МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«УЖГОРОДСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ»
КАФЕДРА ПРИЛАДОБУДУВАННЯ

ДОСЛІДЖЕННЯ ВУЗЛІВ ТА СИСТЕМ АВТОМАТИЗОВАНОГО КЕРУВАННЯ У РОБОТОТЕХНІЦІ.

Методичні вказівки до лабораторних робіт з дисципліни:
«Монтаж та експлуатація робототехнічних комплексів»
(магістерській рівень)

Ужгород – 2023

УДК 004.056.55, 003.26
Іваницький В.П., Мешко Р.О. Дослідження вузлів та систем автоматизованого керування у робототехніці. Методичні вказівки до лабораторних робіт з дисципліни «Монтаж та експлуатація робототехнічних комплексів». Ужгород: в-во УжНУ, 2023, 40 с.

Укладачі:
Валентин ІВАНИЦЬКИЙ – док. фіз.-мат. наук, професор кафедри приладобудування ДВНЗ «УжНУ»
Роман Мешко – старший викладач кафедри приладобудування ДВНЗ «УжНУ»

Відповідальний за випуск – Михайло РЯБОЩУК., канд. фіз.-мат. наук, доцент кафедри приладобудування ДВНЗ «УжНУ».

Методичні рекомендації розглянуто та схвалено на засіданні кафедри приладобудування, протокол № 9 від 21.06.2023 року.

ЗМІСТ

ВСТУП

Метою вивчення навчальної дисципліни «Монтаж та експлуатація технологічних робототехнічних комплексів» є формування у здобувачів вищої освіти умінь і навичок з монтажу, випробовування, експлуатації мехатронних та робото технічних систем. Крім цього формування загального розуміння структури складових РТК, та вміння аналізувати та розробляти необхідну технічну документацію.

Відповідно до освітньої програми, лабораторні роботи з навчальної дисципліни сприяє формуванню у здобувачів вищої освіти таких компетентностей:

- здатність здійснювати автоматизацію складних технічних й технологічних об’єктів і комплексів;

- здатність проектувати та впроваджувати високонадійні технічні системи автоматизації й робототехнічні комплекси та їх прикладне програмне забезпечення.

- здатність застосовувати сучасне спеціалізоване програмне забезпечення для комп’ютерно-інтегрованих технологій та впровадження робототехніки.

Лабораторна робота №1
Програмне керування двигуном постійного струму

Мета роботи: ознайомитися зі способами керування швидкістю обертів двигунів постійного струму. Навчитися створювати програми керування (на базі МК) швидкістю обертів Двигуна Постійоного Струму (ДПС), застосування драйвера. Освоїти навички керування драйверами із цифровим інтерфейсом. Ознайомитись з основними схемо технічними рішеннями реалізації драйверів двигунів постійного струму.

Теоретичні відомості

У мехатронних системах одним з найпоширеніших виконавчих пристроїв є двигун постійного струму (ДПС). Конструкції ДПС відрізняються великою різноманітністю. У лабораторній роботі вивчається метод керування колекторний ДПС, із застосуванням платформи Arduino Uno. Крім цього є можливість застосувати спеціалізовані «шилди» з готовими силовими драйверами ДПС (див рис. 1.1).

Одним із найпростіших варіантів керування ДПС – є застосування силового драйвера на одному транзисторі, який працює у режимі ключа (ON/OFF - режим), або у режимі ШІМ модуляції. Оба методи мають своє місце у привідній техніці у залежності від методу керування.

ON/OFF – режим. Реалізовує керування обертами ON-100%, OFF – 0%.

ШІМ – режим реалізовує плавне регулювання ДПС з дискретністю 1%, (діапазон 0-100%).

Розглянемо технічну реалізацію ШІМ із застосуванням ресурсів МК.

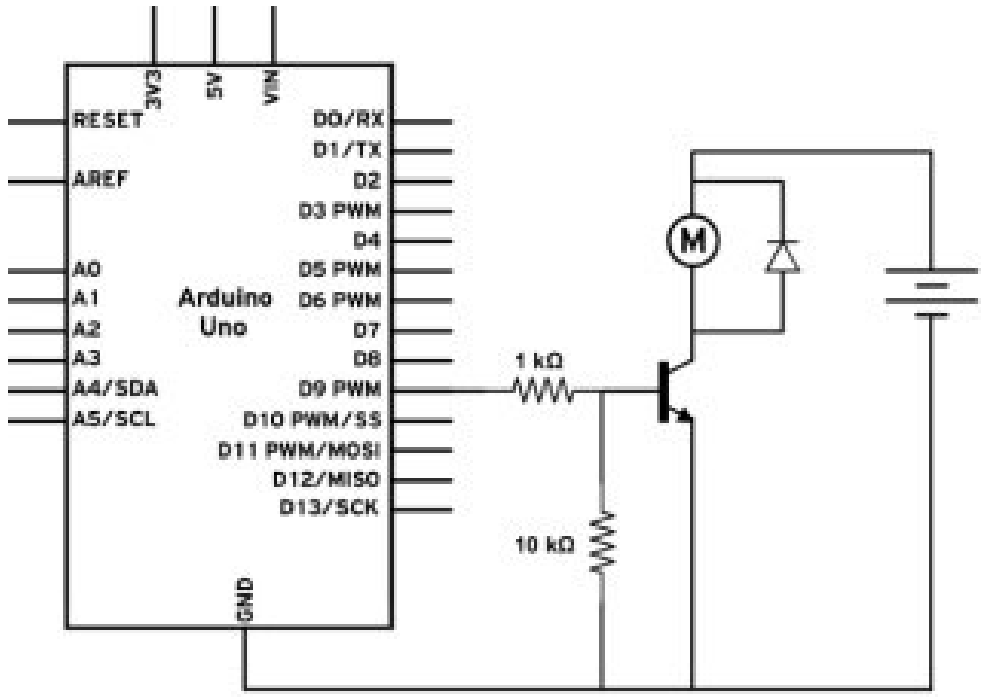

Рисунок 1.1 - Схема включення ДПС за допомогою силового ключа – драйвера.

Широтно-імпульсна модуляція (ШІМ)

Регулювати швидкість обертів ДПС при зміні навантаження на валу можна зміною напруги, що подається на коло якоря.У сучасних мікропроцесорних САР напругу змінюють не безперервно, а дискретно за допомогою широтно-імпульсної модуляції (ШІМ). Принцип ШІМ пояснюється на рис. 1.2. При ШІМ вихідний сигнал являє собою імпульсний сигнал постійної частоти і змінної скважності.

Скважність –це відношення періоду проходження імпульсів Т до їх тривалості. Обернена величина –коефіцієнт заповнення:

$$s = \frac{\tau}{T}.$$

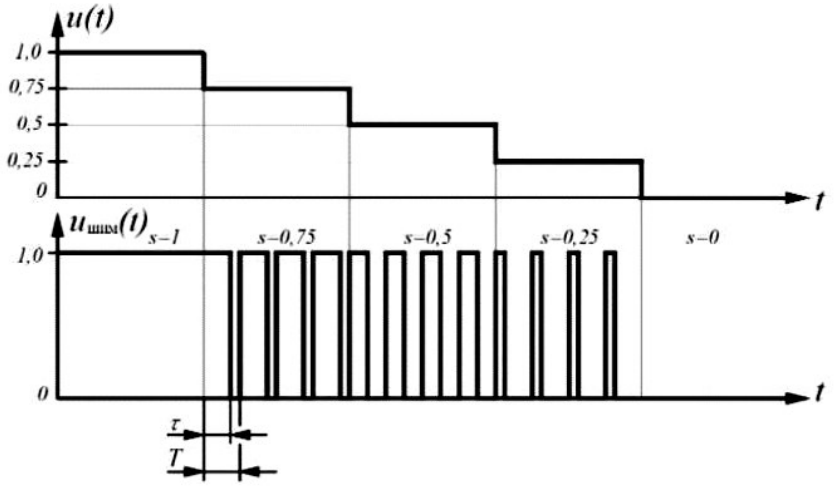

Рисунок 1.2 - Широтно-імпульсна модуляція

Широтно-імпульсна модуляція(ШІМ-англ. pulse-width modulation, PWM), або модуляція за тривалістю імпульсів (англ.pulse-duration modulation, PDM)–процес керування шириною (тривалістю) високочастотнихімпульсівпо закону, який задає низькочастотний сигнал. В електроніці це може бути керування середнім значенням вихідної напругишляхом зміни тривалості замкнутого стану електронного (електромеханічного) ключа.

Аналогова ШІМ.

