

Міністерство освіти і науки України
Національний університет водного господарства та
природокористування

Навчально-науковий механічний інститут
Кафедра будівельних, дорожніх та меліоративних машин

02-01-562М

МЕТОДИЧНІ ВКАЗІВКИ

до виконання практичних робіт та самостійної роботи
з дисципліни: **«Маніпулятори та робототехніка в
машинобудуванні»** для здобувачів вищої освіти першого
(бакалаврського) рівня за освітньо-професійною програмою
«Створення та експлуатація машин і обладнання» спеціальності
133 «Галузеве машинобудування» усіх форм навчання

Рекомендовано науково-
методичною радою з якості ННМІ
Протокол № 2 від 02 жовтня 2024 р.

Рівне – 2024

Методичні вказівки до виконання практичних робіт з дисципліни: «Маніпулятори та робототехніка в машинобудуванні» для здобувачів вищої освіти першого (бакалаврського) рівня за освітньо-професійною програмою «Створення та експлуатація машин і обладнання» спеціальності 133 «Галузеве машинобудування» денної та заочної форм навчання. [Електронне видання] / Серілко Д. Л., Голотюк М. В., Реут Д. Т. – Рівне : НУВГП, 2024. – 52 с.

Укладачі:

Серілко Д. Л. – доцент кафедри будівельних, дорожніх та меліоративних машин, к.т.н., доцент;

Голотюк М. В. – доцент кафедри агроінженерії, к.т.н., доцент;

Реут Д. Т. – доцент кафедри автоматизації, електротехнічних та комп'ютерно-інтегрованих технологій, к.т.н.

Методичні вказівки схвалено на засіданні кафедри будівельних, дорожніх та меліоративних машин

Протокол № 2 від 17 вересня 2024 року

Відповідальний за випуск: Тхорук Є. І. к.т.н., доцент, в.о. завідувача кафедри будівельних, дорожніх та меліоративних машин.

Керівник групи забезпечення спеціальності: Тхорук Є. І., к.т.н., доцент, в.о. завідувача кафедри будівельних, дорожніх та меліоративних машин.

Попередні версії

МВ 02-01-511М

МВ 02-06-10М

© Д. Л. Серілко,

М. В. Голотюк,

Д. Т. Реут, 2024

© НУВГП, 2024

ЗАГАЛЬНІ ПОЛОЖЕННЯ

Дисципліна «Маніпулятори та робототехніка в машинобудуванні» є невід’ємним складником формування професійної компетентності студентів. Програма дисципліни передбачає комплексне вивчення робототехніки в машинобудуванні.

Маніпулятори та робототехніка в машинобудуванні відносно самостійна дисципліна, яка дає загальне уявлення про розвиток техніки. Даний курс покликаний сприяти формуванню у студентів технічних спеціальностей загальної картини розвитку інженерної справи як цілісного процесу, який відбувається закономірно і проходить в органічному взаємозв’язку і взаємодії з історією суспільства.

Курс «Маніпулятори та робототехніка в машинобудуванні» дозволить отримати знання в обсязі, достатньому для самостійного вирішення конструкторських та виробничо-технологічних завдань в галузі конструювання, проектування та сервісного обслуговування робототехнічних систем та комплексів, призначених для автоматизації виробничих (технологічних) процесів. Отримані вміння дозволяють використовувати інженерні методики, аналітичні та числові методи розрахунку для аналізу відомих та розробки нових механізмів, вузлів та комплексів обладнання робототехнічних систем.

Мета навчальної дисципліни «Маніпулятори та робототехніка в машинобудуванні» є надання студентам знань і навиків в області розвитку та використання роботів і роботизованих технологічних комплексів (РТК), ознайомлення з методикою вибору та проектування РТК для вирішення технологічних задач виробництва..

У результаті вивчення навчальної дисципліни «Маніпулятори та робототехніка в машинобудуванні» студент повинен *знати*: типові та сучасні конструкції промислових роботів, принципи їх функціонування та сфери використання, переваги і недоліки; інженерні методики розрахунку та особливості проектування промислових роботів.

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ ДО ВИКОНАННЯ ПРАКТИЧНИХ ЗАНЯТЬ

Практична робота 1. Основи програмування в середовищі Arduino IDE. Вивчення роботи з вхідними та вихідними дискретними сигналами

Мета роботи: ознайомитися з апаратною та програмною частинами середовища Arduino. Вивчити основи роботи з аналоговими і дискретними сигналами в Arduino IDE.

Теоретичні відомості

Arduino – це відкрита електронна платформа, що базується на простому у використанні апаратному та програмному забезпеченні.

Апаратна частина включає мікроконтролерну плату, а програмна – середовище Arduino IDE. Платформа Arduino є відкритою, тому існує ряд аналогів Arduino-сумісних мікроконтролерних плат різного виконання.

Arduino Uno

Arduino Uno – це пристрій на основі мікроконтролера ATmega328P. До його складу входить все необхідне для зручної роботи з мікроконтролером: 14 цифрових входів/виходів (з них 6 можуть використовуватися в якості ШІМ-виходів), 6 аналогових входів, кварцовий резонатор на 16 МГц, роз'єм USB, роз'єм живлення, роз'єм для внутрішньосхемного програмування (ICSP) і кнопка скидання. Для початку роботи з пристроєм достатньо просто подати живлення від АС/DC-адаптера або батарейки, або підключити його до комп'ютера за допомогою USB-кабелю.

Входи і виходи

Кожен з 14 цифрових виводів може працювати в якості входу або виходу. Рівень напруги на виводах обмежений 5В. Максимальний струм, який може віддавати або споживати один вивід, становить 40 мА. Усі виводи з'єднані з внутрішніми підтягуючими резисторами (за замовчуванням відключеними) номіналом 20-50 кОм. Крім цього, деякі виводи Arduino Uno можуть виконувати додаткові функції:

Послідовний інтерфейс: виводи 0 (RX) і 1 (TX). Використовуються для отримання (RX) і передачі (TX) даних по послідовному інтерфейсу. Ці виводи з'єднані з відповідними виводами мікросхеми ATmega8U2, що виконує роль перетворювача USB/UART.

Зовнішні переривання: виводи 2 і 3. Можуть служити джерелами переривань, що виникають при фронті, спаді або при низькому рівні сигналу на цих висновках.

ШИМ: виводи 3, 5, 6, 9, 10 і 11. За допомогою функції analogWrite() можуть виводити 8-бітові аналогові значення у вигляді ШИМ-сигналу.

Інтерфейс SPI: виводи 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Із застосуванням бібліотеки SPI дані виводи можуть здійснювати зв'язок по інтерфейсу SPI.

Світлодіод: 13. Вбудований світлодіод, приєднаний до виводу 13.

В Arduino Uno є 6 аналогових входів (A0 - A5), на кожен з яких можна подати напругу в межах 0 – 5В та отримати аналогово-цифрове перетворення вхідного сигналу у вигляді 10-бітного числа (1024 різних значення). Верхню межу вхідної напруги можна змінити, використовуючи вивід AREF і функцію analogReference().

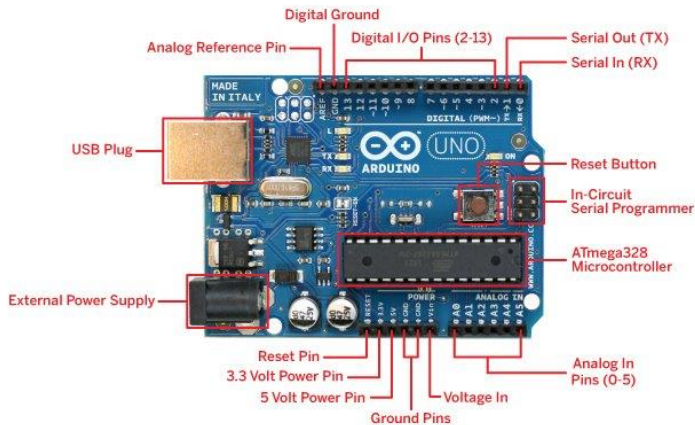


Рис. 1.1. Зовнішній вигляд та призначення виводів плати Arduino Uno

Таблиця 1.1
Основні характеристики Arduino Uno

Мікроконтролер	ATmega328
Робоча напруга	5V
Вхідна напруга (рекомендована)	7-12V
Дискретні (цифрові) входи/виходи	14 (6 із них можуть використовуватися як ШІМ-виходи)
Аналогові входи	6
Максимальний струм одного вивода	40 mA
Максимальний струм вивода 3,3В	50 mA
Флеш-пам'ять	32 KB (ATmega328) з яких 0.5 KB використовуються завантажувачем
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Тактова частота	16 MHz

