

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
імені ВОЛОДИМИРА ДАЛЯ

МЕТОДИЧНІ ВКАЗІВКИ

до виконання лабораторних робіт з дисципліни
**«Мехатронні та робототехнічні системи: базові принципи і
практика»**

*Для здобувачів вищої освіти спеціальності G7 –
«Автоматизація, комп'ютерно-інтегровані технології та
робототехніка»
усіх форм навчання
(електронне видання)*

ЗАТВЕРДЖЕНО:
на засіданні кафедри
комп'ютерно-інтегрованих
систем управління
Протокол №8 від
21.05.2026 р.

м. Київ, 2026

УДК 681.5

Методичні вказівки до лабораторних занять з дисципліни «Мехатронні та робототехнічні системи: базові принципи і практика» (для здобувачів вищої освіти спеціальності G7 – «Автоматизація, комп'ютерно-інтегровані технології та робототехніка») (Електронне видання) / Уклад.: Т.Г. Сотнікова, А.А. Асманкіна. – Київ: вид-во СНУ ім. В. Даля, 2026. – 57 с.

У методичних вказівках викладені матеріали, необхідні для виконання лабораторних робіт з дисципліни «Мехатронні та робототехнічні системи: базові принципи і практика». Методичні вказівки містять в собі необхідні теоретичні відомості за темами лабораторних робіт, приклади проектів на платформі Arduino, що дозволяють одержати поглиблене уявлення про суть розглянутого питання, приклади та контрольні питання для самостійного освоєння матеріалу заняття, а також список літератури і необхідні довідкові матеріали.

Укладачі:

Т.Г. Сотнікова, к.т.н., доцент
А.А. Асманкіна, асистент

Рецензент:

М.Г. Лорія, д.т.н., професор

ЗМІСТ

ВВЕДЕННЯ.....	4
1. ЗАГАЛЬНІ ВКАЗІВКИ ДО ВИКОНАННЯ РОБІТ	5
2. ЗНАЙОМСТВО З ПЛАТФОРМОЮ ARDUINO	8
3. ЛАБОРАТОРНІ РОБОТИ.....	10
4. 3.1 Тема 1. Визначення опору резистора за маркуванням.....	10
5. 3.2 Тема 2. Світлодіод. RGB-світлодіод. Яскравість світлодіода.....	18
6. 3.3 Тема 3. Потенціометр.....	30
7. 3.4 Тема 4. Керування серводвигуном.....	35
8. 3.5 Тема 5. Дзвінок.....	39
9. 3.6 Тема 6. Шумомір.....	45
10. 3.7 Тема 7. Керування шаговим мотором.....	52
СПИСОК ЛІТЕРАТУРИ.	57

ВВЕДЕННЯ

Методичні вказівки складені відповідно до робочої програми з курсу “Мехатронні та робототехнічні системи: базові принципи і практика” зі спеціальності G7 – «Автоматизація, комп’ютерно-інтегровані технології та робототехніка».

Мета дисципліни – підготовка спеціалістів за освітньо-кваліфікаційним рівнем магістра, відповідно до державних стандартів, встановлених освітньо-кваліфікаційною характеристикою та освітньо-професійною програмою підготовки магістрів вищезазначеного фахового спрямування, розвиток інженерного мислення на засадах вивчення методів аналізу й синтезу сучасних систем керування. Відображено освітній стандарт курсу в межах концепції «модель – аналіз – синтез». Разом з основними положеннями класичної теорії керування велика увага приділяється теорії оптимальних й адаптивних систем.

Метою лабораторних занять за дисципліною є набуття знань та навичок роботи з лабораторним устаткуванням, закріплення теоретичних знань, проведення досліджень алгоритмів і формулювання особистих висновків, формування самостійності мислення, розвиток дослідницьких вмінь. Вміння створювати скетчі (проекти) на платформі Arduino дозволяють у повному обсязі зрозуміти на практиці роботу систем керування на виробництві. Arduino – це платформа для прототипів з відкритим вихідним кодом, розроблена на основі популярного сімейства мікроконтролерів, і включає в себе просте середовище розробки програмного забезпечення.

Предметами дисципліни є мехатронні та робототехнічні системи.

Завдання дисципліни – набуття навичок розробки та впровадження систем автоматичного керування на основі сучасних приладів та засобів автоматизації.

1 ЗАГАЛЬНІ ВКАЗІВКИ ДО ВИКОНАННЯ РОБІТ

Робоча програма з дисципліни «Мехатронні та робототехнічні системи: базові принципи і практика» заснована на поєднанні теоретичного вивчення з виробничою і науково-дослідною практикою та взаємозв'язку учбового процесу з науковою роботою через тематику лабораторних робіт у тому числі. Лабораторні практикуми дозволяють набути навичок науково-дослідної роботи, обробки і оформлення результатів експерименту, представлення результатів роботи у формі наукової доповіді і звіту.

Методика учбового експериментального дослідження є мініатюрною моделлю звичайного циклу наукової роботи, хоча в ній відсутня принципова новизна отриманого результату. Що вчиться при виконанні лабораторної роботи в меншій або більшій мірі повинен освоїти ряд елементів аналогічної наукової роботи.

1.1 Загальні положення

Лабораторна робота є невеликим науковим звітом, що узагальнює проведену здобувачем вищої освіти роботу, яку представляють для захисту викладачеві. До лабораторних робіт пред'являється ряд вимог, основним з яких є повний, вичерпний опис всієї виконаної роботи, що дозволяє судити про отримані результати, міру виконання завдань і професійну підготовку здобувачів вищої освіти.

У звіт по лабораторній роботі мають бути включені наступні пункти:

- титульний аркуш;

- мета роботи;
- короткі теоретичні відомості;
- опис експериментальної установки і методики експерименту;
- експериментальні результати;
- аналіз результатів роботи;
- висновки.

1.2 Вимоги до вмісту окремих частин звіту з лабораторної роботи

Титульний аркуш є першою сторінкою будь-якої наукової роботи і для конкретного вигляду роботи заповнюється за певними правилами. Для лабораторної роботи титульний аркуш оформляється таким чином. У верхньому полі аркуша вказують повне найменування учбового закладу і кафедри, на якій виконувалася дана робота.

У середньому полі вказується вид роботи, в даному випадку лабораторна робота з вказівкою дисципліни, за якою вона виконана, і нижче її назва. Назва лабораторної роботи приводиться без слова «Тема» і в лапки не полягає. Далі ближче до правого краю титульного аркуша вказують прізвище, ініціали, курс і групу що вчиться, виконав роботу, а також прізвище, ініціали, вчене звання і посаду викладача, що прийняв роботу.

У нижньому полі аркуша вказується місце виконання роботи і рік її написання (без слова рік). Мета роботи повинна відображати тему лабораторної роботи, а також конкретні завдання, поставлені студентам на період виконання роботи.

За об'ємом мета роботи залежно від складності і багатозадачності роботи складає від декількох рядків до 0,5 сторінки.

Короткі теоретичні відомості. У цьому розділі викладається короткий теоретичний опис явища, що вивчається в роботі, або процесу, приводяться також необхідні розрахункові формули. Матеріал розділу не повинен копіювати вміст методичного посібника або підручника за даною темою, а обмежується викладом основних понять і законів, розрахункових формул, таблиць, потрібних для подальшої обробки отриманих експериментальних результатів. Об'єм літературного огляду не повинен перевищувати 1/3 частини всього звіту.