ШІМ-сигнал генерується аналоговим компаратором, на один вхід якого подається опорний сигнал значно більшої частоти, ніж модулюючий у вигляді «трикутника» або «пилки», а на іншій–модулюючий неперервний аналоговий сигнал. Частота вихідних імпульсів ШІМ відповідає частоті «зубів» пилки. В ту частину періоду, коли сигнал на позитивному вході вище сигналу на негативному вході, на виході виходить одиниця, в іншу, коли сигнал на позитивному вході нижче сигналу на негативному вході-нуль.

Цифрова ШІМ. У двійковій цифровій техніці, виходи в якій можуть приймати тільки одне з двох значень, наближення

бажаного середнього рівня вихідного сигналу за допомогою ШІМ є абсолютно природним. Схема така ж проста: пилкоподібний сигнал генерується N-бітовим лічильником.

Розглянемо кілька прикладів при напрузі живлення Vcc=5В. (рис. 1.3, рис. 1.4).

Рисунок 1.3 - Формування ШІМ сигналу з еквівалентною напругою 2,5VDC

Рисунок 1.4 - Формування ШІМ сигналу з еквівалентною напругою 0,5VDC

Для формуваня ШІМ сигналу певної скважності на дискретний вихід Arduino використовується функція analogWrite().

analogWrite() – формує задану "аналогову" напругу на виводі у вигляді ШІМ-сигналу. Після виклику analogWrite(), на виводі буде безперервно генеруватися ШІМ-сигнал із заданим коефіцієнтом заповнення до наступного виклику функції analogWrite() (або до моменту виклику digitalRead() або digitalWrite(), що взаємодіють з цим же виводом). Частота ШІМ становить приблизно 490 Гц. На платі Uno та подібних виводи 5 і 6 мають частоту ШІМ приблизно 980 Гц.

На більшості плат Arduino (на базі мікроконтролерів ATmega168 або ATmega328) функція analogWrite() працює з виводами 3, 5, 6, 9, 10 і 11. На Arduino Mega функція працює з виводами з 2 по 13 і 44 – 46.

Функція analogWrite() не має нічого спільного з аналоговими входами і функцією analogRead().

Синтаксис:
analogWrite(pin, value)
Параметри:
pin - номер ШІМ-виходу
value - значення скважності від 0 (завжди "0") до 255 (завжди "1").

Найпростіша схема керування двигуном постійного струму складається з польового транзистора, на затвор якого подається ШІМ сигнал (див рис. 1.5). Транзистор в даній схемі виконує роль електронного ключа, який комутує один з виводів двигуна на землю. Транзистор відкривається на час тривалості імпульсу.

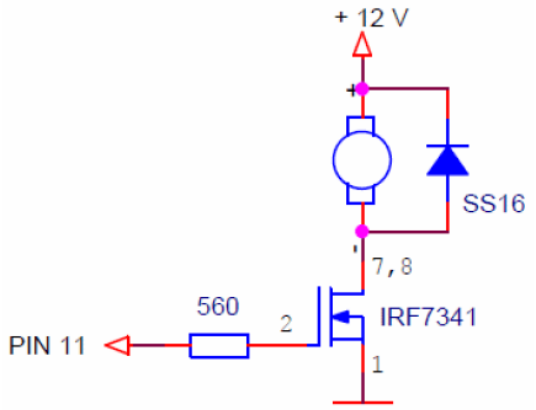

Рисунок 1.5 - Схема керування швидкістю обертів ДПС з використанням польового транзистора

План роботи

1. Провести пошук способів керування швидкістю обертів ДПС.

2. Провести пошук програмних реалізацій керування швидкістю обертів ДПС.

Порядок виконання роботи

1. Ознайомитися з теоретичними відомостями.
2. Створити схему підєднання ДПС, та призначити номер порта.
2. Створити тестову програму у ArduinoIDE.
3. Провести типовий пошук включення драйвера.
4. Розібратись у ключових параметрів програми у AdruinoIDE, прослідкувати логіку залежності параметрів та поведінку конструкції для різних значень ШІМ.

Тестова програма:

```
int led = 9; // ШІМ-вивід, до якого підключений силовий транзистор
int speed= 0; // ця змінна відповідає за швидкість обертання
int fadeAmount = 5; // Крок зміни швидкості
void setup ()
{
pinMode (led, OUTPUT); // Налаштовуємо вивід 9 як дискретний вихід
}
void loop ()
{// Встановлюємо швидкість обертання двигуна
analogWrite(led, speed);// Збільшуємо поточне значення швидкості обертання
speed= speed+ fadeAmount;// Коли швидкість стає максимальною / мінімальною -починаємо її знижувати / підвищувати
if (speed== 0 || speed== 255)
{
fadeAmount = -fadeAmount;
}
// Пауза 30 мілісекунд
```

```
delay (30);
}
```

5. Скомпілювати програму та переконатися у відсутності помилок.

6. Підключити платформа AdruinoUNO до USB-порту комп'ютера. Завантажити програмне забезпечення МК. Провірити на працездатність конструкцію.

7. Самостійно створити програму для задання постійної швидкості обертання ДПС на рівні 95% від верхньої межі діапазону регулювання. Завантажте програму в МК та переконайтеся в правильності її роботи.

8. Оформити звіт про виконання лабораторної роботи. Звіт повинен містити: назву та мету лабораторної роботи; тексти програм з коментарями; висновок про виконання роботи.

Контрольні запитання

1.Яка будова найпростішого колекторного ДПС?

2. Які способи керування швидкістю ДПС Вам відомі?

3. Що таке широтно-імпульсна модуляція?

4. Що таке скважність імпульсів?

5. Як можна змінювати швидкість обертання ДПС за допомогою ШІМ?

Лабораторна робота №2
Інтерфейс керування сервопривода та реалізація позиціонування

Мета роботи: ознайомитися зі способами керування положенням сервоприводів. Навчитися створювати програми керування швидкістю повороту і положенням вихідного вала сервопривода (у нашому випадку даний пристрій ще називають сервомашинкою). Застосовується у якості елемента керування різними моделями.

Теоретичні відомості

Під сервоприводом найчастіше розуміють механізм з електродвигуном, який можна попросити повернутися в заданий кут і утримувати це положення. Однак, це не зовсім повне визначення.Якщо сказати повніше, сервопривід -це привід з управлінням через негативний зворотний зв'язок, що дозволяє точно керувати параметрами руху. Сервоприводом є будь-який тип механічного приводу, що має в складі давач (положення, швидкості, зусилля тощо) і блок керування приводом, який автоматично підтримує необхідні параметри згідно заданому завданню. Тобто сервопривод отримує на вхід значення керуючого параметра, наприклад, кут повороту. Блок управління порівнює це значення зі значенням на своєму датчику. На основі результату порівняння привод виконує певну дію, наприклад: поворот, прискорення або сповільнення так, щоб значення з внутрішнього сенсора стало якомога ближче до значення зовнішнього керуючого параметра.

Структурна схема (див рис.2.1.) складається з генератора опорного імпульсу (ГОП), до якого підключений потенціометр

зворотного зв'язку компаратора (К), пристрої вибірки-зберігання (УВХ) і силового моста, в діагональ якого включений електромотор (М). (бази транзисторів-ключів об'єднані умовно).

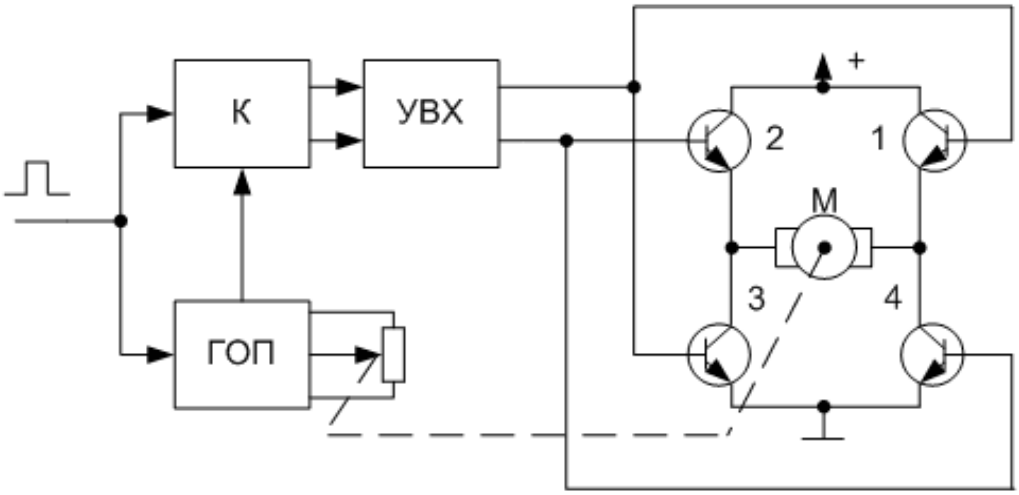

Рисунок 2.1 - Структурна схема сервопривода з датчиком положення та швидкості

Керуючий імпульс приймача приходить на компаратор і одночасно запускає генератор опорного імпульсу. Тривалість опорного імпульсу залежить від положення зворотного потенціометра зв'язку, механічно з'єднаного з вихідним валом. У середньому положенні вихідного вала, тривалість дорівнює 1,5 мс, крайніх положеннях - 0,8 і 2,2 мс відповідно. Керуючий та опорний імпульси порівнюються компаратором за тривалістю. Різнисний імпульс утворюється на верхньому, або нижньому виходах компаратора, залежно від цього, який із порівнюваних імпульсів довше. Довжина різницевого імпульсу визначає величину неузгодженості між "необхідним" і "існуючим" положенням керма моделі. Ця величина вимірюється і запам'ятовується у вигляді постійного потенціалу на час циклу управляючого імпульсу, у пристрої вибірки-зберігання. (Тут також дано спрощення роботи ПВЗ.