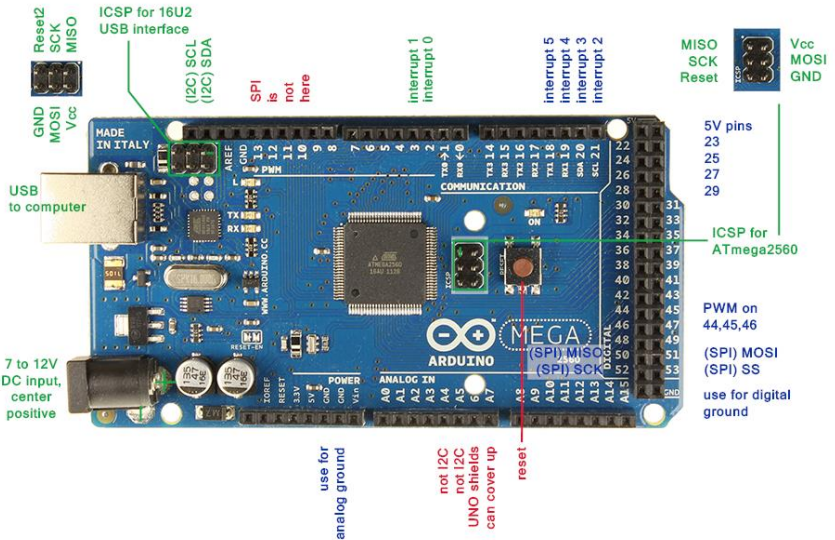


Рис. 1.2. Зовнішній вигляд та призначення виводів плати Arduino Mega 2560

Arduino Mega 2560

Arduino Mega 2560 – це пристрій на основі мікроконтролера ATmega2560.

Основні характеристики плати Arduino Mega 2560 наведені в табл. 1.2.

Таблиця 1.2

Основні характеристики Arduino Mega 2560

Мікроконтролер	ATmega2560
Робоча напруга	5V
Вхідна напруга (рекомендована)	7-12V
Дискретні (цифрові) входи/виходи	54 (15 із них можуть використовуватися як ШІМ-виходи)
Аналогові входи	16
Максимальний струм одного вивода	40 mA
Максимальний струм вивода 3,3В	50 mA
Флеш-пам'ять	256 КВ з яких 8 КВ використовуються авантажувачем
SRAM	8 КВ
EEPROM	4 КВ
Тактова частота	16 MHz

Програмування Arduino

Програмування мікроконтролерних плат Arduino здійснюється у вільно розповсюджваному середовищі Arduino IDE.

Мікроконтролери в платах Arduino випускаються з прошитим завантажувачем (bootloader), що дозволяє завантажувати в

мікроконтролер нові програми без необхідності використання зовнішнього програматора. Взаємодія з ним здійснюється за оригінальним протоколом STK500.

Також мікроконтролер можна прошити і через роз'єм для внутрішньосхемного програмування ICSP (In-Circuit Serial Programming), не звертаючи уваги на завантажувач.

Мова програмування Arduino – це видозмінена мова C/C++ із розширеним набором функцій.

Кожна програма для пристроїв Arduino містить дві основні функції:

1) void setup() – функція, що виконується о дин раз, після кожної подачі живлення або скидання плати Arduino;

2) void loop() – виконується постійно в циклі після виконання функції void setup().

Перед завантаженням програми в мікроконтролер слід спочатку вибрати тип плати Arduino та порт, до якого вона підключена.

Деякі функції для роботи з входами/виходами Arduino

`pinMode(pin, value)` – налаштування виводу з номером `pin` як дискретного входу (`value=INPUT`) або дискретного виходу (`value=OUTPUT`). Наприклад:

```
pinMode(13, OUTPUT);
```

`digitalRead(pin)` – функція зчитує із заданого входу значення HIGH або LOW.

`digitalWrite(pin, value)` – подає на цифровий вхід/вихід значення HIGH або LOW;

`analogRead(pin)` – зчитує величину напруги з аналогового входу і видає результат у вигляді цілого числа від 0 до 1023.

План роботи

1. Ознайомитися з будовою та призначенням елементів плати Arduino Uno.

2. Ознайомитися з принципами програмування в середовищі Arduino IDE.

3. Навчитися підключати датчики з дискретними та аналоговими вихідними сигналами до Arduino та обробляти дані цих датчиків.

Порядок виконання роботи

1. Ознайомитися з теоретичними відомостями.
2. Завантажити середовище Arduino IDE.
3. Ознайомитись зі схемою підключення механічної кнопки (рис. 1.3.). Зібрати її на макетній платі. Використати підтягуючий резистор на 7,5 кОм.

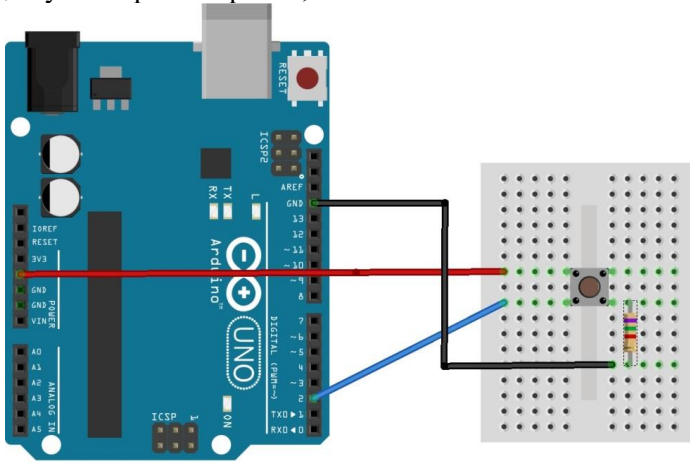


Рис. 1.3. Підключення кнопки до Arduino Uno

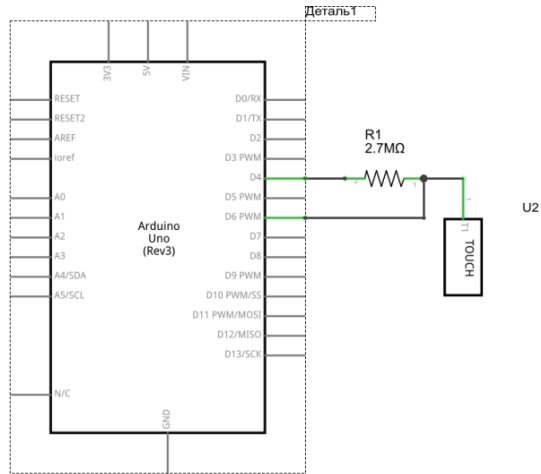
У середовищі Arduino IDE написати програму, що запалюватиме світлодіод при кожному натисканні кнопки.

Скомпілювати програму та переконатися у відсутності помилок.

Підключити мікроконтролерну плату до USB-порта комп'ютера.

У середовищі Arduino IDE вибрати тип плати та порт, до якого вона підключена. Завантажити створену програму в мікроконтролер. Переконатися в коректності роботи програми.

4. Ознайомитися зі схемою підключення ємнісної сенсорної кнопки. Зібрати її на макетній платі. Підключити сенсорну кнопку до 4 (send pin) і 6 (receive pin) дискретних виводів Arduino. Використати резистор 2,5 МОм.



Завантажити бібліотеку Capacitive Sensing

<http://playground.arduino.cc/Main/CapacitiveSensor?from=Main.CapSense> та інстальовати її. Написати програму, що запалюватиме світлодіод при кожному натисканні кнопки.

```
#include <CapacitiveSensor.h>
CapacitiveSensor key1 = CapacitiveSensor(4, 6);
// підключена сенсорна кнопка між 4-м і 6-м пінами
int range = 100; // порогове значення, що
визначає натискання кнопки. Підбирається
експериментально
byte button; // побітове позначення натиснутих
кНОПОК
void setup()
{
  pinMode(13,OUTPUT);
}
void loop()
{
  long total = key1.capacitiveSensor(30);
```

```
button = 0;
  if (total > range) {button=1;}
if (button==1) digitalWrite(13, HIGH);
  else digitalWrite(13, LOW);
}
```

Завантажити програму в мікроконтролер.

Поекспериментувати з чутливістю спрацювання кнопки.

5. Зробити висновки. Звіт повинен містити: титульний лист; тему, мету роботи; порядок виконання; створені програми; висновки.