Опис експериментальної установки і методики експерименту. У даному розділі приводиться схема експериментальної установки з описом її роботи і детально викладається методика проведення експерименту, процес здобуття даних і спосіб їх обробки. Якщо використовуються стандартні пакети комп'ютерних програм для обробки експериментальних результатів, то необхідно обґрунтувати можливість і доцільність їх вживання, а також подробиці обробки даних з їх допомогою.

Експериментальні результати. У цьому розділі наводяться безпосередньо результати, отримані в ході проведення лабораторних робіт: експериментально або в результаті комп'ютерного моделювання певні значення величин, графіки, таблиці, діаграми.

Аналіз результатів роботи. Розділ звіту повинен містити детальний аналіз отриманих результатів, інтерпретацію цих результатів на основі фізичних законів. Слід порівняти

отримані результати з відомими літературними даними, обговорити їх відповідність існуючим теоретичним моделям. Якщо виявлена невідповідність отриманих результатів і теоретичних розрахунків або літературних даних, необхідно обговорити можливі причини цих невідповідностей.

Висновки. У висновках коротко викладаються результати роботи: отримані експериментально або теоретично значення фізичних величин, їх залежності від умов експерименту або вибраної розрахункової моделі, вказується їх відповідність або невідповідність фізичним законам і теоретичним моделям, можливі причини невідповідності.

Звіт з лабораторної роботи оформлюється на папері стандартного формату А4 на одній стороні аркуша, які зшиваються степлером. Допускається оформлення звіту з лабораторної роботи лише в електронному вигляді засобами Microsoft Office.

2 ЗНАЙОМСТВО З ПЛАТФОРМОЮ ARDUINO

2.1 Історія

У 2005 році Массімо Банзі, італійський співзасновник Arduino, розробив технологію для своїх студентів в Інституті дизайну взаємодії Ivrea (IDII). З того часу Arduino перетворилася в одну з найбільших апаратних платформ з відкритим вихідним кодом. Всі програмні компоненти і схеми дизайну Arduino мають відкритий вихідний код, і ви можете купити обладнання за дуже низькою ціною, або навіть можете зробити їх самостійно.

2.2 Переваги Arduino

Основний напрямок спільноти Arduino – постійно вдосконалювати платформу Arduino з огляду на таке:

- платформа Arduino повинна бути доступною;
 - вона повинна бути проста у використанні і легко кодуватися;
 - відкрита вихідна і розширювана програмна платформа;
 - відкрита вихідна і розширювана апаратна платформа;
 - вона повинна мати проекти DIY, підтримувані спільнотою.
- Ці прості, але потужні напрямки зробили Arduino популярної і широко використовуваною платформою для створення прототипів. Arduino використовує мікроконтролери Atmel серії ATmega, засновані на популярній апаратної архітектурі AVR. Величезна підтримка, яка доступна для архітектури AVR, також робить Arduino кращою апаратною платформою. На наступному рисунку показана базова версія плати Arduino, яка називається Arduino Uno (Uno означає один італійською мовою):

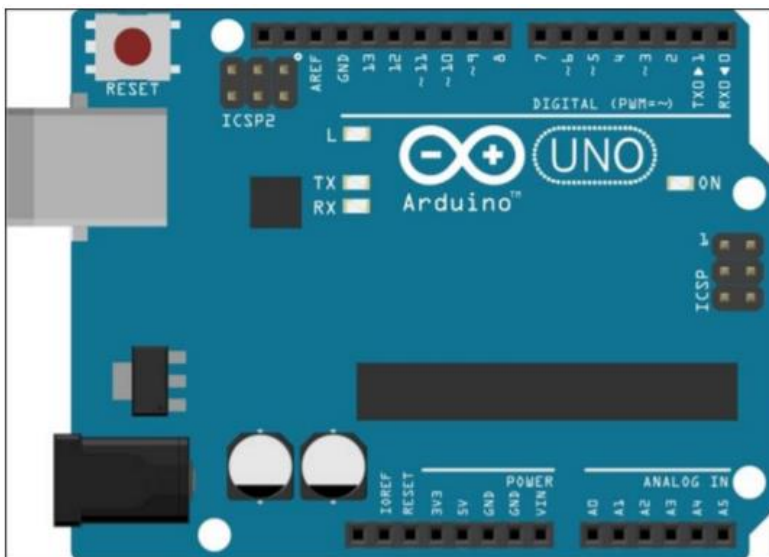


Рис 1. - Базова версія плати Arduino

2.3 Варіанти Arduino

Як і будь-який інший проект, вимоги до обладнання визначаються специфікаціями проекту. Якщо ви розробляєте проект, який вимагає від вас взаємодії з великою кількістю зовнішніх компонентів, вам потрібна платформа прототипування, у якій є достатня кількість входів/виходів (I/O) для сполучення. Якщо ви працюєте над проектом, який повинен виконувати величезну кількість складних обчислень, вам потрібна платформа з великими обчислювальними можливостями. На щастя, плата Arduino існує в 16 різних офіційних версіях, і кожна версія Arduino відрізняється від інших по формфактору, обчислювальній потужності, виводів введення-виведення і іншим вбудованим функціям. Arduino Uno – це основна і найпопулярніша версія, якої достатньо для

простих проектів DIY. Для більшості вправ в цьому посібнику ми будемо використовувати плату Arduino Uno. Ви також можете використовувати інший популярний варіант під назвою Arduino Mega, який представляє більшу плату з додатковими виводами і потужним мікроконтролером. Будь-який варіант збірки може бути запрограмований з використанням загального інтегрованого середовища розробки під назвою Arduino IDE, яка описана в наступних розділах.

3 ЛАБОРАТОРНІ РОБОТИ

3.1 Тема 1. Визначення опору резистора за маркуванням.

Мета роботи: Навчитися визначати опір резистора за його маркуванням для побудови простих схем.

Резистор - це електронний прилад, який має певний опір. Його основне завдання - перетворення сили струму в напругу і навпаки. Зважаючи на малі розміри не завжди вдається нанести і зчитати маркування з резистора — наприклад, **пристрій на 0,25 ват, досить часто застосовується в системотехніці, має довжину не більше 3.2 мм при діаметрі 1,8 мм.** Саме тому і була розроблена кольорова схема маркування. Вона є міжнародною, її затвердила IEC (International Electrotechnical Commission) і вимоги ДСТУ 175-72.

Маркування резисторів смужками

Читати смужки належить зліва направо. Перше кільце наноситься ближче до дротяного виходу з пристрою.

Таблиця кольорів

Для читання маркування резисторів кольоровими смужками можна використовувати цю таблицю:

Таблиця 3.1 Маркування резисторів кольоровими смужками

Колір	Значення		Множник	Допустиме відхилення $\pm \%$	Темп. коеф. опору $\pm 10^{-6}/K$
	1 кільце	2 кільце	3 кільце	4 кільце	Останнє кільце
відсутнє				20	
сріблястий			0,01 Ω	10	
золотистий			0,1 Ω	5	
чорний	0	0	$\times 1 \Omega$	20	200
коричневий	1	1	$\times 10 \Omega$	1	100
червоний	2	2	$\times 100 \Omega$	2	50
оранжевий	3	3	$\times 1 k\Omega$	3	15
жовтий	4	4	$\times 10 k\Omega$	0,1	25
зелений	5	5	$\times 100 k\Omega$	0,5	
голубий	6	6	$\times 1 M\Omega$	0,25	10
фіолетовий	7	7	$\times 10 M\Omega$	0,1	5
сірий	8	8		0,05	1
білий	9	9			

Останні числа використовуються для десяткового множника. Також слід пам'ятати, що існує шість рядів точності, передбачених ГОСТ. Для ряду Е6 допускається

відхилення в 20%, для E12 - в 10%, E24 - 5%, E48 - 2%, E96 - 1%, E 192 - 0,5%.

