

ШИФР

«МОДЕЛЬ СЕРВОПРИВОДА»

АНОТАЦІЯ

Наукова робота складається із вступу, 4 розділів, висновків і додатків. Загальний обсяг роботи становить 47 сторінок, список літератури з 10 найменувань на 1 сторінці і додатків на 10 сторінках.

Актуальність теми. Використання комп'ютерно-інтегрованих засобів дослідження електроприводів є актуальною задачею, зокрема розробка нових та модернізація існуючих стендів для дослідження регуляторів та їх характеристик з використанням сучасних програмних та апаратних можливостей.

Мета і завдання дослідження. Метою роботи є поліпшення процесу дослідження характеристик серводвигунів постійного струму

Об'єкт дослідження – лабораторний стенд для дослідження характеристик сервопривода постійного струму.

Предмет дослідження – динамічні та статичні характеристики замкнутої системи керування електроприводом постійного струму з енкодером.

Сфери застосування: освітня галузь, розробка електроприводів точного позиціонування.

Ключові слова: СЕРВОПРИВОД, ЕНКОДЕР, ЕЛЕКТРОПРИВОД, СТРУКТУРНА СХЕМА, МОДЕЛЮВАННЯ, МІКРОКОНТРОЛЕР.

ЗМІСТ

ВСТУП	4
1 ПОБУДОВА КОНЦЕПЦІЇ СТЕНДА ДЛЯ ДОСЛІДЖЕННЯ СЕРВОПРИВОДІВ	7
1.1 Особливості та технічні характеристики сервоприводів	7
1.2 Постановка завдання роботи.....	11
2 РОЗРАХУНОК ТА ВИБІР ЕЛЕМЕНТІВ ЕЛЕКТРИЧНОЇ СХЕМИ	14
2.1 Розробка електричної схеми стенда	14
2.2 Розрахунок і побудова статичних характеристик електропривода	15
3 СИНТЕЗ ПРОГРАМНИХ ЗАСОБІВ ФУНКЦІОНУВАННЯ СТЕНДА	21
3.1 Розробка алгоритму роботи програми стенда.....	21
3.1 Моделювання роботи програми мікропроцесорної частини стенда	23
4 МОДЕЛЮВАННЯ РЕЖИМІВ РОБОТИ СЕРВОПРИВОДА.....	26
4.1 Цифрова модель сервопривода.....	27
4.2 Неперервна модель сервопривода.....	32
ВИСНОВКИ.....	37
ПЕРЕЛІК ПОСИЛАНЬ	38
Додаток А Лістинг програми мікроконтролера	37

ВСТУП

Розвиток сучасних систем електропривода вимагає постійного підвищення технічного рівня навчальних лабораторій. Використовувані сьогодні лабораторні стенди, що включають до свого складу електромеханічні системи, контроль параметрів яких здійснюється аналоговими приладами, низько інформативні. В лабораторіях по дослідженню електромеханічних систем автоматизації присутній широкий спектр лабораторного устаткування, що включає різноманітні системи електропривода, перетворювальні пристрої й агрегати, а також регулятори які дозволяють керувати електроприводом при різних збуреннях які випадковим чином впливають на систему керування.

В даний час персональний комп'ютер широко використовується як засіб обробки та накопичення результатів вимірювань, а також для управління експериментальними установками в реальному часі. З додаванням зовнішнього модуля збору даних, комп'ютер можна перетворити в багатофункціональний лабораторний стенд.

Для програмування таких пристроїв можна використовувати універсальні мови програмування: C, Pascal та інші, але це досить трудомістке завдання і вимагає високої кваліфікації користувача. Щоб полегшити програмування вимірювальних пристроїв, були створені спеціалізовані системи, засновані на принципах візуального графічного програмування, що містять великий набір готових програмних модулів, для спрощеного програмування багатьох операцій та виконання специфічних задач автоматизації наукових вимірювань та експериментів. Система по замовчуванню сама виконує багато функцій, які зазвичай покладаються на програміста, звільняючи користувача від оперування низькорівневими деталями роботи програми.

Тому актуальним є розробка нових та модернізація існуючих стендів для дослідження регуляторів та їх характеристик з використанням сучасних програмних та апаратних можливостей.

Метою роботи є поліпшення процесу дослідження характеристик серводвигунів постійного струму

Для досягнення поставленої мети необхідно розв'язати такі задачі:

- провести аналіз існуючих підходів до побудови цифрових систем керування електроприводами з використанням серводвигунів постійного струму ;
- провести розробку схеми електричної структурної стенда;
- здійснити практичну реалізацію програмної частини стенда та провести його тестування. (статичні та динамічні режими роботи, імітація технологічного режиму).
- провести розробку комп'ютерної моделі серводвигунів постійного струму.

У процесі дослідження застосовувалися: методи теорії автоматичного керування для структурного аналізу системи електропривода та синтезу регуляторів; теорія алгоритмів для написання та налагодження програми роботи мікроконтролера.

Практичне значення одержаних результатів полягає в тому, що на основі отриманих теоретичних положень розроблено програмні і апаратні засоби, зокрема:

- розроблена принципова електрична схема стенда для дослідження динамічних характеристик сервопривода постійного струму з цифровим енкодером, що передбачають цифрове регулювання положення виконавчого механізму;

- здійснено вибір механічних та електронних компонентів необхідних для побудови стенда із можливістю регулювання параметрів його налаштування;
- розроблено інформаційно-вимірювальну систему для візуалізації характеристик системи електропривода;
- модернізовано експериментальну дослідну установку.

1 ПОБУДОВА КОНЦЕПЦІЇ СТЕНДА ДЛЯ ДОСЛІДЖЕННЯ СЕРВОПРИВОДІВ

1.1 Особливості та технічні характеристики сервоприводів

Сервопривод - механізм, що складається із двигуна, механічної передачі та має у своєму складі спеціальний сенсор положення, який призначений для перетворення значення управляючого сигналу в пропорційне переміщення або швидкість обертання.

Від звичайного електродвигуна сервопривод відрізняється тим, що можна задати точне положення вала в градусах. Сервоприводи - це будь-які механічні приводи, які включають в себе сенсор деякого параметра і блок управління, який здатний автоматично підтримувати необхідні параметри, що відповідають певним зовнішнім значенням.

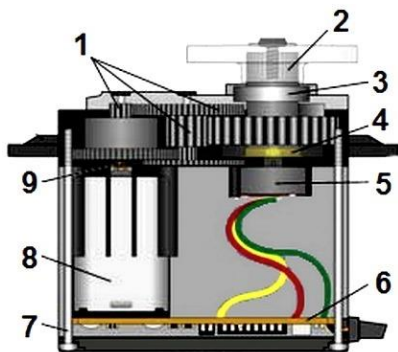


Рисунок 1.1 – Основні елементи серводвигуна

На рисунку 1.1: 1 – шестерні редуктора; 2 – вихідний вал; 3 – підшипник; 4 – нижня втулка; 5 – потенціометр (сенсор положення); 6 – плата управління; 7 – гвинти корпусу; 8 – електродвигун постійного струму; 9 – ведуча шестерня двигуна.

Для перетворення електричної енергії в механічний рух, необхідний електродвигун. Приводом є редуктор з електродвигуном. Редуктор потрібний для зниження швидкості двигуна, так як швидкість занадто велика для

застосування в режимі відпрацювання кута повороту. Редуктор складається з корпусу, в якому розташовані вали з шестернями, здатними перетворювати і передавати крутний момент.

Шляхом запуску і зупинки електродвигуна можна приводити в рух вихідний вал редуктора, який пов'язаний з шестернею сервоприводу. До валу можна приєднувати пристрій або механізм, яким потрібно керувати. Крім цього для контролю положення вала потрібна наявність сенсора зворотного зв'язку. Цей сенсор може перетворити кут повороту знову в сигнал електричного струму.

Такий сенсор отримав назву енкодера. Як енкодера може застосовуватися потенціометр. Якщо бігунок потенціометра повертати, то буде змінюватися його опір. Значення цього опору прямо пропорційно залежить від кута повороту потенціометра. Таким чином, є можливість домогтися установки певного положення механізму.

Крім вище названого потенціометра, редуктора і електродвигуна, сервоприводи оснащені електронною платою, яка обробляє зовнішні сигнали контрольованого параметра від потенціометра, порівнює, і відповідно до результату порівняння запускає або зупиняє електродвигун. Іншими словами ця електронна начинка відповідає за підтримку негативного зворотного зв'язку.

Підключення сервоприводу здійснюється трьома провідниками, два з яких подають живлення напругою електродвигуна, а по третьому провіднику надходить сигнал управління, за допомогою якого виконується установка положення вала двигуна.

Крім електродвигуна, роль приводу може відігравати і інший механізм, наприклад пневматичний циліндр зі штоком. Як сенсор зворотного зв'язку застосовують також сенсори кута повороту, або сенсори, що працюють на ефекті Холла. Керуючий блок є сервопідсилювачем, частотним

перетворювачем або індивідуальним інвертором. Він може містити також і задатчик сигналу управління.

При необхідності створення плавного гальмування або розгону для запобігання надмірним динамічним навантажень двигуна, виконують схеми більш складних мікроконтролерів управління, які можуть контролювати позицію робочого елемента набагато точніше. Подібним чином виконано пристрій приводу установки позиції головок в комп'ютерних жорстких дисках.

Основні параметри, які характеризують сервоприводи:

- обертовий момент, це найбільш важливий параметр сервопривода. У паспортних даних найчастіше вказується кілька значень моменту для різних величин напруги;
- швидкість повороту також є важливою характеристикою. Вона вказується в еквіваленті часу, необхідному для зміни позиції вихідного валу привода на 60 градусів. Цей параметр також можуть вказувати для кількох значень напруги;
- за типом системи керування бувають аналогові та цифрові сервоприводи;
- кут повороту - це найбільший кут, на який вихідний вал здатний повернутися. Найчастіше цей параметр дорівнює 180 або 360 градусів. Також поширені двигуни постійного обертання. При необхідності звичайний сервопривід можна модернізувати для постійного обертання.

Сервоприводи обертального руху використовуються для промислових робіт, станків з ЧПУ, поліграфічних верстатів, промислових швейних машин, пакувальних верстатів, приладів, авіамоделювання.

Іншим варіантом точного позиціонування без сенсора зворотного зв'язку є застосування крокового двигуна. У цьому випадку схема управління відраховує необхідну кількість імпульсів (кроків) від положення репера. При

цьому точне позиціонування забезпечується параметричними системами з негативним зворотним зв'язком, які утворюються взаємодіючими між собою відповідними полюсами статора і ротора крокового двигуна. Система управління кроковим двигуном, активізує відповідний полюс статора, формує сигнал завдання для відповідної параметричної системи.

Так як сенсор зазвичай контролює положення приводного елемента, електричний сервопривод має наступні переваги перед кроковим двигуном:

- не пред'являється специфічні вимоги до електродвигуна і редуктора - вони можуть бути практично будь-якого потрібного типу і потужності (а крокові двигуни, як правило, малопотужні і тихохідні);
- гарантує максимальну точність, автоматично компенсуючи: механічні (люфти в приводі) або електронні збої привода;
- широкий діапазон регулювання швидкості переміщення елемента (у крокової двигуна найменша максимальна швидкість в порівнянні з іншими типами електродвигунів);
- витрати енергії пропорційні опору зовнішнього механізму (на кроковий двигун постійно подається номінальна напруга з запасом на можливе перевантаження);

Недоліки сервопривода в порівнянні з кроковим двигуном:

- необхідність в додатковому елементі – сенсорі положення;
- складніша реалізація блоку управління і логіка його роботи (потрібна обробка результатів датчика і вибір управляючого впливу, а в основі контролера крокового двигуна – простий лічильник);
- проблема фіксування вихідного вала в статичному положення: зазвичай вирішується постійним пригальмовуванням переміщуваного елемента або вала електродвигуна (що веде до втрат енергії) або застосування черв'ячних / гвинтових передач (ускладнення конструкції) (в кроковому двигуні кожен крок фіксується самим двигуном);

- менший момент на низьких швидкостях обертання;
- сервоприводи, як правило, дорожче крокових.

1.2 Постановка завдання роботи

На рисунку 1.2 зображено стенд для дослідження серводвигуна постійного струму в лабораторії спеціальних електричних машин.