Контрольні запитання

1. Яка функція налаштовує режим роботи піна на вхід?
2. Яка функція дозволяє зчитати стан цифрового входу?
3. Яка функція дозволяє змінити стан цифрового виходу?

Практична робота 2. Організація зчитування сигналів з датчиків

Мета роботи: навчитися програмувати плату Arduino Uno для зчитування сигналу з датчиків.

Теоретичні відомості

Мікроконтролер в Arduino Uno містить 3 таймери. Вони можуть використовуватись для отримання імпульсного сигналу від датчиків.

Мікроконтролер ATmega328P містить два 8-розрядні таймери та один 16-розрядний. Таймер 1 має регістр захоплення OCR1, куди записується вміст таймера TCNT1 по події захоплення - зміні сигналу на вході ICP1 або виході аналогового компаратора.

Режим захоплення (capture). У режимі захвату 16-ти бітне значення таймера (TCNT1) захоплюється в регістр OCR1 при кожній події на вході ICP1 або виході аналогового компаратора. Подія для захоплення задається в регістрі TCCR1B (біт ICES1) та ACSR (біт ACIC).

Режим захоплення використовується для вимірювання тривалості між двома подіями, наприклад періоду, тривалості імпульсу, скважності і т.д.

Приклад 1. Вимірювання періоду дискретного сигналу (рис. 2.5)

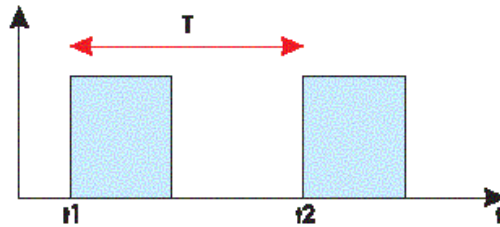


Рис. 2.5. Вимірювання періоду

Приклад 2. Вимірювання періоду з усередненням результату (рис. 2.6)

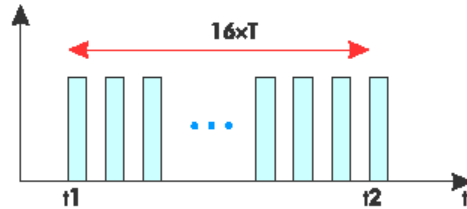


Рис. 2.6. Вимірювання періоду з усередненням

Приклад 3. Вимірювання тривалості імпульсу (рис.2.7)

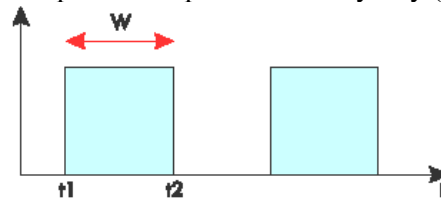


Рис. 2.7. Вимірювання тривалості імпульсу

Приклад 4. Вимірювання скважності імпульсів (рис. 2.8)

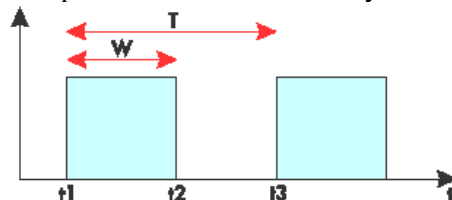


Рис. 2.8. Вимірювання скважності імпульсів

Аналогово-цифровий перетворювач (АЦП, англ. Analog-to-digital converter, ADC) - пристрій, що перетворює вхідний аналоговий сигнал в цифровий код (цифровий сигнал).

У мікроконтролері ATmega328P є 10-розрядний АЦП. Вхідна напруга конвертується в 10-розрядне двійкове значення. Мінімальне значення відповідає 0, а максимальне - опорній напрузі. Отриманий результат перетворення записується в 2 регістра: ADCH і ADCL та повертається функцією analogRead(pin) як число в діапазоні 0...1023. Результат перетворення можна розрахувати за формулою

$$ADC = \frac{V_{in} \cdot 1024}{V_{ref}}$$

В ATmega328P АЦП реалізовано на основі ЦАП та компаратора. На вхід ЦАП подається зростаючий цифровий код, на виході отримуються зростаюча напруга. Коли напруга з виходу ЦАП зрівнюється з вимірюваною напругою з обраного каналу АЦП, процес перетворення можна вважати закінченим та код, що в цей момент був на вході ЦАП, є результатом аналогово-цифрового перетворення. Внаслідок перехідних процесів швидкість АЦП є обмеженою.

План роботи

1. Навчитися підключати датчики з дискретними вихідними сигналами до Arduino та обробляти дані цих датчиків.
2. Навчитися підключати датчики з аналоговими вихідними сигналами до Arduino та обробляти дані цих датчиків.

Порядок виконання роботи

1. Підключити до плати Arduino ультразвуковий датчик вимірювання відстані HC-SR04.



- Вивід Echo датчика підключити до виводу 12 плати Arduino;
вивід Trigger – до виводу 11;
вивід Vcc – до виводу 5 В;
вивід GND – до одного з виводів GND на платі Arduino.

Завантажити в МК наступну програму:

```
#define echoPin 12 // Echo Pin
#define trigPin 11 // Trigger Pin
#define LEDPin 13 // Onboard LED
char rec;

int maximumRange = 200; // максимальна
Відствнь
int minimumRange = 0; // Minimum range
needed
float duration, distance; // Duration used
to calculate distance

void setup() {
  Serial.begin (9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(LEDPin, OUTPUT); // Use LED
indicator (if required)
}

void loop() {
  /* The following trigPin/echoPin cycle is
used to determine the distance of the nearest
object by bouncing soundwaves off of it. */
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);

  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);

  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);

  //Calculate the distance (in cm) based on
the speed of sound.
  distance = duration/58.2;
```

```

    if (distance >= maximumRange || distance <=
minimumRange){
    /* Send a negative number to computer and
Turn LED ON to indicate "out of range" */
    Serial.println("error");
    Serial.print("\n\r");
    digitalWrite(LEDpin, HIGH);
    delay(50);
    }
    else {
    /* Send the distance to the computer using
Serial protocol, and
turn LED OFF to indicate successful
reading. */
    Serial.println(distance);
    Serial.print("\n\r");
    digitalWrite(LEDpin, LOW);
    delay(250);
    }

    //Delay 50ms before next reading.
    delay(150);
    byte l[255];
    byte count=0;
    int i;
    if (Serial.available()>0){
    while(Serial.available()>0){
        digitalWrite(LEDpin, HIGH);
        l[count] = Serial.read();
        count++;}
    for(i=0;i<count;i++){
        Serial.write(l[i]);};
    Serial.println();
    Serial.print("\n\r");
    delay(500);}
}

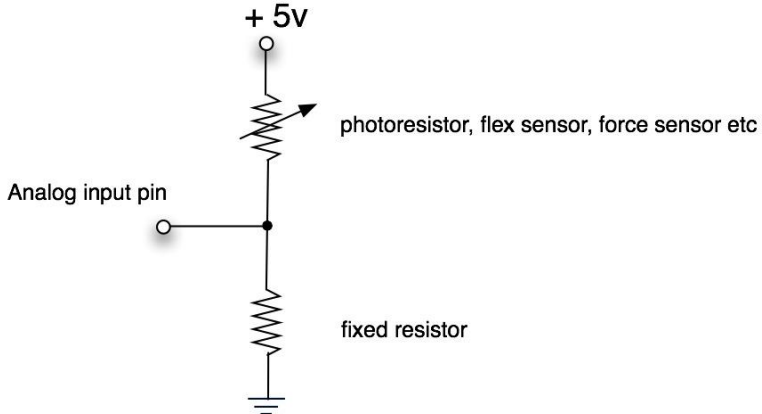
```

Відкрити монітор порту. Піднести руку навпроти давача

на відстані 20 – 50 см. Спостерігати за вимірним значенням відстані, що виводиться на екран.

2. Підключити підстроювальний або змінний резистор за схемою подільника напруги (потенціометра).

Variable resistor connected to the analog input of the arduino



Відповідно до напруги на ковзному контакті (0...5 В) передавати у послідовний інтерфейс значення змінної *norm*, яка зберігає число від 0 до 100, яке прямопропорційно залежить від напруги.

```
int sensorPin = A0;    // аналоговий вхід
int sensorValue = 0;  // значення сигналу
від давача
int norm=0; // нормоване значення сигналу з
давача
void setup() {
  Serial.begin(9600);    // налаштуємо
послідовний порт для зв'язку з ПК
}

void loop() {
  sensorValue = analogRead(sensorPin);    //
зчитуємо значення з давача
  norm=map(sensorValue, 0, 1023, 0, 100); //
```



```

нормуємо сигнал в межах від 0 до 100
Serial.println(norm); // виводимо
нормоване значення сигналу в порт
delay(500); // затримка 0,5 с
}

```

3. Замість змінного резистора підключити до аналогового входу подільник, утворений постійним резистором й давачем освітленості (фоторезистором). Змінюючи освітленість фоторезистора, спостерігати за значенням змінної `norm`.