Правила маркування

Класична маркування резисторів за кольором складається з 3-6 смуг / кілець. Чим більше смуг, тим більше точність вимірювання. Розберемо найбільш популярні варіанти.

Пристрої з трьома смужками:

Подібну маркування застосовують тільки для тих елементів, які мають "планові" відхилення не більше 20%. Цифри, належать до кольорів, можна взяти з наведеної вище таблиці. Перший і другий круг показує опір пристрою, третя - показник множника. Якщо позначити першу смужку D1, другу D2, третю E, то формула розрахунку опору буде виглядати так:
 $R = (10D1 + D2) * 10^E$

Наприклад, на резисторі перша смуга червона, друга зелена, третя - жовта. Шукаємо опір $(10 * 2 + 5) * 10^4 = 25 * 10^4$ в 4 ступеня = 250000 Ом або 250 кОм.

Елемент з 4 полосками

Використовуються для пристроїв з точністю до 5-10% (ряд E12 і E24 по маркуванню ГОСТ). Схема маркування опорів за кольорами залишається тим самим: перші два кільця - номінал опору, третє - десятковий множник, четверте - допуск. Золотистий допуск - 5% (відноситься до ряду E24), сріблястий - 10% (ряд E 12). У цьому випадку формула виглядає наступним чином: **$R = (10D1 + D2) * 10^E \pm S$** , де перша смуга - D1, друга - D2, третя - E, четверта - S.

*Приклад: якщо ви бачите пристрій з 4 смугами зеленого, оранжевого, червоного і золотого кольору, то опір дорівнюватиме $R = (50 + 3) * 10$ другого ступеня = 5300 Ома $\pm 5\%$ або 5.3 кОм $\pm 5\%$.*



Рис 2. - Резистори з 4 смужками

Пристрої з 5 смужками

Подібна маркування резисторів по смужках застосовується для смуг E48 - 2%, E96 - 1%, E 192 - 0,5%. Техніка підрахунку перших трьох смуг залишається попередньою, четверта позначає десятковий множник, п'ята - рівень допуску. Формула виглядає наступним чином: $R = (100D1 + 10D2 + D3) * 10^E \pm S$, де D1, D2 і D3 - перші три кола, E-четвертий, S - п'ятий. Допуски позначаються наступним чином:

- E48 (2%) - червоний;
- E96 (1%) - коричневий;
- E192 (0,5%) - зелений;
- 0,25% - синій;

- 0,1% - фіолетовий;
- 0,05% - сірий.

Пристрої з 6 смужками

Професійні ремонтники знають, що у деяких резисторів є так званий коефіцієнт температурного опору або коротко - ТКС. Цей параметр показує, на яку величину підвищується / зменшується опір елемента при зміні температури на 1 градус. Цей коефіцієнт вимірюється в ppm / ОС (parts per million або мільйонна частина від наявного номіналу, поділена на кількість градусів). Розберемо позначення резисторів за кольорами на шостому кільці:

1. Коричневий колір - 100 ppm / ОС.
2. Червоний - 50 ppm / ОС.
3. Жовтий - 25 ppm / ОС.
4. Помаранчевий - 15 ppm / ОС.
5. Синій - 10 ppm / ОС.
6. Фіолетовий - 5 ppm / ОС.
7. Білий - 1 ppm / ОС.

Розберемо приклад визначення резистора по кольоровому маркуванню на 6 кілець. Наприклад, ми маємо резистор з червоною, зеленою, фіолетовою, жовтою, коричневою і помаранчевою смугою. Опір дорівнюватиме $(100 * 2 + 10 * 5 + 7) * 10^4 \pm 1\%$ (15ppm / ОС) або ж $2570000 \pm 1\%$ (15ppm / ОС) або $2,57 \pm 1\%$ (15ppm / ОС) МОм.

Увага: шосте кільце часто використовується для підрахунку коефіцієнта надійності елемента. Якщо воно

стандартної ширини, то визначає коефіцієнт ррт / ОС, якщо воно ширше в півтора рази, то показує відсоток відмов елемента на одну тисячу годин роботи.

Кольорові позначення в цьому випадку такі:

1. Коричневий колір - до 1 відсотка відмов.
2. Червоний колір - не більше 0,1% відмов.
3. Помаранчевий колір - не більше 0,01% відмов.
4. Жовтий - не більше 0,001% відмов за 1000 годин роботи.

В якості робочої таблиці для визначення опору можна використовувати наступний варіант:

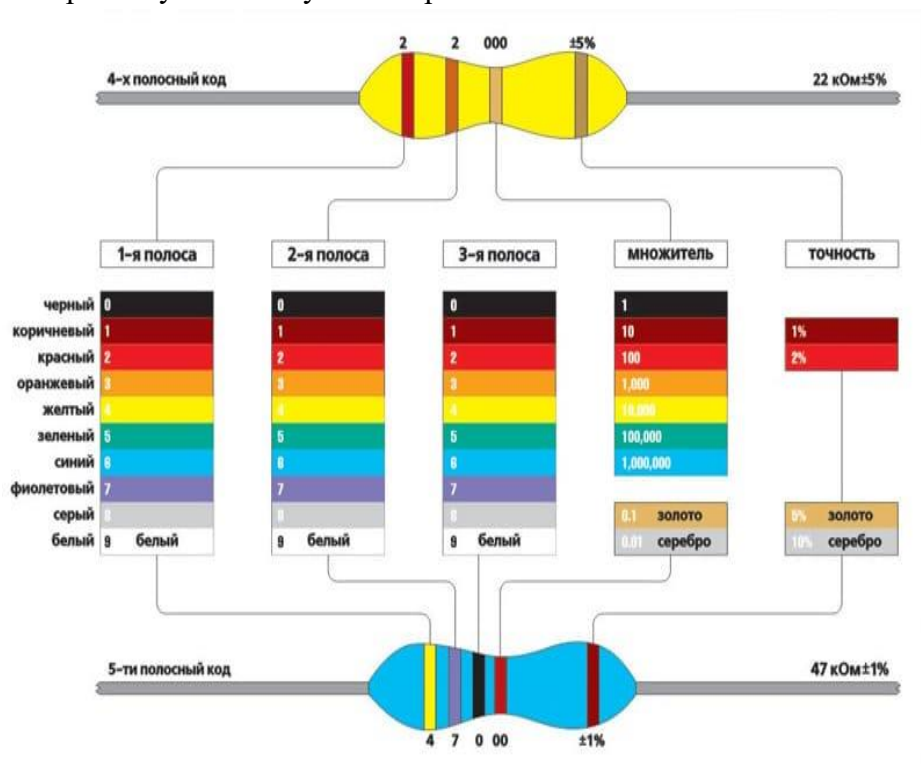


Рис. 3. - Таблица для чтения номиналу резистора

Дротяні резистори

Для дротяних резисторів прийняті трохи інша розшифровка резисторів за кольором. Першою смугою в будь-якому випадку буде широка біла смужка, яка говорить про технологію виготовлення (дротяний). На них не може бути більше 4 смуг, останнє кільце говорить про властивості мікроелемента.

Завдання: розібрати резистори з лабораторних наборів за опором.

Контрольні питання:

Що таке резистор?

Що таке платформа Arduino?

Навіщо потрібно досить точно вміти визначати опір резисторів у роботі з Arduino?

3.2 Тема 2.1 Світлодіод.