Насправді постійний потенціал відкриває ключі лише при великому різницевому імпульсі, а при малих його значеннях ключами управляє пропорційно подовжений різницевий імпульс). Виходи останнього керують ключами силового моста.

Найбільш поширені сервоприводи, які утримують заданий кут і сервоприводи, що підтримують задану швидкість обертання.

Тремтіння (Jitter) і "мертва зона" сервопривода

Щоб розглянути більш детальні характеристики сервомашини розглянемо її роботу в другому наближенні. Для цього звернемося до графіка переорієнтації (рис. 2.2):

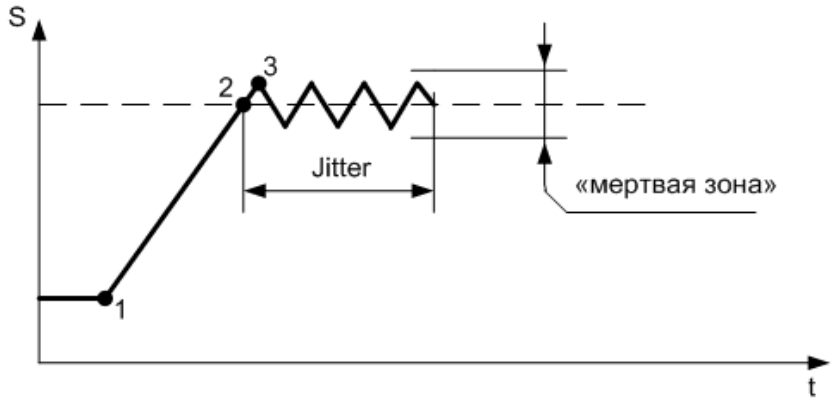

Рисунок 2.2 - Графік швидкості позиціонування / утримання сервопривода.

На графіку (рис. 2.2), швидкість переміщення S, t – час позиціонування. У момент 1 приходить команда на переорієнтацію сервомашинки починає її відпрацювання. У момент 2 тривалості керуючого та опорного імпульсів у керуючій електроніці сервомашинки порівнюються. Але механіка має властивість інерції та проскакує задане значення. У точці 3 електроніка, що управляє, відпрацьовує мотором назад і все повторюється. В результаті спостерігається тремтіння вала біля положення, що вимагає

керуючого імпульсу. Це тремтіння отримало назву jitter. Для боротьби з ним в електроніку запроваджується зона нечутливості до точності позиціонування. Технічно це виглядає так: пристрій вибірки-зберігання, оцінюючи тривалість різницевих імпульсів після компаратора, сприймає їх малі значення як нулі. Тобто вводиться інтервал, в межах якого зміна керуючого імпульсу відносно опорного не включає двигун сервомашинки. Цю зону називають ще "мертвою зоною". Її величина визначає точність позиціонування вихідного вала сервомашинки відносно керуючого сигналу.

Занадто велика "мертва зона" – негативно впливає з точки зору точності керування. Занадто мала - з'явиться тремтіння / вібрація виконавчого елемента, що також погіршує точність відпрацювання команди і різко підвищує енергоспоживання сервомашини. Її допустима величина визначається точністю виконання механіки редуктора сервомашинки та досконалістю керуючої електроніки. Причиною тремтіння справної сервомашинки може бути вихід напруги живлення за межі, що допускаються виробником. Це є наслідком того, що силові та швидкісні параметри сервомашини залежать від напруги її живлення. Виробник оптимізує параметри керуючої електроніки під заданий інтервал, знаходячи компроміс між точністю відпрацювання та відсутністю тремтіння. За його межами, "мертва зона" може зрости – не критично, а може і зменшитися, – з'явиться тремтіння.

Отже, сервопривод – це регульований редукторний електродвигун. Він зазвичай складається з приводного механізму з двигуном постійного струму, плати управління і потенціометра (сенсора положення), котрий забезпечує зворотний зв'язок (див рис. 2.3).

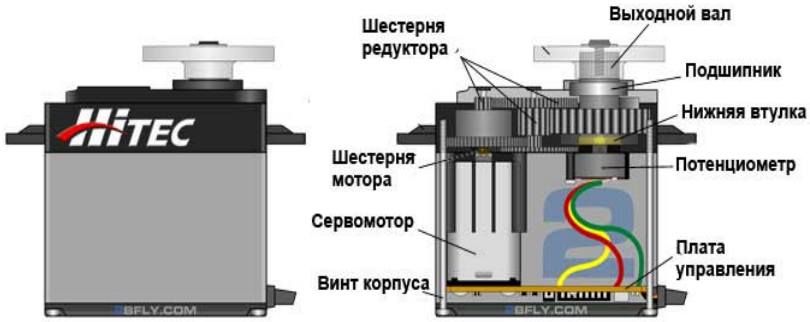

Рисунок 2.3 - Будова сервопривода у розрізі

Управління сервоприводом. Інтерфейс керуючих сигналів

Керуючий сигнал для сервопривода – імпульси постійної частоти і змінної ширини. Те, яке положення повинен зайняти сервопривод, залежить від довжини імпульсів. Коли сигнал надходить в керуючу схему, наявний у ній генератор імпульсів виробляє свій імпульс, тривалість якого визначається через потенціометр. Інша частина схеми порівнює тривалість двох імпульсів. Якщо тривалість різна, включається електродвигун. Напрямок обертання визначається тим, який з імпульсів коротший. Якщо довжини імпульсів рівні, електродвигун зупиняється. Найчастіше в аматорських сервоприводів імпульси виробляються з частотою 50 Гц. Це означає, що імпульс випускається і приймається раз в 20 мс. Зазвичай при цьому тривалість імпульсу 1520 мкс означає, що сервопривод повинен зайняти середнє положення. Збільшення або зменшення довжини імпульсу змусить сервопривод повернутися за годинниковою або проти годинникової стрілки відповідно (рис. 2.4). При цьому існують верхня і нижня межі тривалості імпульсу.

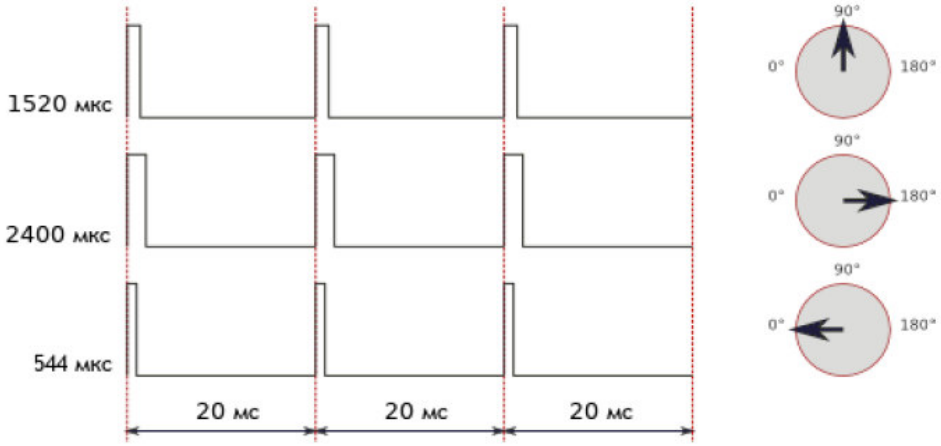

Рисунок 2.4 - Залежність кута повороту сервопривода від трвалості керуючих імпульсів

У бібліотеці Servo для Arduino за замовчуванням виставлені наступні значення довжин імпульсу: 544 мкс –для 0° і 2400 мкс –для 180°.На конкретному пристрої заводські налаштування можуть відрізнятися від стандартних. Також варто відзначити, що це всього лише загальноприйняті довжини.

