}
}
//////////////////ручне управління _X//////////////////
if(!digitalRead(butPin_X))
{
delay(50);
if(!digitalRead(butPin_X))
{
Serial.println("G91 G0 X-1");
do{
}while(!digitalRead(butPin_X));
}
}
```

```
//////////////////ручне управління Y//////////////////
if(!digitalRead(butPinY))
{
delay(50);
if(!digitalRead(butPinY))
{
Serial.println("G91 G0 Y1");
do{
}while(!digitalRead(butPinY));
}
}
//////////////////ручне управління _Y//////////////////
if(!digitalRead(butPin_Y))
{
delay(50);
if(!digitalRead(butPin_Y))
{
Serial.println("G91 G0 Y-1");
do{
}while(!digitalRead(butPin_Y));
}
}
//////////////////ручне управління _0//////////////////
if(!digitalRead(butPin_0))
{
delay(50);
if(!digitalRead(butPin_0))
{
Serial.println("G90 G28 X0 Y0");
do{
}while(!digitalRead(butPin_0));
}
}
if (t==10)
{
t=0;
Serial.print("?");
}
t++;
incomingByte = Serial.read();
if (incomingByte=='I')
{
File dataFile = SD.open("GCODE.TXT", FILE_READ);
if (dataFile)
{
while (dataFile.available() && value!='G' && value!='M')
{
if (value=='g')
{
Serial.write('G');
}
if (value=='m')
{
Serial.write('M');
}
```

```
dataFile.seek(pos);
value = dataFile.read();
if (value!='G' && value!='M')
{
Serial.write(value); }
pos ++;
delay (3);
}
dataFile.close();
}
if (value=='G')
{
value='g';
}
if (value=='M')
{
value='m';
}
}
```

}[29][30]

1.8 Драйвер для управління кроковим двигуном

Існує величезна різноманітність крокових двигунів різних розмірів і характеристик. Щоб управляти ними потрібний спеціальний пристрій – драйвер. Для роботи драйвера необхідно подавати 3 сигнали

Enable - дозвіл на обертання

Step - кількість кроків

Dir - напрям обертання

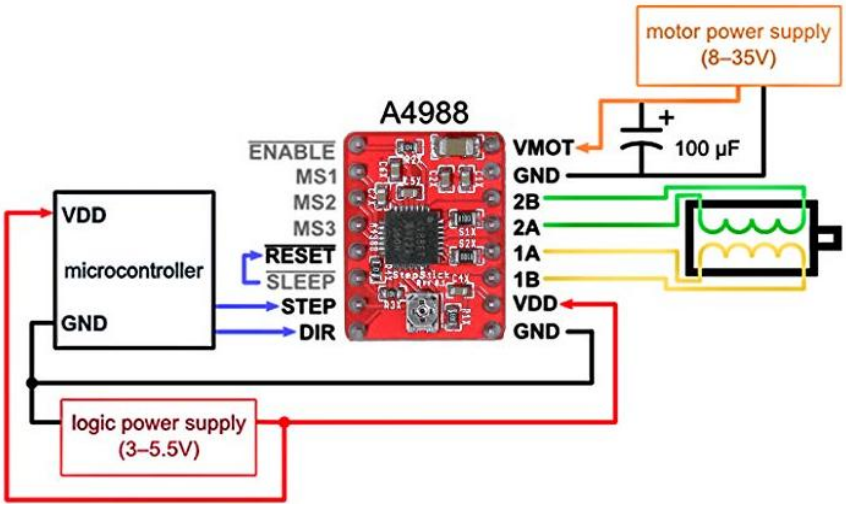

Рисунок 1.25 – Драйвер крокового двигуна

Для лазерного гравера, що розробляється, стане в нагоді недорогий компактний драйвер родини A4988 він має наступні характеристики:

модель: A4988;

напруга живлення: від 7 до 34 В;

можливість подрібнення кроку: від 1 до 1/16 кроку;

напруга логіки: 5 В;

захист від перегріву;

максимальний струм на фазу: 2 А з радіатором;

відстань між рядами ніжок: 11 мм;

розмір плати: 21 х 16 мм;

габарити драйвера: 21 х 16 х 15 мм;

габарити радіатора: 10 х 6 х 10 мм;

вага з радіатором: 4 г;

Плата створена на базі мікросхеми A4988 родини - драйвери біполярного крокового двигуна. Особливостями A4988 є регульований струм, захист від перевантаження і перегріву, драйвер також має п'ять варіантів мікрокроку (аж до 1/16-кроку). Він працює від напруги 7 - 34 В і може забезпечити струм 2 А на кожну обмотку.

Опис:

Цей драйвер дозволить управляти біполярним крововим двигуном з вихідним струмом до 2 А на обмотку (для отримання додаткової інформації дивіться розділ про розсіювання потужності). Нижче приведені ключові особливості драйвера:

1. Простий інтерфейс управління кроком і напрямом обертання електродвигуна, п'ять різних дозволів переміщення: повний крок, 1/2-кроку, 1/4- кроку, 1/8- кроку, 1/16- кроку.
2. Регульований контроль струму за допомогою потенціометра, дозволить встановити максимальний вихідний струм. Це дасть вам можливість використовувати напругу вище допустимого діапазону для досягнення вищої кутової швидкості кроку двигуна.
3. Інтелектуальне управління автоматично вибирає режим регулювання загасання струму (повільний і швидкий режими).

4. Захисне відключення при перегріві і перевантаженні по струму, а також блокування живлення при зниженій напрузі.
5. Захист від короткого замикання на землю, захист від замикання в навантаженні

Використання: З'єднання з джерелом живлення:

Для роботи з драйвером необхідне живлення логічного рівня (3 -5,5 В), що подається на виводи VDD і GND, а також живлення двигуна (8 - 35 В) на виводи VMOT і GND. Щоб забезпечити необхідний споживаний струм (в піку до 4 А), необхідно поставити конденсатори якомога ближче до плати.

Розмір кроку (і мікрокроку):

У крокових двигунів зазвичай встановлена конкретна величина (наприклад 1,8° або 200 кроків на оборот), при якій досягається повний оборот в 360°. Мікрокроковий драйвер, такий як A4988 дозволяє збільшити кількість, за рахунок можливості управління проміжними кроками. Це досягається шляхом збудження обмоток середньої величини струму. Наприклад, управління мотором в режимі чверті кроку дасть двигуну з величиною 200-кроків-за-оберт вже 800 мікрокроків при використанні різних рівнів струму.

Дозвіл (розмір кроку) задається комбінаціями перемикачів на входах (MS1, MS2, і MS3). З їх допомогою можна вибрати п'ять різних кроків, відповідно до таблиці нижче. На входи MS1 і MS3 перемикача встановлено 100 кОм резистори, що підтягають на землю, а на MS2 - 50 кОм, і якщо залишити їх не підключеними, двигун працюватиме в повнокроковому режимі. Для правильної роботи в режимі мікрокроку необхідний слабкий струм, який забезпечується обмежувачами по струму, інакше, проміжні рівні некоректно сприйматимуться, і двигун пропускатиме мікрокроки.

Таблиця 1.4 – Настройки кроку управління

MS1	MS2	MS3	Дозвіл мікрокроку
Низький	Низький	Низький	Повний крок
Високий	Низький	Низький	1/2 кроку

Низький	Високий	Низький	1/4 кроку
Високий	Високий	Низький	1/8 кроку
Високий	Високий	Високий	1/16 кроку

Виводи управління:

Кожен імпульс на вході STEP відповідає одному мікрокроку двигуна, напрям обертання якого залежить від сигналу на виводі DIR. Зверніть увагу, що виводи STEP і DIR не підтягнуті до якої-небудь конкретної внутрішньої напруги, тому ви не повинні залишати ці виводи плаваючими при створенні пристроїв. Якщо ви просто хочете обертати двигун в одному напрямі, ви можете з'єднати DIR безпосередньо з VCC або GND. Чіп має три різні входи для управління станом живлення: RST, SLP і EN. Зверніть увагу, що вивод RST плаває; якщо ви його не використовуєте, ви можете підключити його до сусіднього контакту SLP на друкарській платі, щоб подати на нього високий рівень і включити плату.

Обмеження струму:

Для досягнення високої швидкості кроку, живлення двигуна, як правило, подають набагато вищу, ніж це було б допустимо без активного обмеження струму. Наприклад, типовий крокový двигун може мати максимальний струм 1А з 5 Ом; опором обмотки, звідси максимально допустиме живлення двигуна рівне 5 В (U=I*R). Використання такого двигуна з живленням 12 В дозволить підвищити швидкість кроку. Проте щоб запобігти пошкодженню двигуна, необхідно обмежити струм до рівня нижче 1 А.

A4988 підтримує активне обмеження струму, яке можна встановити регульованим потенціометром на платі. Один із способів встановити граничний струм - підключити драйвер в повнокроковий режим і вимірювати струм, що протікає через одну обмотку двигуна без синхронізації по входу STEP. Зміряний струм буде рівний 0,7 частин граничного струму (оскільки обидві обмотки завжди обмежуються приблизно на 70% від поточної настройки граничного струму в

повнокроковому режимі). Врахуйте, що при зміні логічної напруги Vdd, на інше значення, змінить граничний струм, оскільки напруга на виведенні "ref" є функцією Vdd.

Ще один спосіб встановити граничний струм – зміряти напругу на виводі "ref" і обчислити отримане обмеження струму (резистори SENSE рівні 0,05 Ом). Напруга виводу доступна через металізований крізний отвір (на шовкографії друкованої плати). Обмеження струму відноситься до опорної напруги таким чином:

Current Limit = VREF 2,5

Наприклад: опорна напруга рівна 0,3 В, граничний струм 0,75 А. Як згадувалося вище, в режимі повного кроку, струм через котушки обмежений 70% від поточної межі, тому, щоб отримати повний крок струму котушки в 1 А, поточна межа повинна бути 1 А / 0,7 = 1,4 А, що відповідає VREF 1,4 А / 2,5 = 0,56 В. Дивіться специфікацію A4988 для отримання додаткових відомостей.

Рекомендації по розсіюванню потужності:

Максимально допустимий струм що подається на обмотку, у мікросхемах A4988 рівний 2 А. Фактичний струм, який можна подати на плату, залежить від якості охолоджування мікросхеми. Плата розроблена з урахуванням відведення тепла від мікросхеми, але при струмі вище 1 А на обмотку, необхідний тепловідвід або інше додаткове охолоджування.

Зверніть увагу, що струм, заміряний на джерелі живлення, як правило, не відповідає величині струму на обмотці. Оскільки напруга що подається на драйвер, може бути значно вище за напругу на обмотці, то, відповідно, вимірюваний струм на джерелі живлення може бути трохи нижче, ніж струм на обмотці (драйвер і обмотка в основному працюють як кероване джерело з покроковим пониженням живлення). Крім того, якщо напруга живлення набагато вище необхідного двигуну рівня для досягнення необхідного струму, то скважність буде дуже низькою, що також приводить до істотних відмінностей між середнім і RMS струмом (середньоквадратичне значення змінного струму).

Час перемикання драйвера з режиму Disabled в режим Enabled складає 15

мілісекунд. Таким чином, після подачі на вхід ENBL драйвера A4988 сигналу, що включає управління, необхідно зробити паузу 20 мілісекунд, і лише після цього посилати сигнали, що управляють, на PUL.

- Якщо контролювати включення-виключення управління кроковим двигуном, то на вхід ENBL драйвера A4988 можна подати +5В від виходу Arduino. Також, якщо взагалі нічого не підключати до входу ENBL драйвера TB6560, то він буде постійно включений (Enable).

- У режимі 200 кроків на оберт між імпульсами на вході PUL необхідно зробити паузу 2 мілісекунди, щоб дати крововому двигуну відреагувати на команду переміщення ротора. Тобто, якщо на PUL передати наступні сигнали:

HIGH - LOW - [пауза 2мс] HIGH – LOW - [пауза 2мс] HIGH – LOW - [пауза 2мс], то ротор ШД зробить 3 кроки за 6 мілісекунд.