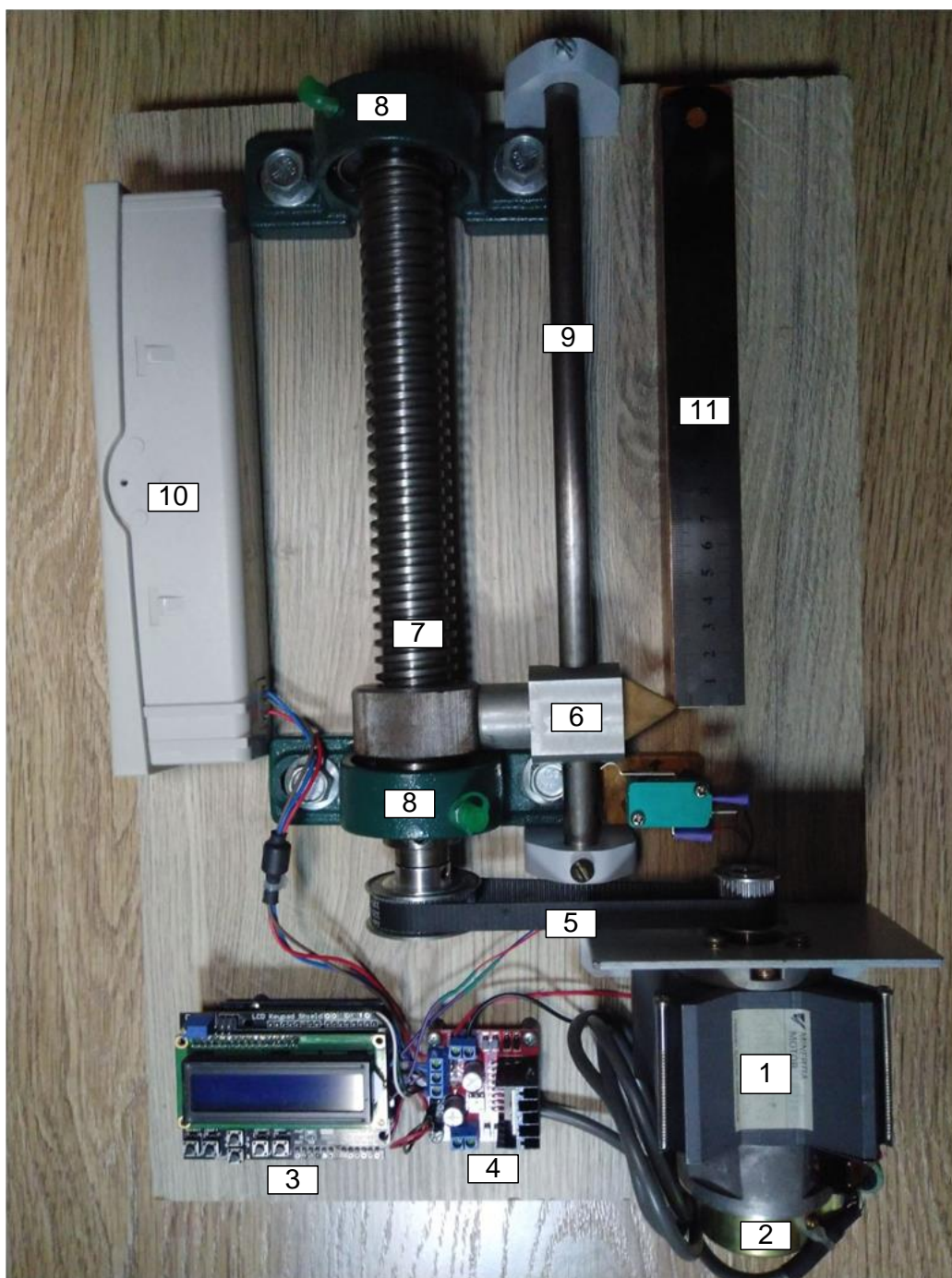


Рисунок 1.2 – Стенд для дослідження серводвигуна

Основними елементами стенда є: 1 – двигун постійного струму з постійними магнітами; 2 – вбудований енкодер двигуна; 3 – контролер керування з дисплеєм та кнопками керування; 4 – реверсивний драйвер двигуна; 5 – зубчата ремінна передача; 6 – стрілка позиціонування; 7 – гвинтова передача; 8 – підшипники; 9 – горизонтальна направляюча; 10 – блок живлення; 11 – лінійка.

Кінематична схема стенда для дослідження сервоприводів наведена на рисунку 1.3.

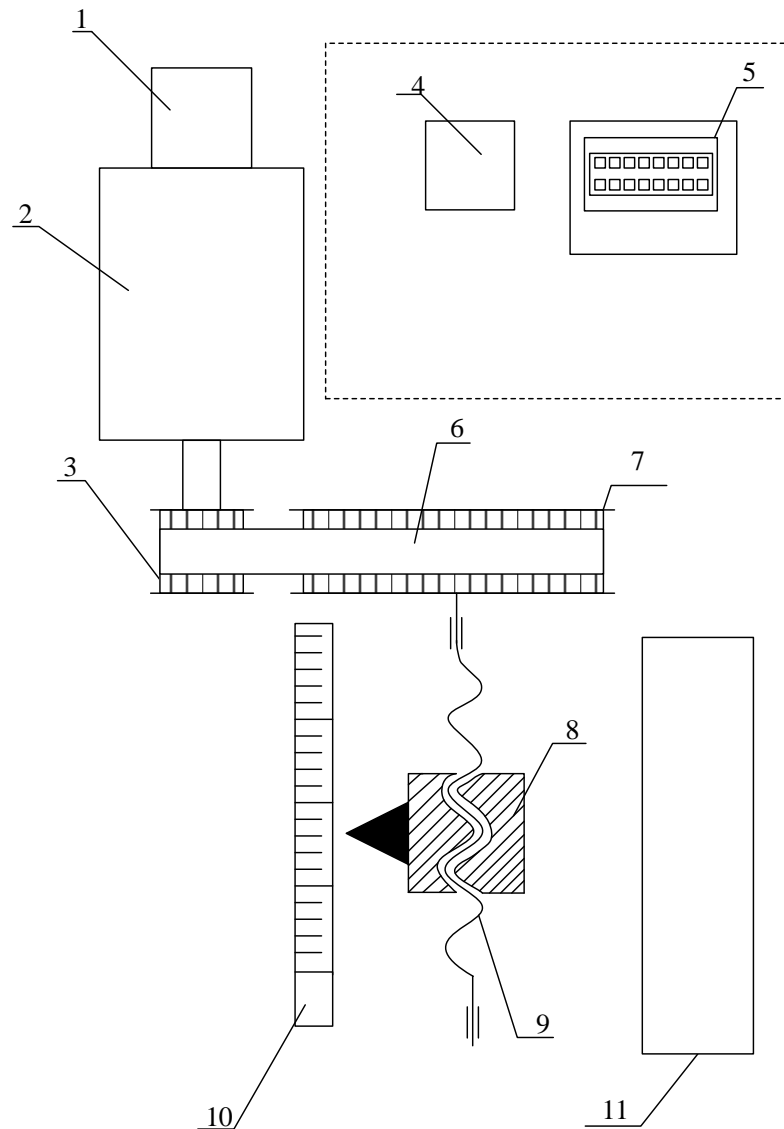


Рисунок 1.3 – Кінематична схема стенда для дослідження
сервоприводів

На рисунку 1.3: 1 – енкодер; 2– двигун; 3 – шків двигуна; 4 – драйвер двигуна I298n; 5 – плата керування ArduinoUNO; 6 – ремінь; 7– шків; 8 – гайка; 9– вал; 10– лінійка; 11 – блок живлення.

Необхідність модернізації даного стенда впливає із наявних недоліків:

1. Програмне забезпечення не дозволяє зберігати налаштування та поточну позицію в енергонезалежній пам'яті, що призводить до помилок при вимкненні чи скиданні контролера.
2. Програма керуючого контролера не передбачає зміну налаштувань (швидкість переміщення, зона нечутливості, параметри регулятора) окрім заданого значення координати.
3. Відсутній механізм автоматичного калібрування та пошуку нульової точки.
4. Відсутні комп'ютерні моделі для віддаленого налаштування програмного та апаратного забезпечення стенда.

2 РОЗРАХУНОК ТА ВИБІР ЕЛЕМЕНТІВ ЕЛЕКТРИЧНОЇ СХЕМИ

2.1 Розробка електричної схеми стенда

Схема електрична структурна стенда для дослідження сервоприводів з використанням запропонованого підходу зображена на рисунку 2.1.

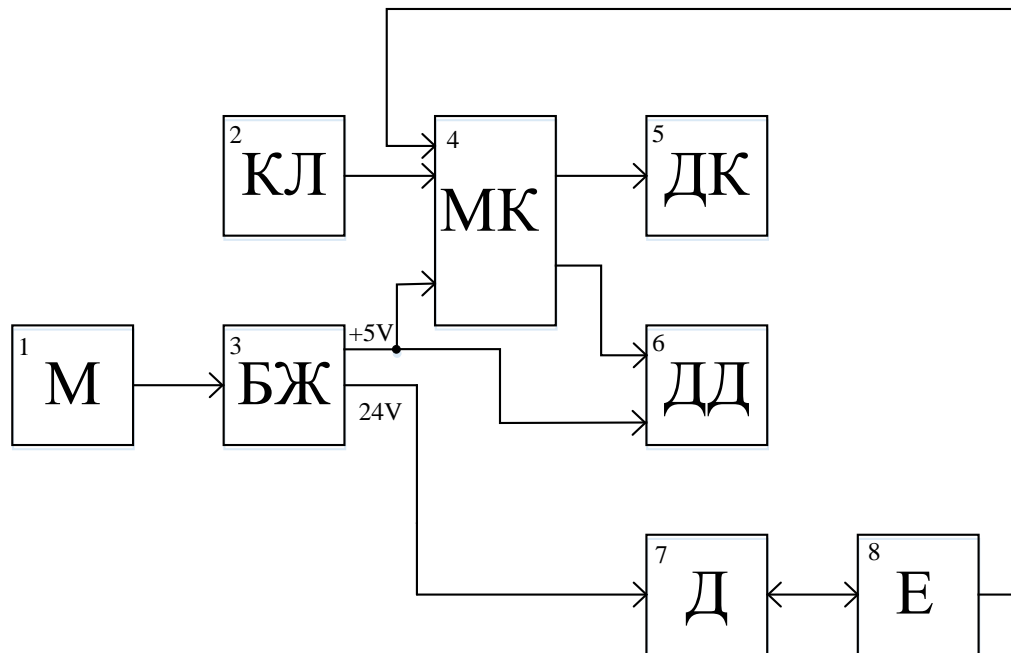


Рисунок 2.1 - Схема електрична структурна стенда для дослідження сервопривода

На рисунку 2.1: М – мережа ; БЖ – блок живлення; КЛ – клавіатура; МК – мікроконтролер Arduino Uno; К – клавіатура; МК – мікроконтролер; ДД – драйвер двигуна; Д – двигун; Е – енкодер; ДК – дисплей користувача.

Основою стенда є мікроконтролер Arduino Uno. Мікроконтролер Arduino Uno керування роботою серводвигуна постійного струму за рахунок дії на драйвер двигуна. Також контролер зчитує інформацію про положення використовуючи енкодер. Також передбачено наявність дисплея користувача з клавіатурою для введення параметрів регулювання[4].

Для живлення мікропроцесорних пристроїв використовується

стабілізований блок живлення (БЖ).

В якості вимірювальної платформи в стенді для дослідження характеристик сервоприводів використана апаратна платформа Arduino [1].

Arduino – це електронний конструктор і зручна платформа швидкої розробки електронних пристроїв. Платформа користується величезною популярністю в усьому світі завдяки архітектурі і програмному коду.

Середовище розробки Arduino складається з вбудованого текстового редактора програмного коду, поля повідомлень, вікна виводу тексту (консолі) та панелі інструментів найчастіше вживаних команд та декількох меню.

В даному випадку обрано апаратну платформу типу Arduino Uno. З врахуванням вибраних елементів схема електрична принципова стенда матиме вигляд, що приведено на рисунку 2.2.