Навіть у рамках однієї і тієї ж моделі сервоприводу може існувати похибка, що допускається при виробництві, яка призводить до того, що робочий діапазон довжин імпульсів трохи відрізняється. Для точної роботи кожен конкретний сервопривод повинен бути відкалібрований: шляхом експериментів необхідно підібрати коректний діапазон, характерний саме для нього. При формуванні сигналу керування для сервопривода важливою є довжина імпульсів, а не частота їх появи. 50 Гц –це норма, але сервопривод буде працювати коректно і при 40, і при 60 Гц. При цьому слід мати на увазі, що при сильному зменшенні частоти він може працювати ривками і на зниженій потужності, а при сильному завищенні частоти може перегрітися і вийти з ладу.

Сервоприводи малої потужності (наприклад, SG90) можна безпосередньо підключати до плати Arduino, для

підключення більш потужних –слід використовувати зовнішнє джерело живлення (див рис.2.5).

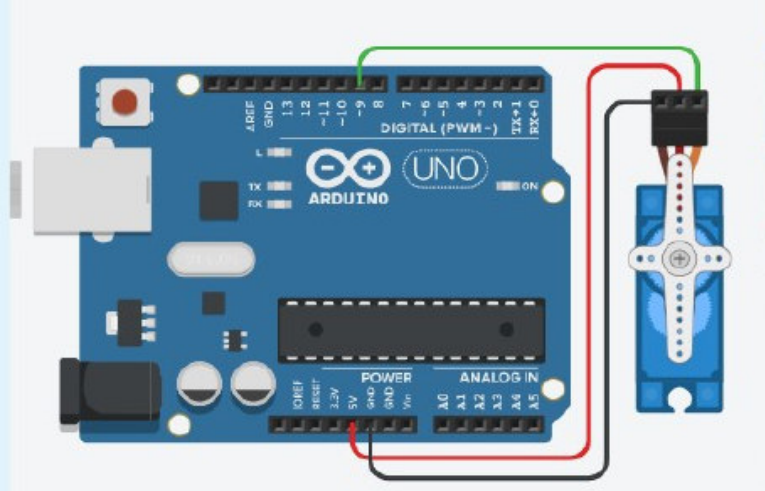

Рисунок 2.5 - Підключення сервопривода малої потужності до платформи ArduinoUno

Бібліотека Servodозволяє здійснювати програмне керування сервоприводами. Управління здійснюється наступними функціями:

attach() –приєднує об’єктдо конкретного виводу плати. Можливі два варіанти синтаксису для цієї функції: servo.attach(pin) і servo.attach(pin, min, max). При цьому pin-номер виводу, до якого приєднують сервопривод, mini max-довжини імпульсів в мікросекундах, що відповідають за кути повороту 0° і 180°. За замовчуванням виставляються рівними 544 мкс і 2400 мкс відповідно.

write() -віддає команду сервоприводу прийняти деяке значення параметра. Синтаксис наступний: servo.write(angle), де angle-кут, на який повинен повернутися сервопривод.

writeMicroseconds() -віддає команду надіслати на сервопрівід імпульс певної довжини, є низькорівневим аналогом попередньої команди. Синтаксис наступний: servo.writeMicroseconds(uS), де uS–довжина імпульсу в мікросекундах.

read() - читає поточне значення кута, в якому знаходиться сервопривод. Синтаксис наступний: servo.read(), повертається ціле значення від 0 до 180.

attached() - перевірка, чи була приєднаний об’єкт до конкретного виводу. Синтаксис наступний: servo.attached(), повертається true, якщо змінна була приєднана до якого-небудь виводу, або false в зворотному випадку.

detach() - виконує дію, зворотну дії attach(), тобто від'єднує об’єкт від виводу, з яким він був асоційований. Синтаксис наступний: servo.detach().

Приклад програми з використанням бібліотеки Servo

```
#include<Servo.h>
Servo myservo;
void setup()
{
myservo.attach(9);
}
voidloop()
{
myservo.write(90);// встановлюємо сервопривод в середнє положення
delay(500);
myservo.write(0); // встановлюємо сервопривод в крайнє ліве положення
delay(500);
myservo.write(180);// встановлюємо сервопривод в крайнє праве положення
delay(500);
}
```

План роботи

1.Скласти програму для керування положенням сервопривода.

2. Реалізувати циклограму положення вала сервопривода.

3. Програмно обмежити швидкість повороту вала сервопривода.

Порядок виконання роботи

1. Складіть схему за рис. 2.5, використовуючи сервопривод SG90.

2. Завантажте програму з прикладу бібліотеки Servo.

Напишіть програму для реалізації наступної циклограми (рис.2.6).

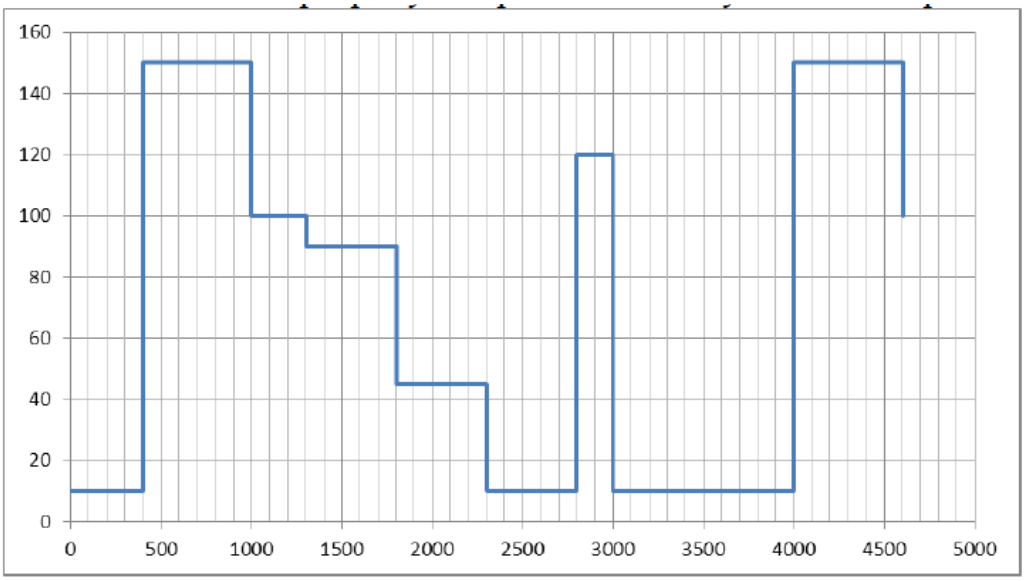

Рисунок 2.6 - Задана циклограма роботи

Вісь абсцис – час в мілісекундах, вісь ординат – положення вала сервопривода в градусах.

3. Для того, щоб мати змогу змінювати швидкість обертання використайте бібліотеку VarSpeedServo. Завантажити в МК приклад VarSpeedServo-Sweep, змінюючи значення швидкості. Переконатися в правильності роботи програми та точності позиціювання сервопривода.

4. Оформити звіт про виконання лабораторної роботи. Звіт повинен містити: назву та мету лабораторної роботи; тексти програм з коментарями; висновок про виконання роботи.

Контрольні запитання

1. Що таке сервопривод?
2. Назвіть основні елементи сервопривода.
3. Як реалізоване керування положенням вала сервопривода, що використано в цій лабораторній роботі?
4. Яка бібліотека дає змогу обмежувати швидкість повороту сервопривода?

Лабораторна робота №3
Керування маніпулятором з цифровим управлінням

Мета роботи: Ознайомитися з принципом управління роботом-маніпулятором з 6-ма ступенями свободи. Навчитися створювати програми керування для дистанційного управління маніпулятором.

Теоретичні відомості

У процесі автоматизації промислових підприємств важливим є використання роботизованих комплексів, що складаються з механічних маніпуляторів та систем управління ними. Застосування промислових роботів-маніпуляторів дозволяє виключити вплив людського фактора на виробництві, підвищити точність виконання технологічних операцій, певною мірою зменшити вплив шкідливих факторів на персонал, скоротити площу виробничих приміщень і забезпечити безперебійну роботу виробництва.

Промисловий робот–автоматична машина, що складається з маніпулятора і пристрою програмного керування його рухом, призначений для заміни людини при виконанні основних і допоміжних операцій у виробничих процесах.