Мета роботи: Знайомство з принципами розробки найпростіших електронних схем на базі макетної плати.

Завдання роботи

1. Розробити скетч для миготіння світлодіодом через цифровий пін №4 на платі Arduino.
2. Зібрати схему з світлодіодом на макетній платі.

Інструменти для виконання роботи

1. Комп'ютер з підключенням до мережі Internet.

2. Плата Arduino з USB виходом (наприклад, Arduino Leonardo).

3. Макетна плата для безпечної монтажу.

4. Світлодіод.

5. Резистор на 10-20 КОм.

6. Провід тато-тато 2 шт. Теоретическая часть

Цифрові Піни на мікроконтролерах Arduino можуть працювати в трьох режимах: INPUT, INPUT_PULLUP і OUTPUT.

У режимі OUTPUT можна програмно встановлювати на пін цифровий сигнал за допомогою функції digitalWrite. Для установки рівня використовуються зумовлені константи HIGH і LOW. Прийнято, що HIGH позначає високий рівень напруги (3.3 або 5 вольт в залежності від робочої напруги плати), а LOW - низький (близько 0). При цьому можливе живлення пристроїв зі споживанням струму до 40 мА. У режимі INPUT можна подавати цифровий сигнал на пін і програмно зчитувати його в мікроконтролері за допомогою функції digitalRead. При зчитуванні логічного значення буде отримано значення HIGH, якщо рівень поданої на пін напруги вище 2 вольт для 3.3 вольтової плати і 3 вольт для 5 вольтової плати. Для установки режиму роботи цифрового Піна використовується функція pinMode.

Завдання: описати, як працює і для чого потрібен режим INPUT_PULLUP

Для розробки простих електронних схем часто використовуються макетні плати. Вони бувають двох основних видів - для паяння та для монтажу без спаювання.

У даній роботі передбачається використання плати для монтажу без спаювання. Контакти цієї плати з'єднані всередині особливим чином, як показано на рисунку:

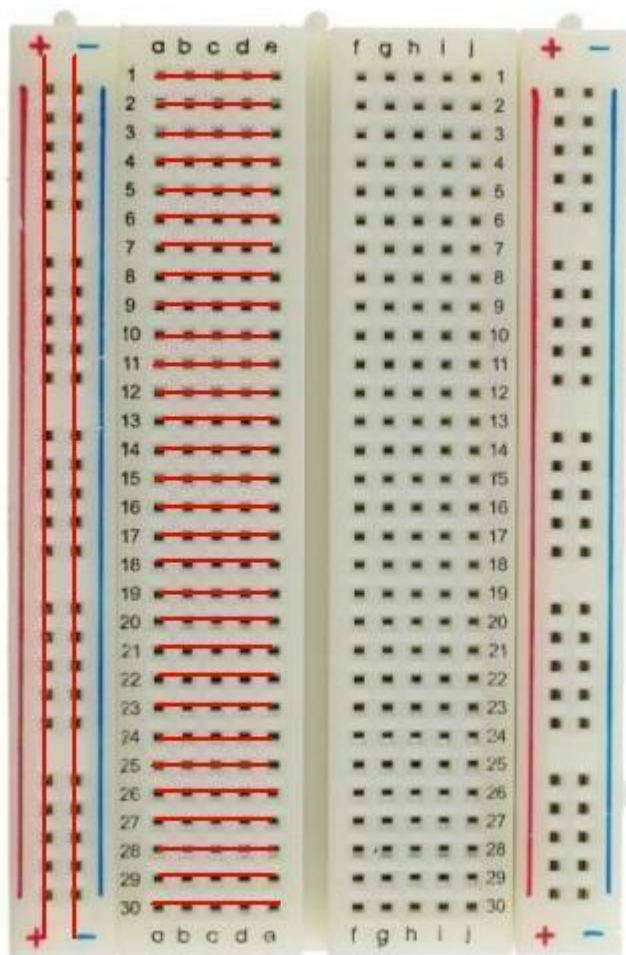


Рис. 4. - Плата для монтажу без спаювання.

Виконання роботи

Розробка скетчу для миготіння світлодіодом Через цифровий пін №4 на платі Arduino

Для керування світлодіодом через цифровий пін потрібно виконати дві дії:

1. Налаштувати цифровий пін в функції setup ():

```
pinMode(4,OUTPUT);
```

2. У функції loop змінювати стан піна з затримкою таким чином, щоб отримати ефект миготіння:

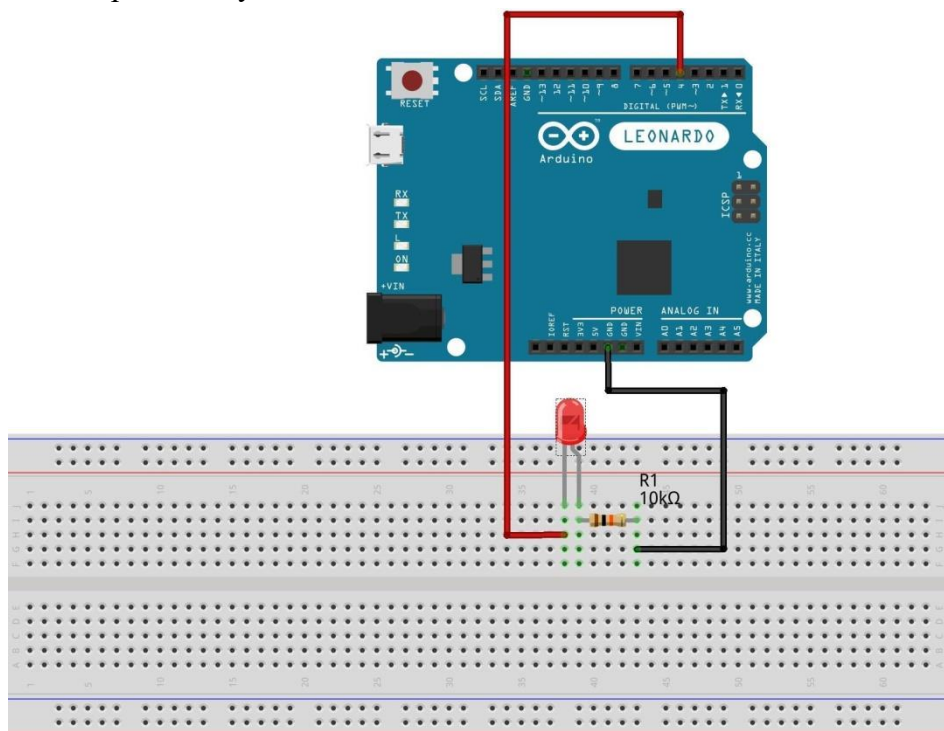
```
digitalWrite(4,HIGH);  
delay(1000);  
digitalWrite(4,LOW);  
delay(1000);
```

Завдання: перерахуйте функції стандартної бібліотеки для роботи з часом з описом їх роботи (в тому числі використовувану в скетчі функцію delay)

Збірка схеми з світлодіодом на макетній платі

Для того, щоб блимати світлодіодом, потрібно зібрати схему, в якій світлодіод підключений послідовно з резистором на 10-20 КОм, щоб не спалити світлодіод.

Зберіть схему:



fritzing

Рис. 5. – Зображення схеми

Завдання: додайте фотографію зібраної схеми

Завантажте скетч на плату, переконайтеся, що світлодіод блимає. Якщо він не блимає, спробуйте перевернути його (поміняти місцями ніжки).

Для довідки: для розробки схем підключення модулів до Arduino зручно використовувати безкоштовну програму Fritzing.