1.9 Розробка схеми управління

Для роботи лазерного гравера була розроблена схема підключення всіх необхідних елементів, двох драйверів управління крокових двигунів осі X і осі Y, а також 2 мікроконтролерів в одній буде прошивка grbl-master яка перетворить G-code у машинний для управління драйверів, а в іншій програма для читання з SD карти траєкторій і передачі їх в перший (Рисунок 1.26).

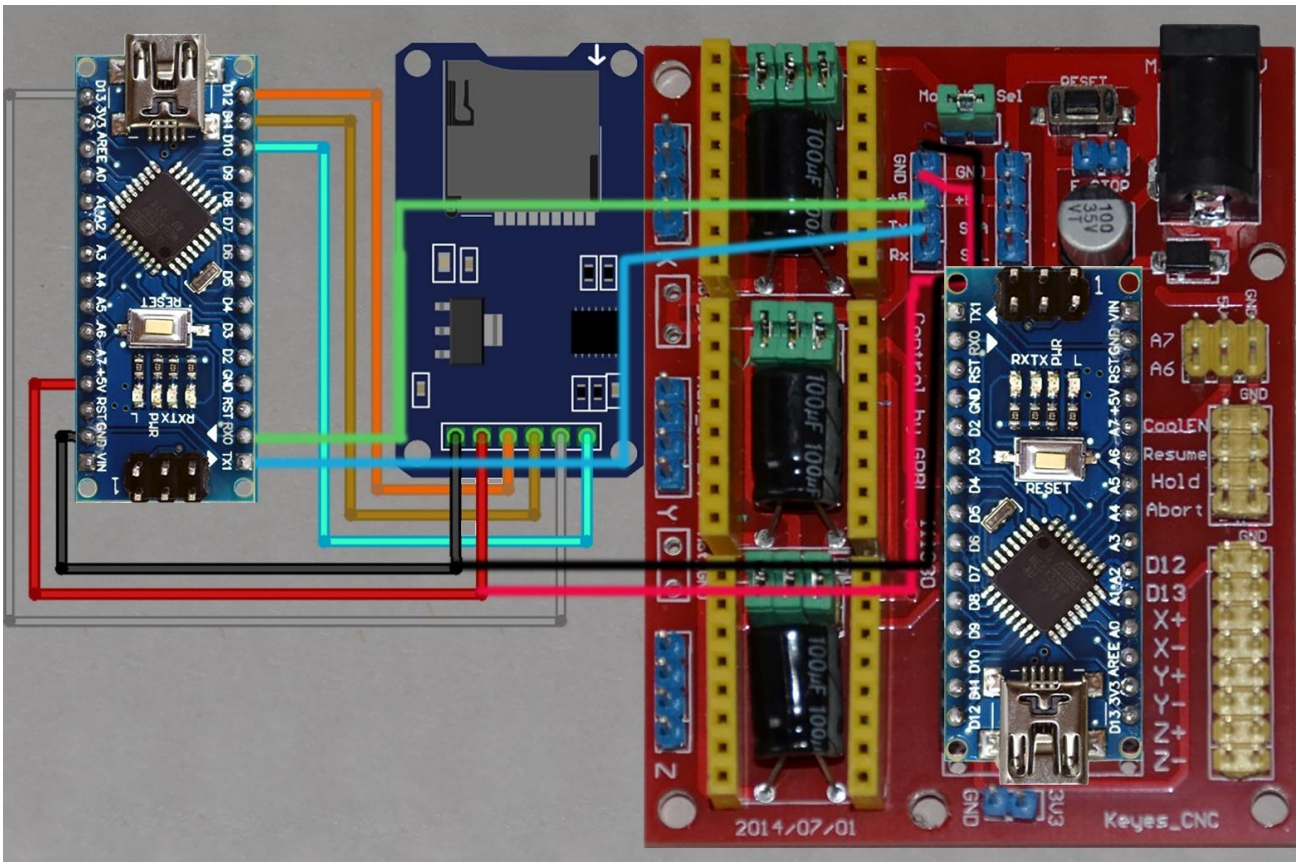

Рисунок 1.26 – Схема підключення автономного лазерного гравера

Таблиця 1.5 – Підключення SD карти

micro SD card	Контакт ATMEGA328
2	D4
3	D11
4	5v
5	D13
7	D12
8	GND

Таблиця 1.6 – Підключення мікроконтролерів до драйверів ШД

1й МК atmega328	2й МК atmega328	1й Драйвер ШД	2й Драйвер ШД
tx	rx		
rx	tx		
	D2	dir	
	D3		dir
	D5	step	
	D6		step

Висновок до розділу 1

В розділі було розглянуто загальну інформацію про мікроконтролери, їх типи, їх структуру, вибір платформи, та середовища в якому буде відбуватись програмування, відбулось перше знайомство з ними, і написана перша програма для управління верстатом.

Також було розглянуто принцип роботи верстата з числовим програмним управлінням, система команд, компенсація відхилення інструменту, способи задання координат, і ознайомлення з мовою програмування G-code.

Експериментально було розроблено схему підключення модулів верстата з числовим програмним управлінням, і підключення до нього автономної системи. Був написаний експериментальний шматочок коду для поєднання двох незалежних систем.

РОЗДІЛ 2
РОЗРОБКА ВЕРСТАТУ З ЧИСЛОВИМ ПРОГРАМНИМ УПРАВЛІННЯМ

2.1 Конструкторська частина

2.1.1 Призначення лазерного пристрою

На сьогоднішній день все частіше застосовують нанесення маркування на деталі, за допомогою лазера, або фрези, для гравіювання фрезою потрібна досить масивна і міцна станина, тому свій вибір спинив саме на гравіюванні лазером. Це проста процедура, надійна технологія із стійким результатом. Для нанесення маркування на поверхні металу потрібний лазер з потужністю близько 100-200 Вт. Проте такі лазери мають високу вартість і як наслідок і сам верстат має велику ціну. Для нанесення маркування також можна використовувати і малопотужні лазери, але при цьому потрібне застосування спеціальних покриттів, які мають хороші адгезійні властивості.

Для здійснення процесу нанесення маркування за допомогою малопотужного лазера із застосуванням полімерного порошку потрібно було сконструювати спеціальний робочий макет верстата.

Даний макет призначений для лазерної маркування деталей з різними габаритами. Конструкція лазерного пристрою дозволяє наносити маркування на будь-яких поверхнях.

2.1.2 Опис конструкції

В умовах лабораторії був спроектований лазерний гравер з ЧПУ для лазерного маркування деталей.

Перед проектуванням були розглянуті аналоги верстатів для лазерного маркування. За основний аналог взятий верстат з рухомим порталом, що забезпечує переміщення лазерної головки в подовжньому і поперечному напрямах. Для фокусування лазера по висоті є вертикальне переміщення лазера.

Для виключення перекосу і заклинювання руху порталу в подовжньому напрямі необхідна наявність двох крокових приводів [17]. Тому з метою спрощення кінематики верстата було ухвалено рішення про застосування одного двигуна, встановленого на рухомому столі. При цьому портал залишити

нерухомим в подовжньому напрямі. У поперечному напрямі рух відбувається переміщенням лазерної головки. У вертикальному напрямі переміщення відбувається підйомом і опусканням радіатора охолоджування з лазером.

Однією з основних вимог установки є можливість транспортування одній людині. Тому маса верстата повинна бути не більше 5 кг.

Як матеріал для виготовлення станини, була прийнята профільна квадратна труба з алюмінієвого сплаву, оскільки цей матеріал задовольняє по масі, легко піддається механічній обробці і має широке розповсюдження.

Електронна частина розташована в підставці лазерної установки.

Arduino Nano, побудована на мікроконтролері ATmega328 (Arduino Nano), має невеликі розміри завдяки чому може використовуватися в невеликих за розміром пристроях. У даній платі встановлений інтерфейсний чіп FTDI CH340. Arduino Nano може отримувати живлення через підключення MINI-B USB, або стабілізованої напруги 5 В зовнішнього джерела живлення. Автоматично вибирається джерело з найвищою напругою. На платформі Nano встановлено 8 аналогових входів, кожен дозволом 10 битий (тобто може приймати 1024 різних значення).

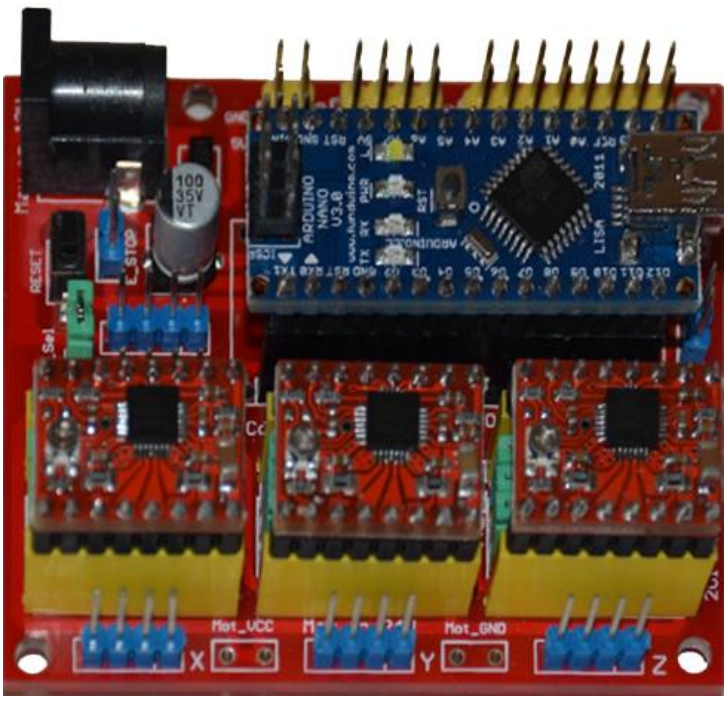

Рисунок 2.1 – Arduino Nano

На платформі Arduino Nano встановлено декілька пристроїв для здійснення зв'язку з комп'ютером, іншими пристроями Arduino або мікроконтролерами. ATmega328 підтримує послідовний інтерфейс UART TTL (5 В), зв'язок здійснюється через сигнали 0 (RX) і 1 (TX). Встановлена на платі мікросхема FTDI CH340 направляє даний інтерфейс через USB, а драйвери FTDI надають віртуальний COM порт програмі на комп'ютері. Моніторинг послідовної шини (Serial Monitor) програми Arduino дозволяє посилати і отримувати текстові дані при підключенні до платформи. Світлодіоди RX і TX на платформі мигатимуть при передачі даних через мікросхему FTDI або USB підключення. Проте це не завжди доцільно, не завжди є можливість підключити гравер до комп’ютера, тому була розроблена система автономної роботи верстата, з карти пам’яті.

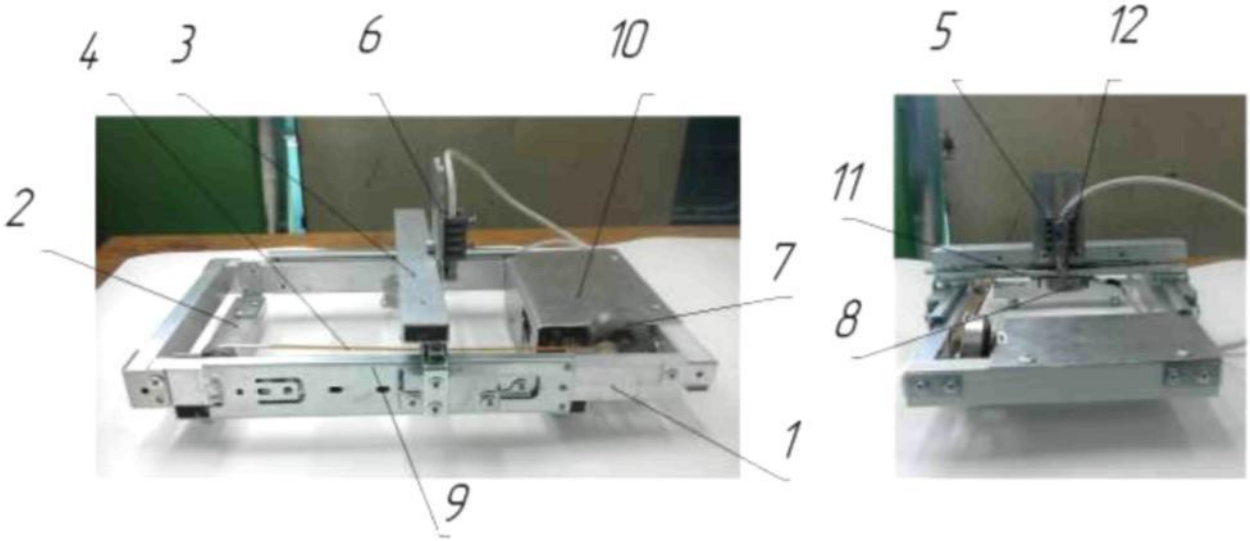

Рисунок 2.2 – Загальний вид макету верстата

Верстат складається з двох основних частин, таких як станина 1, і рухомого порталу 9. Станина служить для установки і закріплення рухомої частини, а так само служить корпусом для закріплення і установки всієї електронної апаратури.