Маніпулятор – сукупність просторового важільного механізму і системи приводів, що здійснює під керуванням програмованого автоматичного пристрою чилюдини-оператора дії (маніпуляції), аналогічні діям руки людини.

Промислові роботи призначені для заміни людини при виконанні основних і допоміжних технологічних операцій у процесі промислового виробництва. Гнучкі автоматизовані виробництва, які створюються на базі промислових роботів,

дозволяють вирішувати задачі автоматизації на підприємствах із широкою номенклатурою продукції при дрібносерійному і штучному виробництві.

Необхідність дослідження та вдосконалення систем управління маніпуляційними роботами на сам перед з умовлена їх широким застосуванням. Подібні пристрої використовуються в будівельній галузі (крани-маніпулятори), металургії (прокатні стани, кувальні маніпулятори), гірничодобувній промисловості (бурильні машини), хімічній промисловості (маніпулятори для роботи з токсичними і радіоактивними матеріалами), суднобудівній та авіаційній галузях (зварювальні та складальні маніпулятори, маніпулятори для різання металів) тощо.

Маніпулятор промислового робота повинен забезпечувати рух вихідної ланки і, закріпленогов ній, об'єкта маніпулювання в просторі за заданою траєкторією і з заданою орієнтацією. Промисловий робот із шістьма ступенями свободи є складною автоматичною системою. У реальних конструкціях промислових роботів часто використовуються також механізми зі ступенями свободи менше шести. Найбільш прості маніпулятори мають три, рідше два, рухи. Такі маніпулятори значно дешевші у виготовленні та експлуатації, але вимагають специфічних вимог до організації робочого простору.

Розглянемо для прикладу структурну і функціональну схеми промислового робота з трьома рухомими ланками. Основний механізм руки маніпулятора складається з нерухомої ланки 0 і трьох рухомих ланок 1, 2 і 3 (рис. 3.1). Механізм цього маніпулятора відповідає циліндричній системі координат. У цій системі ланка 1 може обертатися відносно ланки 0 (відносне кутове переміщення ф10), ланка 2 переміщається по вертикалі відносно ланки 1 (відносне

лінійне переміщення S21) і ланка 3 переміщається в горизонтальній площині відносно ланки 2 (відносне лінійне переміщення S32). На кінці ланки 3 закріплений захоплюючий пристрій (захват), призначений для захоплення й утримання об'єкта маніпулювання. Ланки основного важільного механізму маніпулятора утворюють між собою три кінематичні пари (одну обертальну А і дві поступальні В та С) і можуть забезпечити переміщення об'єкта в просторі без керування його орієнтацією.

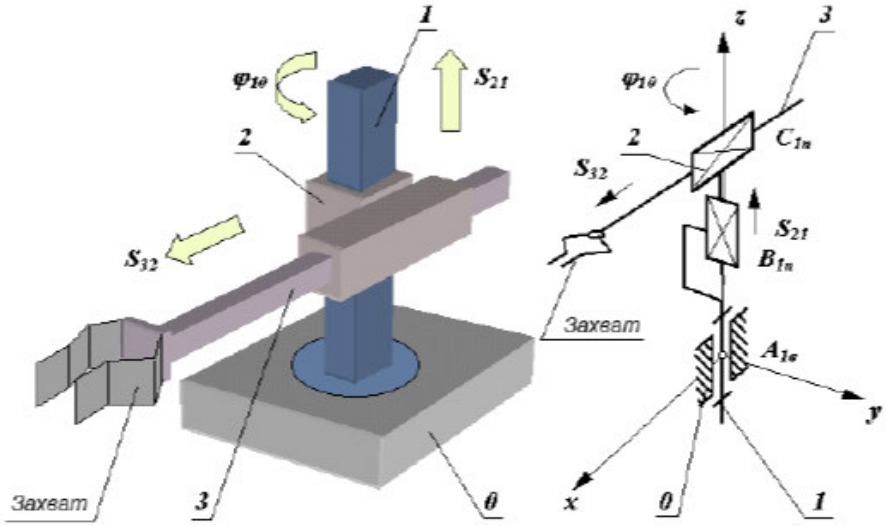

Рисунок 3.1 - Структурна і функціональна схеми промислового маніпулятора

Для виконання кожного з трьох відносних рухів маніпулятор повинен бути оснащений приводами, що складаються з двигуна, редуктора і системи давачів зворотного зв’язку. Оскільки рух об'єкта здійснюється за заданим законом руху, то в системі повинні бути пристрої, що зберігають і задають програму руху. Перетворення заданої програми руху в сигнали керування приводами *u* іздійснюється системою керування. При необхідності вона коректує ці впливи за сигналами Δ*x*і, що надходять до неї з давачів зворотного зв'язку (рис. 3.2).

Отже, маніпуляційний робот складається з декількох ступенів рухливості (ланок) і приводів, що призводять ланки в рух. У якості приводів робота найчастіше використовуються крокові двигуни або сервоприводи різної потужності. Кроковим двигунам надається перевага, якщо швидкість переміщення ланок робота не є критичним параметром. Наприклад, такий тип приводів може використовуватися при побудові вантажних маніпуляторів. Якщо ж потрібно забезпечити високу швидкість руху робота, то найбільш доцільно використовувати сервоприводи.

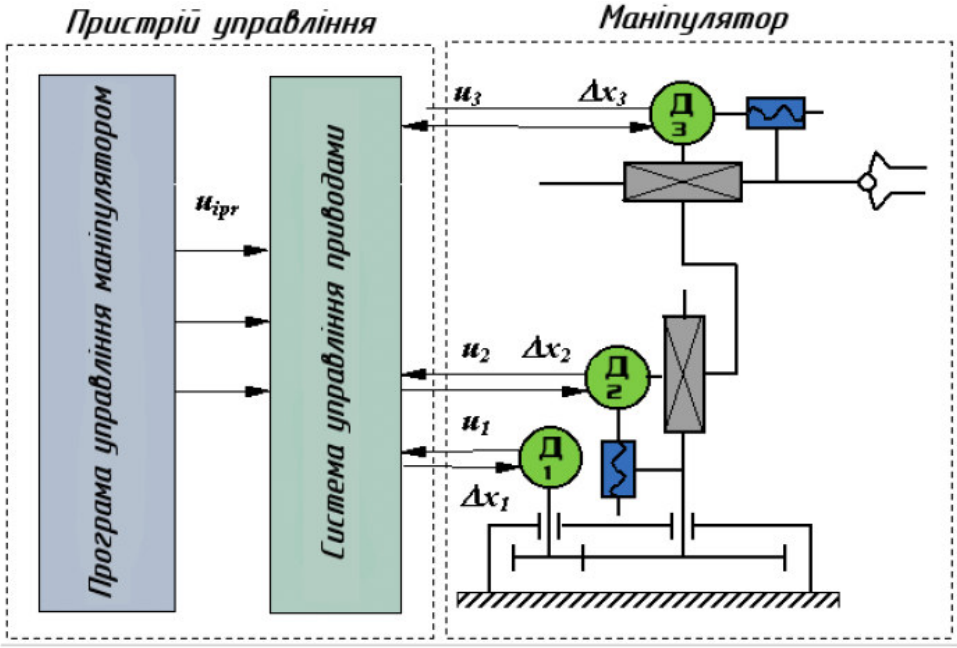

Рисунок 3.2 - Функціональна схема промислового робота

Процес розробки робота-маніпулятора складається з двох етапів:

-розробка механічної частини робота (вибір матеріалу для виготовлення складових частин, а також типу виконавчих механізмів);

-розробка системи управління маніпулятором (вибір контролера, вибір засобів програмування, розробка алгоритмів керування).

Швидкодію промислових роботів визначають за максимальною швидкістю лінійних переміщень центра захвату маніпулятора. Розрізняють промислові роботи з малою (VM<0,5м/с), середньою (0,5<VM<1,0м/с) і високою (VM>1,0м/с) швидкодією. Сучасні промислові роботи мають в основному середню і високу швидкодію.

Точність маніпулятора промислового робота характеризується абсолютною лінійною похибкою позиціювання центра захвату. Промислові роботи поділяються на групи з малою (Δ rM<1 мм), середньою (0,1 мм< Δ rM<1 мм) і високою (Δ rM<0,1 мм) точністю позиціювання.

Опис лабораторної установки

Досліджуваний маніпулятор, зображений на рис. 3.3, має 6 ступенів свободи. Для приведення в рух ланок маніпулятора в кожному із суглобів встановлено сервопривод MG996R. Перший ступінь рухомості забезпечує основа маніпулятора, другий –плече, третій –лікоть, четвертий –обертання кисті, п’ятий –поворот кисті, шостий –захват.