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. Скільки таймерів має мікроконтролер ATmega328P?
2. Яка розрядність модуля АЦП мікроконтролера ATmega328P?
3. Яка кількість каналів модуля АЦП в Arduino Uno?
4. Якою функцією можна виміряти тривалість імпульсу?
5. Якою функцією можна отримати результат АЦП?

Практична робота 3. Реалізація програмного керування двигуном постійного струму

Мета роботи: ознайомитися зі способами керування швидкістю обертів двигунів постійного струму. Навчитися створювати програми керування швидкістю обертів ДПС для мікроконтролера.

Теоретичні відомості

У мехатронних системах одним з найпоширеніших виконавчих пристроїв є двигун постійного струму (ДПС). Конструкції ДПС відрізняються великою різноманітністю. У цій лабораторній роботі розглядається колекторний ДПС, будову якого схематично показано на рис. 3.1.

Найпростіший колекторний ДПС з двополюсним статором і з двополюсним ротором, складається з одного постійного магніту на статорі, з одного електромагніту з явно вираженими полюсами на роторі (якорі), щітково-колекторного вузла з двома пластинами (ламельями) і двома щітками.

Широтно-імпульсна модуляція

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).

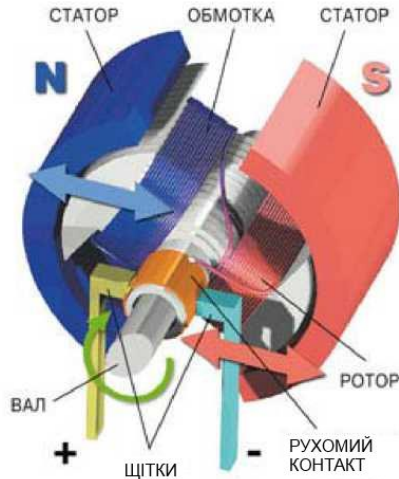


Рис. 3.1. Будова колекторного ДПС з двополюсним статором і двополюсним ротором

Принцип ШІМ пояснюється на рис. 3.2. При ШІМ вихідний сигнал являє собою імпульсний сигнал постійної частоти і змінної скважності. Скважність – це відношення періоду проходження імпульсів T до їх тривалості τ :

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

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

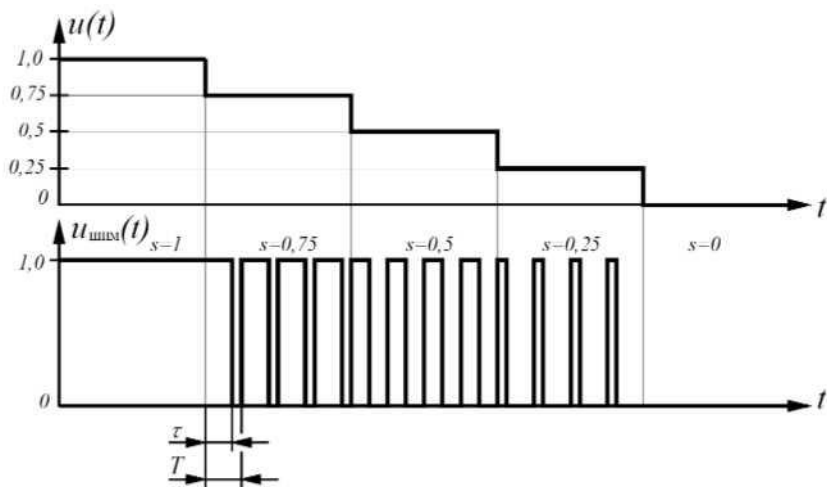
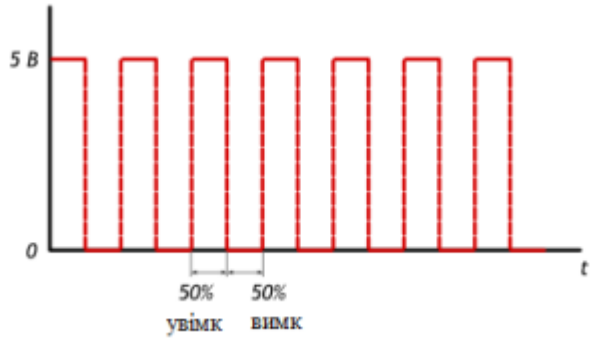


Рис. 3.2. Широтно-імпульсна модуляція

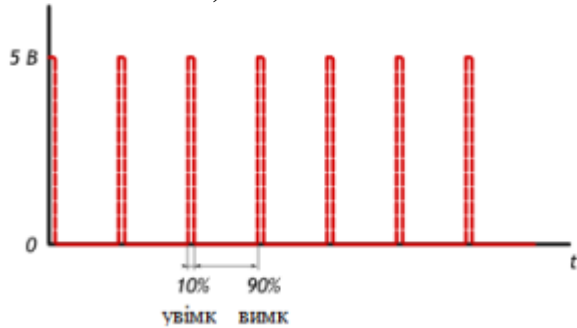
Аналогова ШІМ. ШІМ-сигнал генерується аналоговим компаратором, на один вхід якого подається опорний сигнал значно більшої частоти, ніж модулюючий у вигляді «трикутника» або «пилки», а на іншій – модулюючий безперервний аналоговий сигнал. Частота вихідних імпульсів ШІМ відповідає частоті «зубів» пилки. 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.

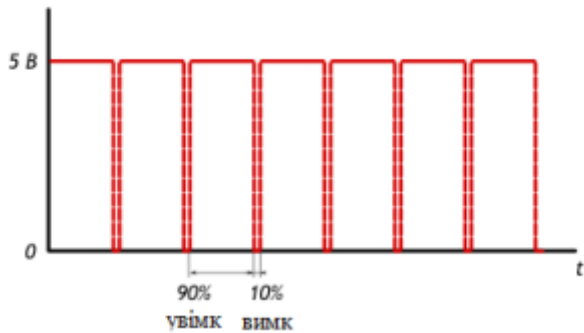
Розглянемо кілька прикладів при напрузі живлення V_{cc} рівній 5 вольтам.
50% — еквівалент 2,5 В



10% — еквівалент 0,5 В



90% — еквівалент 4,5 В



Для подачі 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").

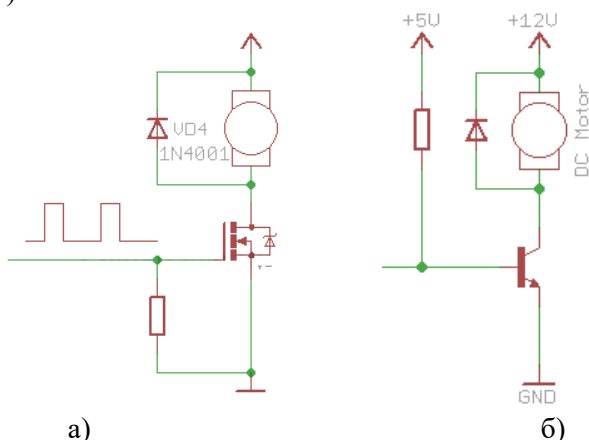


Рис. 3.3. Схема керування швидкістю обертів ДПС з використанням: а) польового транзистора; б) біполярного транзистора і діода

Найпростіша схема керування двигуном постійного

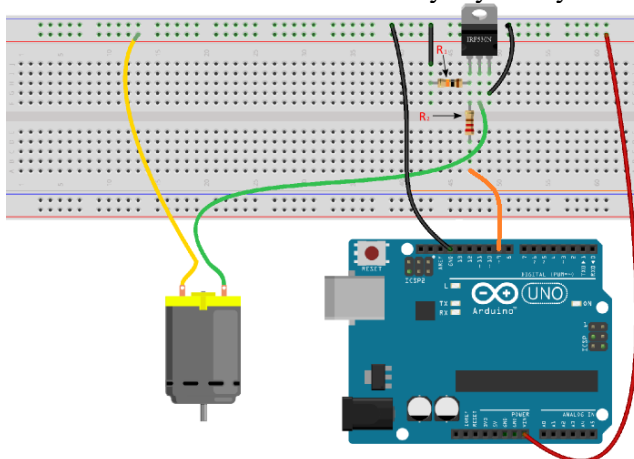
струму складається з польового транзистора, на затвор якого подається ШІМ сигнал. Транзистор в даній схемі виконує роль електронного ключа, який комутує один з виводів двигуна на землю. Транзистор відкривається на момент тривалості імпульсу.