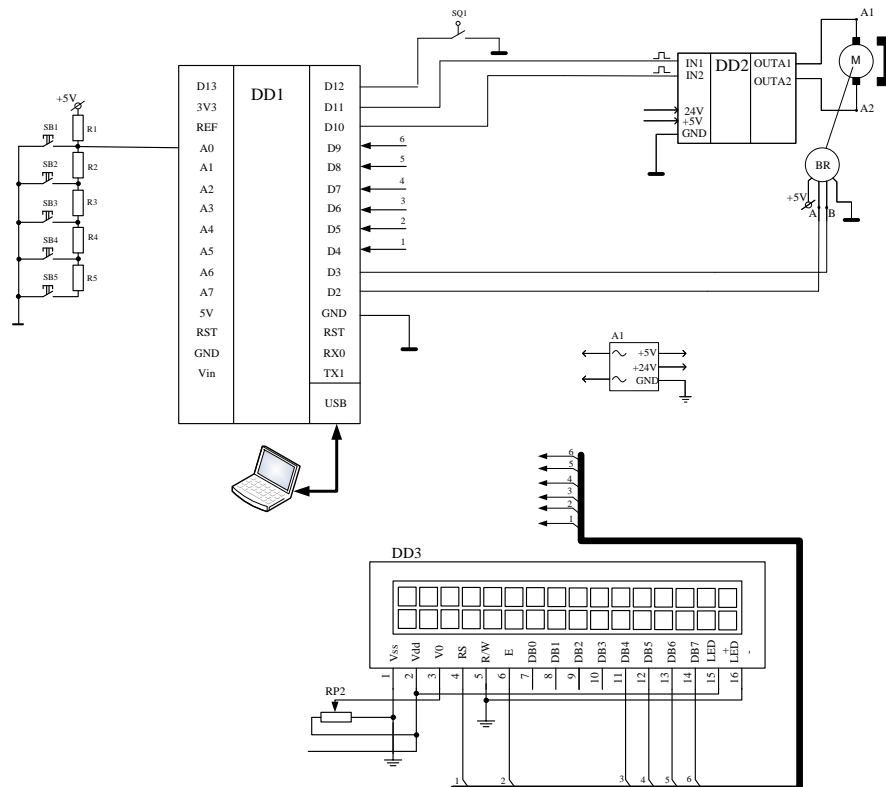


Рисунок 2.2 - Схема електрична принципова стенда

2.2 Розрахунок і побудова статичних характеристик

електропривода

Побудуємо і розрахуємо електромеханічну характеристику двигуна ЕМ-115 номінальні параметри якого наведені в таблиці 2.1.

Таблиця 2.1 – Паспортні дані двигуна ЕМ-115

	Параметр	Значення
1	Напруга живлення, В	36.5
2	Номінальний струм, А	1,5
3	Максимальна швидкість обертання, об/хв	3000
4	Привід	біполярний привід постійного струму

Електромеханічною характеристикою називається залежність кутової швидкості ω від струму якоря двигуна I_a . Електромеханічна характеристика двигуна буде мати вигляд:

$$\omega = \frac{U}{k\Phi} - \frac{I_a \cdot R_{\Sigma}}{k\Phi}, \quad (2.1)$$

де k – коефіцієнт, що залежить від конструктивних даних двигуна;

ω – кутова швидкість двигуна;

Φ – магнітний потік.

Номінальна кутова швидкість

$$\omega_n = 2\pi n / 60, \quad (2.2)$$

$$\omega_n = 209,333 \text{ (с}^{-1}\text{)}.$$

За номінальними параметрами можемо розрахувати такі величини:

Знайдемо R_{Σ} – сумарний опір якірного кола. Для цього спочатку знайдемо номінальний опір:

$$R_H = U_H / I_H, \quad (2.3)$$

$$R_H = 36,5 / 1,5 = 24,33 \text{ (Ом)}.$$

Тоді:

$$R_{\Sigma} = 0,5(1 - \eta)R_H, \quad (2.4)$$

$$R_{\Sigma} = 0,5(1 - 0,475) \cdot 24,33 = 6,38 \text{ (Ом)}.$$

Розрахуємо коефіцієнт потоку:

$$k\Phi = \frac{U_H - I_{я} \cdot R_{я}}{\omega_H}, \quad (2.5)$$

$$k\Phi = \frac{36,5 - 1,5 \cdot 6,38}{209,333} = 0,128.$$

Розрахуємо кутову швидкість холостого ходу:

$$\omega_0 = \frac{U_H}{k\Phi}, \quad (2.6)$$

$$\omega_0 = \frac{36,5}{0,128} = 285,15 \text{ (с}^{-1}\text{)}.$$

Можемо розрахувати падіння кутової швидкості:

$$\Delta\omega = \frac{I_{я} \cdot R_{я}}{k\Phi}, \quad (6.7)$$

$$\Delta\omega = \frac{1,5 \cdot 6,38}{0,128} = 74,76 \text{ (с}^{-1}\text{)}.$$

Побудуємо електромеханічну характеристику, для цього визначимо точки:

Точка 1: $I_{я} = 0$; $\omega = 0$; $\omega_0 = 285,15 \text{ с}^{-1}$;

Точка 2: $\omega = 0$; $I_{кз.} = U/R_{я} = 36,5/6,38 = 5,72 \text{ А}$.

Електромеханічна характеристика двигуна постійного струму показана на рисунку 2.3.

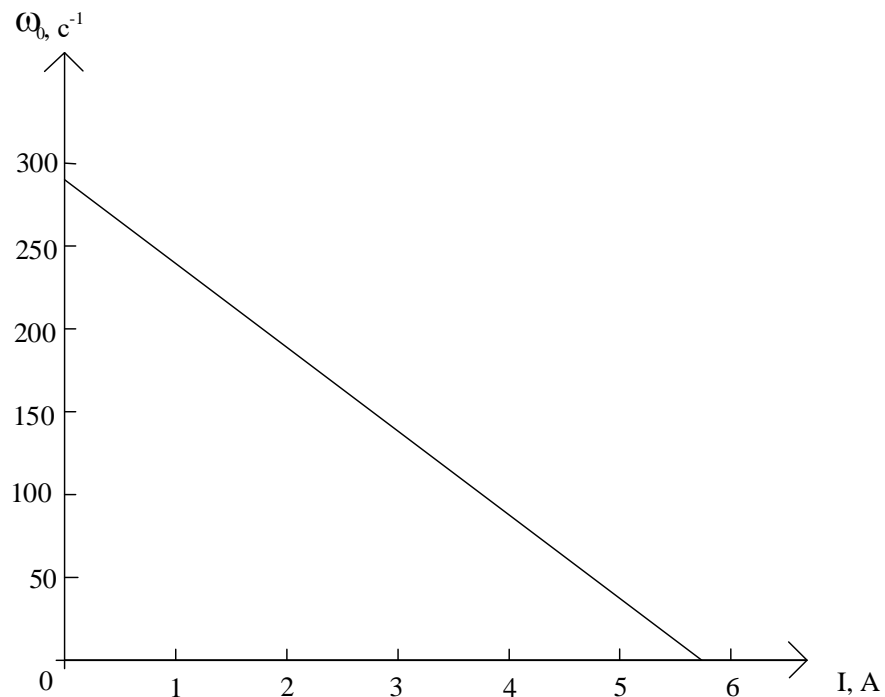


Рисунок 2.3 – Електромеханічна характеристика двигуна

Розрахуємо та побудуємо механічну характеристику двигуна. Механічною характеристикою електричного двигуна називається залежність кутової швидкості і струму в обмотках. Механічна характеристика двигуна буде мати вигляд:

$$\omega = \frac{U}{k\Phi} - \frac{M_{\text{д}} \cdot R_{\Sigma}}{(k\Phi)^2}. \quad (2.8)$$

Всі номінальні величини залишаються такими самими, як і для електромеханічної характеристики двигуна. Для побудови механічної характеристики визначимо точки:

Точка 1: $M = 0$, $\omega = U / k\Phi = \omega_0 = 285,15 \text{ c}^{-1}$,

Точка 2: $\omega_0 = 0$, $M_{кз} = (U / R_{я}) \cdot K\phi = (36,5/6,38) \cdot 0,128 = 0,73 \text{ Нм}$.

По даним точкам побудуємо механічну характеристику на рисунку 2.4.

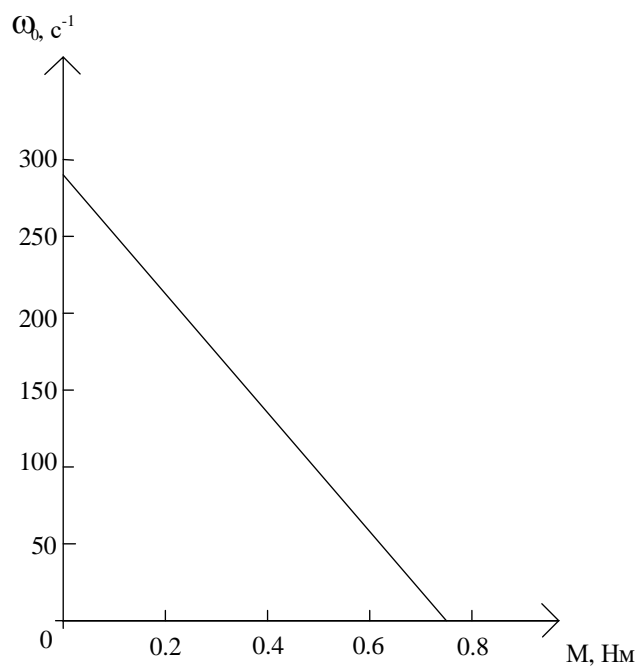


Рисунок 2.4 – Механічна характеристика електродвигуна

Дозвіл інкрементних енкодерів вимірюється в імпульсах за оберт. На рисунку 2.5 зображена дозвільна здатність інкрементного енкодера, яка показує, що на один повний оберт енкодера припадає 282 імпульса.

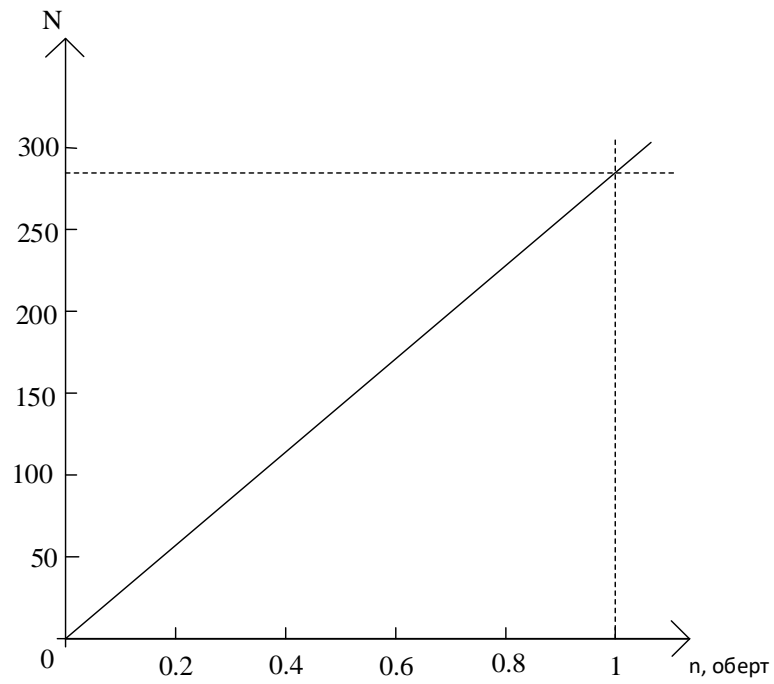


Рисунок 2.5 – Дозвільна здатність інкрементного енкодера

3 СИНТЕЗ ПРОГРАМНИХ ЗАСОБІВ ФУНКЦІОНУВАННЯ СТЕНДА

3.1 Розробка алгоритму роботи програми стенда

Алгоритм роботи регулятора положення сервопривода зображено на рисунку 3.1.