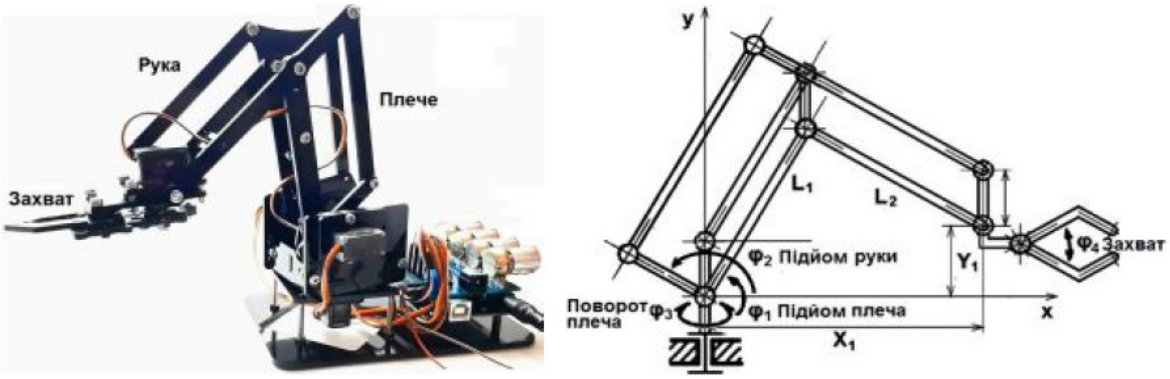

Рисунок 3.3 - Зовнішній вигляд досліджуваного маніпулятора

Параметри сервоприводів MG996Rподані нижче (рис. 3.4).

Рисунок 3.4 - Зовнішній вигляд та розміри сервопривода MG996R

Підключення сервопривода MG996R, та типова характеристика управління (рис 3.5).

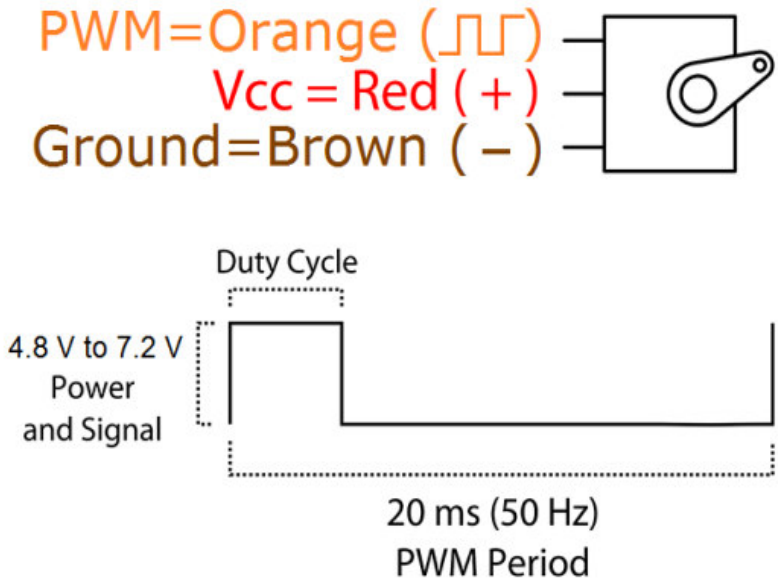

Рисунок 3.5 - Підключення сервопривода MG996R, та типова характеристика управління.

Основні характеристики:

Розміри: 40.7 мм x 19,7 мм x 42.9 мм.

Маса: 55 г

Робоча швидкість: 0.17с / 60 градусів (4.8В без навант.)Робоча швидкість: 0.14с/ 60 градусів (6.0В без навант.)

Обертальний момент: 9.4 кгс/см (4.8В), 11 кгс/см (6 В)

Робоча напруга: 4.8 –7.

Робочий струм при русі: 500 –900 мА (6 В).

Струм утримання(момент навантаження дорівнює крутному моменту двигуна): 2.5 А (6 В).

Часова діаграма сигналів управління та позиціонування сервопривода (рис. 3.6).

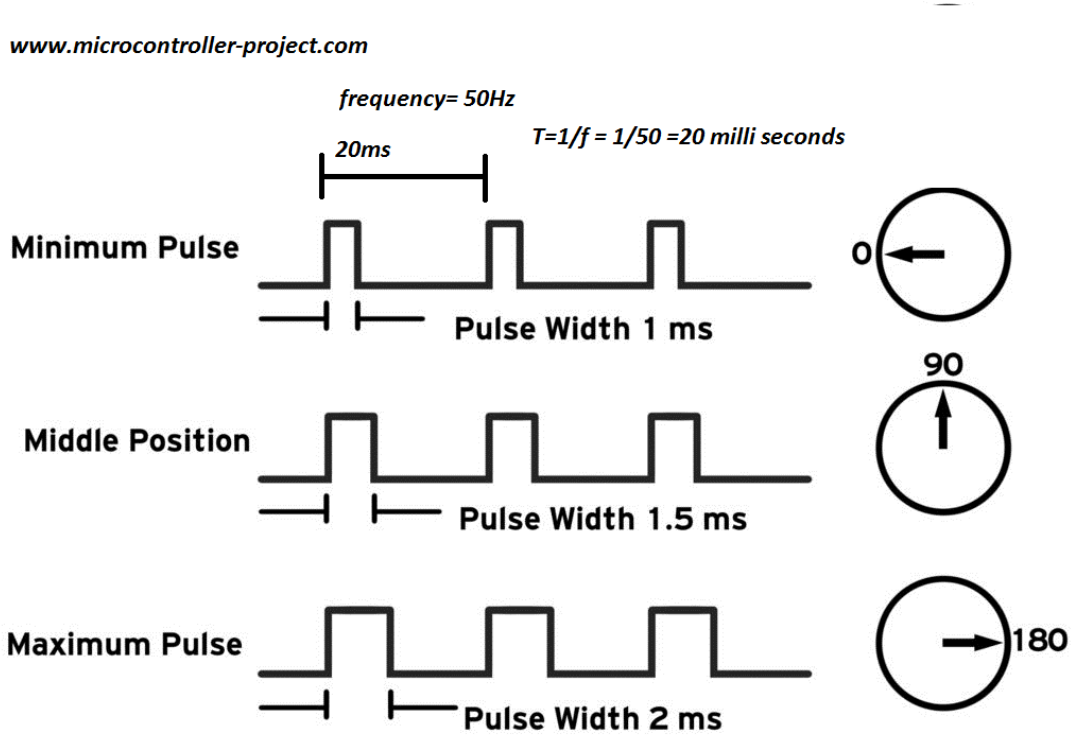

Рисунок 3.6 - Часова діаграма сигналів управління та позиціонування сервопривода [].

Для живлення сервоприводів у лабораторній роботі використовується блок живлення 5В, 1А. Живлення використовується від USB порту ПК.

Система управління роботом- маніпулятором реалізована на основі мікроконтролерної плати Arduino Uno.

План роботи

1. Ознайомитися з будовою та принципами управління роботами-маніпуляторами.
2. Ознайомитися з будовою лабораторної установки.
3. Визначити межі безпечних кутів повороту (робочу зону) маніпулятора.