Контрольні питання:

Що таке пін?

Скільки пінів має платформа Arduino?

Що таке світлодіод?

В чому вимірюється яскравість світло діодів?

3.2 Тема 2.2 RGB-світлодіод. Яскравість світлодіода (Частина 2)

Мета роботи: дослідити RGB-світлодіод

Завдання:

1. Зібрати установку
2. Написати програму, що управляє світлодіодом
3. Виконати всі завдання

RGB-світлодіод поєднує в одному корпусі світлодіод червоного (red), зеленого (green) і синього (blue) кольорів. Такий світлодіод має чотири контакту, три з яких є анодами (кожен з них відповідає своєму кольору), і один загальний катод.

При роботі з трибарвним світлодіодом без вбудованих резисторів можна визначити призначення ніжки по її довжині. Найдовша - земля, коротше - зелений, ще коротше - блакитний, найкоротша - червоний. Необхідно використовувати три резистори - по одному на кожен анод.

Компоновка лабораторної установки

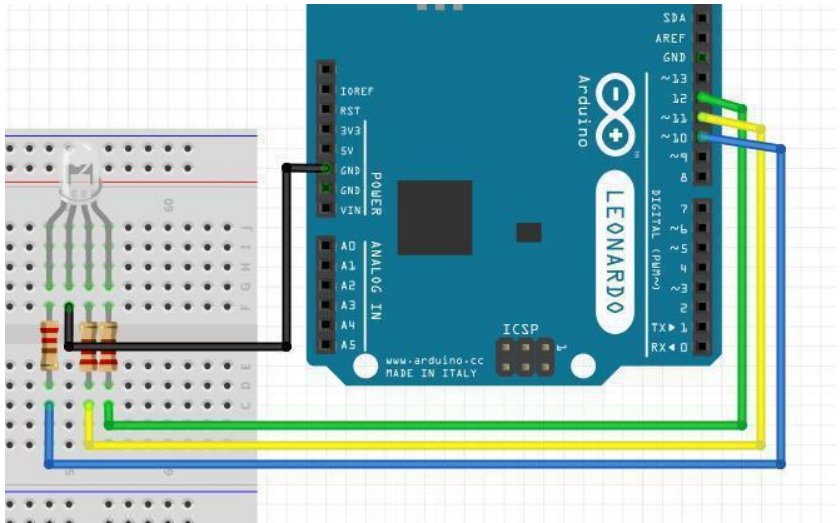


Рис.6. - Схема лабораторної установки.

Підготуємо всі необхідні нам деталі.

Компоненти лабораторної установки:

1. Контролер
2. Макетна плата
3. RGB-світлодіод
4. Резистори 220 Ом - 3 шт.
5. Дроти

Алгоритм виконання роботи:

1. з'єднуємо з виведенням №12
2. Катод RGB-світлодіоди підключаємо до землі через резистор 220 Ом;

Робота в середовищі програмування Arduino IDE

Напишемо код програми.

Оголосимо Піни, до яких буде підключений світлодіод, як

```
const int red = 12;  
const int green = 11;  
const int blue = 10;
```

константи.

У функції `setup ()` ініціалізуємо піни, підключені до світлодіоду, як вихід.

```
pinMode(red, OUTPUT);  
pinMode(green, OUTPUT);  
pinMode(blue, OUTPUT);
```

Перед тим як перейти до функції `loop ()`, розберемо функцію `showColors ()`. Вона нічого не повертає і нічого не приймає. У цій функції буде змінюватися колір світлодіода за рахунок зміни подачі напруги на кожен з кольорів.

```
void showColors()  
{  
  
}
```

Нічого не горить: на 3 Піна подається 0v.

```
digitalWrite(red, LOW);  
digitalWrite(green, LOW);  
digitalWrite(blue, LOW);
```

Чекаємо 1,5 секунди.

```
delay(1500);
```

Червоний: на пін, який відповідає за червоний колір, подається HIGH, а на інші - LOW.

```
digitalWrite(red, HIGH);  
digitalWrite(green, LOW);  
digitalWrite(blue, LOW);
```

Чекаємо 1,5 секунди.

```
delay(1500);
```

Синій: на пін, який відповідає за синій колір, подається HIGH, а на інші - LOW.

```
digitalWrite(red, LOW);  
digitalWrite(green, LOW);  
digitalWrite(blue, HIGH);
```

Чекаємо 1,5 секунди.

```
delay(1500);
```

Завдання: аналогічним чином продемонструйте синій колір. Що Ви додасте в код?

Чекаємо 1,5 секунди.

```
delay(1500);
```

Блакитний: на пін, який відповідає за червоний колір, подається LOW, а на інші - HIGH.

```
digitalWrite(red, LOW);  
digitalWrite(green, HIGH);  
digitalWrite(blue, HIGH);
```

Чекаємо 1,5 секунди.

```
delay(1500);
```

Фіолетовий: на пін, який відповідає за зелений колір, подається LOW, а на інші - HIGH.

```
digitalWrite(red, HIGH);  
digitalWrite(green, LOW);  
digitalWrite(blue, HIGH);
```

Чекаємо 1,5 секунди.

```
delay(1500);
```

Завдання: аналогічним чином продемонструйте жовтий колір. Що Ви додасте в код?

Білий: все горять.

```
digitalWrite(red, HIGH);  
digitalWrite(green, HIGH);  
digitalWrite(blue, HIGH);
```

Чекаємо 1,5 секунди.

```
delay(1500);
```

Функція `loop ()` буде містити тільки виклик процедури

```
void loop()  
{  
  showColors();
```

```
showColors().}
```

В результаті отримуємо наступний код:

```

const int red = 12;
const int green = 11;
const int blue = 10;

void setup()
{
  pinMode(red, OUTPUT);
  pinMode(green, OUTPUT);
  pinMode(blue, OUTPUT);
}

void loop()
{
  showColors();
}

void showColors()
{
  //Ничего не горит
  digitalWrite(red, LOW);
  digitalWrite(green, LOW);
  digitalWrite(blue, LOW);
  delay(1500);
  //Красный
  digitalWrite(red, HIGH);
  digitalWrite(green, LOW);
  digitalWrite(blue, LOW);
  delay(1500);
  //Зеленый
  digitalWrite(red, LOW);
  digitalWrite(green, HIGH);
  digitalWrite(blue, LOW);
  delay(1500);
  //Синий *необходимо дополнить*
  delay(1500);
  //Голубой (зеленый + синий)
  digitalWrite(red, LOW);
  digitalWrite(green, HIGH);
  digitalWrite(blue, HIGH);
  delay(1500);
}

```

```

    //фиолетовый (красный + синий)
    digitalWrite(red, HIGH);
    digitalWrite(green, LOW);
    digitalWrite(blue, HIGH);
    delay(1500);
    //Желтый (красный + зеленый) *необходимо дополнить*
    delay(1500);
    //Белый (все горят)
    digitalWrite(red, HIGH);
    digitalWrite(green, HIGH);
    digitalWrite(blue, HIGH);
    delay(1500);
}