План роботи

1. Ознайомитися зі способами керування швидкістю обертів ДПС.
2. Скласти програму для керування швидкістю обертів ДПС.

Порядок виконання роботи

1. Ознайомитися з теоретичними відомостями.
2. Завантажити середовище Arduino IDE.
3. Скласти на макетній платі наступну схему



Резистор R1 підтягує до "землі" затвор транзистора. Номінал не принциповий - можна використовувати будь-які резистори в діапазоні від 1 до 10 кОм. Резистор R2 служить для захисту виводу мікроконтролера. Діапазон, приблизно, від 10 до 500 Ом.

4. В середовищі Arduino IDE створити нову програму.

```
int led = 9; // ШІМ-вивід, до якого
підключений затвор транзистора
int speed = 0; // ця змінна відповідає за
швидкість обертання
int fadeAmount = 5; // Крок зміни швидкості
```

```

void setup ()
{ // Налаштовуємо вивід 9 як дискретний
вихід
  pinMode (led, OUTPUT);
}
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. Скопіювати програму та переконатися у відсутності помилок.

6. Підключити мікроконтролерну плату до USB-порта комп'ютера.

7. У середовищі Arduino IDE вибрати тип плати та порт, до якого вона підключена. Завантажити створену програму в мікроконтролер. Переконатися в коректності роботи програми.

8. 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.

9. 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?

Практична робота 4. Дослідження роботи сервоприводів та реалізація циклограми

Мета роботи: ознайомитися зі способами керування положенням сервоприводів. Навчитися створювати програми керування швидкістю повороту і положенням вихідного вала сервопривода.

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 давача стало якомога ближче до значення зовнішнього керуючого параметра. Найбільш поширені сервоприводи, які утримують заданий кут і сервоприводи, що підтримують задану швидкість обертання.

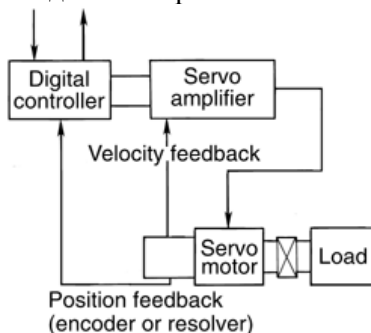


Рис. 4.1. Структурна схема сервопривода з давачем швидкості обертів



Рис. 4.2. Сервоприводи

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

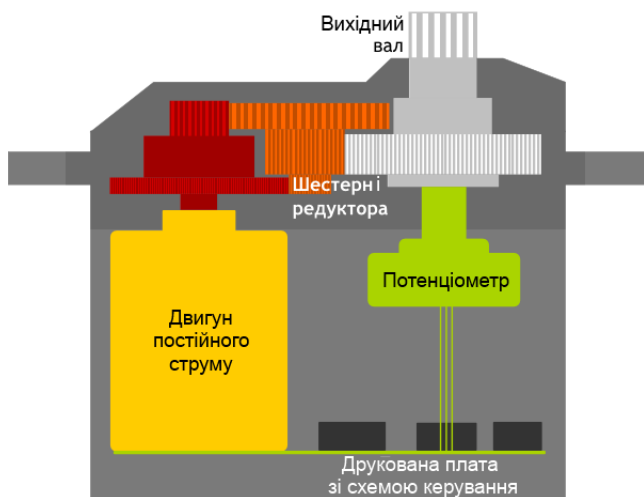


Рис. 4.3. Будова сервопривода

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. При цьому існують верхня і нижня межі тривалості імпульсу.

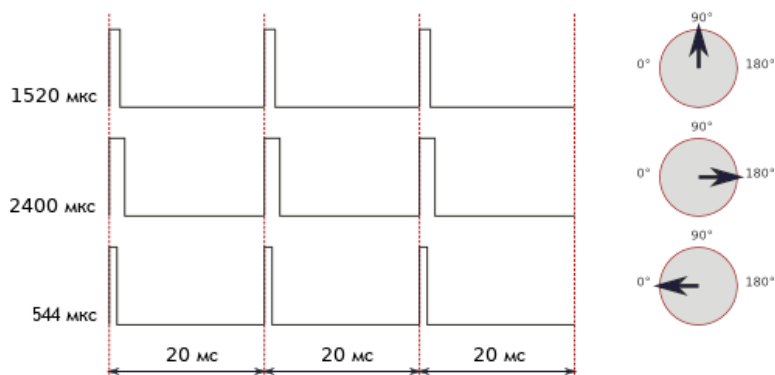


Рис. 4.4. Залежність кута повороту сервопривода від тривалості керуючих імпульсів

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.

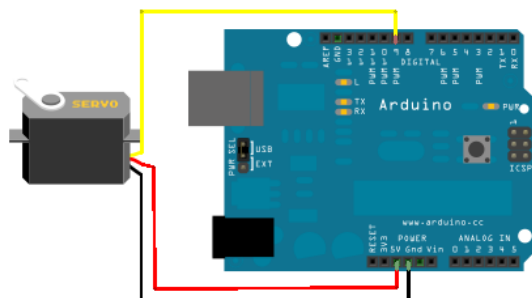


Рис. 4.5. Підключення сервопривода малої потужності до мікроконтролерної плати Arduino Uno

Бібліотека Servo дозволяє здійснювати програмне керування сервоприводами. Управління здійснюється наступними функціями:

`attach()` – приєднує змінну до 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, `min` і `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 сервопривод 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 приєднана змінна 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 змінну від виводу, до якого вона була приписана. Синтаксис наступний: `servo.detach()`.

Приклад програми з використанням бібліотеки

Servo

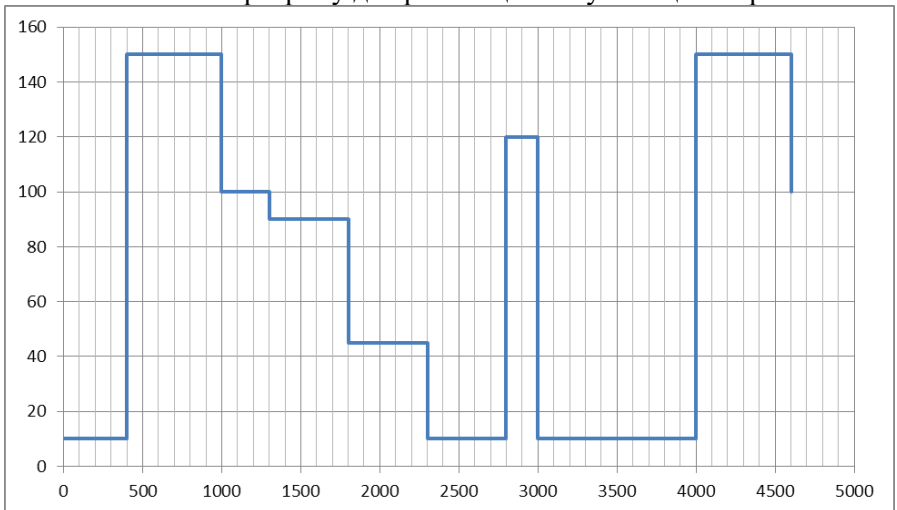
```
#include <Servo.h>
Servo myservo;
void setup()
{
  myservo.attach(9);
}
void loop()
{
  myservo.write(90); // встановлюємо
сервопривод в середнє положення
  delay(500);
  myservo.write(0); // встановлюємо
сервопривод в крайнє леве положення
  delay(500);
  myservo.write(180); // встановлюємо
сервопривод 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. 4.5, використовуючи сервопривод SG90.
2. Завантажте програму з прикладу бібліотеки Servo.
3. Напишіть програму для реалізації наступної циклограми.



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

3. Для того, щоб мати змогу змінювати швидкість обертання використайте бібліотеку VarSpeedServo. Завантажити в МК приклад VarSpeedServo, змінюючи значення швидкості. Переконайтеся в правильності роботи програми та точності позиціонування 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?