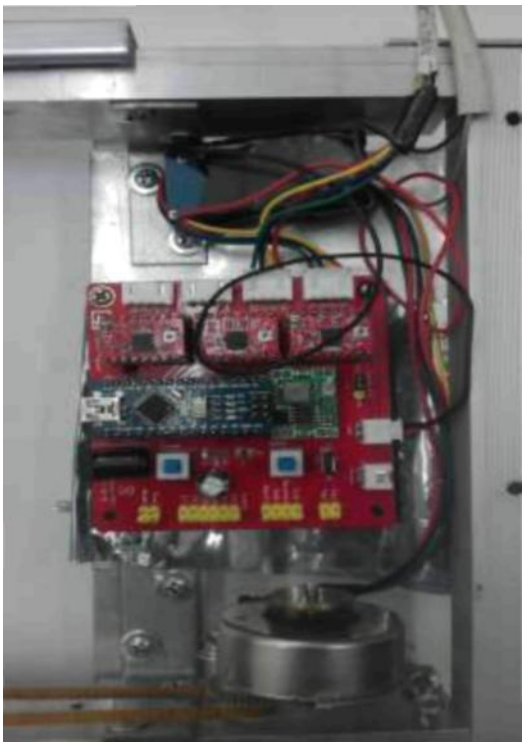

Рисунок 2.3 – Зовнішній вигляд розміщення електронної апаратури.

Макет верстата має переміщення в трьох координатах X, Y, Z.

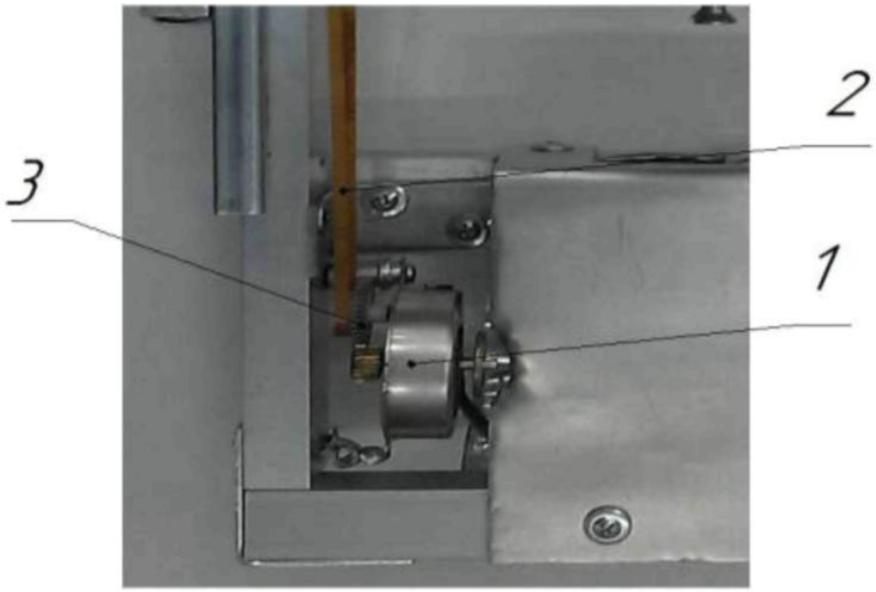

Рисунок 2.4 – Зовнішній вигляд переміщення в подовжньому напрямі

Переміщення в подовжньому напрямі здійснюється кроковим двигуном 1 через зубчате колесо 3 і зубчатим ременем 2. Як приводи були вибрані крокові двигуни з потужністю 7 Вт. Натягнення ремінної передачі забезпечується пружиною. Для поперечної подачі також використовується кроковий двигун із зубчатою рейкою.

Рисунок 2.5 – Зовнішній вигляд лазера, встановленого в макеті верстата

Рух у вертикальному напрямі здійснюється в ручну, переміщенням радіатора руху з лазером вгору-вниз встановлений на плиті, ослабивши гвинт М6. Охолоджування крокових приводів - повітря.

Для кріплення лазера на лазерному пристрої з ЧПУ для лазерної маркування деталей необхідно було спроектувати пристосування для кріплення лазера на вертикальну вісь макету верстата.

На першому етапі були розглянуті аналоги конструкцій для кріплення подібних деталей. Найраціональніше кріпити лазер в циліндрове пристосування так як корпус лазера має циліндрову форму. Це дає якнайкраще центрування і кріплення лазера на макеті верстата. Оскільки з лазера виділяється тепло, було ухвалено рішення сконструювати радіатор охолоджування.

Так же розглядалося декілька варіантів матеріалів виготовлення пристосування. У зв'язку з тим, що основною вимогою, що пред'являється до пристосування, є вага конструкції, хороша теплопровідність, то в якості основного матеріалу був прийнятий алюмінієвий сплав.

Як основна заготовки був прийнятий прокат звичайної точності розміром 32х100 мм з алюмінієвого сплаву Ад31т. Даний сплав задовольняв всім умовам, що пред'являються до пристосування.

Дане пристосування складається з двох основних елементів: плита і радіатор охолоджування.

Радіатор охолоджування встановлюються в уступ плити і закріплюється гвинтом на М6.

Основними вимогами, що пред'являються до плити, є шорсткість базового переміщення і точне розташування двох отворів під установку у вертикальній підставці верстата. До радіатора охолоджування також пред'являються вимоги до шорсткості, в базовому отвору під установку лазера, розташовано отвір під кріплення радіатора охолоджування до плити.

Базування пристосування на вертикальну вісь верстата забезпечується двома отворами під гвинти. Кріплення лазера в радіаторі охолоджування проводиться двома гвинтами з різьбою М5х4.

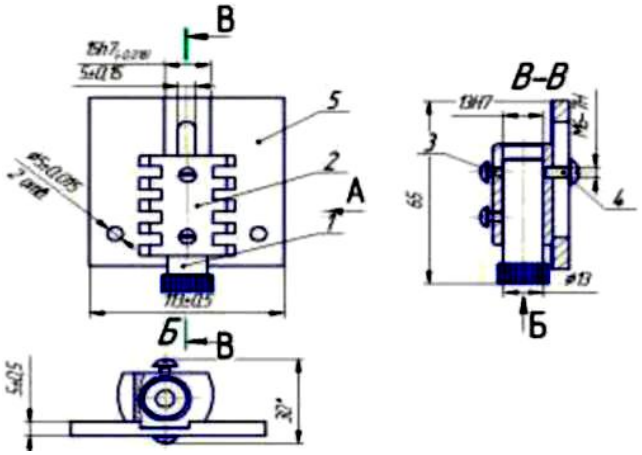

Рисунок 2.6 – Пристосування для кріплення лазера в зборі: 1-лазер, 2-радиатор охолоджування, 3- 2 гвинти М5х6, 4- гвинта М6х8, 5-плита.

2.1.3 Розрахунок площі охолоджування тепловідводу

Площа охолоджування тепловідводу розраховується чисто математично, змірявши основні розміри радіатора:

$$S = ((ax2) + ((b+c + t + до)xn) + l-d)xh \quad (2.1)$$

де: а - товщина підставки;

b і t - висота ребра з обох боків;

з - ширина верхівки ребра;

до - відстань між ребрами радіатора;

L - довжина радіатора;

n - кількість ребер на радіаторі;

h - висота радіатора.

S = ((5x2) + ((8 + 4 + 8 + 4)x10) + 36-4)x21 = 5922 мм^2.

Радіатор площею тепловідводу 5922 мм^2 з розрахунку, має тепловий опір Q = 2,3 °С/Вт. При допустимому перегріві 45°С отримуємо потужність розсіяння рівна 45/2,3=19,57 Вт.

$$M = \frac{G}{W} = \frac{45}{2,3} = 19,57 Bm \quad (2.2)$$

При допустимому перегріві 45°С отримуємо потужність розсіяння 19,57 Вт. Потужність розсіяння малопотужного лазера, який в нас використовується, рівна 8,5 Вт. Таким чином, радіатор охолоджування виконує умови тепловідводу.

2.1.4 Оцінка точності кінематики верстата

На рисунку 2.7. представлена схема впливу перекосу що направляють на діаметр світивши. З рисунку 2.7 видно, що на довжині між крайніми положеннями перекос складає 1 мм. Зміну діаметрів при перекосі тих, направляючих в один бік складає 0,005 мм. Відповідно при менших переміщеннях лазерної головки перекос направляючих складатиме менше значення, і відповідно змінюватиметься діаметр променю.

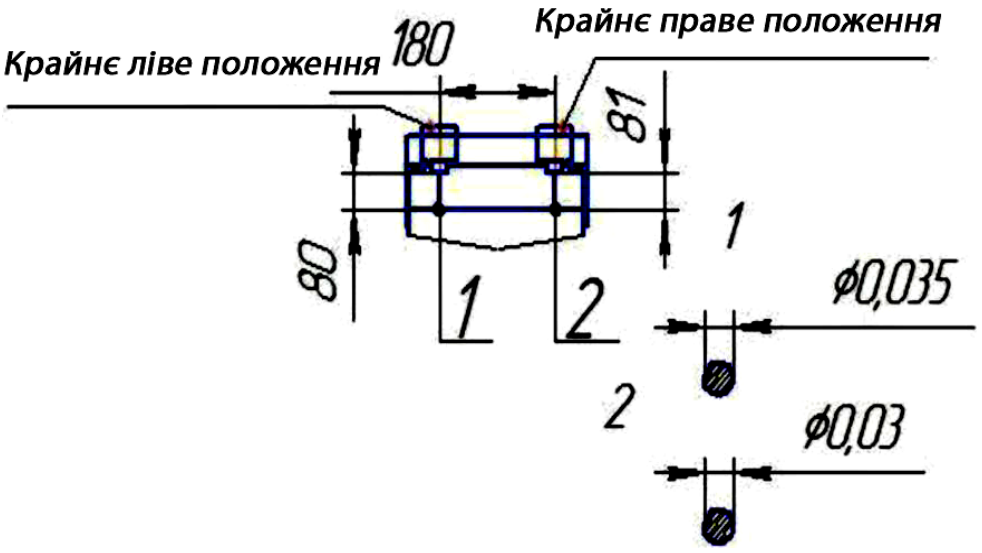

Рисунок 2.7 – Вплив перекосу направляючих ь порталу на діаметр променю

На рисунку 2.7 представлений вплив не перпендикулярності порталу до направляючих стала. З рисунка видно, що на граничному переміщенні столу в 580 мм перекос складає 2 мм. При найменших переміщеннях столу погрішність перекосу зменшуватиметься.

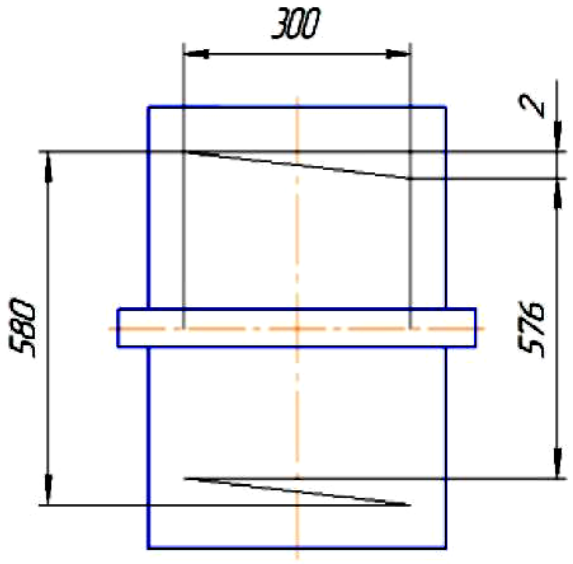

Рисунок 2.8 – Не перпендикулярність порталу, що до його направляючих

Розрахунок погрішності базування установки пристосування на каретці верстата не потрібно, оскільки кутове положення пристосування забезпечується штифтами, а положення в площині каретки може бути будь-яке, оскільки настройка нуля деталі проводиться системою ЧПУ.