```

Тепер напишіть цю програму:

```

sketch_sep05b
void setLeds(int r,int g,int b)
{
    digitalWrite(10,r);
    digitalWrite(11,g);
    digitalWrite(12,b);
    delay(1500);
}

void setup() {
    pinMode(10,OUTPUT);
    pinMode(11,OUTPUT);
    pinMode(12,OUTPUT);
}

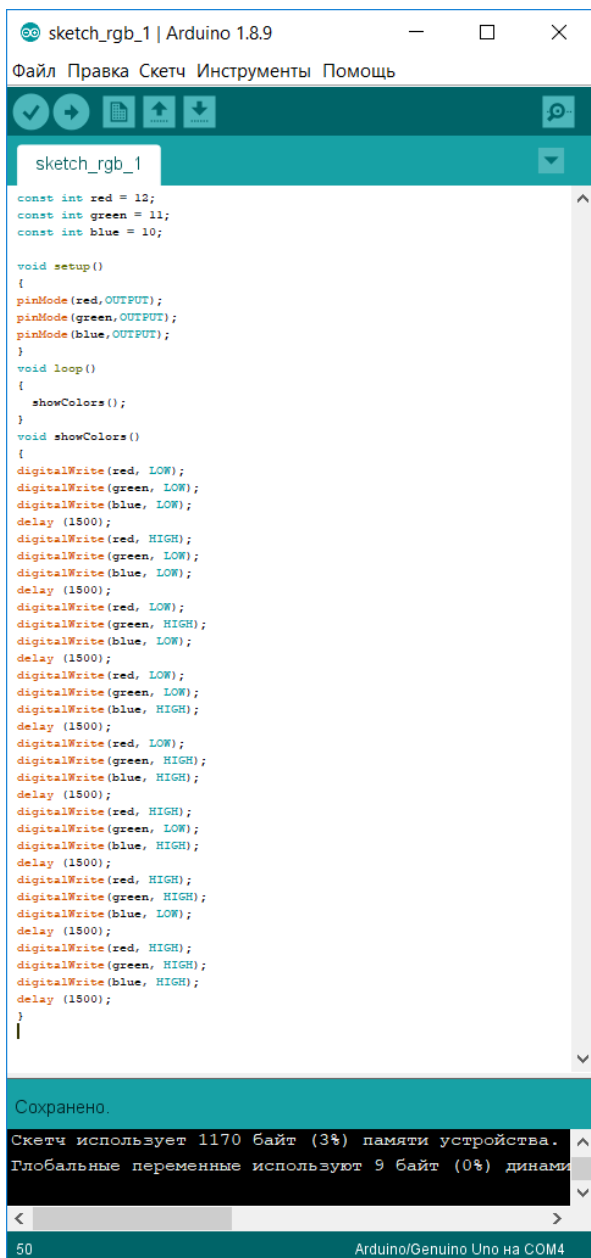
void loop() {
    setLeds(LOW,LOW,LOW);
    setLeds(HIGH,LOW,LOW);
    setLeds(LOW,HIGH,LOW);
    setLeds(LOW,LOW,HIGH);
    setLeds(LOW,HIGH,HIGH);
    setLeds(HIGH,LOW,HIGH);
    setLeds(HIGH,HIGH,LOW);
    setLeds(HIGH,HIGH,HIGH);
}

```

Сохранено.

Скетч использует 1102 байт (3%) памяти устройства.
Глобальные переменные используют 9 байт (0%) динамической памяти.

24 Arduino/Genuino Uno на COM4



Контрольні питання. Порівняйте дві програми і опишіть їх різницю, коли може бути актуальна одна програма, а коли друга - в чому перевага маленького скетчу, в чому перевага великого. (Це питання на оцінку не впливає, велике прохання не цікавитися один у одного відповідями, цікавий хід думки).

3.3 Тема 3. Потенціометр

Мета роботи: дослідити потенціометр

Завдання:

1. Зібрати установку
2. Написати програму
3. Виконати всі завдання

Потенціометр - це змінний резистор з регульованим опором. З його допомогою ми будемо регулювати яскравість світлодіода.

Завдання: зберіть інформацію про види потенціометрів.

Збірка лабораторної установки

Компоненти лабораторної установки:

1. Контролер
2. Макетна плата
3. Потенціометр
4. Резистор 220 Ом
5. Дроти

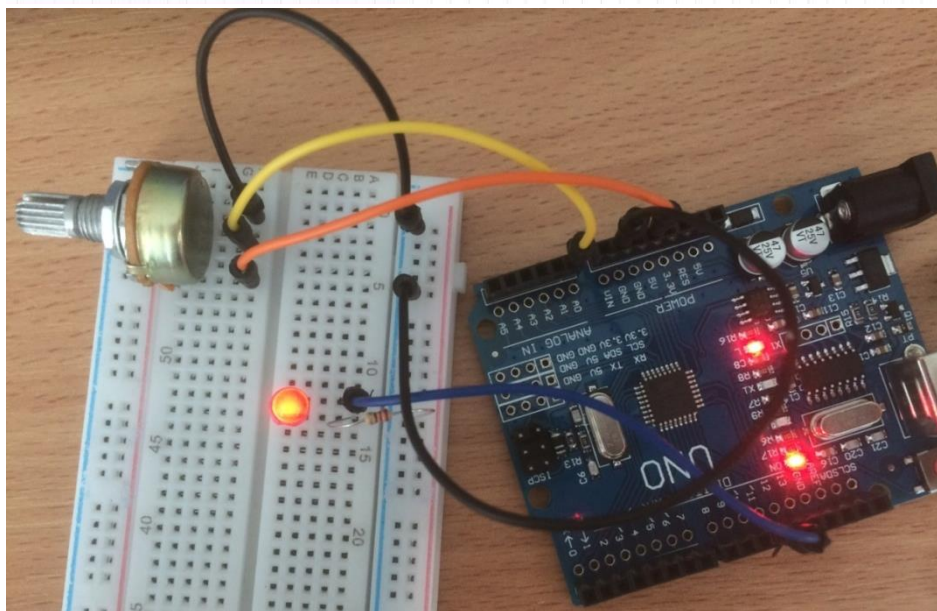
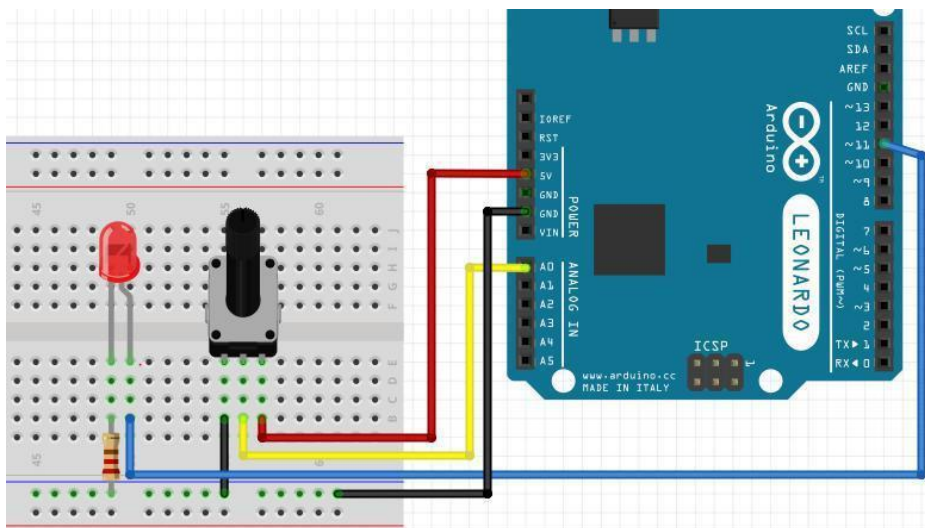


Рис. 7. - Схема лабораторної установки

Алгоритм виконання роботи:

1. Анод світлодіода з'єднуємо з піном №11
2. Катод підключаємо до землі через резистор 220 Ом;
3. Підключаємо потенціометр. Потенціометр має 3 конектори.
 - а) Перший конектор підключаємо до землі.
 - б) Центральний конектор - до аналогового входу A0.
 - в) Третій - до живлення.