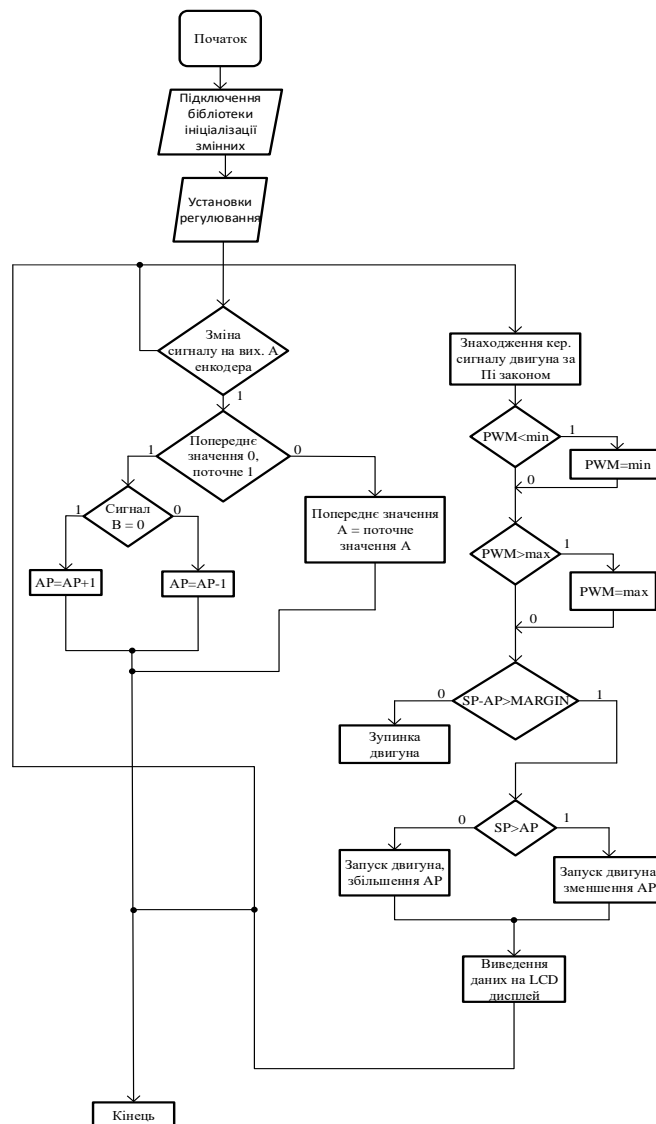


Рисунок 3.1 – Алгоритм роботи програми

Значення блоків на блок-схемі алгоритму:

- блок «Початок» - підготовка системи, підключення та виконання налаштувань;

- блок «Установка регулювання» - відбувається налаштування регулювання;
- блок «Зміна сигналу на виході “А” енкодера?» - відбувається перевірка про зміну сигналу з енкодера;
- блок «Попереднє значення “0”, поточне “1”?» - якщо попереднє значення “0” а поточне “1” то зчитується значення з вихода “В”;
- блок «Сигнал $V = 0$?»- якщо так, то до поточного значення додається одиниця, якщо ні, то віднімається одиниця;
- блок «Знаходження керуючого закону двигуна» - знаходження керуючого закону двигуна за П законом;
- блок « $PWM < min$?» - якщо значення менше мінімального воно порівнюється до мінімального;
- блок « $PWM > max$?» - якщо значення більше максимального воно порівнюється до максимального;
- блок « $SP - AP > MARGIN$?» - якщо різниця між поточним та заданим входить в межі допуску двигун зупиняється;
- блок « $SP > AP$?» - якщо поточне значення більше за задане, двигун запускається та збільшує/зменшує задане значення;
- блок «Виведення даних на LCD» - виведення даних поточного та заданого значення на LCD;
- блок «Кінець» - завершення алгоритму.

При ввімкненні відбувається ініціалізація змінних та установка регулювання. Якщо не відбувається зміна сигналу знаходиться керуючий сигнал двигуна. Якщо значення менше мінімального, то воно порівнюється до мінімального, якщо більше максимального тоді порівнюється до максимального. Під час вимірювання різниця заданого та поточного не виходить за допустимі межі двигун зупиняється, в іншому випадку двигун

запускається зменшуючи або збільшуючи задане значення до поточного. Далі дані виводяться на дисплей.

Якщо все ж таки відбувається зміна сигналу та попереднє значення змінилось, до поточного віднімається або додається деяке необхідне значення до встановлення заданого. Програма для мікроконтролера була розроблена в програмному середовищі ArduinoIDE. Лістинг програми з коментарями наведено в додатку.

3.1 Моделювання роботи програми мікропроцесорної частини стенда

Для налаштування параметрів сервопривода та програмного коду його роботу промодельовано в середовищі Proteus (рисунок 3.2).

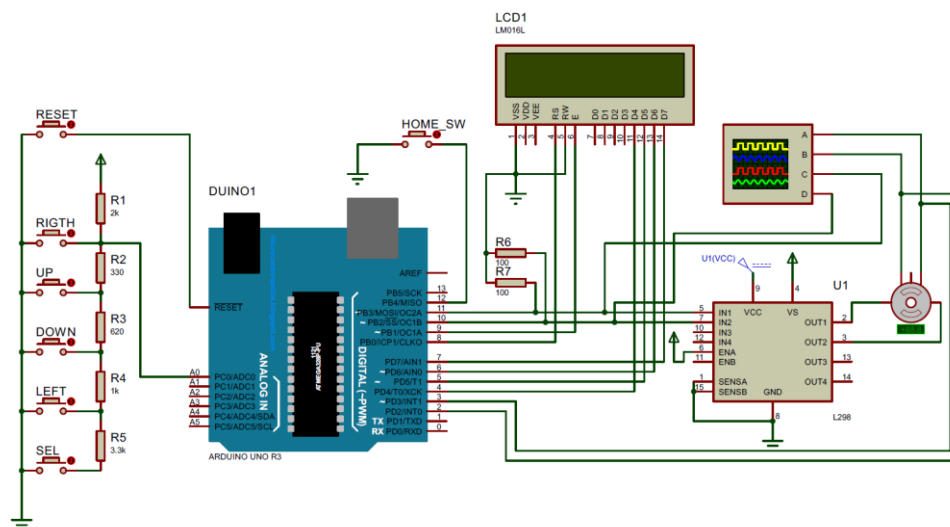


Рисунок 3.2 – Модель сервопривода в середовищі Proteus

Зокрема здійснено моделювання (рисунок 3.3) показує адекватність відпрацювання програми та наявні перехідні процеси. На рисунку 3.3 наведено графіки імпульсних сигналів виходів енкодера та ШІМ сигналів включення обох напрямків транзисторного моста, як можна побачити є незначне перерегулювання, параметри якого залежать від пропорційного

коефіцієнта регулятора, інерції привода, максимальної та мінімальної напруги.

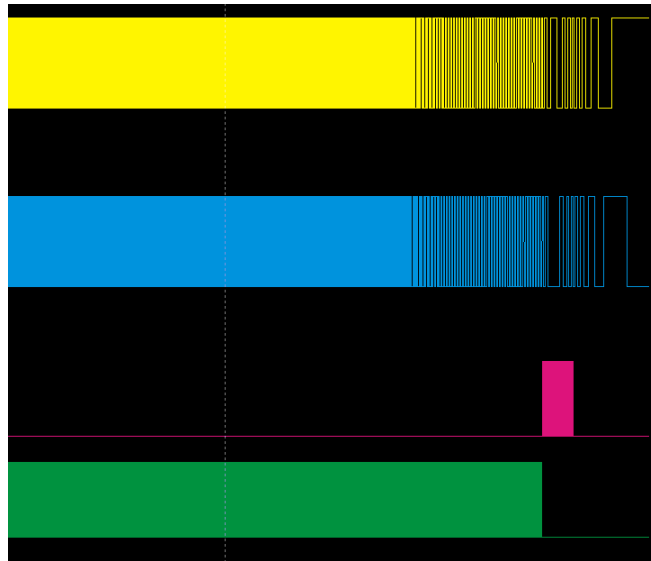


Рисунок 3.3 –Моделювання сервопривода в середовищі Proteus

В роботі розроблено меню користувача, з відображенням інформації на дисплеї з можливостями налаштування основних параметрів привода, а саме: заданої позиції, мінімальної та максимальної швидкості переміщення, зони нечутливості регулятора, та окреме меню налаштування пошуку нульової точки. Слід відмітити, що введені параметри та поточне значення координати зберігаються в енергонезалежній пам'яті, тому навіть після вимкнення живлення дані налаштувань та поточна координата відтворюються з пам'яті при новому завантаженні. Задане значення вводиться в міліметрах, а відображення поточного і заданого значення на головному меню в імпульсах, які надійшли з енкодера за одинарним алгоритмом зчитування. Ці значення є пропорційними, зокрема для механіки стенда, що модернізується цей коефіцієнт складає 96, тобто переміщення робочого органу на 1 мм відповідає 96 імпульсам на виході енкодера. Переходи по пунктам меню в графічному вигляді показано на рисунку 3.4. Слід зауважити, що збереження поточної позиції відбувається після її відпрацювання, через деякий проміжок

часу, для підвищення надійності та зменшення похибки накопичення, що спричинюється перехідним процесом. При відпрацювання пошуку нульової точки привод вмикається на понижених обертах в сторону кінцевого вимикача до його спрацювання.

4.1 Цифрова модель сервопривода

Всі елементи стенда змонтовані на ДСП розміром 500x400 з використанням болтового з'єднання. Для можливості віддаленого налаштування параметрів сервопривода створені комп'ютерні моделі сервопривода в середовищі Matlab Simulink. На рисунку 4.1 представлено цифрову модель сервопривода. Модель представляє собою релейний регулятор положення із рівномірним темпом нарощення чи зменшення поточної координати який задано частотою тактового генератора.

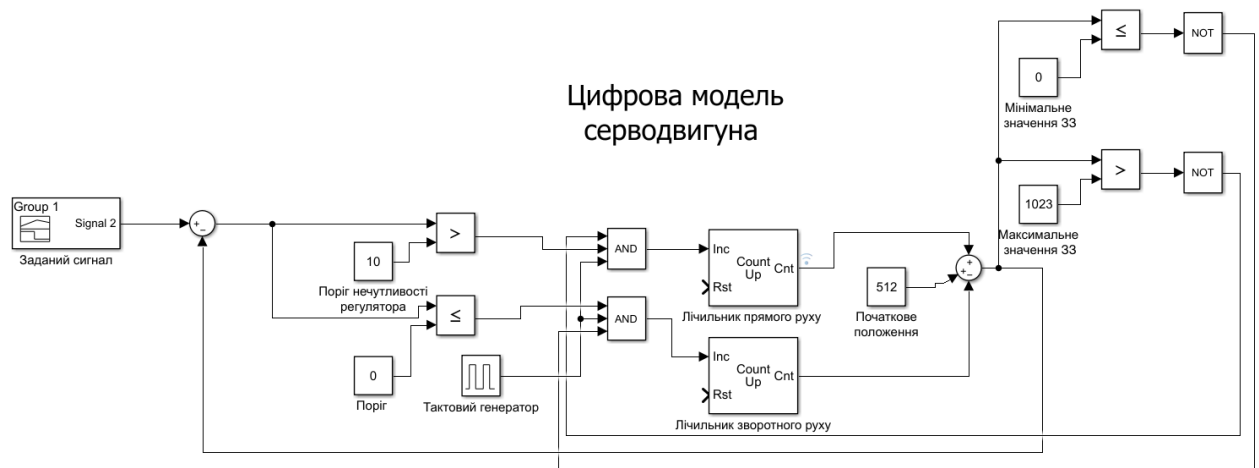


Рисунок 4.1 – Цифрова модель сервопривода

Положення сервопривода в моделі обмежено в діапазоні 0...1000, якому пропорційно відповідає положення або кут повороту. Значення координати положення формується на виході суматора на протилежні за знаком входи якого подані виходи лічильників прямого та зворотного руху. Нарощування вмісту лічильників відбувається з частотою тактового генератора в залежності від керуючого сигналу релейного регулятора. Так коли помилка регулювання більша порогу нечутливості, вмикається нарощення лічильника прямого руху, а коли значення помилки має негативне значення – вмикається нарощення лічильника зворотного руху.

Проведемо моделювання для різної форми вхідних сигналів задання, приймемо початкове положення вала 500, тобто центральне положення. На рисунку 4.2 наведено форму тестового сигналу задання вихідної координати.