Практична робота 5

Тема: Розробка моделі робота-маніпулятора

Мета роботи: Вивчити методику розробки моделі робота-маніпулятора

Теоретичні відомості

Маніпулятор робота за своїм функціональним призначенням повинен забезпечувати рух вихідної ланки і, закріпленого в ньому, об'єкта маніпулювання в просторі по заданій траєкторії і з заданою орієнтацією. Для повного виконання цієї вимоги основний механізм важеля маніпулятора повинен мати не менше шести степеней вільності, причому рух по кожній з них має бути керованим. Робот з шістьма степенями вільності є складною автоматичною системою. Ця система складна як у виготовленні, так і в експлуатації. Тому в реальних конструкціях роботів часто використовуються механізми з числом подвижностей менше шести. Найбільш прості маніпулятори мають три, рідше дві, степені вільності. Такі маніпулятори значно дешевше у виготовленні та експлуатації, але висувають специфічні вимоги до організації робочого середовища.

Модель маніпулятора складена на основі аналізу існуючих конструкцій побутових роботів. Основним режимом роботи маніпулятора є безперервне стеження за об'єктом.

Таким чином, завдання даної роботи полягає в аналітичному рішенні прямої та зворотної задачі кінематики у вигляді, придатному для алгоритмічного опису та написання керуючої програми, виконання якої відбувається в режимі реального часу, вкладається в проміжки між отриманими даних про об'єкт від відеокамери.

На основі проведеного аналітичного огляду сформовано

технічне завдання на конструювання робота-маніпулятора для позиціонування датчиків на виробах зі складною геометрією (рис.5.1.).

Для реалізації задуманої конструкції робота-маніпулятора проаналізовано роботи-маніпулятори промислові, та зпроектувати прототип промислового робота-маніпулятора. Даний робот-маніпулятор розроблений для проведення певних операцій, які автоматизують промислові процеси. Також можливе використання даного роботу у процесі контролю, при оснащенні активної частини робота-маніпулятора відповідними датчиками, наприклад ультразвуковими.



Рис. 5.1. Модель робота-маніпулятора

Для зручності реалізації поставленої задачі було вирішено розділити конструкцію робота-маніпулятора на три частини:

- активна частина робота-маніпулятора;
- рукоятка;
- робоча зона або зона захвату.

Розділення на частини даної моделі обумовлено характеристиками 3D-принтера на якому було заплановано друк деталей, а саме: розмірні характеристики робочої поверхні

принтера.

До активної частини робота-маніпулятора входить стаціонарна зона, у якій розміщується серводвигун та підшипники. Разом ці деталі забезпечують переміщення рукоятки разом з робочою частиною навколо своєї осі. Окрім цього, активна частина відповідає за положення рукоятки у просторі. На рисунку 5.2 показано внутрішні складові активної частини.



Рис. 5.2. Внутрішні складові активної частини робота-маніпулятора

Для реалізації створення активної частини робота-маніпулятора створити її модель в програмному середовищі SolidWorks, та розділити на складові частини, які згодом роздруковуються на 3D-принтері. На рисунку 5.3 зображено деталі – дискові основи активної частини, які скріплюються між собою гвинтами та містять всередині підшипник, який згодом буде частиною обертального руху активної частини.

Для повного функціонування на зборі активної частини моделі робота- маніпулятора необхідні такі матеріали:

- Кроковий двигун Nema 17;

- Гвинти М3 х 10мм та М3 х 8мм;
- Шурупи М3 х 3 мм та М4 х 3 мм;
- Підшипник 10 мм х 30 мм х 9 мм, 6200zz
- Сталеві кульки 10 мм.



Рис. 5.3. Складові активної частини після друку на 3D-принтері

Для створення 3D-моделей було застосувати програмне середовище SolidWorks. А для адаптування моделі під параметри 3-Д принтера, та задання розмірів, матеріалу та щільності друку застосувати програмний продукт Ultimaker Cura.

На рисунку 5.4 зображено змодельований вигляд активної частини робота-маніпулятора, в якому всередині встановлено кроковий двигун, підшипник та 10 сталевих кульок.

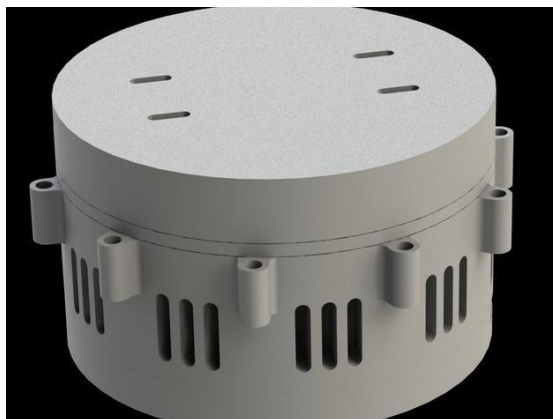


Рис. 5.4. Змодельована активна частина робота-маніпулятора

Сталеві кульки призначені для оберту верхньої частини конструкції по своїй осі, таким чином забезпечуватиметься рух рукоятки на 360°. Також верхня та нижня частина конструкції з'єднуються між собою по колу гвинтами, в сумарній кількості присутні 10 з'єднань по колу (рис. 5.5).

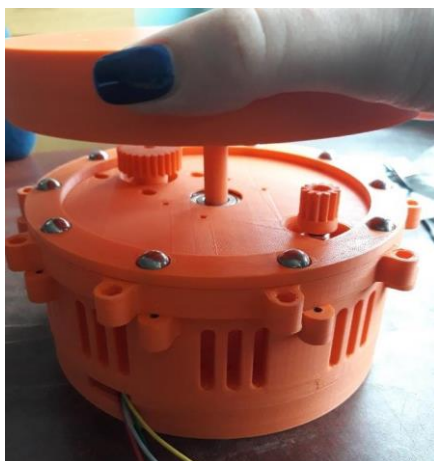


Рис. 5.5. Роздрукована на 3D-принтері та зібрана активна частина робота-маніпулятора

При друку на 3D-принтері деталей застосовано пластик PLA, зтакими механічними властивостями:

- міцність на розтяг 51 МПа;
- відносне подовження при розриві 30%;
- модуль пружності при розтягу 2300 МПа;
- модуль пружності при згинанні 1440;
- міцність при згинанні 80 Мпа.

Даний матеріал було обрано з розрахунку міцності подальшої конструкції. За своїми властивостями пластик типу PLA має більш міцну характеристику аніж пластик типу ABS.

Кінцевий вигляд 1/3 конструкції робота-маніпулятора, а саме активної частини робота-маніпулятора зображено на рисунку 5.6.

Всередині активної частини розміщено такі елементи активного переміщення як: кроковий двигун Nema17 та

потенціометр BOURNS 3590S.

Для роботи рукоятки в парі з захватом та активною частиною встановлено 5 серводвигунів типу MG995. Даний двигун було підбрано на основі проаналізованих прикладів роботів-маніпуляторів, а також у задоволенні характеристиками двигуна вимог, які було висунуто до даної частини конструкції.



Рис. 5.6. Активна частина робота-маніпулятора в зібраному стані

Основною задачею рукоятки та її плечей є передача крутного моменту відактивної частини до захвату, таким чином робот-маніпулятор матиме 5 ступенів свободи. Для реалізації складання даної конструкції необхідно було використати такі матеріали:

- Гвинти М3 x 8мм, М3 x 10мм, М3 x 63мм та М3 x 16мм;
- Підшипники типу 688zz з розміром 8 мм x 16 мм x 5 мм;
- Гайка М3 10 мм;
- Серводвигун MG995.

Модель надруковано на 3-Д принтері деталі, як зображено на рисунку 5.7.

Змодельований вигляд рукоятки та плечей представлено на рисунку 30.

Всі отвори було зашліфовано так як при друку неможливо досягнути точного розміру отвору під гвинт та підшипники. При друку деталей рукоятки щільність заповнення деталей матеріалом була виставлена не менше ніж 50%.