2.2 Технологічна частина

2.2.1 Службове призначення і технічні характеристики складальної одиниці.

У магістерській дипломній роботі розробляється технологічний процес на виготовлення радіатора охолоджування.

Радіатор охолоджування входить до складу пристосування для кріплення малопотужного лазера на вертикальну вісь макету верстата. Радіатор охолоджування складається з двох деталей, втулки і плити. У даний виріб встановлюються два гвинти М5 для кріплення лазера, і один гвинт М6 для закріплення радіатора на плиті. Плита кріпиться двома гвинтами М6 на

вертикальній підставі макету верстата. Даний виріб виконує функцію базуючого елементу для малопотужного лазера, регулювання висоти фокусної відстані і утримання його в макеті верстата.

Даний виріб виготовлятиметься з алюмінієвого сплаву Ад31т оскільки цей матеріал задовольняє по масі і по обробці в порівнянні із сталлю. Ад31т ГОСТ 4784-97 застосовується у вигляді прокату.

Хімічний склад алюмінієвого сплаву приведений в таблиці 2.1 [1].

Таблиця 2.1 – Хімічний склад алюмінієвого сплаву Ад31т

Основні компоненти %				Домішки % (не більш)				
Al	Mg	Si	Fe	Cu	M n	Zn	Ti	інші
Останнє	0,45-0,9	0,22-0,6	0,35	0,1	0,1	0,1	0,1	0,15

Алюмінієвий сплав, що деформується, використовується в машинобудуванні для виготовлення деталей невисокої міцності і високої корозійної стійкості, що працюють в інтервалі від мінус 70 до плюс 50 град.

Механічні властивості сталі приведені в таблиці 2.2 [1].

Таблиця 2.2 – Фізико-механічні властивості алюмінію

$\sigma_в$,Мпа	$\sigma_{0,2}$ Мпа	KCU, Дж/см^2	ψ, %	НВ, 10^{-1} МПа
250	210	0,5	70	80

Технологічні властивості для Ад31т по ГОСТ 4784-97 не нормуються.

Найбільш відповідальними поверхнями є паз на втулці шириною 15 мм, отвір діаметром 13 мм під кріплення лазера, з квалітетом точності Н9, висока шорсткість на поверхнях пазу. До розмірів середньої точності відносяться довжина і глибина пазів, отвори під різьблення М5-7н і М8-7н, а також діаметр 30h11. До решти розмірів вимоги не пред'являються.

З оброблених поверхонь відносять поверхні з Ra=3,2 мкм (5 клас шорсткості). Поверхні, що мають параметр Ra=6,3 мкм, виходять після механічної обробки точінням.

2.2.2 Виробнича програма випуску і визначення типу виробництва

Деталь має об'єм випуску 1000 шт. в рік. По навчальному посібнику

заздалегідь визначаємо, що тип виробництва середньосерійний [1].

Виробнича програма випуск представлена в таблиці 2.3.

Таблиця 2.3 – Річна програма випуску виробу

Найменування виробу	Число виробів на програму	Маса, кг	
		Вироби	На річну програму
Втулка	1000	1.5	600

Для малого- і середньосерійного виробництва розраховуємо розмір партії запуску по навчальному посібнику [2]:

$$n=NaF \quad (2.3)$$

де F - число робочих днів в році;

а = 3, 6, 12, 24 - періодичність запуску в днях;

N - річна програма випуску виробів, шт.

2.2.3 Аналіз типового технологічного процесу.

Як заготовка для отримання радіатора охолоджування, що входить до складу лазерного верстата прийнятий сортовий прокат звичайної точності з алюмінієвого сплаву Ад31т ГОСТ 4789-97. Цей вид отримання заготівки є оптимальним для даної конструкції деталі, дозволяє отримувати розміри, близькі до розмірів деталі, що зменшує припуски на механічну обробку, а також однорідніщу структуру металу, що підвищує його механічні властивості.

Базовий технологічний процес виготовлення радіатора охолоджування розроблений для дрібносерійного виробництва і представлений в таблиці 2.4.

Таблиця 2.4 – Базовий технологічний процес

№ Обпер.	Найменування операції	Устаткування пристосування ріжучий і вимірник інструмент
1	2	3
005	Токарна 1 точити в Ш30 Н14 2 відрізувати заготівку в розмір 36-0,39 мм	Токарний верстат
010	Свердлувальна 1 свердлити в Ш13 Н9 згідно ескізу	Токарний верстат

015	Слюсарна 1 видалити задирки, притупити гострі кромки після фрезерування	Верстак Лещата Напилок

Продовження таблиці 2.4

1	2	3
020	Фрезерна 1 фрезерувати 2 площини згідно ескізу(з переустановленням)	Верстат 6Р82Г Лещата (підкладки)
025	Слюсарна 1 видалити задирки, притупити гострі кромки після фрезерування	Верстак Лещата Напилок
030	Фрезерна 1 фрезерувати уступ па розмір 2+0,1	Верстат 6Р81Г Лещата (підкладки)
035	Фрезерна 1 фрезерувати 8 пазів в розмір 4+0,12 мм (з переустановленням) 2 фрезерувати 2 фаски в розмір 0,5Ч45°	Верстат 6Р13 Лещата (підкладки)
	Свердлувальна 1 свердлити отвори Ш4,95 2 свердлити отвір Ш5,7	Верстат 2Н125 Лещата (підкладки)
045	Слюсарна 1 видалити задирки, притупити гострі кромки після свердлення 2 нарізувати різьблення М5-1,25-7н 3 нарізувати різьблення М6-1,5-7н	Верстак Мітчик М5-1,25-7н Мітчик М6-1,5-7н
050	Контрольна 1 перевірити деталь згідно вимогам креслення і техпроцесу	Плита
055	Консервація 1 поверхні деталей покрити тонким шаром машинного масла 2 оформити приймання деталей	Ділянка

При аналізі даного технічного процесу необхідно змінити послідовність операцій щоб скоротити тривалість технологічного процесу при виготовленні даного виробу.

В результаті аналізу можна прийняти такі шляхи поліпшення

технологічного процесу:

- застосувати прогресивніше устаткування;
- застосування концентрованого технологічного переходу;
- використовувати заготівку у вигляді прокату.

2.2.4 Аналіз технологічності об'єкту виробництва

Бази в технологічному процесі вибрані раціонально, дотримані правила і принципи базування. Засоби технологічного оснащення відповідають даному типу виробництва, устаткування, оснащення і вимірювальний інструмент універсальні, інструмент стандартизований, спеціальний інструмент не використовується. На кресленні достатньо видів, розмірів, перетинів, що дають повне уявлення про конструкцію деталі, а постановка розмірів раціональна. Вказані всі необхідні вимоги шорсткості, відхилення форми і розташувань поверхонь, зведення про матеріал виробу, застосуванні захисних покриттів, маса, а так само спосіб отримання заготівки.

2.2.5 Якісна оцінка технологічності

Технологічним є те, що деталь має хороші базові поверхні для установки на верстаті, дозволяє застосовувати високопродуктивні режими обробки. Деталь не має отворів, розташованих не під прямими кутами. Всі отвори розташовані під прямим кутом один до одного. Конструкція деталі дозволяє вести обробку площин на прохід. Матеріал обробляється стандартними інструментами з швидкорізальної сталі оснащеними твердосплавними пластинами. Взаємне розташування поверхонь деталі не викликає трудності при підводі ріжучого інструменту. Алюмінієвий сплав Ад31т володіє хорошою технологічністю. Отримувана заготовка раціональна.

Не технологічною є велика кількість пазів, оскільки їх обробка збільшує трудомісткість виготовлення деталі. Конструкція деталі вимагає розмічальних операцій, що є не технологічним. Багато слюсарних операцій. Не всі розміри можна проконтролювати за допомогою універсального вимірювального інструменту.

2.2.6 Кількісна оцінка технологічності

Кількісна оцінка технологічності розраховується по навчальному посібнику [3].

Коефіцієнт використання матеріалу обчислюється за формулою:

$$K_{им} = \frac{m_д}{m_з}, \tag{2.4}$$

де $K_{им}$ - коефіцієнт використання матеріалу,

$m_д$ - маса деталі, кг;

$m_з$ - маса заготівки, кг

$K_{им} \frac{0,36}{0,5} = 0,72.$

Оскільки $K_{им}$ більше 0,7, та умова виконується, деталь по цьому показнику технологічна.

Коефіцієнт уніфікації конструктивних елементів обчислюється за формулою:

$$K_{уд} = \frac{Q_{уд}}{Q_э}, \tag{2.5}$$

де $K_{уд}$ - коефіцієнт уніфікації конструктивних елементів;

$Q_{уд}$ - кількість уніфікованих елементів, шт.

$Q_э$ - кількість елементів всього у виробі, шт.

$K_{уд} = \frac{14}{37} = 0,37$

Оскільки $K_{уд}$ менше 0,6, то деталь по цьому показнику технологічна. Коефіцієнт точності обробки обчислюється за формулою:

$$K_{то} = 1 - \frac{1}{A_{ср}} = 1 - \frac{1}{3} = 0,66 \geq 0,5 \tag{2.6}$$

де $K_{то}$ - коефіцієнт точності обробки.

Оскільки Хто більше 0,5, то деталь по цьому показнику технологічна. Середній квалітет точності знаходимо по формулі:

$$A_{ср} = П_1 + 2П_2 + 3П_3 + ... + 14П_{14} / П_\Sigma \tag{2.7}$$

де $A_{ср}$ - середній квалітет точність мкм.;

$П_1$. $14П_{14}$ - кількість розмірів з квалітетом, шт.;

$П_{\Sigma}$ - число поверхонь деталі, шт.;

$A_{ср}1+2\cdot 2/2=3$

По цьому показнику деталь технологічна, оскільки середній квалітет точності більше 0,5.

Коефіцієнт шорсткості обчислюється за формулою:

$$K_{шр}=\frac{1}{Б_{ср}}, \tag{2.8}$$

де $K_{шр}$ - коефіцієнт шорсткості.

де $Б_{ср}$ - середня шорсткість поверхонь;

Середня шорсткість поверхонь знаходиться по формулі:

$$Б_{ср}=\frac{\sum_{i=1} Ra_i \cdot nRa_i}{n_{\sum}}=\frac{(3{,}2\cdot 3)+(6{,}3\cdot 3)+2{,}5\cdot 5}{18}=2{,}27, \tag{2.9}$$

$\sum Ra_i$ - задана шорсткість, мкм.;

nRa_i - кількість поверхонь тих, що мають шорсткість, шт.;

$n_{\sum}$ - сумарна кількість поверхонь, шт.

$$K_{шр}=\frac{1}{2{,}27}=0{,}44\le 0{,}32$$

Оскільки $K_{шр}$ більше 0,32, то, по цьому показнику деталь не технологічна.

В ході аналізу кількісної оцінки технологічності з'ясувалося, що всі показники відповідає вимогам технологічності. Це викликано особливостями конструкції виробу. В цілому можна укласти, то що по кількісній і якісній оцінках деталь є технологічна.

2.2.7 Економічна ефективність

Розрахунок собівартості Повну вартість комплектуючих виробів визначили по таблиці 2.5.

Таблиця 2.5 – Вартість компонентів

Найменування	Марка, розмір	ГОСТ, ТУ	Кількість,	Ціна за	Витрати

виробу			шт.	одиницю (крб.)	(крб.)