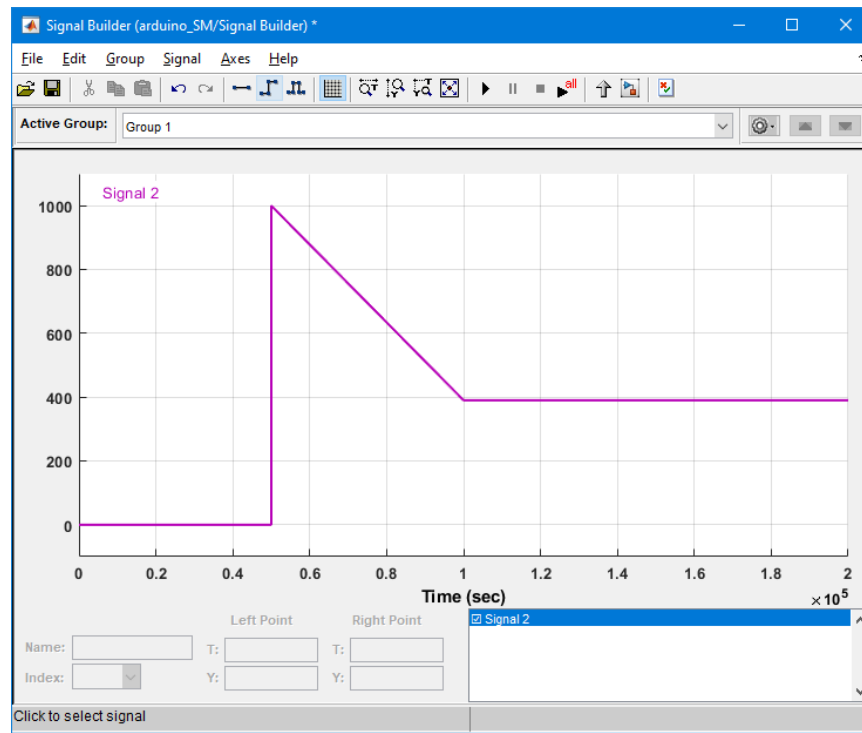


Рисунок 4.2 – Тестовий сигнал

Відпрацювання координати задання наведено на рисунку 4.3

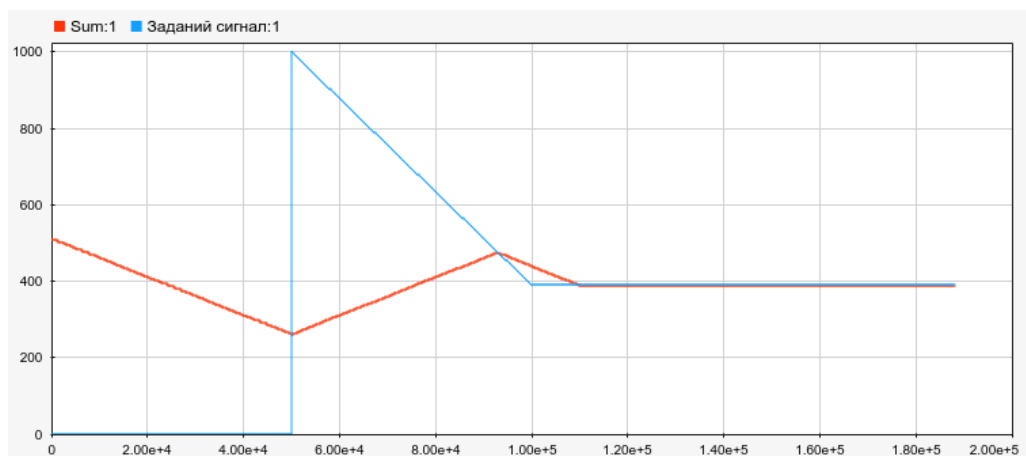


Рисунок 4.3 – Відпрацювання сигналу задання

Промодельюємо відпрацювання положення сервоприводом для стрибкоподібного сигналу (рисунк 4.4)

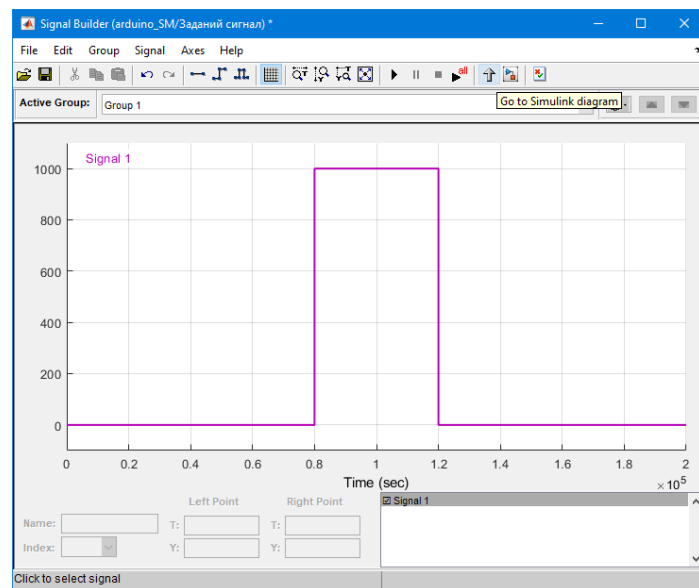


Рисунок 4.4 – Тестовий стрибкоподібний сигнал

Як видно із результатів моделювання (рисунк 4.5) сервопривод коректно відпрацьовує задане положення, але його динаміки не достатньо для його повного відтворення.

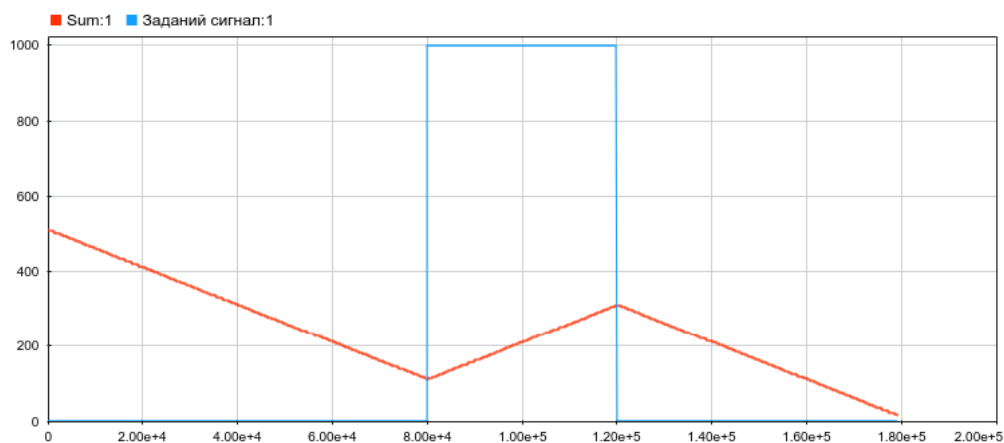


Рисунок 4.5 – Тестовий стрибкоподібний сигнал

Змінимо налаштування генератора тактових імпульсів збільшивши частоту в 10 разів. Відповідні налаштування тактового генератора наведені на рисунку 4.6.

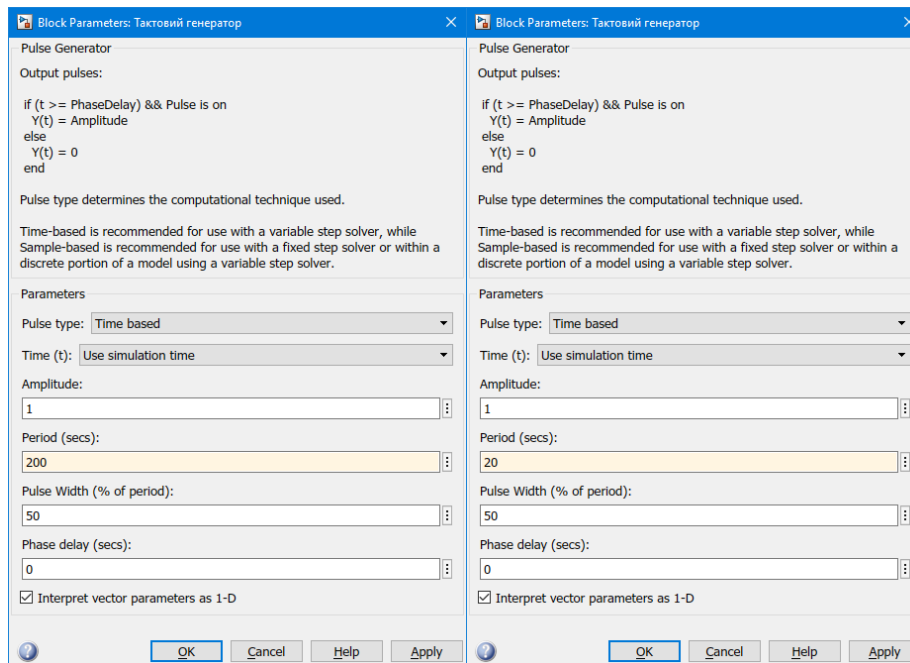


Рисунок 4.6 – Параметри налаштування тактового генератора

Відповідно відпрацювання координати сервоприводом відбувається із більшою ступінню подібності (рисунок 4.7).

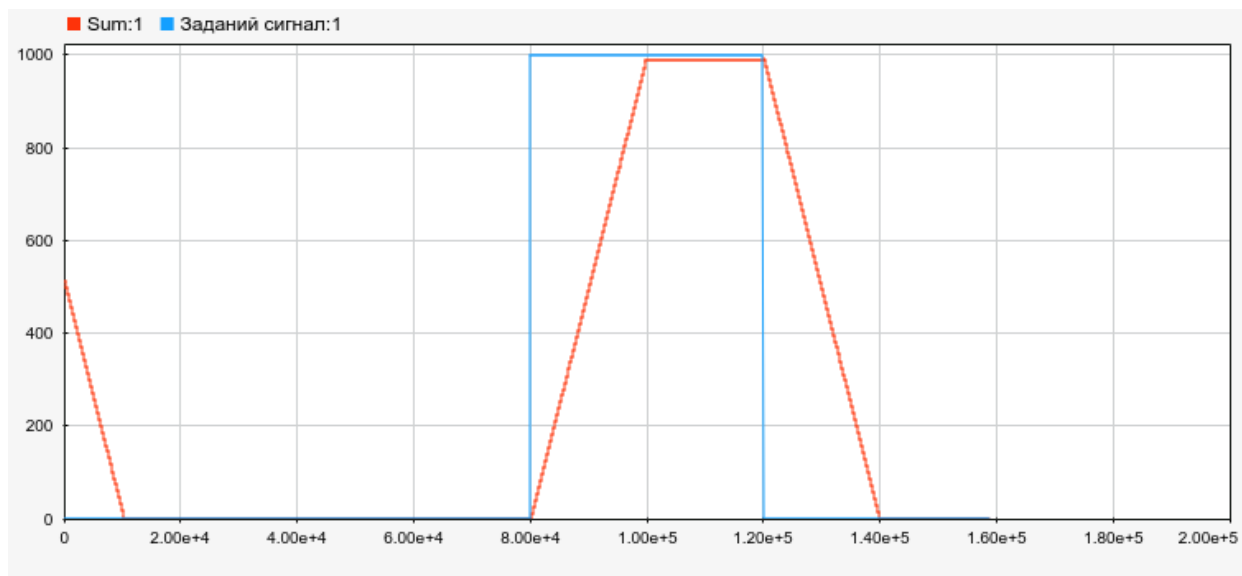


Рисунок 4.7 – Відпрацювання тестового стрибкоподібного сигналу

Промодельюємо відпрацювання положення сервоприводом для синусоїдального сигналу (рисунок 4.8).