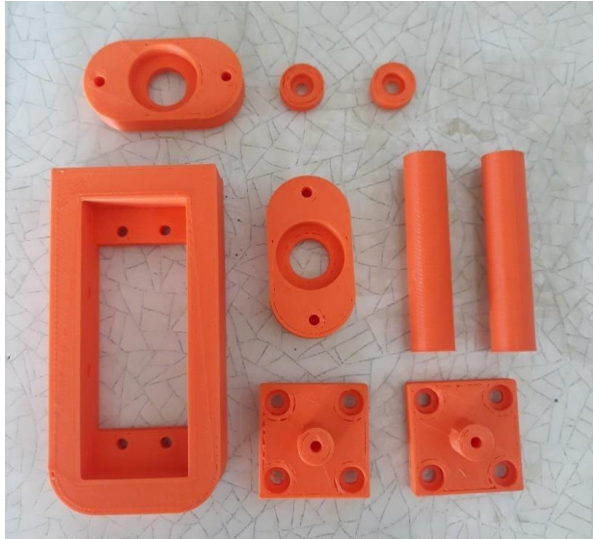


Рис. 5.7. Надруковані деталі рукоятки робота-маніпулятора



Рис. 5.8. Модель рукоятки та плечей

Кінцевий вигляд іншої частини конструкції робота-маніпулятора, а саме рукоятки робота-маніпулятора зображено на рисунку 5.9.

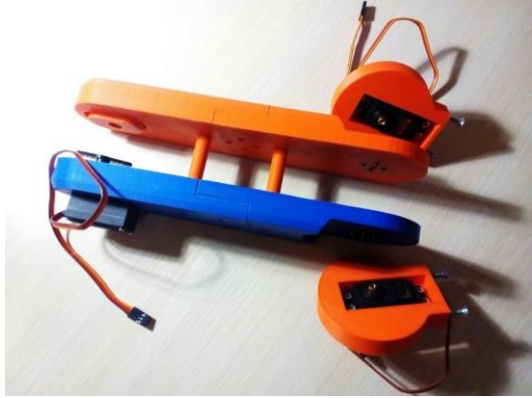


Рис. 5.9. Рукоятка робота-маніпулятора в зібраному стані

Як видно з рисунку 5.9 встановлено 3 серводвигуна, які будуть забезпечувати переміщення рукоятки.

Проте, однією з найважливіших елементів робота-маніпулятора є зона захвату. Для реалізації роботи зони захвату було застосовано такі матеріали:

- Гвинти М3 x 25мм, М3 x 10 мм, М3 x 8мм;
- Підшипники типу 688zz
- Серводвигун MG995.

Роздруковану зону захвату зображено на рисунку 5.10. Всі отвори зашліфовано так як при друку неможливо досягнути точного розміру отвору під гвинт та підшипники. При друку деталей рукоятки щільність заповнення деталей матеріалом була виставлена не менше ніж 70%.

З'єднання зони захвату виконане у вигляді двох деталей з перфорованою внутрішньою частиною, що необхідне для чіткого захвату деталей при роботі з роботом-маніпулятором.

Дана частина робота-маніпулятора з'єднується з рукояткою завдяки окремому елементу, який є з'єднувальною частиною між зоною захвату та рукояткою (рис.5.11). Сама зона захвату з'єднується за допомогою гвинтів.



Рис. 5.10. Зона захвату робота-маніпулятора в зібраному стані



Рис. 5.11. Зона захвату з кріпильними елементами в зібраному стані

Після друку на 3D-принтері всіх деталей, їх шліфування та збору отримано завершену модель робота-маніпулятора для позиціонування датчиків на виробах зі складною геометрією, яку зображено на рисунку 5.12.



Рис. 5.12. Модель робота-маніпулятора в зібраному стані

Завершена модель зібраного робота-маніпулятора готова до програмування та подальшого використання згідно запланованих цілей застосування.

В роботі описано процес створення конструкції моделі робота-маніпулятора. Обрано комплектуючі елементи та надано опис їх застосування у моделі. Також за допомогою системи автоматичного проектування SolidWorks розроблено 3D модель приладу, яку згодом було використано як основу для створення макету. Деталі надруковано на 3D-принтері та зібрано згідно запланованій конструкції. Показано основні вузли робота-маніпулятора та описано принцип їх з'єднання та взаємодії. Конструкція зібрана та готова до програмування.

Контрольні запитання

1. Яке основне завдання маніпулятора робота та скільки ступенів вільності йому необхідно для повноцінної роботи?
2. Чому в реальних конструкціях часто використовуються механізми з числом ступенів вільності менше шести?
3. Які основні частини включає конструкція маніпулятора, і яка роль активної частини?
4. Чому для друку деталей було обрано пластик PLA, і які його основні властивості?
5. Які програмні засоби використовуються для створення 3D-моделі маніпулятора та підготовки деталей до друку?

Практична робота 6. Дослідження роботи маніпулятора з дистанційним 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 чи людини-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 насамперед зумовлена 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, закріпленого в 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 нерухокої ланки 0 і трьох рухомих ланок 1 , 2 і 3 (рис. 6.1).

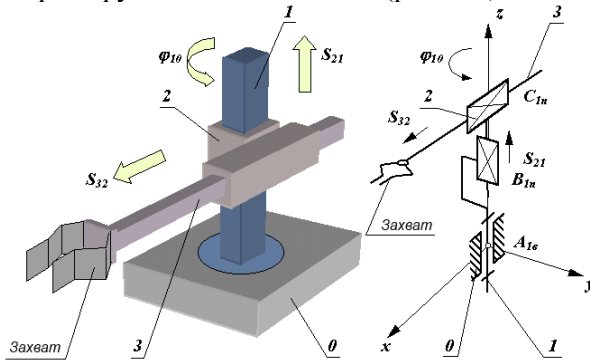


Рис. 6.1. Структурна і функціональна схеми промислового маніпулятора

Механізм цього маніпулятора відповідає циліндричній системі координат. У цій системі ланка 1 може обертатися відносно ланки 0 (відносно кутового переміщення φ_{10}), ланка 2 переміщається по вертикалі відносно ланки 1 (відносно лінійне переміщення S_{21}) і ланка 3 переміщається в горизонтальній площині відносно ланки 2 (відносно лінійне переміщення S_{32}).

На кінці ланки Z закріплений захоплюючий пристрій (захват), призначений для захоплення й утримання об'єкта маніпулювання. Ланки основного важільного механізму маніпулятора утворюють між собою три однорухливі кінематичні пари (одну обертальну A і дві поступальні B та C) і можуть забезпечити переміщення об'єкта в просторі без керування його орієнтацією.

Для виконання кожного з трьох відносних рухів маніпулятор повинен бути оснащений приводами, що складаються з двигунів, 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_i здійснюється системою керування. При необхідності вона коректує ці впливи за сигналами Δx_i , що надходять до неї з давачів зворотного зв'язку (рис. 6.2).

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.

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).

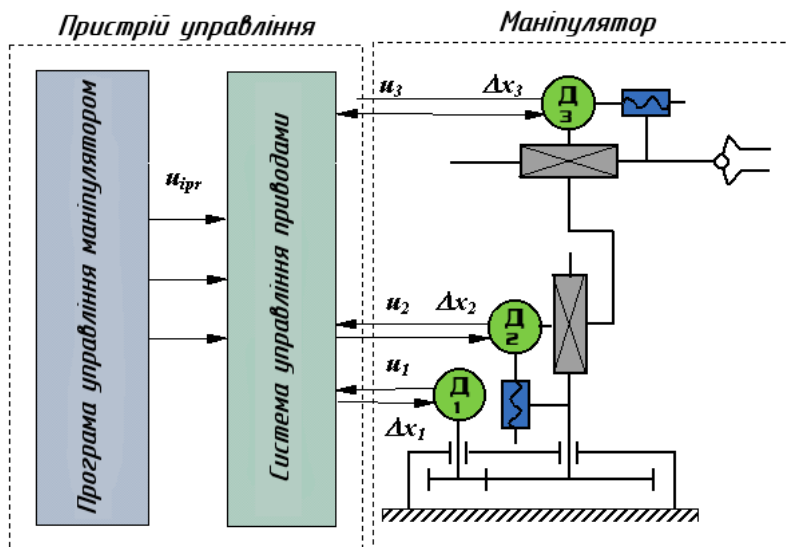


Рис. 6.2. Функціональна схема промислового робота

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

Точність маніпулятора промислового робота характеризується абсолютною лінійною похибкою позиціонування центра захвату. Промислові роботи поділяються на групи з малою ($\Delta r_M < 1$ мм), середньою ($0,1$ мм $< \Delta r_M < 1$ мм) і високою ($\Delta r_M < 0,1$ мм) точністю позиціонування.

Опис лабораторної установки

Досліджуваний маніпулятор, зображений на рис. 6.3, 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.