1	2	3	4	5	6
Мікросхеми	A3967 driver chip	Гост17467-79	3	50	150
	Atmega328		2	100	200
	LM317MDCYR		3	15	45
	CH340		2	20	40

Продовження таблиці 2.5

Найменування Виробу	Марка, розмір	ГОСТ, ТУ	Кількість, шт	Ціна за одиницю (крб.)	Витрати (крб.)
1	2	3	4	5	6
Діоди стабілітрони світлодіоди оптопари	LED RED 591NM 0603 SMD (2mA 1.8V)	ТУ 362.029	2	10	20
	LED YELLOW 591NM 0603 SMD (2mA 1.8V)		3	11	33
	LED BLUE 591NM 0603 SMD (2mA 1.8V)		2	15	30
	LED GREEN 591NM 0603 SMD (2mA 1.8V)		2	12	24
Найменування виробу	Марка, розмір	ГОСТ, ТУ	Кількість, шт.	Ціна за одиницю (крб.)	Витрати (крб.)
	1N4004		2	2	4
	1N4148		2	2	4
	MBR0520		4	2	8
Конденсатори	CAP 100UF 35V ELECT MZA SMD	Ожо.464. 214ТУ	3	20	60
	CAP 1UF 10V CER Y5V SMD0603		3	5	15
	CAP 680PF 50V CERAMIC X7R 0603		3	5	15
	22пФ		3	5	15

	100нФ		4	5	20
	CAP CER .10UF 50V Y5V 0603		8	5	40

Продовження таблиці 2.5

Найменування Виробу	Марка, розмір	ГОСТ, ТУ	Кількість, шт	Ціна за одиницю (крб.)	Витрати (крб.)
1	2	3	4	5	6
Резистори	RES 20K OHM 1/10W 5% 0603 SMD	ГОСТ 7113-77	3	2	6
	RES .75 OHM 1/4W 1% 0805 SMD		3	2	6
	RES 5.1K OHM 1/10W 5% 0603 SMD		3	2	6
	RES 10K OHM 1/10W 5% 0603 SMD		13	2	26
	RES 1.0K OHM 1/10W 5% 0603 SMD		13	2	26
	RES 240 OHM 1/10W 5% 0603 SMD		3	2	6
	RES 715 OHM 1/10W 1% 0603 SMD		3	2	6
	RES 910 OHM 1/10W 5% 0603 SMD		3	2	6
	RES 4.7K OHM 1/10W 1% 0603 SMD		2	2	4
Склотекстоліт	СФ-2-35		1	120	120
Кабель екранований	КГВЕВ 4х0,5		1	50	50
Корпус	ВОХ-103		1	70	70

Припой	Sn60/ Pb40		1	50	50
Флюс	ЛТІ-120		1	30	30
Разом					1224

Вартість закупівлі комплектуючих виробів з урахуванням транспортно-заготовчих витрат:

$$C_{закуп} = \sum_{i-1}^{n} S_{закуп} \cdot (1 + K_{тз})$$

де $K_{тз}$ - коефіцієнт транспортно-заготовчих витрат; $K_{тз}$ =0,04

$S_{закуп}$ – вартість купувальних комплектуючих виробів, грн.

$C_{закуп}$ =1224х(1+0,04)=1272,96 грн.

Ціни актуальні на 22.05.2021р. Валютний курс складав 27 грн. за 1 долар. Закупівля проводилася в магазинах: http:\www. chipidip.ru http:\\www.ebay.com

2.2.8 Виготовлення друкованої плати

Конструкторсько-технологічний розділ

Склали принципову схему верстата з числовим програмним управлінням. Відповідно до принципової схеми зробили розводку друкованої плати для розроблюваного пристрою.

У верстаті застосовується двостороння друкована плата, виконана на односторонньому склотекстоліті. Струмопровідні шари плати робляться комбінованим методом. Провідники виготовляються методом витравлення фольгованої підкладки, а отвори з металізацією - методом електрохімії. Виведення елементу і кріплення провідників на друковані платі роблять за допомогою контактного майданчика, виконаних потовщеною ділянкою з більшою шириною.

Розміри верстата, його конфігурація і фіксація друкованої плати робляться виходячи з його розрахункових розмірів, паяння, елементної бази, контрольованих даних і технічних і економічних показників.

Зовнішні розміри друкованої плати приймаються з вибором класу точності, обов'язково враховується те, що з укрупненням розмірів, будуть рости погрішності, допуск при виконання конструктивних елементів для фіксації їх на друкованій платі. Друкована плата, має форму у вигляді простого прямокутника.

Верстат є конструкцією з друкованою платою розмірами 200 х 150 мм з

розміщеними на ній електронними деталями і контактами для приєднання джерела живлення і силових ланцюгів.

Для конструювання електрохімічним методом друкованих плат береться листовий матеріал з ізолюючим прошарком і нанесеною на нього металевою фольгою з двох сторін. У даній магістерській роботі вибираємо склотекстоліт фольгований марки СФ-2-35. Потрібними параметрами номінальної товщини двосторонніх друкарських плат є: 0,5; 1; 1,5; 2; 2,5; 3 (мм). Беремо значення $H_т$ рівне 2 мм.

Монтажний отвір знаходиться зоні контактного майданчика, в яке вставлятиметься виводи елементу схеми. Монтажний отвір має металізовані бічні стінки. Застосування отворів з металізацією необхідне для більшої надійності друкованої плати.

Діаметр самих отворів зроблених в друкарській платі повинен бути більше діаметру, виводу елементу що вставляється, це дає можливість безперешкодного монтажу встановлюваного елементу на плату. Розмір отвору з металізацією залежить від товщини використовуваної плати. Це пов'язано з тим, що при осадженні металу з допомогою гальваніки на стінках максимального розміру, виконаного з товстою підставою, нерівномірною виходить товщина шару металізації, непокритими можуть залишитися місця з великим співвідношенням довжини до діаметру отворів. Розмір отвору з металізацією необхідно вибирати при умові, що він складатиме половину від товщини плати.

Розмір допустимої мінімальної ширини провідника на друкованій платі вибираємо рівне 1мм, мінімальна відстань в межах допуску між елементами вибираємо рівне 1,5мм.

Провідники на друкованій платі виготовляються з однаковою шириною у всьому діапазоні розводки. Сильноточні ланцюги силових елементів, і управління, виконуються за іншим принципом, виходячи з навантаження на їх струмопровідні ланцюги.

Друковані провідники розміщуються рівномірним монтажем по всій поверхні друкованої плати, враховуються наступні вимоги:

- координатна сітка повинна бути паралельною або під кутом, кратним 15°;
- напрям руху хвилі припою повинен бути паралельний, і не повинен перевищувати більш 30° з боку паяння елементів, за умови що провідник не покритий захисною маскою з боку паяння;
- до контуру дротового майданчику він повинен знаходиться перпендикулярно дотичній.
- Розташування електрокомпонентів на друкованій платі вибирається згідно друкованому вузлу і його конструктивним вимогам. При розміщенні електрокомпонентів враховуємо: їх взаємне розташування, раціональну компоновку, логічніше трасування прибирає паразитний вплив на слабкострумові елементи управління;
- забезпечити основні технологічні вимоги, що пред'являються до верстатів (автоматизована збірка і паяння, контроль вузлів); даними заходами забезпечити підвищену надійність, виконання високих масогабаритних показників, забезпечити високу швидкодію, вибрати оптимальний температурний режим із застосуванням тепловідводу, передбачити можливість хорошої ремонтопридатності.

Краї друкованої плати виготовляються паралельно відповідно до ліній координатної сітки пристрою.

Отвори і електрокомпоненти струмопровідного мідного шару розміщуються на друкованій платі відповідно до бази координат сітки. Електрокомпоненти струмопровідного мідного шару розміщуються з краю друкованої плати на відстані більш ніж товщина самої плати з урахуванням допусків і погрішностей при проектуванні і обліком лінійних розмірів друкованої плати.

Координатна сітка наноситься з кроком 2,54 мм безпосередньо на креслення. Центри отворів для монтажу розташовуються в перехресті сітки координат. При установці на друковану плату електрокомпоненту що має два виводи і більш, за умови що отвори між його виводами кратно координатній сітці пристрою, елемент розміщуємо виводами в отвори на вузлах сітки пристрою.

Розміщуваного електрокомпоненту який не має виводів, розмір між якими кратний кроку координатної сітки пристрою, то у вузлі координатної сітки розташовується один вивід, а центр отвору на горизонтальній і вертикальній лініях координатної сітки під інший вивід.

Виготовлення друкованих плат - це дуже складний технологічний процес. Розрахунок конструкції друкованої плати пристрою і її креслення відправляються в лабораторію, там виготовляються фотозразки на твердому носієві.

У лабораторії розміщені пристрої які дозволяють масштабувати розмір плати у декілька разів.

Плати надалі перевозять в цех спеціальної обробки, там вони доводяться до необхідних розмірів відповідно до тех.завдання. Виконується свердління отворів і різні операції з матеріалами, електрокомпонентами друковані плати.

Як тільки друкована плата пройшла технічну обробку плати потрапляє в серію на виробництві, автоматизованими лініями роблять металізацію отворів, на отвори осаджується мідь. Потім плата лудиться і покривається свинцем і оловом, надалі відбувається травління. У закритих ємностях з друкованої плати знімається поверхневий шар, на друкованій платі залишається лише метал - струмопровідні доріжки.

Потім ця друкована плата із струмопровідними доріжками відправляється на відділення для вирізання, там виконуються роботи на гільйотині згідно з розмірами друкованої плати.

Заздалегідь оброблені друковані плати відправляються на ділянку оксидування, надалі поступають на ділянку фрезерування, де відбувається їх обробка по контуру відповідно до розрахованих розмірів.

Після повної обробки плати, відбувається установка електрокомпонентів. Електрокомпоненти на платі розміщуються як вручну, або із застосуванням "лазерного щупа": плати розміщується в спеціальному пазу монтажного столу, лазерним променем підсвічуються місця на друкованій платі де будуть розміщуватися електрокомпоненти одного і того ж типу.

Паяння електрокомпонентів відбуватиметься хвилею припою, на ділянку з

рідкою розплавленою сумішшю свинцю і олова занурюється друковані плати зі встановленими електрокомпонентами і при певній температурі, регульованій спеціальними автоматами, в отворах друкованої плати виконується паяння виводів електрокомпонентів.

Для захисту друкованої плати від зовнішніх шкідливих дій навколишнього середовища і інших агресивних середовищ, наноситься захисна маска. Друкарська плата маркується і передається на контроль в технічний відділ ВТК.

Висновок до розділу 2

В розділі було розглянуто варіанти механіки верстата з числовим програмним керуванням, розроблено і прораховано точність та надійність, розроблені та витравлені друковані плати. Також були прораховані системи охолодження, і технологічності даного проекту.

РОЗДІЛ 3
ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ

Для перевірки працездатності системи, самий перший макет верстата було зібрано на Arduino Nano. Вона має достатній об’єм пам’яті, щоб вмістити велику керуючу програму GRBL об’ємом 80КБ. Вибір програми безальтернативний: для відносно дешевих налагоджувальних плат Arduino Pro Mini (120 грн.) було два варіанти, більш затратний RIBS [31], який коштує 1950 грн., і безкоштовний GRBL [32], якому було надано перевагу.

Розглядався також третій варіант – збирати верстат під керуванням професійної програми mach3 [33], але вона досить дорога (5304 грн.). Для керуючої програми апаратна частина досить проста, так як в верстаті знаходяться лише двигуни і кінцевики, а всіма розрахунками і керуванням займається програма встановлена на комп’ютері. Це створює багато незручностей: для роботи верстата потрібно, щоб був постійно увімкнений комп’ютер. Краще, щоб у верстата був свій ПК, який буде постійно споживати електроенергію. Також в mach3 не найкращий варіант інтерфейсу. Він побудований на LPT порті стандарту IEEE 1284, що дає змогу не ставити мікроконтролер, але змушує вести товстий жмут дроту від верстата до комп’ютера (близько десятка), які ловлять перешкоди, що спричиняє помилки в роботі обладнання.