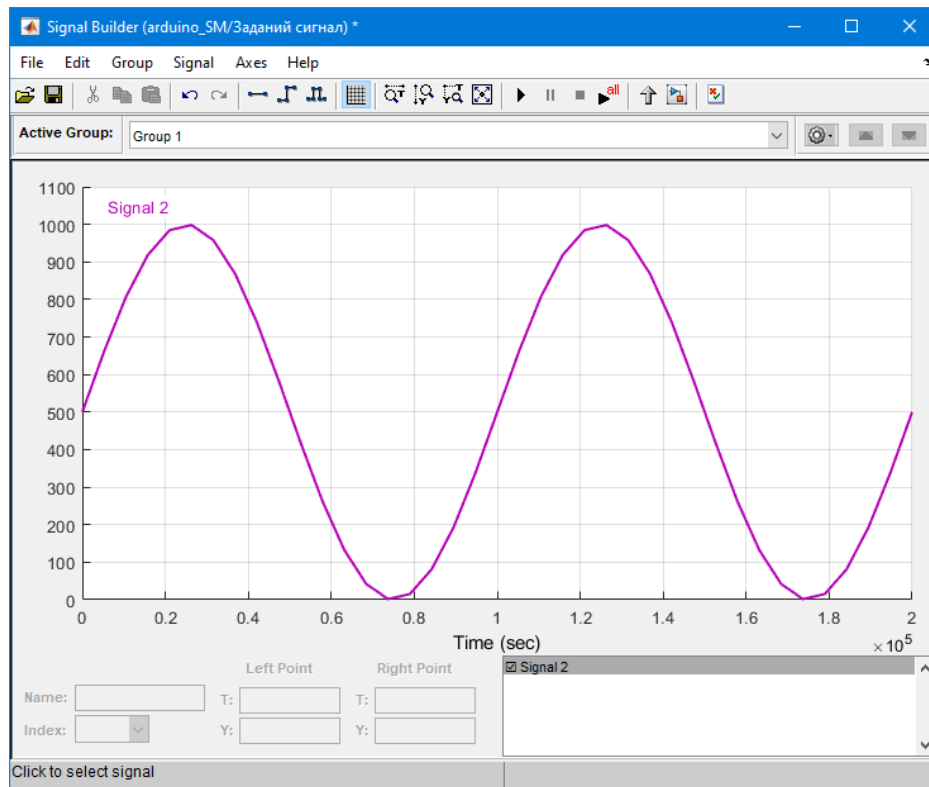


Рисунок 4.8 – Тестовий синусоїдальний сигнал

Відповідно відпрацювання координати сервоприводом відбувається коректно із врахуванням динаміки сервопривода (рисунок 4.9). На рисунку 4.10 наведено графік відпрацювання координати для синусоїди при нульовому початковому положенні.

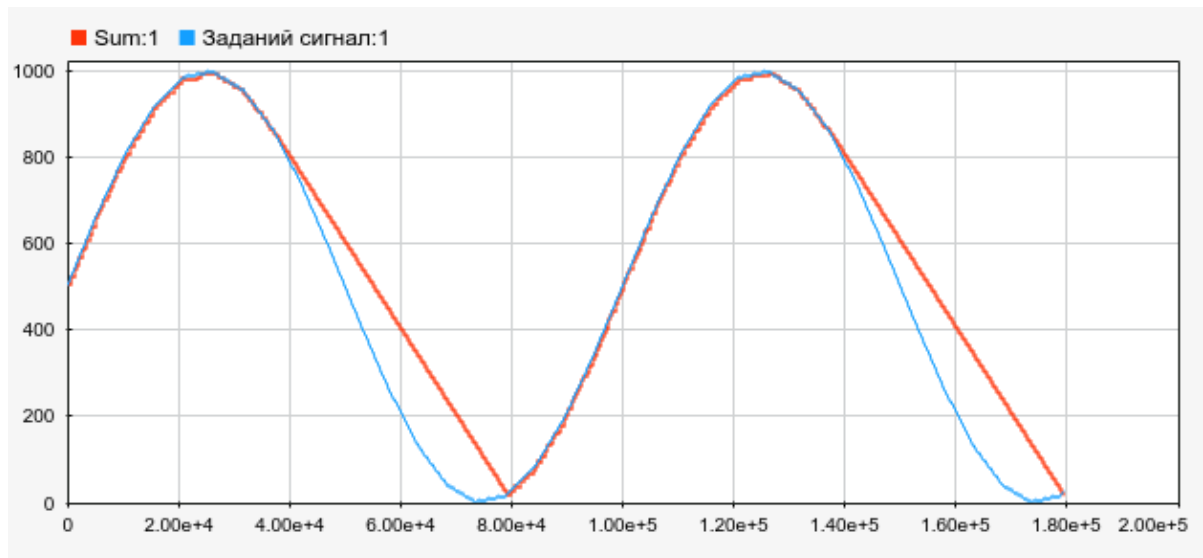


Рисунок 4.9 – Відпрацювання тестового синусоїдального сигналу

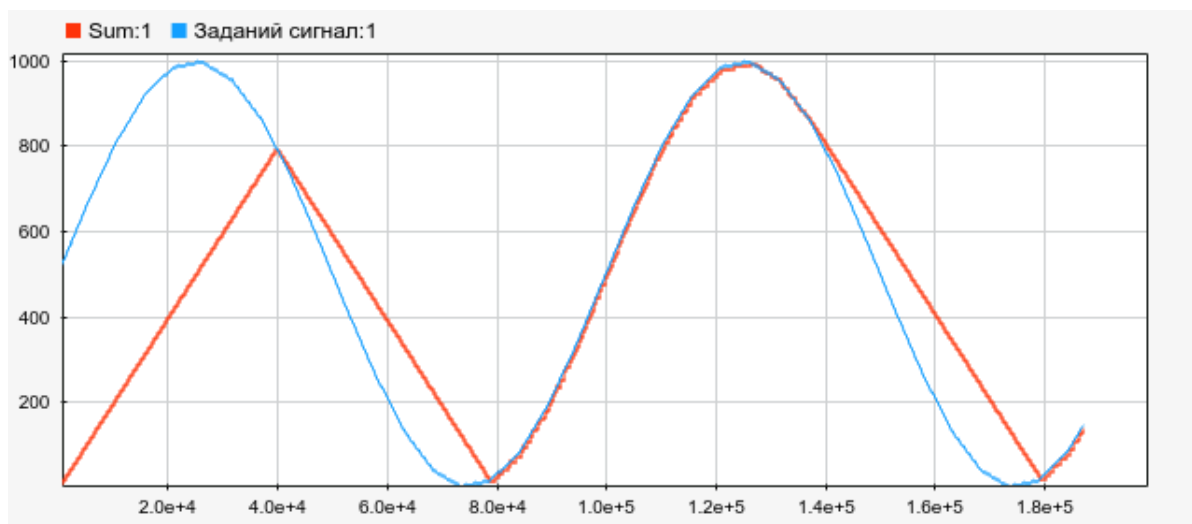


Рисунок 4.10 – Відпрацювання синусоїдального сигналу при нульовому початковому положенні

4.2 Неперервна модель сервопривода

Також запропонована неперервна модель сервопривода (рисунок 4.11) де положення визначається як вихідний сигнал інтегратора з обмеженням.

Задане значення регулятора будемо формувати вручну через елемент керування.

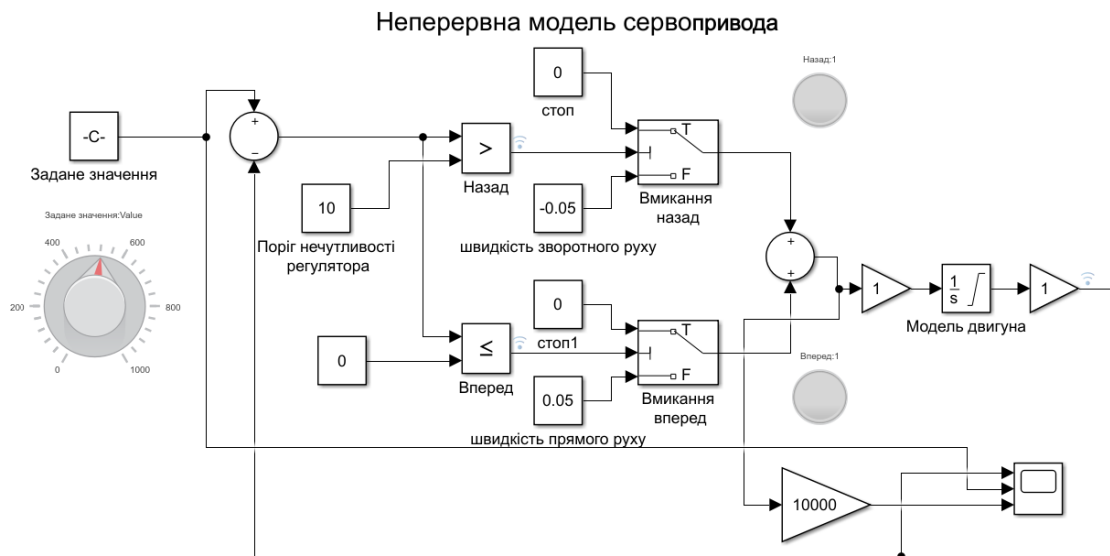


Рисунок 4.11 – Неперервна модель сервопривода

Регулятор працює наступним чином, на вхід будь-якого регулятора приходить сигнал різниці між задаючим значенням і тим, що є насправді. Тобто від того значення, яке ми хочемо задати ми віднімаємо те, що є в дійсності. В залежності від цього, потрібно рухатися вперед, щоб цю помилку компенсувати, всі регулятори компенсують помилку, для цього потрібно рухатися або вперед, або назад.

Представимо графік для кращого розуміння процесу регулювання (рисунок 4.12).

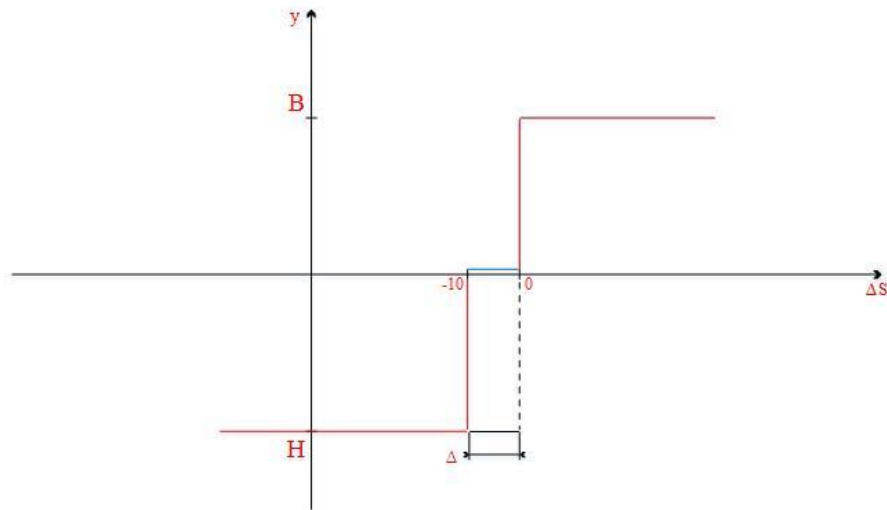


Рисунок 4.12 – Вихідна характеристика релейного регулятора із зоною нечутливості

Якщо різниця становить додатне число, то рух буде відбуватися вперед, тобто відняли від заданого значення зворотній зв'язок. З часом різниця буде зменшуватися поки не буде дорівнювати нулю. Простий релейний регулятор при від'ємній різниці відразу вмикається зворотну сторону але тоді буде виникати не стійкий режим роботи тому, що є інерція і за рахунок неї пристрій може переїжджати, і тому для цього роблять зону нечутливості (на графіку позначена синім кольором). Тобто в цій зоні він буде вимкнений, навіть якщо прилад трішки переїде, наприклад -5, то нічого не відбудеться тому, що в цій зоні він буде вимкнений, і вже коли ця різниця буде досить велика, більша чим -10 то потім відбувається рух в іншу сторону.

Також це можливо коли міняється задане значення. Коли зворотній зв'язок більший за задане значення знову відбувається рух у відповідну сторону і різниця починає зменшуватися, і в той момент коли вони становить -10 прилад знову вмикається.

Варто сказати, що чим більша величина Δ , тим більша стійкість але недолік, що це абсолютна похибка, тобто:

На рисунку 4.13 показано відпрацювання стрибкоподібного сигналу.