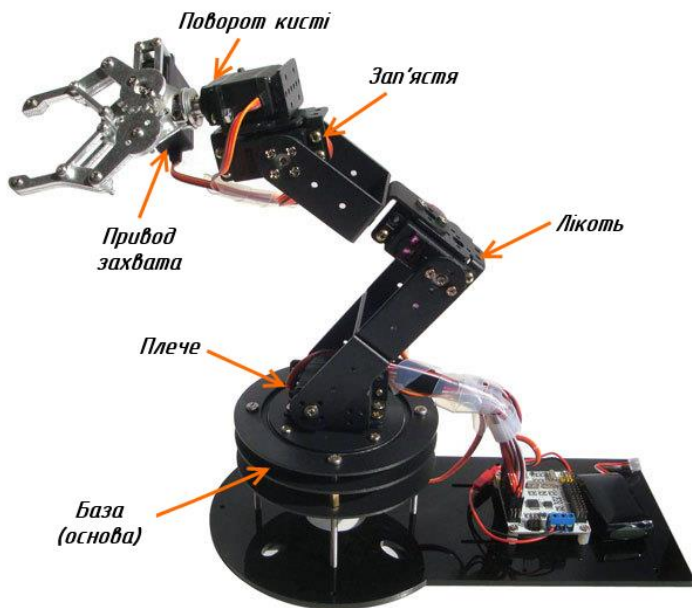


Рис. 6.3. Зовнішній вигляд досліджуваного маніпулятора

Параметри сервоприводів MG996R подані нижче.



Рис. 6.4. Зовнішній вигляд та розміри сервопривода MG996R

Основні характеристики:
З'єднувальний провід: довжина 300мм.

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)

Джерело живлення: через зовнішній адаптер.

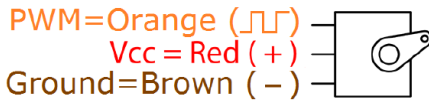
Робоча напруга: 4.8 – 7.2V

Редуктор : металеві шестерні.

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

Струм при стабільному навантаженні: 2.5 А (6 В).

Підключення:



Для живлення сервоприводів у лабораторній роботі використовується блок живлення 5В, 10А.

Система управління роботом-маніпулятором реалізована на основі мікроконтролерної плати Arduino Mega 2560.

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 Arduino IDE.
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(2,740,2365); // attaches
the servo on pin 2 to the servo object
    myservo2.attach(3,760,2365); // attaches
the servo on pin 3 to the servo object
    myservo3.attach(4,720,2365); // attaches
the servo on pin 4 to the servo object
    myservo4.attach(5,720,2300); // attaches
the servo on pin 5 to the servo object
    myservo5.attach(10,670,2300); // attaches
the servo on pin 10 to the servo object
    myservo6.attach(11,720,2300); // attaches
the servo on pin 11 to the servo object
    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(my servo3.read()); // Movement
Data
        Serial.print(",");
        Serial.print(my servo4.read()); // Movement
Data
        Serial.print(",");
        Serial.print(my servo5.read()); // Movement
Data
        Serial.print(",");
        Serial.print(my servo6.read()); // Movement
Data
        Serial.println();
        delay(20);
    }

```

4. Скомпілювати програму та переконалися у відсутності помилок.

5. Підключити мікроконтролерну плату до USB-порту комп'ютера.

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

7. Завантажити середовище Processing. Відкрити скетч Scrollbar1.pde та запустити його на виконання.

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. 6.1.

Таблиця 6.1

№ з/п	Сервопривод	Нижня межа кута повороту	Верхня межа кута повороту
1	База		
2	Плече		
3	Лікоть		
4	Зап'ястя		
5	Поворот кисті		
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 3.2; висновки.

Контрольні запитання

1. Що таке промисловий робот?
2. Що таке маніпулятор?
3. Яка сфера застосування промислових роботів і маніпуляторів?
4. Наведіть приклад функціонально схеми промислового робота?
5. Які типи приводів використовують у промислових маніпуляторах?
6. Як класифікують промислові роботи за точністю позиціонування?
7. Що таке ступені свободи маніпулятора?

Практична робота 7. Розроблення програмного забезпечення роботизованої ділянки

Мета роботи: ознайомитися з принципами програмно-логічного управління роботизованими ділянками. Написати програму для автоматичного управління маніпулятором.

Теоретичні відомості

У процесі програмування роботизованих ділянок виробництва потрібно враховувати взаємний вплив кожної ланки у ділянці на попередню, наприклад, неможливість здійснення зупинки одного конвеєра без зупинки попереднього конвеєра, з якого в нормальному режимі роботи повинні забиратись об'єкти на конвеєр, який необхідно зупинити.

Приклад автоматизованої виробничої ділянки показано на рис. 7.1. Заготовка, що прибуває у ділянку на вхідному конвеєрі, переміщається у робочу зону верстата. При цьому слід передбачити зупинку конвеєра до того часу, поки маніпулятор не захопить заготовку. Робот виконує спочатку операцію переміщення заготовки на вхід у верстат. Потім відбувається обробка заготовки у верстаті. Після цього заготовка переміщається на вихід верстата. Потім, якщо робот вільний, він переносить оброблену заготовку на вихідний конвеєр

виробничої ділянки.

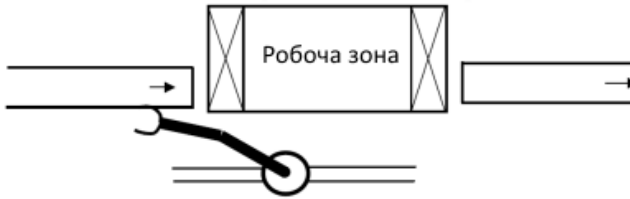


Рис. 7.1. Схема досліджуваної роботизованої ділянки

План роботи

1. Ознайомитися з будовою роботизованої ділянки в лабораторній установці.
2. Написати та відлагодити програму для керування роботизованою ділянкою.

Порядок виконання роботи

1. Ознайомитися з схемою підключення приводів роботизованої ділянки до плати Arduino Mega та силовими елементами для керування приводами.
2. Написати програму для керування маніпулятором роботизованої ділянки.
3. У середовищі Arduino IDE вибрати тип плати та порт, до якого вона підключена. Завантажити створену програму в мікроконтролер.
4. З дозволу викладача увімкнути живлення стенда та відлагодити роботу програми.
5. Написати та відлагодити програму для керування конвеєром або рухомою кареткою роботизованої ділянки.
6. На підставі розроблених програм створити програму, що керуватиме роботою всієї ділянки: і робота-маніпулятора, і конвеєра або рухомої каретки.
7. Перевірити правильність роботи програми та відлагодити її.
8. Додати в програму функцію блокування роботи всієї механічної складової при виявленні руху датчиком HC-SR501.
9. 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. У чому відмінність розробки програм керування окремими мехатронними елементами та роботизованими ділянками?
2. Якими способами можна виявити надходження об'єкта на початок ділянки?
3. Які способи керування двигном постійного струму ви знаєте?
4. Чим спричинено використання зворотно включених діодів паралельно якорям двигунів постійного струму під час керування їх швидкістю за допомогою ШІМ сигналу, що надходить на затвор польового транзистора?

Список рекомендованої літератури

1. Ловейкін В. С., Ромасевич Ю. О., Крушельницький В. В. Мехатроніка : підручник. К., 2020. 404 с.
2. Синтез робототехнічних систем в машинобудуванні / Л. Є. Пелевін, К. І. Почка, О. М. Гаркавенко та ін. К. : Інтерсервіс, 2016. 258 с.
3. Margolis Michael. Arduino Cookbook. O'Reilly Media, 2011. 662 p.
4. Evans B. Arduino programming notebook. First edition. 2007. 38 p. URL: https://playground.arduino.cc/uploads/Main/arduino_notebook_v1-1.pdf.
5. Методичні вказівки до виконання практичних робіт з навчальної дисципліни «Робототехніка в машинобудуванні» для здобувачів вищої освіти першого (бакалаврського) за освітньо-професійною програмою «Галузеве машинобудування» спеціальності 133 «Галузеве машинобудування» всіх форм навчання [Електронне видання] / Гололюк М. В., Реут Д. Т., Марчук Н. М. Рівне : НУВГП, 2021. 35 с.
6. Ніколайчук В. М. Основи робототехніки : навч. посіб. Рівне : НУВГП, 2008. 76 с. URL: <http://ep3.nuwm.edu.ua/2243/>.

7. Пелевін Л. Є., Балака М. М., Аржаєв Г. О. Механотронні системи гідропневмоавтоматики. К. : Аграр Медіа Груп, 2014. 192 с.

8. Автоматизація виробничих процесів / Ельперін І. В., Пупена О. М., Сідлецький В. М., Швед С. М.. К. : Ліра-К, 2021, 378 с.