Порядок виконання роботи

1. Ознайомитися з теоретичними відомостями.
2. Завантажити середовище ArduinoIDE.
3. Створити нову програму.

```
#include <VarSpeedServo.h>
VarSpeedServo myservo1, myservo2, myservo3, myservo4,
myservo5, myservo6;  // create servo object to control a servo
byte a1,a2,a3,a4,a5,a6; //angle of Servo
byte oa1,oa2,oa3,oa4,oa5,oa6; //Old angle of Servo
boolean serialflag = 0;
void setup()
{
// initialize serial:
Serial.begin(9600);
myservo1.attach(2,900,2365);
myservo2.attach(3,720,1000);
myservo3.attach(4,720,2365);
myservo4.attach(5,720,2300);
myservo5.attach(10,670,2300);
myservo6.attach(11,720,2300);
oa1 = myservo1.read();
oa2 = myservo2.read();
oa3 = myservo3.read();
oa4 = myservo4.read();
```

```
oa5 = myservo5.read();
oa6 = myservo6.read();
}
void loop()
{
if (a1 != oa1){setPosServo(myservo1,a1);oa1 = a1;printserial(); }

if (a2 != oa2){setPosServo(myservo2,a2);oa2 = a2;printserial(); }
if (a3 != oa3){setPosServo(myservo3,a3);oa3 = a3;printserial(); }
if (a4 != oa4){setPosServo(myservo4,a4);oa4 = a4;printserial(); }
if (a5 != oa5){setPosServo(myservo5,a5);oa5 = a5;printserial(); }
if (a6 != oa6){setPosServo(myservo6,a6);oa6 = a6;printserial(); }
if (serialflag == 1) {printserial(); serialflag = 0;}
//printserial(); delay(200);
}
void setPosServo(VarSpeedServo myservo,int angle){
int i = 1;
int oldangle = myservo.read();
if (oldangle > angle) {i=-1;}
for (int a = oldangle; a!= angle; a += i ){
myservo.slowmove(a,15);     // tell servo to go to position in
variable 'pos'
delay(15);
}
}
void serialEvent() {
byte myservo;
while (Serial.available()) {
// get the new byte:
byte inByte = (byte)Serial.read();
if ((inByte >= 201) && (inByte<=206)  ) {
myservo = inByte -200;
```

```
}
if (inByte <= 180) {
if (myservo == 1){a1 = inByte;}
if (myservo == 2) {a2 = inByte;}
if (myservo == 3) {a3 = inByte;}
if (myservo == 4) {a4 = inByte;}
if (myservo == 5) {a5 = inByte;}
if (myservo == 6) {a6 = inByte;}
serialflag = 1;

myservo = 0;
}
}
}
void printserial() {
delay(20);
Serial.print(myservo1.read()); //  Movement Data
Serial.print(",");
Serial.print(myservo2.read()); //  Movement Data
Serial.print(",");
Serial.print(myservo3.read()); //  Movement Data
Serial.print(",");
Serial.print(myservo4.read()); //  Movement Data
Serial.print(",");
Serial.print(myservo5.read()); //  Movement Data
Serial.print(",");
Serial.print(myservo6.read()); //  Movement Data
Serial.println();
delay(20);
}
```

4. Скомпілювати програму та переконатися у відсутності помилок.

5. Підключити мікроконтролерну плату до USB-портукомп'ютера.

6. У середовищі ArduinoIDE вибрати тип плати та порт, до якого вона підключена. Завантажити створену програму в мікроконтролер.

7. Запустити утиліту RoboArm Control 2.

8. З дозволу викладача ввімкнути живлення стенда.

9. Плавно переміщуючи повзунки смуг прокрутки, спостерігати за переміщенням маніпулятора.

10. Встановити межі кута повороту кожного сервопривода. Результати записати в табл. 3.1.

Таблиця 3.1. Результати

№з/п	Сервопривод	Нижня межа кута повороту	Верхня межа кута повороту
1	База		
2	Плече		
3	Лікоть		
4	Зап'ястя		
5	Поворот		
6	Захват		

11. Вимкніть живлення стенда.

12. Від'єднайте плату Arduino від комп'ютера.

13. Зробити висновки. Звіт повинен містити: титульний лист; тему, мету роботи; порядок виконання; створені програми; заповнену таблицю 3.1; висновки.

Лабораторна робот 4
Дослідження електропривода на базі крокового двигуна.

Мета роботи: Ознайомитись базовими методами керування кроковими двигунами типу NEMA за допомогою драйвера крокового двигуна TB6600 (або A4988), освоїти базові методи побудови інтерфейса керуваня електроприводом.

Теоретичні відомості

Драйвер крокового двигуна TB6600

Arduino Uno - це базова і найпопулярніша версія платформи на базі мікроконтролера ATmega324. Вона володіє достатньою потужністю практично для будь-яких проектів. З нею дуже зручно працювати завдяки тому, що піни розпаяні однорядними коннекторами типу «мама». Зазвичай цю плату використовують для прототипування проектів, а збирають готовий пристрій на базі більш дрібних плат Ардуіно, таких як Arduino Nano і Arduino pro. Це легко зробити так як прошивки сумісні і в більшості випадків номера пинов не відрізняються. Для Arduino Uno існує безліч плат розширення (Шілд), таких як Ethernet shield, motor shield, servo shield і інші. Цей мікроконтролер забезпечений чіпом ATmega328 з тактовою частотою 16 МГц. Так само на платі розташовані: порт USB, роз'єм живлення, кварцовий резонатор, стабілізатори напруги на 5 вольт і на 3.3 вольта, світлодіоди і кнопка перезавантаження.

TB6600 - це простий у використанні професійний драйвер крокової двигуна, який може управляти двофазним кроковим двигуном. Він сумісний з Arduino та іншими микроконтроллерами, які можуть видавати цифровий

імпульсний сигнал 5 В. TB6600 з харчуванням 9 - 40 В постійної напруги призначений для використання з двигунами типу NEMA42 - NEMA86 з максимальним струмом фази до 4А. Піковий струм в 4А достатній для більшості крокових двигунів. Драйвер підтримує управління швидкістю і напрямом руху. Ви можете встановити його мікрокрок і вихідний струм за допомогою перемикача. Існує 7 видів мікрокроків (1, 2 / A, 2 / B, 4, 8, 16, 32) і 8 видів контролю струму (від 0.5 до 4 А) (рис. 4.1).

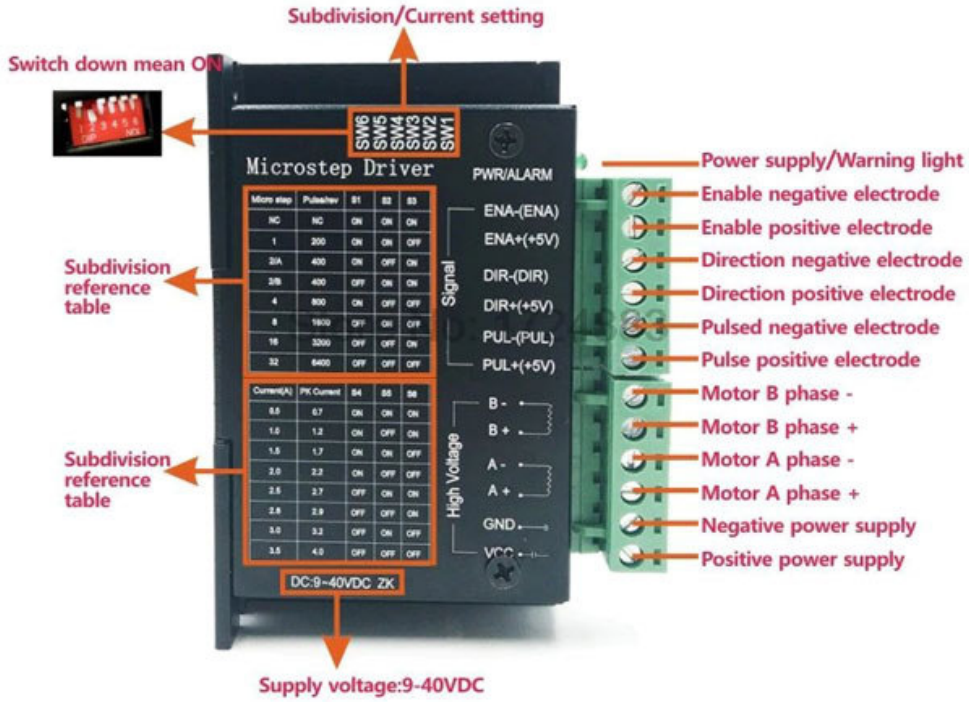

Рисунок 4.1 - Крокový двигун NEMA і драйвер TB6600

Драйвер крокового двигуна A4988

Модуль розроблено на базі чіпі A4988 (рис. 4.2). Не дивлячись на свій малий розмір (всього 0,8” × 0,6”), але має чудові технічні характеристики.

Драйвер крокового двигуна A4988 має високу вихідну потужність (до 35 В , до 2 А) і дозволяє керувати одним біполярним крокових двигуном з вихідним струмом до 2 А на фазу, наприклад NEMA 17.

Для зручності роботи драйвер має вбудований інтерфейс (Step/Dir). Використання даного інтерфейса дозволило зменшити кількість керуючих контактів до 2, один для управління кроками, а інший для управління напрямом обертання.