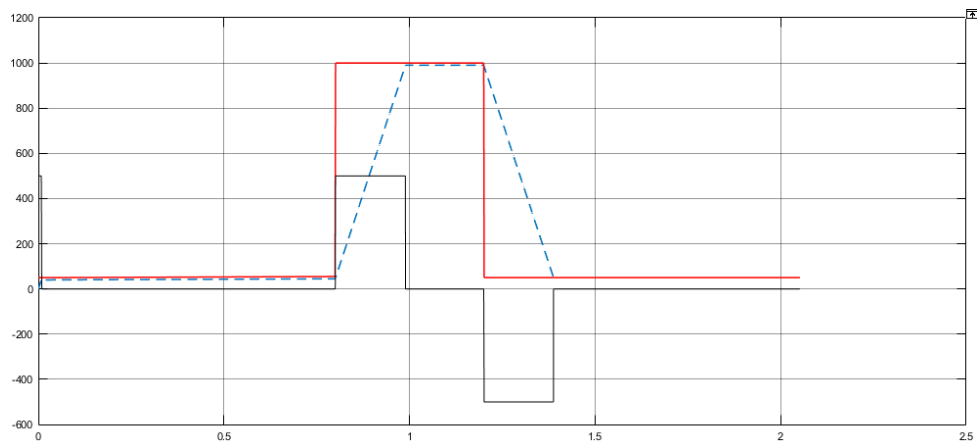


Рисунок 4.13 – Відпрацювання тестового стрибкоподого сигналу

На рисунку 4.14 показано відпрацювання змінного сигналу задання

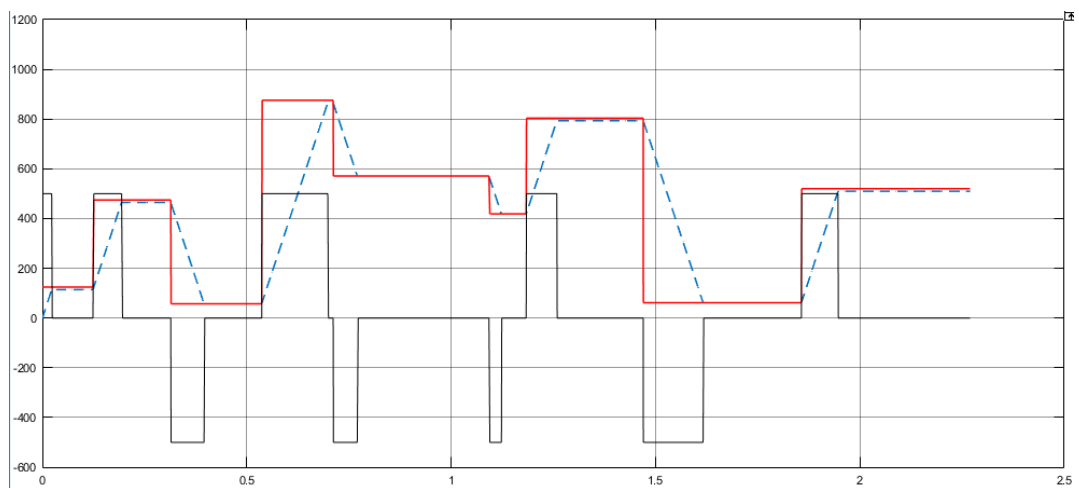


Рисунок 4.14 – Відпрацювання змінного сигналу задання

На рисунку 4.15 наведено графік відпрацювання координати для синусоїди при нульовому початковому положенні.

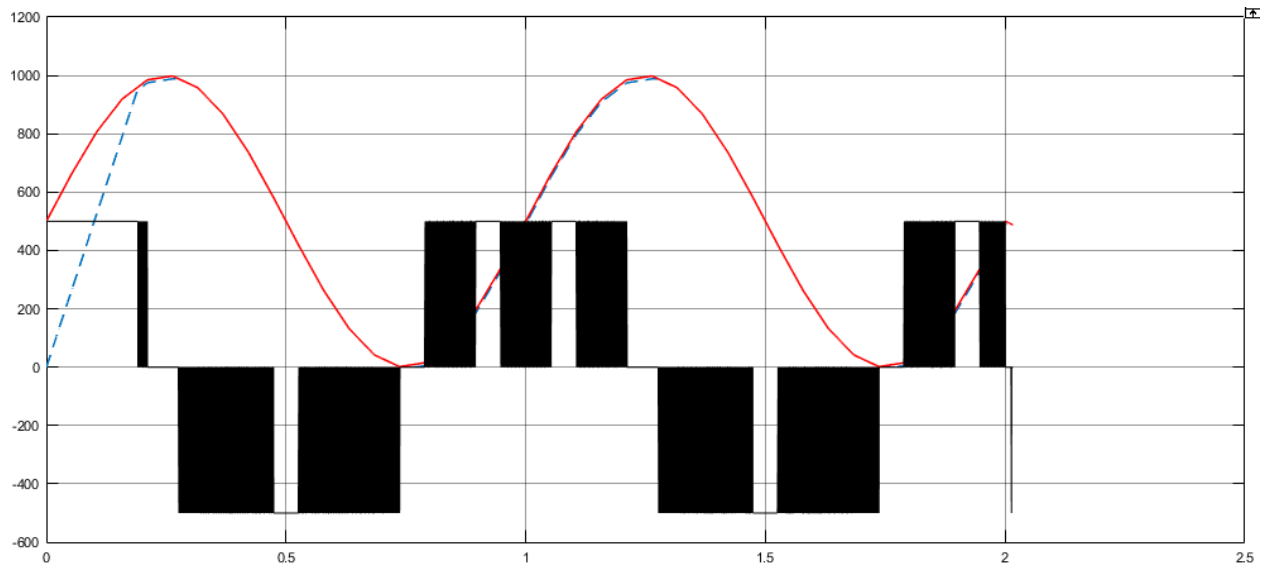


Рисунок 4.15 – Відпрацювання синусоїдального сигналу при нульовому початковому положенні

Як видно із графіка 4.15 неперервна модель малопридатна для відпрацювання неперервного сигналу задання.

ВИСНОВКИ

Здійснено комплекс робіт по модернізації стенда для дослідження електропривода позиціювання (сервопривода) з використанням енкодерів на базі мікроконтролерної платформи Arduino, яка виконує функції пристрою керування.

Основну роль відіграє можливість розширення функціональних можливостей стенда за рахунок введення інформаційно-вимірювальної системи, яка дозволяє знімати статичні та динамічні характеристики роботи електропривода позиціювання.

Розроблено програмне забезпечення не дозволяє зберігати налаштування та поточну позицію в енергонезалежній пам'яті, що призводить відсутності помилок при вимкненні чи скиданні контролера. Програма керуючого контролера передбачає зміну налаштувань (швидкість переміщення, зона нечутливості, параметри регулятора) окрім заданого значення координати. Розроблено механізм автоматичного калібрування та пошуку нульової точки. Синтезовано комп'ютерні моделі для віддаленого налаштування програмного та апаратного забезпечення стенда.

В роботі в повному обсязі проведено весь комплекс проектних робіт, зокрема розроблено структурну та електричну принципову схему лабораторної установки, а також був описаний їх принцип роботи, та розраховано і вибрано елементи для даної схеми. Було проведено вибір мікроконтролера для системи керування. Був розроблений алгоритм програми, яка програмується в мікроконтролер на основі алгоритму була розроблена програма.

Результати роботи засвідчені практичною реалізацією та апробацією роботи на стенді. Даний стенд буде корисним під час вивчення курсу

«Спеціальні електричні машини» для студентів освітньої програми «Електромеханіка»

ПЕРЕЛІК ПОСИЛАНЬ

1. Алгоритм роботи енкодерів. – [Електронний ресурс]. – Режим доступу: <https://tutorial.cytron.io/2012/01/17/quadrature-encoder/>;
2. Arduino IDE – [Електронний ресурс]. - Режим доступу: https://biblprog.org.ua/ua/arduino_ide/;
3. Arduino UNO. – [Електронний ресурс]. - Режим доступу: <https://arduinomaster.ru/platy-arduino/plata-arduino-uno/>;
4. LCD 1602. – [Електронний ресурс]. - Режим доступу: <http://robotchip.ru/obzor-lcd-displeya-1602a/>;
5. Драйвер L298N. – [Електронний ресурс]. - Режим доступу: <https://arduinomaster.ru/datchiki-arduino/drajver-dvigatelya-i-motor-shield-arduino/>;
6. Система керування серводвигуном - [Електронний ресурс]. - Режим доступу: http://util.oem.se/pdf/Electromen_EM-115_DC_motor_control_1159756-500630.pdf;
7. Технічні характеристики серводвигуна. – [Електронний ресурс]. - Режим доступу: <https://www.printerbrain.xyz/epson-dfx-8000/carriage-motor-controldrive-circuit.html>;
8. Інкрементний енкодер. – [Електронний ресурс]. - Режим доступу: <http://www.servotechnica.ru/catalog/type/index.pl?id=138>;
9. Абсолютний енкодер. – [Електронний ресурс]. - доступу <http://www.servotechnica.ru/catalog/type/index.pl?id=139>;
10. Коваль, С. В. Arduino: проектування та програмування мікроконтролерних систем [Текст] / С. В. Коваль, І. М. Чорнобай. – Львів: Новий Світ – 2000, 2020. – 368 с.

Додаток А

Лістинг програми мікроконтролера

```
//*****
// Підключення бібліотек та ініціалізація змінних
//*****
#include <EEPROM.h>
#include<util/delay.h>
#include<LiquidCrystal.h>
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);
int stepStatus = 0, adc_key_in = 0, lcd_key = 0;
byte sensorStatusA = 0;
byte oldSensorStatusA = 0;
byte sensorStatusB = 0, Menu = 0, count;
unsigned long dutyCycle = 0;
signed long setPoint = 0, setPoint_D = 0;
signed long actualPoint = 0;
signed long readByte = 0;
unsigned long millisNow = 0;
byte MIN_DUTYCYCLE; // Значення мінімальної ШІМ (60)
byte MAX_DUTYCYCLE; // Значення максимальної ШІМ (180)
byte STEP_MARGIN; //Зона нечутливості (4)
byte enable_write_position = 0, enable_position = 0, enable_home;
//*****
// Установки змінних і налаштувань
//*****
#define btnRIGHT 0// коди кнопки вправо
#define btnUP 1// коди кнопки вверх
#define btnDOWN 2// коди кнопки вниз
#define btnLEFT 3// коди кнопки вліво
#define btnSELECT 4// коди кнопки вибір
#define btnNONE 5// пусто
#define SENSOR_A 3 // піни підключення енкодера
#define SENSOR_B 2
#define MOTOR_DIRECTION 10 //ШІМ піни підключення драйвера
двигуна
#define MOTOR_PWM 11
#define HOME_SW 12 //Кінцевий вимикач
#define P_FRACTION 1.9 //Пропорційний коефіцієнт 0.001 - 10.0 (1.5)
#define CALIBR 96 // кількість імпульсів на 1 мм
//*****
```

```

//          Функція зчитування кнопок
//*****
int read_LCD_buttons()
{
    adc_key_in = analogRead(0);
    if (adc_key_in > 1000) return btnNONE;
    if (adc_key_in < 50)  return btnRIGHT;
    if (adc_key_in < 250) return btnUP;
    if (adc_key_in < 450) return btnDOWN;
    if (adc_key_in < 650) return btnLEFT;
    if (adc_key_in < 850) return btnSELECT;
    return btnNONE;
}
//*****

//          Функція запису long в eeprom
//*****
//This function will write a 4 byte (32bit) long to the eeprom at
//the specified address to address + 3.
void EEPROMWritelong(int address, long value)
{
    byte four = (value & 0xFF);
    byte three = ((value >> 8) & 0xFF);
    byte two = ((value >> 16) & 0xFF);
    byte one = ((value >> 24) & 0xFF);
    //Write the 4 bytes into the eeprom memory.
    EEPROM.write(address, four);
    EEPROM.write(address + 1, three);
    EEPROM.write(address + 2, two);
    EEPROM.write(address + 3, one);
}
//*****

//          Функція читання long з eeprom
//*****
//This function will return a 4 byte (32bit) long from the eeprom
//at the specified address to address + 3.
long EEPROMReadlong(long address)
{
    //Read the 4 bytes from the eeprom memory.
    long four = EEPROM.read(address);
    long three = EEPROM.read(address + 1);
    long two = EEPROM.read(address + 2);
    long one = EEPROM.read(address + 3);
    //Return the recomposed long by using bitshift.