Робота в середовищі програмування Arduino IDE

Напишемо код програми.

Оголосимо Піни, до яких буде підключено потенціометр і світлодіод, як константи.

```
const int potentiometr = 0;  
const int ledLamp = 11;
```

Задамо змінну, в яку буде зчитуватися значення потенціометра.

```
int potentiometrValue;
```

У функції setup () встановлюємо пін ledlamp в якості вихідного.

```
pinMode(ledLamp, OUTPUT);
```

Переходимо до функції loop ().

```
loop()  
{  
  
  
}
```

Зчитуємо значення аналогового Піна potentiometer.

```
potentiometrValue = analogRead(potentiometr);
```

```
analogWrite(ledLamp, potentiometrValue / 4);
```

В результаті отримаємо наступний код:

```

const int potentiometr = 0;
const int ledLamp = 11;
int potentiometrValue;

void setup()
{
  pinMode(ledLamp, OUTPUT);
}

void loop()
{
  potentiometrValue = analogRead(potentiometr);
  analogWrite(ledLamp, potentiometrValue / 4);
}

```

Зібрана установка повинна змінювати стан світлодіода раз в 2 секунди.

Збірка лабораторної установки

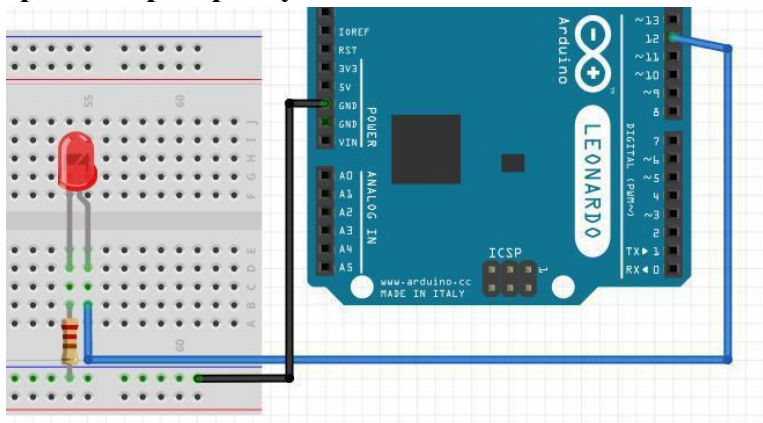


Рис. 8. - Схема лабораторної установки.

Алгоритм виконання роботи:

1. Анод з'єднуємо з виведенням №12
2. Катод світлодіода підключаємо до землі через резистор 220 Ом;

3.

Робота в середовищі програмування Arduino IDE

Напишемо код програми.

Оголосимо пін, до якого буде підключений світлодіод.

```
int ledLamp = 12;
```

У функції `setup ()` ініціалізуємо пін, підключений до світлодіоду, як вихід.

```
pinMode(ledLamp, OUTPUT);
```

Переходимо до функції `loop ()`.

```
loop()
```

```
{
```

```
}
```

Для включення світлодіода необхідна команда `digitalWrite (pin, value)`. Вона встановлює цифровий вивід на потрібний пін. Параметр `pin` означає номер Піна для запису, а `value` - значення запису. Другий параметр може приймати тільки 2 значення: `HIGH` (5v або 3.3v) або `LOW` (0v).

Включимо світлодіод.

```
digitalWrite(ledLamp, HIGH);
```

Також скористаємося функцією `delay (value)`, яка здійснює затримку на потрібний час. У параметр передається значення часу затримки в мілісекундах.

Виконаємо паузу на 2 секунди.

```
delay(2000);
```

Тепер вимкнемо світлодіод і зробимо затримку на 2 секунди.

```
digitalWrite(ledLamp, LOW);
```

```
delay(2000);
```

В результаті отримуємо наступний код:

```

int ledLamp = 12;

void setup() {

    pinMode(ledLamp, OUTPUT); // режим светодиода - вивод

}

void loop() {

    digitalWrite(ledLamp, HIGH); // включение светодиода
    delay (2000);                //задержка
    digitalWrite(ledLamp, LOW);  //выключение светодиода
    delay (2000);

}

```

Завдання: змінити код програми так, щоб затримка збільшувалася з кожним разом в 2 рази. Як він зміниться?

Контрольні питання:

Що таке потенціометр та для чого він потрібен?

Що таке контролер?

Що таке сервопривід?

3.4 Тема 4. Керування серводвигуном

Мета роботи: навчання Arduino

Завдання: навчитися керувати серводвигуном

Збірка лабораторної установки

Для досягнення поставленої задачі, підготуємо всі необхідні нам деталі.

Компоненти лабораторної установки:

1. Контролер Arduino UNO
2. Сервопривід
3. Провід для підключення

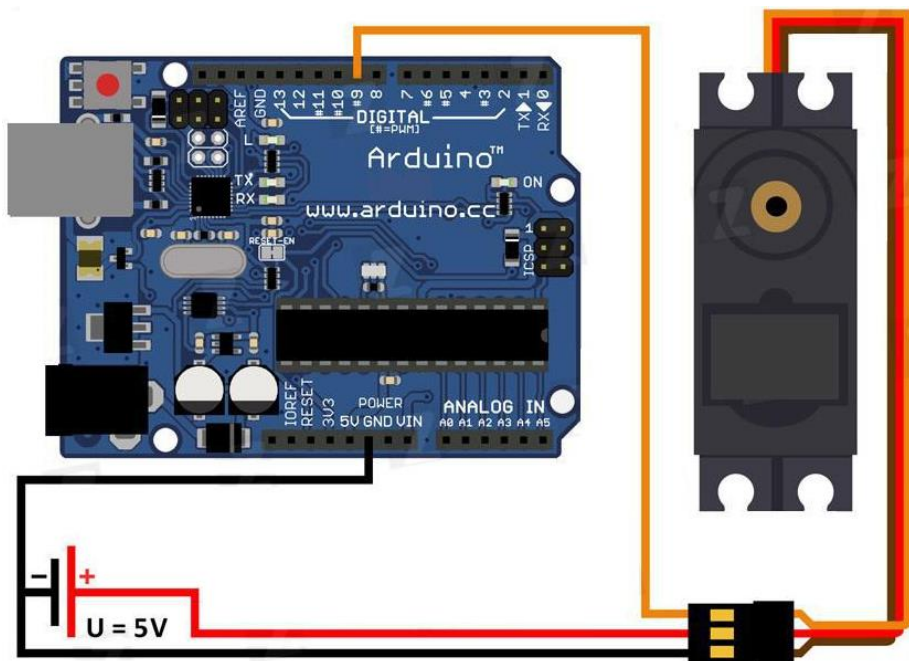


Рис. 9. - Схема лабораторної установки.

Алгоритм виконання роботи:

1. Підключаємо пін, підключаючи помаранчевий провід до піну №9
2. Підключаємо напруга (червоний)
3. Підключаємо заземлення (чорний)
4. Підключаємо мікроконтролер до мережі.