Драйвер пропонує 5 різних дозволів кроку, а саме:

- Повний крок
- 1/2 крока
- 1/4 крока
- 1/8 крока
- 1/16 крока

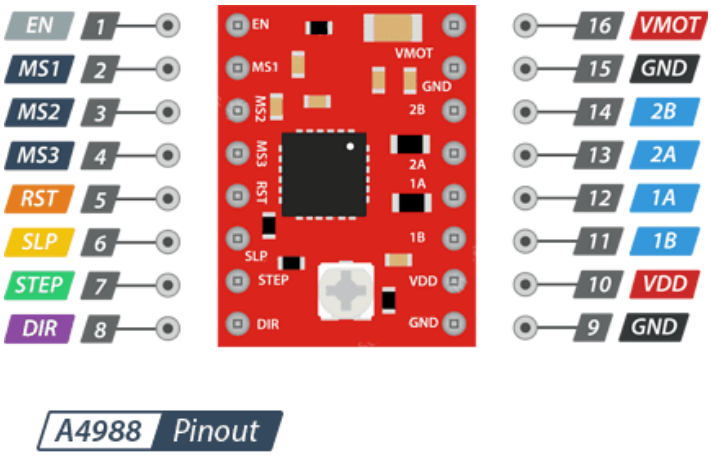

Рисунок 4.2 - Призначення виводів драйвер крокового двигуна A4988

Драйвер A4988 має два управляючі входи, а саме: STEP і DIR.

STEP - керує мікрокроком двигуна. Кожен імпульс, що надсилається цей вивід, приводить двигун у дію кількістю заданих мікрокроків, за допомогою виводів Microstep Selection (MS1, MS2 і MS3). Чим швидше імпульси, тим швидше обертатиметься двигун.

DIR - керує напрямком обертання двигуна. Якщо на нього подати високий рівень, двигун буде обертатися за годинниковою стрілкою, а якщо низький - проти годинникової стрілки.

Робота з компонентами

В даному прикладі підключимо кроковий двигун 17HS4401 до драйверу TB6600 (рис. 4.3) (A4988) (рис. 4.4). В результаті вал двигуна буде здійснювати цикл в 200 кроків (у повно кроковому режимі) в одну сторону, потім у зворотній (рис. 4.3).

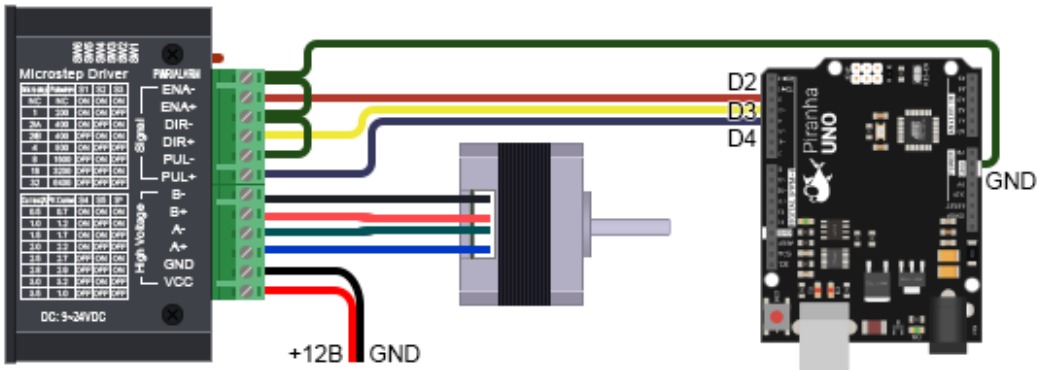

Рисунок 4.3 - Підключення драйверу TB6600 до плати Arduino UNO

Запуск апаратної платформи

Підключаємо драйвер TB6600 (або A4988) до заданих в скетчі пінів, також під'єднуємо кроковий двигун до драйверуВаріант драйвера використовується згідно індивідуального завдання. Після підключаємо плату до ПК та за допомогою програмного застосунку Arduino IDE створюємо та завантажуємо скетч до плати Arduino Uno.

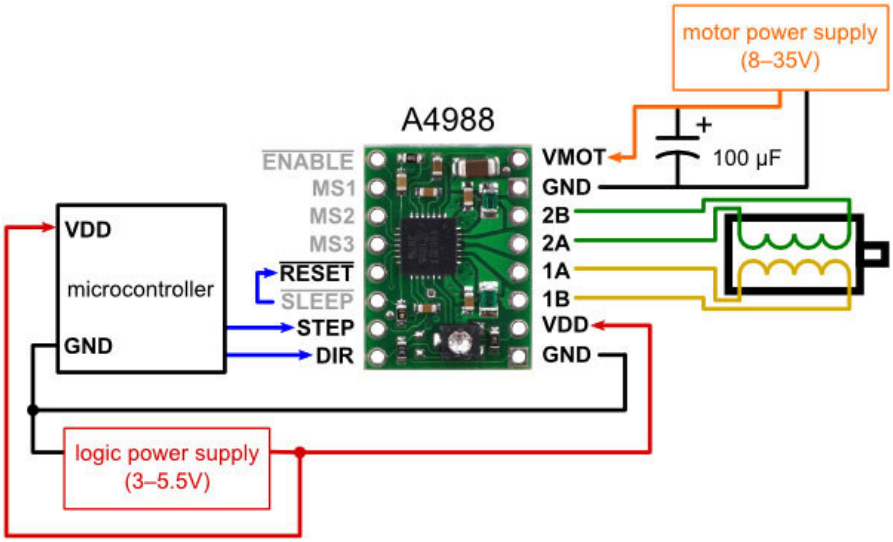

Рисунок 4.4 - Підключення драйверу A4988 до плати Arduino UNO

Заготовка тестової програми:

```
const int dirPin = 2; // налаштування виводів керування
const int stepPin = 3;
const int stepsPerRevolution = 200; //калібрування (число імпульсів на оберт, повний крок
// 1/8Градуса)
    void setup()
{
  pinMode(stepPin, OUTPUT); // налаштування портів МК як виходи
 pinMode(dirPin, OUTPUT);
}
void loop()
{
  digitalWrite(dirPin, HIGH); // налаштування обертання двигуна по часовій стрілці
  for(int x = 0; x < stepsPerRevolution; x++); // повільне обертання двигуна
 {
  digitalWrite(stepPin, HIGH);
```

```
   delayMicroseconds(2000); //часова затримка
   digitalWrite(stepPin, LOW);
   delayMicroseconds(2000);
  }
  delay(1000); // затримка 1сек
     digitalWrite(dirPin, LOW); // налаштування обертання
двигуна проти часової стрілки
   for(int x = 0; x < stepsPerRevolution; x++)// швидке обертання
двигуна
  {
   digitalWrite(stepPin, HIGH);
   delayMicroseconds(1000);
   digitalWrite(stepPin, LOW);
   delayMicroseconds(1000);
  }
  delay(1000); // затримка 1сек.
}
```

За даним скетчем, двигун виконав 200 кроків в одну сторону, після чого таку саму кількість кроків в зворотню, налаштування привода без застосування режиму дроблення кроку.

Порядок виконання роботи

1. Відкрити середовище програмування Arduino IDE.
2. Створити новий проєкт, обрати плату Arduino.
3. Записати базовий скетч наведений у пункті 4.3. Скомпілювати скетч, перевірити помилки та виправити у разі наявності.
4. Розробити персональний скетч керування кроковим двигуном (змінити кількість кроків, напрямки руху, додати паузи, тощо).
5. Оформити звіт з індивідуальним завданням.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ ТА РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. Навчальний посібник з дисципліни Маніпулятори та промислові роботи. Для студентів бакалаврів, спеціальності: 131 - Прикладна механіка, 133 – Галузеве машинобудування, / Укладачі.: Михайлов Є. П., Лінгур В.М. – Одеса: ОНПУ, 2019. - 233 с.

2. Цвіркун Л.І. Робототехніка та мехатроніка: навч. посіб. / Л.І. Цвіркун, Г. Грулер ; під заг. ред. Л.І. Цвіркуна ; М-во освіти і науки України, Нац. гірн. ун-т. –3-тє вид.,переробл. і доповн. – Дніпро: НГУ, 2017. – 224 с.

3. Михайлов Є. П. Навчальний посібник з дисципліни "Мобільні роботи" для студентів за фахом 131 - Прикладна механіка – спеціалізація - Мехатроніка та промислові роботи / Укладач: Михайлов Є. П. Одеса: ОНПУ. 2016, – 239 с. Укладач: МихайловЄ. П. Одеса: ОНПУ, 2016, – 239 с. Рег.ном. НП07368 01.06.2016, №3765-РС-2016

Формат 60x84/16. Умовн. друк. арк. 2,55. Зам. № 27. Наклад 50 прим.
Видавництво УжНУ «Говерла».
88000, м. Ужгород, вул. Капітульна, 18. E-mail: goverla-print@uzhnu.edu.ua

Свідоцтво про внесення до державного реєстру
видавців, виготівників і розповсюджувачів видавничої продукції –
Серія Зт № 32 від 31 травня 2006 року