Звісно можна у виробника програмного забезпечення придбати і апаратну частину за 2652 грн. купити двигуни і таке інше. Тоді певно простіше було б купити готовий верстат, але на жаль цей варіант підходить не для всіх. Тому було вирішено будувати верстат на платформі Arduino Pro Mini під керівництвом GRBL.

3.1 Операційна система GRBL

Операційна система GRBL передбачає підключення до комп’ютера по USB або UART інтерфейсу, але в цьому випадку буде 3-4 дроти. Їх можна екранувати від перешкод. Також можна поставити автономну систему, яка буде зчитувати G-код з карти пам’яті і відправляти в послідовний порт (лістинг в додатку 1).

Не маючи жодного досвіду в будуванні верстатів і не розуміючи як буде працювати кінематика, було вирішено зібрати першу версію з механіки від CD-ROM, де вже готова кінематика, стоїть кроковий двигун, ходовий гвинт і зібраний привод на каретку (Рисунок 3.1). Для гравіювання можна використати лазер, знятий з комп'ютерного DVD-RW привода, який коштує недорого.

Рисунок 3.1 – Кінематика CD-ROM

З механіки від CD-ROM було зібрано маленьку версію верстата. Мініатюрний лазер виявився на 0,5 Вт. Щоб не спалити його, лазер був увімкнений на потужність 0,3 Вт. і, навіть в такому режимі, він був здатний палити дерево та пластик.

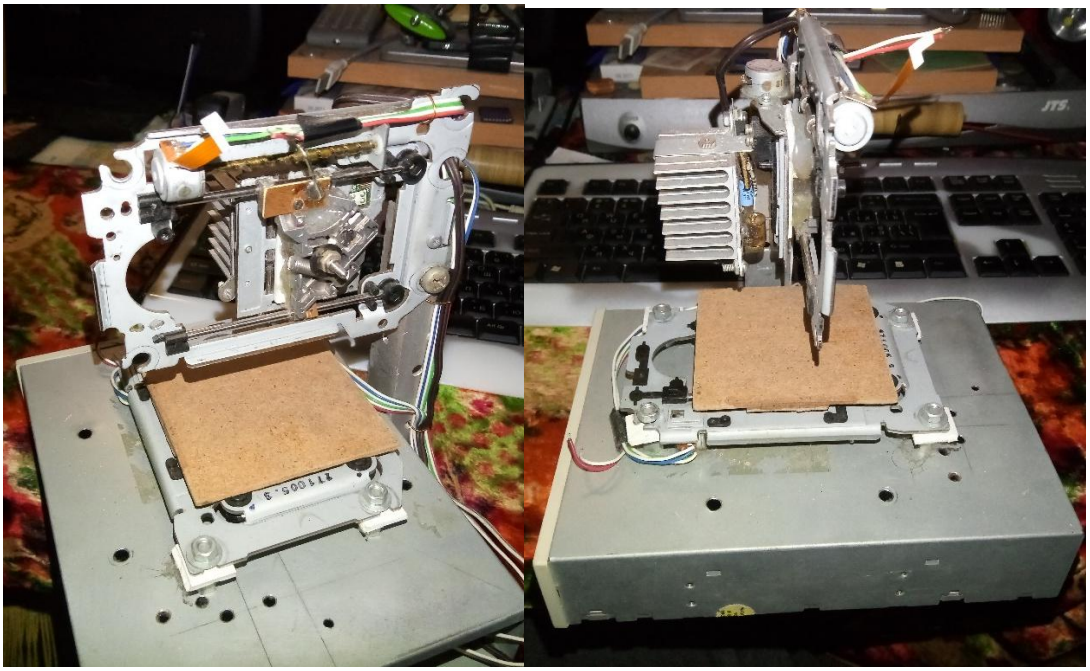

Рисунок 3.2 – Пробна версія верстата

Для першого запуску було використано драйвери від флоппі-дисків (FDD) (Рисунок 3.3).

Рисунок 3.3 – Драйвер FDD

Потужності драйверів не вистачало, адже вони були створені для руху маленької каретки FDD. Було встановлено значно потужніші двигуни, проте перші малюнки були не якісні, з малою роздільною здатністю (Рисунок 3.4).

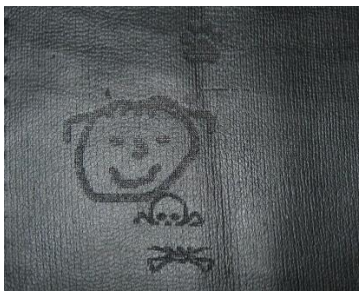

Рисунок 3.4 – Перші спроби гравіювання

На жаль, макет виявився досить примітивним. Було вирішено зібрати верстат з значно більшим полем і значно вищою роздільною здатністю. За основу було взято механіку від струменевих принтерів Lexmark Z615. Це дешеві та разові принтери. При виході з ладу двох картриджів, купівля нових перевищувала вартість принтера з цими картриджами, тому було використано направляючі каретки як направляючі для координат та крокові двигуни з Lexmark Z615.

Рисунок 3.5 – Lexmark Z615

В результаті вийшла така конструкція (Рисунок 3.6).

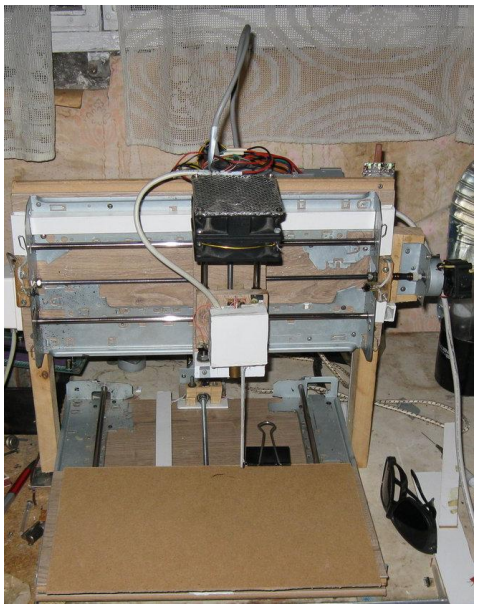

Рисунок 3.6 – Верстат з більшою роздільною здатністю

Цей верстат мав поле А4 формату. В якості ходового гвинта була використана будівельна шпилька на М5. Драйвери від FDD не могли рухати столом через велике тертя та великої ваги координат до 0,5 Кг. Було придбано спеціалізовані драйвери для керування кроковими двигунами А4988, які виявились найдешевшими (40 грн./шт.).

Для перевірки працездатності нових драйверів з мікроконтролером, було створено шилд на макетній платі, в якому можна було закріпити компоненти, адже їх стало більше. Для запобігання псування механіки були встановлені кінцеві вимикачі для того, щоб сповістити контролер про нештатну ситуацію та зупинити верстат. В результаті отримано таку плату (Рисунок 3.7).

Рисунок 3.7 – Плата керування верстатом

Рисунок 3.8 – Результат роботи верстата

Верстат показав значно кращу якість гравіювання, значно вищу швидкість роботи, але зі збільшенням складності пристрою, зменшувалась і надійність, навіть при виготовленні промисловим способом. Під час вмикання і вимикання світла в кімнаті, виникали збої в роботі верстата, тому було придбано дешеві шилди для верстатів з числовим програмним керуванням (92 грн.), виконані з екрануванням, відповідно до всіх правил побудови. Довелось замінити плату мікроконтролера на Arduino Nano, яка має як USB, так і UART. Як джерело живлення використано комп’ютерний ATX блок живлення.

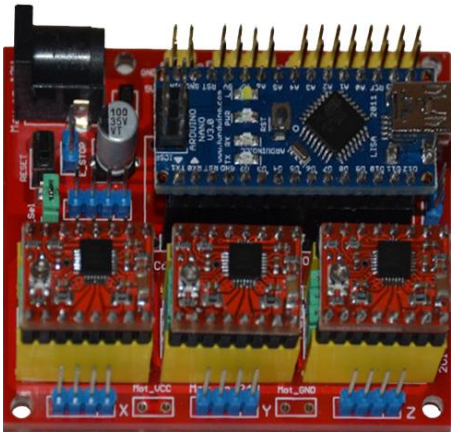

Рисунок 3.9 – Плата розширення верстата

Верстат чудово працював, але лише підключений до комп’ютера. Комп’ютер повинен бути постійно увімкнений. В той час, коли працює верстат, на комп’ютері краще не працювати, адже при підвисанні комп’ютера, або його перезавантаженні, малюнок на дошці буде зіпсовано, і тому потрібно починати гравіювання спочатку. Було прийнято рішення написати програму, яка буде автономно керувати верстатом, усунувши комп’ютер, і дати можливість користуватись ПК на власний розсуд.

Було вирішено використати Arduino Pro Mini. Система має зчитувати з карти пам’яті G-код і відправляти його у верстат і контролювати виконання команд. Для цього потрібен мікроконтролер, картрідер і кнопка, яка запускає процес гравіювання. Пристрій було зібрано навісним монтажем.

Рисунок 3.10 – Автономна система

До автономної системи було написано код:

```
#include <SPI.h>
#include <SD.h>
#include <SoftwareSerial.h>
SoftwareSerial cnc(8, 9);
const int chipSelect = 10;
//#define CMD_MAX_TIME 10000
File logFile;
void setup()
{
 pinMode(SS, OUTPUT);
 SD.begin(chipSelect);
 Serial.begin(115200);
 cnc.begin(115200);
```

```
}

void loop()
{
 File dataFile = SD.open("cat.cnc");
 if (dataFile){
 while (dataFile.available())
 {
 char rByte = dataFile.read();
 Serial.write(rByte);
 cnc.print(rByte);
 if (rByte == 13)
 {
 rByte = Serial.read();
 if (rByte != -1)
 {
 cnc.print(rByte);
 }
 while (rByte != 111 ) { // Вихід по timeOut
 rByte = Serial.read();
 if (rByte != -1)
 {
 cnc.print(rByte);
 }
 }
 cnc.print(rByte);
 }
 }
 dataFile.close();
 }
 else {Serial.println("error opening cat.cnc");}
}
```

Програма деякий час працювала, проте не стабільно. Отже, було знайдено більш стабільну версію програми:

```
  // include the SD library:
  #include <SPI.h>
  #include <SD.h>
  // set up variables using the SD utility library functions:
  Sd2Card card;
  // change this to match your SD shield or module;
  // Arduino Ethernet shield: pin 4
  // Adafruit SD shields and modules: pin 10
  // Sparkfun SD shield: pin 8
  // MKRZero SD: SDCARD_SS_PIN
  const int chipSelect = 10;
  int Pin = 14;
  File myFile;
  String inStringSerial = ""; // вхідне повідомлення від Serial
  void setup()
  {
   pinMode (Pin, INPUT_PULLUP);
   while (digitalRead(Pin) == HIGH) {
   }
   Serial.begin(9600);
```

```
 while (!Serial) {
 ; // wait for serial port to connect. Needed for native USB port only
 }
 //Serial.print("\nInitializing SD card...");
 if (!SD.begin(chipSelect)) {
 //Serial.println("initialization failed!");
 while (1);
 }
 //Serial.println("initialization done.");

 // we'll use the initialization code from the utility libraries
 // since we're just testing if the card is working!
 if (!card.init(SPI_HALF_SPEED, chipSelect)) {
 //Serial.println("initialization failed. Things to check:");
 //Serial.println("* is a card inserted?");
 //Serial.println("* is your wiring correct?");
 //Serial.println("* did you change the chipSelect pin to match your shield or module?");
 while (1);
 } else {
 //Serial.println("Wiring is correct and a card is present.");
 }
 }
void loop()
{
 String t = "";
 // re-open the file for reading:
 myFile = SD.open("1.txt");
 int temp=0;
 temp = myFile.available();
 //Serial.println(temp);
 if (myFile) {
 //Serial.println("test.txt:");
unsigned long pos = 0;
 // read from the file until there's nothing else in it:
 unsigned long t1 = millis();
 while (myFile.available()) {
 //Serial.write(myFile.read());
 myFile.seek(pos);
 int c;
 c = myFile.peek();
 t += char(c);
 if(t.lastIndexOf("\n")>-1)
 {
 Serial.print(t);
 t = "";
 //Serial.println(micros()-t1);
 waitOK();
 //t1 = micros();
 }
 pos++;
 }
 // close the file:
 myFile.close();
 } else {
 // if the file didn't open, print an error:
 //Serial.println("error opening test.txt");
```

```
 }

 while(1);
 /*
 if(Serial.available())
 {
 fastReadSerial();
 }
 */
}
void slowReadSerial()
{
 inStringSerial = "";
 inStringSerial = Serial.readString();
 inStringSerial.trim();
 Serial.println(inStringSerial);
}
void fastReadSerial()
{
 inStringSerial = "";
 int c;
 while(Serial.available())
 {
 c = Serial.read();
 inStringSerial += char(c);
 delay(3);
 }
 inStringSerial.trim();
 Serial.println(inStringSerial);
}
void waitOK()
{
 bool outWhile = 1;
 while(outWhile)
 {
 if(Serial.available())
 {
 while(Serial.available())
 {
 Serial.read();
 delay(1);
 }
 outWhile = 0;
 }
 }
}
```

В результаті вийшов цілком працездатний автономний верстат, який може працювати як сам по собі, так і підключений до комп’ютера та може бути керований CAM системою Grbl Controller, SerialComCNC, або використовувати спеціалізований слайсер для 3D принтера Repetier-Host. Він може керувати і лазерним гравером.