```



```

    return ((four << 0) & 0xFF) + ((three << 8) & 0xFFFF) + ((two << 16) &
0xFFFFFFFF) + ((one << 24) & 0xFFFFFFFF);
}
//*****
//Функція установок
//*****
void setup() {
    pinMode(HOME_SW, INPUT_PULLUP);// вхід кінцевого вимикача
+5В
    MIN_DUTYCYCLE = EEPROM.read(1); // читаем настройки
    MAX_DUTYCYCLE = EEPROM.read(2); // читаем настройки
    STEP_MARGIN = EEPROM.read(3); // читаем настройки
    actualPoint = EEPROMReadlong(4);// читаем настройки початкової
позиції
    setPoint = actualPoint;// задане значення = поточному збереженому
    setPoint_D = setPoint / CALIBR;
    analogWrite(MOTOR_PWM, 0);// Зупинка двигуна
    delay(100);
    attachInterrupt(0, change_sensor_A, CHANGE);// Вмикання переривань
для сигналів енкодера
    attachInterrupt(1, change_sensor_B, CHANGE);
    sensorStatusA = digitalRead(SENSOR_A);//Початкові рівні сигналів
енкодера
    sensorStatusB = digitalRead(SENSOR_B);
    pinMode(MOTOR_DIRECTION, OUTPUT);// Налаштування входів-
виходів
    pinMode(MOTOR_PWM, OUTPUT);
    analogWrite(MOTOR_PWM, 0);
    analogWrite(MOTOR_DIRECTION, 0);
    lcd.begin(16, 2);          // Запуск та виведення на екран
    lcd.setCursor(0, 0);
    lcd.print(" SERVO MOTOR "); //інформаційного повідомлення
    lcd.setCursor(1, 1);
    lcd.print("CONTROL SYSTEM");
    delay (3000); // на 3 секунди
    lcd.clear();
}
//*****
// Функція визначення положення ротора
//*****
void change_step() {
    sensorStatusA = digitalRead(SENSOR_A);
    if ((oldSensorStatusA == LOW) && (sensorStatusA == HIGH)) {

```

```

    if (sensorStatusB == 0) {
        actualPoint++;
    }
    else {
        actualPoint--;
    }
}
oldSensorStatusA = sensorStatusA;
}
//*****
void change_sensor_A() { // Вимірювання за сигналом А 1 раз на період
    change_step();
}
//*****
void change_sensor_B() { // За сигналом В тільки поточне зчитування
    sensorStatusB = digitalRead(SENSOR_B);
}
//*****
//          Основний цикл
//*****
void loop() {
    lcd_key = read_LCD_buttons(); // перебираємо меню при натисканні
    кнопки SELECT
    if (lcd_key == btnSELECT) {
        Menu++;
        lcd.clear();
    }
    if (Menu != 0) LCD_out(); //Якщо обрано не основне меню вивод
    прискорений
    if (Menu == 0 && (millis() - millisNow) > 500) { // На головному меню
    вивод раз в 0,5с.
        millisNow = millis();
        LCD_out();
    }
    //Визначення напруги що подається на двигун
    //*****
    dutyCycle      =      (double)(abs(setPoint      -      actualPoint))      *
    (double)P_FRACTION; // П-регулювання
    if (dutyCycle < MIN_DUTYCYCLE) { // Мінімальне обмеження
    керуючого сигналу
        dutyCycle = MIN_DUTYCYCLE;
    }
    if (dutyCycle > MAX_DUTYCYCLE) { // Максимальне обмеження

```

```

керуючого сигналу
    dutyCycle = MAX_DUTYCYCLE;
}
// Якщо поточне і задане значення зрівноважені
//*****
    if (abs(setPoint - actualPoint) < STEP_MARGIN) { //Обмеження по
чутливості керуючого сигналу
        analogWrite(MOTOR_PWM, 0); // Зупиняємо двигун
        analogWrite(MOTOR_DIRECTION, 0);
        count++; // якщо відрацьована введена позиція та пройшов час
        if (enable_write_position == 1 && count == 10) {
            EEPROMWritelong(4, actualPoint); // записуємо поточну позицію
            count = 0;
            enable_write_position = 0;
        }
        if (enable_home == 1) { // якщо включено пошук поч. точки
            while (digitalRead(HOME_SW) == 0) { // рухаємось до нуля
                analogWrite(MOTOR_DIRECTION, 0); //поки не спрацює кінцевик
                analogWrite(MOTOR_PWM, 120); // на пониженій швидкості
            }
            actualPoint = 0; // все зануляємо
            setPoint = 0;
            setPoint_D = 0;
            enable_home = 0;
            EEPROMWritelong(4, actualPoint); // і записуємо
            lcd.clear();
        }
    }
//*****
else {
    if (actualPoint < setPoint) { // проста реверсивна логіка
        analogWrite(MOTOR_DIRECTION, dutyCycle);
        analogWrite(MOTOR_PWM, 0);
    }
    if (actualPoint > setPoint) {
        analogWrite(MOTOR_DIRECTION, 0);
        analogWrite(MOTOR_PWM, dutyCycle);
    }
}
//*****
//          Виведення повідомлень на екрані
//*****

```

```

void LCD_out() {
    // Головне меню
    /*******
    if (Menu == 0) {
        if (enable_position) {
            setPoint = setPoint_D * CALIBR;
            enable_position = 0;
        }
        lcd.setCursor(0, 0);
        lcd.print("SP="); // Виведення заданої кількості імпульсів
        lcd.setCursor(3, 0);
        lcd.print(setPoint);
        if (setPoint / 10 == 0) {
            lcd.setCursor(4, 0);
            lcd.print(" ");
        }
        if (setPoint / 100 == 0) {
            lcd.setCursor(5, 0);
            lcd.print(" ");
        }
        lcd.setCursor(9, 0);
        lcd.print("(inp)");
        lcd.setCursor(0, 1);
        lcd.print("АСР="); // Виведення поточного положення в імпульсах
        lcd.setCursor(4, 1);
        lcd.print(abs(actualPoint));
        if (actualPoint / 10 == 0) {
            lcd.setCursor(5, 1);
            lcd.print(" ");
        }
        if (actualPoint / 100 == 0) {
            lcd.setCursor(6, 1);
            lcd.print(" ");
        }
        if (actualPoint / 1000 == 0) {
            lcd.setCursor(7, 1);
            lcd.print(" ");
        }
        lcd.setCursor(9, 1);
        lcd.print("(inp)");
        delay (250);
    }
    // Меню задання заданого значення

```

```

//*****
if (Menu == 1) {
    lcd.setCursor(0, 0);
    lcd.print(" SETPOINT SERVO "); //Повідомлення
    lcd.setCursor(2, 1);
    lcd.print("SP=");
    lcd.setCursor(11, 1);
    lcd.print("(mm)");
    lcd.setCursor(6, 1);
    lcd.print(setPoint_D);
    if (setPoint_D / 10 == 0) {
        lcd.setCursor(7, 1);
        lcd.print(" ");
    }
    if (setPoint_D / 100 == 0) {
        lcd.setCursor(8, 1);
        lcd.print(" ");
    }
    if (setPoint_D / 1000 == 0) {
        lcd.setCursor(9, 1);
        lcd.print(" ");
    }
    }
    delay (200);
    lcd_key = read_LCD_buttons(); // Керування збільшенням чи
зменшенням уставки
    if (lcd_key == btnUP)setPoint_D++;
    if (lcd_key == btnDOWN)setPoint_D--;
    if (setPoint_D < 0)setPoint_D = 0;// мінімум
    if (setPoint_D > 170)setPoint_D = 170;//максимум
    if (lcd_key == btnRIGHT) {
        Menu = 0;
        enable_write_position = 1;
        enable_position = 1;
        lcd.clear();
    }
}
// Меню налаштування мінімальної швидкості
//*****
if (Menu == 2) {
    lcd.setCursor(0, 0);
    lcd.print(" MIN_DUTYCYCLE"); // print a simple message
    lcd.setCursor(2, 1);
    lcd.print("MiN_N="); // print a simple message

```

```

lcd.setCursor(11, 1);
lcd.print("\n"); // print a simple message
lcd.setCursor(8, 1);
lcd.print(MIN_DUTYCYCLE);
if (MIN_DUTYCYCLE / 10 == 0) {
    lcd.setCursor(9, 1);
    lcd.print(" ");
}
if (MIN_DUTYCYCLE / 100 == 0) {
    lcd.setCursor(10, 1);
    lcd.print(" ");
}
delay (150);
lcd_key = read_LCD_buttons(); // read the buttons
if (lcd_key == btnUP) MIN_DUTYCYCLE = MIN_DUTYCYCLE + 1;
if (lcd_key == btnDOWN) MIN_DUTYCYCLE = MIN_DUTYCYCLE - 1;
if (MIN_DUTYCYCLE < 20) MIN_DUTYCYCLE = 20;
if (MIN_DUTYCYCLE > 255) MIN_DUTYCYCLE = 254;
if (lcd_key == btnRIGHT) {
    EEPROM.write(1, MIN_DUTYCYCLE);
    lcd.clear();
    lcd.print(" MINIMUM_PWM "); // print a simple message
    lcd.setCursor(5, 1);
    lcd.print("SAVED"); // print a simple message
    delay (2000);
    Menu = 0;
    lcd.clear();
}
}
// Меню налаштування максимальної швидкості
//*****
if (Menu == 3) {
    lcd.setCursor(0, 0);
    lcd.print(" MAX_DUTYCYCLE"); // print a simple message
    lcd.setCursor(2, 1);
    lcd.print("MAX_N= "); // print a simple message
    lcd.setCursor(12, 1);
    lcd.print("\n"); // print a simple message
    lcd.setCursor(9, 1);
    lcd.print(MAX_DUTYCYCLE);
    if (MAX_DUTYCYCLE / 10 == 0) {
        lcd.setCursor(10, 1);
        lcd.print(" ");
    }
}

```

```

    }
    if (MAX_DUTYCYCLE / 100 == 0) {
        lcd.setCursor(11, 1);
        lcd.print(" ");
    }
    delay (300);
    lcd_key = read_LCD_buttons(); // read the buttons
    if (lcd_key == btnUP) MAX_DUTYCYCLE = MAX_DUTYCYCLE + 1;
    if (lcd_key == btnDOWN) MAX_DUTYCYCLE = MAX_DUTYCYCLE -
1;
    if (MAX_DUTYCYCLE < 20) MAX_DUTYCYCLE = 100;
    if (MAX_DUTYCYCLE > 255) MAX_DUTYCYCLE = 255;
    if (lcd_key == btnRIGHT) {
        EEPROM.write(2, MAX_DUTYCYCLE);
        lcd.clear();
        lcd.print(" MAXIMUM_PWM "); // print a simple message
        lcd.setCursor(5, 1);
        lcd.print("SAVED"); // print a simple message
        delay (2000);
        Menu = 0;
        lcd.clear();
    }
}
// Меню налаштування зони нечутливості
//*****
if (Menu == 4) {
    lcd.setCursor(0, 0);
    lcd.print(" STEP_MARGIN "); // print a simple message
    lcd.setCursor(2, 1);
    lcd.print("STEPN= "); // print a simple message
    lcd.setCursor(12, 1);
    lcd.print("(n)"); // print a simple message
    lcd.setCursor(9, 1);
    lcd.print(STEP_MARGIN);
    if (STEP_MARGIN / 10 == 0) {
        lcd.setCursor(10, 1);
        lcd.print(" ");
    }
    if (STEP_MARGIN / 100 == 0) {
        lcd.setCursor(11, 1);
        lcd.print(" ");
    }
    delay (300);

```

```

    lcd_key = read_LCD_buttons(); // read the buttons
    if (lcd_key == btnUP)STEP_MARGIN = STEP_MARGIN + 1;
    if (lcd_key == btnDOWN)STEP_MARGIN = STEP_MARGIN - 1;
    if (STEP_MARGIN < 1)STEP_MARGIN = 1;
    if (STEP_MARGIN > 255)STEP_MARGIN = 255;
    if (lcd_key == btnRIGHT) {
        EEPROM.write(3, STEP_MARGIN);
        lcd.clear();
        lcd.print(" STEP_MARGIN"); // print a simple message
        lcd.setCursor(5, 1);
        lcd.print("SAVED"); // print a simple message
        delay (2000);
        Menu = 0;
        lcd.clear();
    }
}
// Меню для налаштування нульової точки
//*****
if (Menu == 5) {
    lcd.setCursor(0, 0);
    lcd.print(" HOME_POSITION "); // print a simple message
    lcd.setCursor(2, 1);
    lcd.print(" PRESS LEFT"); // print a simple message
    delay (300);
    lcd_key = read_LCD_buttons(); // read the buttons
    if (lcd_key == btnLEFT) {
        enable_home = 1;
        Menu = 0;
        lcd.clear();
    }
}
//*****
if (Menu == 6) {
    Menu = 1;
    lcd.clear();
    delay (200);
}
}

```