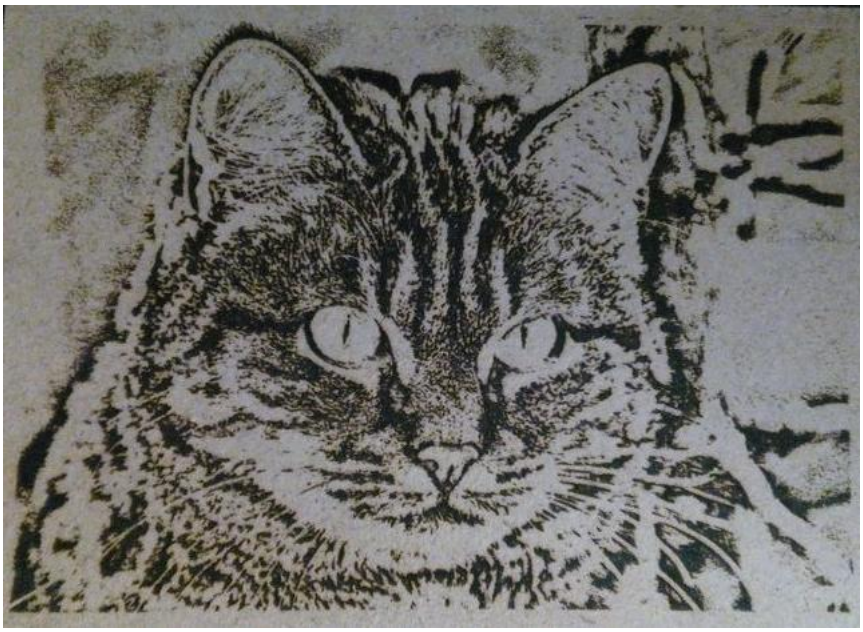

Рисунок 3.11 – Результат гравіювання верстата в автономному режимі

Результат гравіювання виявився задовільним, проте кожного разу доводиться фокусувати лазер. Неможливо точно позиціонувати лазер, і якщо все ж невдача, продовжити гравіювання, тому що не можливо попасти по тих самих координатах, по-перше через люфт в гвинтовій парі, а по-друге тому, що початок координати виставляється вручну. Дорогі 3D принтери роблять позиціонування автоматично, знаходять початок координат і виставляють відстань від сопла до столу. Автономна система там реалізована, також є можливість керувати координатами не з комп'ютера, а саме з кнопок в автономному режимі. Є можливість щось виправити в налаштуваннях. Також в мережі Інтернет є доступ до безкоштовної операційної системи для 3D принтерів Marlin. Marlin також випускають різне обладнання з ЧПУ [34].

3.2 Операційна система для 3D принтера Marlin

Завдяки експлуатаційним характеристикам програмного забезпечення Marlin, її вдалось адаптувати до гравіювання лазером, зроблено точнішу механіку. Було придбано платформу Arduino Mega 2560 R3 (Рисунок 3.12), оскільки в іншу неможливо залити 16МБ коду, плата RAMPS 1.4 для Arduino Mega 2560, RAMPS LCD 2004.

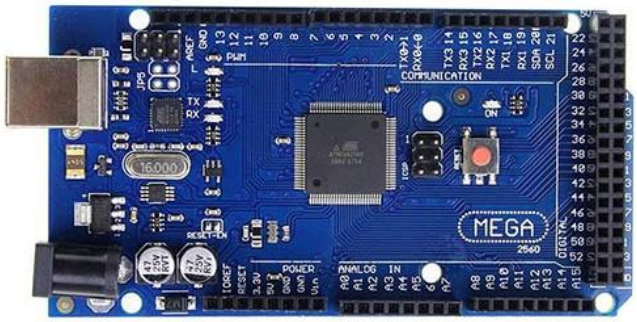

Рисунок 3.12 – Arduino Mega 2560 R3

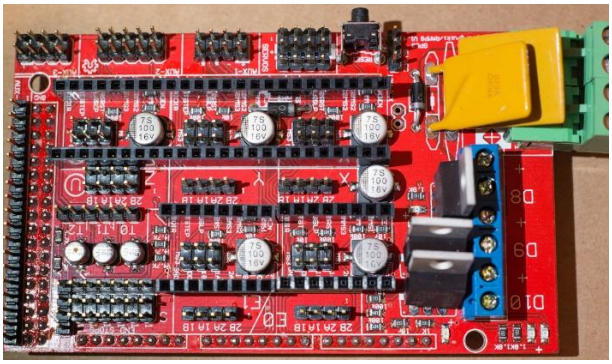

Рисунок 3.13 – Плата RAMPS 1.4 для Arduino Mega 2560

Рисунок 3.14 – RAMPS LCD 2004

Також було вирішено зменшити люфт в гвинтовій парі, замінивши будівельну шпильку на ходовий гвинт з трапецеїдальною різьбою і підігнаною під неї бронзовою гайкою, КГП (ШВП) (кульково-гвинтова пара), звісно краще, але вона перевищить по ціні сумарно всі спроби побудови верстату.

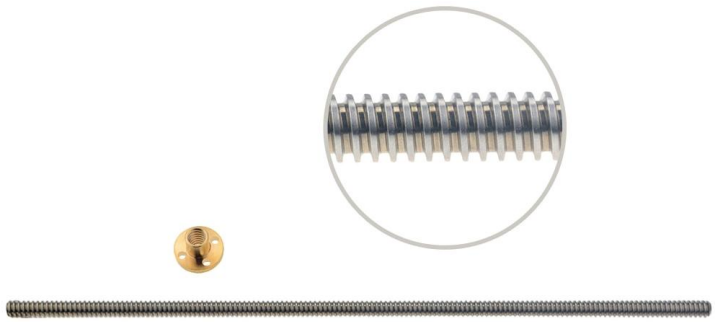

Рисунок 3.15 – Ходовий гвинт з трапецеїдальною різьбою Т8

Направляючі були взяті з радянського принтера, крокові двигуни з нього ж. сам корпус був зварений з профільних труб, зібраний на гвинтове з’єднання, гвинтами М8, лінійні підшипники для здешевлення проекту були надруковані з пластику, встановлений лазер на 30Вт. був спроектований датчик рівня столу. Лістінг коду операційної системи наведено в додатку 2.

В результаті розроблено такий верстат, і на даний момент в ньому реалізовано всі потрібні функції: гравіювання органічних матеріалів, таких як деревина, пластик, фанера, а також скло, покрите фарбою, але результаті фарба і шар скла вигорають. Також є можливість різати тонку фанеру.

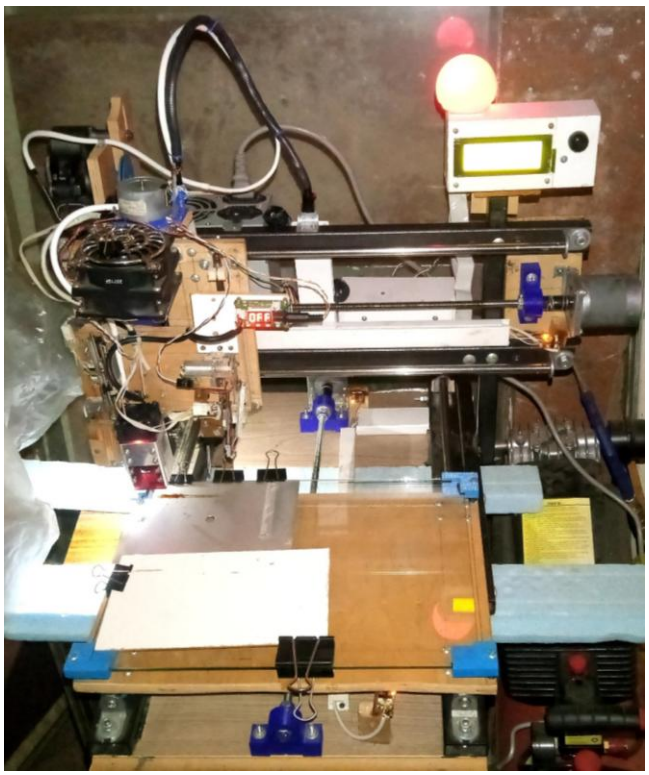

Рисунок 3.16 – Верстат під керівництвом Marlin

Для тесту було вигравіювано кілька портретних робіт, проте з етичних міркувань їх продемонструвати не можу, хіба що не надто вдалий портрет невідомої дівчини, зроблений на проміжній стадії налаштування.

Рисунок 3.17 – Тестове гравіювання на дереві

Висновок до розділу 3

Цей розділ повністю присвячений дослідженням. Розглянуто типи операційних систем для управління верстатами і їх відмінності, спробував їх на тестових версіях верстатів. В домашніх умовах зібрано пробні моделі, спочатку на макетній платі, сам верстат зібрано з підручних засобів. Потім зібрано на дешевих китайських модулях, для розуміння в чому різниця.

Далі зібрано верстат на більш надійні елементній базі, з надійнішою і дорожчою механікою, використано складніші і дорожчі модулі, розроблено повнофункціональний пульт керування верстатом, вивчено складну операційну систему для управління 3D принтерами, і переконфігуровано її для роботи з лазерним гравером. Було отримано багато позитивних результатів, і набуто навички з розробки і використання верстатів з ЧПУ.

ЗАГАЛЬНІ ВИСНОВКИ

У результаті проведених досліджень і розрахунків був спроектований пульт управління для верстату з числовим програмним управлінням. Згідно технічного завдання даний пристрій здатний виконувати зчитування робочої програми із зовнішніх носіїв і перетворювати їх в машинні коди для управління робочим механізмом верстата. Це підтверджено практично рисунками з результатами робіт верстата.

В ході проектування був використаний мікроконтролер архітектури ARM, що дозволив ефективно використовувати обчислювальну потужність мікроконтролера. Було розглянуто принцип роботи верстата, вибраний і прошитий мікроконтролер, розроблена схема і керуюча програма для пульту керування верстатом, і наприкінці, знайдено альтернативні варіанти готових рішень, що значно спростили б дослідження. Обидва результати відповідають технічним вимогам, і можуть бути використані в подальшій роботі. Як апаратні, так і програмні засоби добре себе зарекомендували, і показали чудові результати.

В подальшому заплановано встановити на каретку окрім лазера, ще й шпиндель, що дозволить не лише маркувати лазером, а й гравіювати фрезою, або фрезерувати друковані плати, що може дуже спростити їх виготовлення в домашніх умовах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Автоматизация проектно-конструкторских работ и технологической подготовки производства в машиностроении / Под ред. О. И. Семенкова. Минск : Высшая школа, 2005. Т. 1. 352 с.
2. Бабенко М. І., Стаценко В. В. Дослідження систем збору даних та керування електромеханічними пристроями на базі сучасних мікроконтролерів. *Технології та дизайн*. 2019. № 2. URL: http://nbuv.gov.ua/UJRN/td_2019_2_6 (дата звернення: 12.09.2021).
3. Бачинський Р. В. Контролер керування кроковими двигунами. *Вісник Національного університету «Львівська політехніка». Комп'ютерні системи та мережі*. 2013. № 773. С. 3–7. URL: http://nbuv.gov.ua/UJRN/VNULPKSM_2013_773_3 (дата звернення: 12.09.2021).
4. Бондар М. Ю., Єськін М. Ю., Заєць С. С., Максимчук І. В. Діагностика процесу обробки кінцевими фрезами, на фрезерних верстатах з числовим програмним управлінням. *Технологічні комплекси*. 2014. № 1. С. 230–233. URL: http://nbuv.gov.ua/UJRN/Tehkom_2014_1_40 (дата звернення: 12.09.2021).
5. Глушаков С. И., Сурядный А. С. Программирование на Visual Basic 6.0. Харьков : Фолио, 2004. 512 с.
6. Гольдштейн А. И., Молочник В. И. О внутренней структуре постпроцессоров. *Повышение эффективности использования станков с ЧПУ*. Киев : Знание, 2006. С. 25–26.
7. Дубинин П. И. К вопросу классификации обработки природного камня. *Горный информационно-аналитический бюллетень*. 2006. №5. С. 36–48.
8. Дубинина А. П. Влияние технологических режимов на качество обрабатываемых поверхностей алмазов при групповой огранке. *Горный информационно-аналитический бюллетень*. 2005. №9. С. 321–325.
9. Дубинина А. П. Метод групповой огранки мелких алмазов. *Сборник научных трудов IV Международной научно-технической конференции «Добыча,*

обработка и применение природного камня». Магнитогорск : МагГТУ, 2004. С. 224–227.

10. Дубинина А. П. Особенности технологического процесса групповой огранки алмазов в режиме пластического шлифования. *Труды VII Всероссийский научной конференции «Дизайн и технология художественной обработки материалов».* Челябинск : Издательство ЮУрГУ, 2004. Вып. 8. С. 49–53.
11. Заяць В. М., Колосов В. Р. Алгоритми і програмні засоби виконання арифметичних операцій над числами у форматі часу мікроконтролера сімейства AVR. *Вісник Національного університету «Львівська політехніка». Інформаційні системи та мережі*. 2013. № 770. С. 50–56. URL: http://nbuv.gov.ua/UJRN/VNULPICM_2013_770_9 (дата звернення: 12.09.2021).
12. Кавин Я. М. Лазерна обробка зернистих структур – метод аналізу поширення теплового поля в об’єкті. *Поліграфія і видавнича справа*. 2018. № 1. С. 109–113. URL: http://nbuv.gov.ua/UJRN/Pivs_2018_1_14 (дата звернення: 12.09.2021).
13. Кондрашов Ю. Н. Visual Basic для Windows. Формы и элементы управления. Учебное пособие. Москва : Академия бюджета и казначейства, 1997. 242 с.
14. Кондрашов Ю. Н., Мещерякова И. А., Лебедев В. М. Visual Basic 6.0. Описание языка. Основные элементы управления : учебное пособие. Москва : Академия бюджета и казначейства, 2003. 77 с.
15. Королев Н. AVR: аппаратные средства разработчика. *Компоненты и технологии*. 1999. №1 . С. 30–32.
16. Королев Н., Королев Д. AVR-микроконтроллеры второго поколения: новые аппаратные возможности. *Компоненты и технологии*. 2003. № 4. С. 112–117.
17. Королев Н., Королев Д. AVR-микроконтроллеры: большое в малом. *Схемотехника*. 2001. № 5. С 44–46.
18. Королев Н., Королев Д. AVR-микроконтроллеры: программные средства. *Компоненты и технологии*. 2000. № 4. С. 24–26.
19. Королев Н., Королев Д. AVR-микроконтроллеры второго поколения: средства разработчика. *Компоненты и технологии.* 2003. № 7. С. 124–128.

20. Косиловой А. Г. Справочник технолога-машиностроителя: в 2 т. Т. 2 / под ред. А. Г. Косиловой, Р. К. Мещерякова, А. Г. Суслова. 5-е изд. исправл. Москва : Машиностроение-1, 2003. 944 с.
21. Костючко С. М., Склянчук О. М., Ілюшик Р. С. Математичні основи та програмування мікроконтролера ATMEGA328 з використанням POLOLU 3PI ROBOT. *Комп'ютерно-інтегровані технології: освіта, наука, виробництво*. 2017. № 28-29. С. 5–9. URL: http://nbuv.gov.ua/UJRN/Kitonv_2017_28-29_3 (дата звернення: 12.09.2021).
22. Кривонос О. М., Кузьменко Є. В., Кузьменко С. В. Огляд та перспективи використання платформи Arduino Nano 3.0 у вищій школі. *Інформаційні технології і засоби навчання*. 2016. Т. 56, вип. 6. С. 77–87. URL: http://nbuv.gov.ua/UJRN/ITZN_2016_56_6_9 (дата звернення: 12.09.2021).
23. Крупський С. А., Гуда А. І. Ефективність мікроконтролерного управління двигуна постійного струм. *Молодий вчений*. 2019. № 1(2). С. 270–272. URL: http://nbuv.gov.ua/UJRN/molv_2019_1%282%29__6 (дата звернення: 12.09.2021).
24. Лебедев В. М., Мещерякова Н. А., Распутин А. П. Основные возможности Visual Basic 6.0 для работы с файлами, графикой и базами данных: Учебное пособие. Омск : ООИПКРО, 2004. 88 с.
25. Лісовець С. М., Дідович І. О. Швидкісні інтерфейси для зв'язку мікроконтролерів із периферійними пристроями. *Технології та дизайн*. 2016. №2. URL: http://nbuv.gov.ua/UJRN/td_2016_2_5 (дата звернення: 12.09.2021).
26. Ловыгин А. А., Теверовский Л. В. Современный станок с ЧПУ и CAD/CAM-система. Москва : ДМК Пресс, 2015. 280 с.
27. Лотиш Я. В. Система управління сервоприводами на базі контролера Arduino UNO. *Комп'ютерно-інтегровані технології: освіта, наука, виробництво*. 2013. № 12. С. 111–114. URL: http://nbuv.gov.ua/UJRN/Kitonv_2013_12_24 (дата звернення: 12.09.2021).

28. Механика промышленных роботов : Учеб. пособие для вузов. В 3 кн. / Под ред. К. В. Фролова, Е. И. Воробьева. Москва : Высш. шк., 1988. Кн. 1: Кинематика и динамика / Е. И. Воробьев, С. А. Попов, Г. И. Шевелева. 304 с.
29. Мещерякова Н. А. Алгоритмизация и обьектно-ориентированное программирование в курсе информатики высших учебных заведений экономического профиля. *Компьютеризация обучения и проблемы гуманизации образования в техническом вузе: материалы Международной научно-методической конференции*. Пенза, 2003. С. 301–305.
30. Микитин О. В., Черенюк М. С. Огляд та аналіз малогабаритної мікроконтролерної техніки з великими можливостями. *Вісник Криворізького національного університету*. 2014. Вип. 37. С. 122–126. URL: http://nbuv.gov.ua/UJRN/Vktu_2014_37_28 (дата звернення: 12.09.2021).
31. Морозов В. И., Дубинин П. И., Дубинина А. П. Тенденции развития художественной обработки природного камня в России. *Труды XI Международного симпозиума «GEOTECHNIKA - GEOTECNTCS 2004», Польша, г. Устронь*. Устронь, 2004. С. 51–58.
32. Паладійчук Ю. Б., Руткевич В. С., Зінєв М. В., Лісовий І. О. Перспективи використання відкритого програмного комплексу arduino для вивчення технічних дисциплін. *Збірник наукових праць Кіровоградського національного технічного університету. Техніка в сільськогосподарському виробництві, галузеве машинобудування, автоматизація*. 2018. Вип. 31. С. 158–164. URL: http://nbuv.gov.ua/UJRN/znpkntu_2018_31_21 (дата звернення: 12.09.2021).
33. Пашкевич К. Л. Нові технології: 3D принтер. *Легка промисловість*. 2014. № 1. С. 22–25. URL: http://nbuv.gov.ua/UJRN/legpro_2014_1_13 (дата звернення: 12.09.2021).
34. Петраков Ю. В., Писаренко В. В., Мацківський О. С. Напрями розвитку CAM-систем. *Журнал інженерних наук*. 2015. Т. 2, № 2. С. A7–A13. URL: http://nbuv.gov.ua/UJRN/VSU_tekh_2015_2_2_4 (дата звернення: 12.09.2021).
35. Плясунов А. В. 06 одном подходе к решению задач двухуровневого

программирования. *Труды XII Байкальской между нар. конф. «Методы оптимизации и их приложении».* Иркутск, 2001. С. 227–231.

36. Професійні верстати плазмової різки металу з ЧПК. URL: https://marlincnc.com.ua/uk/ (дата звернення: 12.09.2021).
37. Розорінов Г. М., Труш О. В. Дослідження можливостей реалізації оптимального по швидкодії управління кроковим двигуном приводу обертання CD, DVD. *Наукові записки Українського науково-дослідного інституту зв'язку*. 2012. № 3. С. 22–27. URL: http://nbuv.gov.ua/UJRN/Nzundiz_2012_3_6 (дата звернення: 12.09.2021).
38. Сальніков О. В., Мартинюк О. С., Шолом П. С. Технології виготовлення та використання 3D-принтера. *Комп'ютерно-інтегровані технології: освіта, наука, виробництво*. 2016. № 23. С. 37–43. URL: http://nbuv.gov.ua/UJRN/Kitonv_2016_23_8 (дата звернення: 12.09.2021).
39. Сильченко О. Б., Дубинина А. П. Возможности применения метода пластических деформаций в мезообъемах для групповой огранки алмазов в бриллианты. *Горный информационно-аналитический бюллетень*. 2004. №4. С. 275–276.
40. Сильченко О. Б., Дубинина А. П. Критическая технология размерно-регулируемой бездефектной обработки твердоструктурных минералов микро-шлифованием. *Сб. науч. тр. Международной конференции «VI Школа геомеханики», Польша, г. Устронь*. Устронь, 2003. С. 167–180.
41. Сильченко О. Б., Дубинина А. П. Критические технологии обработки сверхтвёрдых материалов. *Горный информационно-аналитический бюллетень*. 2004. №3. С. 139–141.
42. Сильченко О. Б., Дубинина А. П. Особенности алмазно-абразивной обработки минералов в режиме пластического шлифования на станке АН15Ф4. *Драгоценные металлы. Драгоценные камни*. 2004. №6 (126). С. 94–96.
43. Сильченко О. Б., Попов В. Л., Дубинина А. П. Повышение производительности и экологичности производства бриллиантов за счет применения группового метода огранки алмазов. *Материалы конференции «VII SZKOLA GE-OMECHANIKI», Польша, г. Устронь*. Устронь, 2005. С. 493–497.

44. Сильченко С. Б., Дубинина А. П. Особенности групповой алмазноабразивной обработки минералов в режиме пластического шлифования на стайке АН15Ф4. *Труды XI Международного симпозиума «GEOTECHNIKA – GEOTECIINICS 2004», Польша, г. Устронь*. Устронь, 2004. С. 85–89.
45. Софт для выжигания. URL: https://sites.google.com/site/nikromsoft/ribs (дата звернення: 14.09.2021).
46. Способ определения допустимой скорости резания при механической обработке детали твердосплавным инструментом: пат. 2063307 Россия, С1 В23В25/06. № 94010673/08; заявлено 29.03.1994; опубл. 10.07.1996, Бюл. № 19. 6 с.
47. Старков В. К. Обработка резанием. Управление стабильностью и качеством в автоматизированном производстве. Москва : Машиностроение, 1989. 296 с.
48. Фурман І. О., Піскарьов О. М. Особливості створення мікроконтролерної системи визначення параметрів робочих органів сільськогосподарських машин. *Праці Таврійського державного агротехнологічного університету*. 2014. Вип. 14, т. 2. С. 74–77. URL: http://nbuv.gov.ua/UJRN/Ptdau_2014_14_2_13 (дата звернення: 12.09.2021).
49. Automatically send GCode codes to the CNC machines. URL: https://grbl-controller.software.informer.com (дата звернення: 14.09.2021).
50. The next level of CNC control. URL: https://www.machsupport.com (дата звернення: 14.09.2021).