Робота в середовищі програмування Arduino IDE

Тепер напишемо код програми:

```
#include <Servo.h> // Підключення бібліотеки Servo.h
Servo servoPrivod; // Створюємо об'єкт типу Servo для
роботи з сервоприводом

int grad = 0; // Кут повороту сервоприводу
void setup ()
{
    servoPrivod.attach (9); // Підключаємо сервопривід до піну 9
}

void loop () {
    servoPrivod.write (0); // Ставимо кут повороту: 0
    delay (2000); // Чекаємо 2 секунди
    servoPrivod.write (90); // Ставимо кут повороту: 90
    delay (2000); // Чекаємо 2 секунди
    servoPrivod.write (180); // Ставимо кут повороту: 180
    delay (2000); // Чекаємо 2 секунди
    servoPrivod.write (0); // Ставимо кут повороту: 0
    delay (2000); // Чекаємо 2 секунди
    servoPrivod.writeMicroseconds (1500); // Встановимо
тривалість імпульсу
    grad = servoPrivod.read (); // Прочитуємо значення
поточного положення сервоприводу
    for (int i = grad; i > 0; i - = 1)
    { // Цикл від grad до 0 градусів з кроком 1
        servoPrivod.write (i); // Передати на сервопривід кут
        повороту
        delay (100); // Затримка на 0,1 сек
    }
    delay (1000); // Затримка на 1 сек
```

}

Так само Вам можуть знадобитися наступні команди:
`myservo.attached ()`; // Перевіряємо, прив'язаний чи сервопривід. Повертає логічне значення bool.

`myservo.detach ()`; // Вимикає сервопривід від Піна.

Контрольні питання. 1. Виконайте завдання: напишіть рядок програми, в якій буде присутній підключення сервоприводу до пину №10.

2. В яких межах можна задавати кут повороту за допомогою функції `write ()`?

- а) від 0 до 360 градусів
- б) від -180 до 180 градусів
- в) від 0 до 180 градусів

3. Якого типу значення повертає функція `read ()`?

4. Заповніть пропуск: за допомогою стандартної бібліотеки `Servo` можна управляти не більше, ніж ... сервоприводами

5. На яких пінах немає ШІМ?

- а) 8, 9, 10
- б) 9, 10
- в) 8, 10

6. З чого складається сервопривід?

7. Принцип роботи сервоприводу?

8. Чи можна підключати сервопривід від джерела зовнішнього живлення?

3.5 Тема 5. Дзвінок

Мета роботи: навчитися керувати зумером

Завдання:

1. Зібрати установку
2. Написати програму
3. Виконати всі завдання

У цій лабораторній роботі зберемо простий варіант дзвінка, який буде дзвеніти при натисканні кнопки.

Збірка лабораторної установки

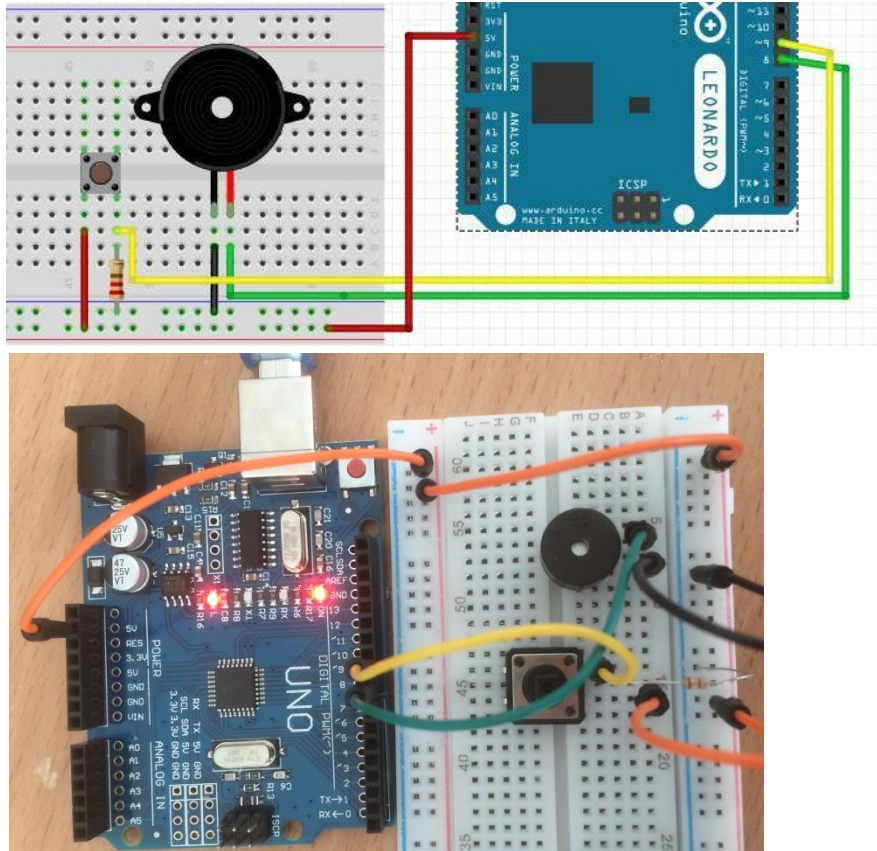


Рис. 10 - Схема лабораторної установки.

Алгоритм виконання роботи:

1. Підключаємо кнопки:
 - а) З'єднуємо пін 9 з контактом тактовою кнопки;
 - б) До цього ж контакту підключаємо резистор 10 кОм, який з'єднуємо з землею;
 - в) Інший вихід кнопки з'єднуємо з живленням 5 v;
2. Підключаємо зумер
 - а) З'єднаємо позитивний вивід зумера з піном 8;
 - б) Виведемо негативний вивід зумера на землю

Робота в середовищі програмування Arduino IDE

Напишемо код програми.

Задаємо постійні Піни для зумера і кнопки.

```
const int buzzer = 8;  
const int button = 9;
```

У функції setup () встановлюємо пін кнопки на вхід, а пін зумера на вихід.

```
pinMode(buzzer, OUTPUT);  
pinMode(button, INPUT);
```

Переходимо до функції loop ().

```
loop()  
{  
  
}
```

Якщо кнопка натиснута, то подаємо на зумер звук частоти 294 Гц на 100 мілісекунд.

```
if (digitalRead(button) == HIGH)  
tone(buzzer, 294, 100);
```

В результаті отримуємо наступний код:

```
const int buzzer = 8;
const int button = 9;

void setup()
{
  pinMode(buzzer, OUTPUT);
  pinMode(button, INPUT);
}

void loop()
{
  if (digitalRead(button) == HIGH)
    tone(buzzer, 294, 100);
}
```

Завдання: зберіть інформацію про функції tone (...) і noTone ().

Контрольні питання:

Що таке зумер?

Для чого в роботі використовується кнопка?

3.6 Тема 6. Шумомір

Мета роботи: вивчити датчик звуку

Завдання:

1. Зібрати шумомір
 2. Написати програму, що управляє датчиком звуку
 3. Виконати всі завдання
- блок схема

Завдання: зібрати інформацію про принципи виготовлення датчиків звуку, приведіть порівняльну таблицю про різні мікрофони.

У цій лабораторній роботі ми познайомимося з аналоговим датчиком визначення звуку і створимо на його основі шумометр. Результати вимірювань будуть виводитися за допомогою світлодіодів - чим вище рівень звуку, тим більше світлодіодів в лінійці буде загорятися.

Складається датчик з плати, на якому змонтовані виходи, підсилювача звуку, підлаштування резистора і електронного мікрофона. Регулятором чутливості можна вибирати, від якого звуку буде спрацьовувати датчик. Дана плата дозволяє перевести звукові коливання в цифровий сигнал. При коливанні мембрани в мікрофоні від звукових хвиль, змінюється ємність його конденсатора, внаслідок чого виявляється зміна напруги на виходах датчика звуку, відповідне звуковим сигналом.

Збірка лабораторної установки

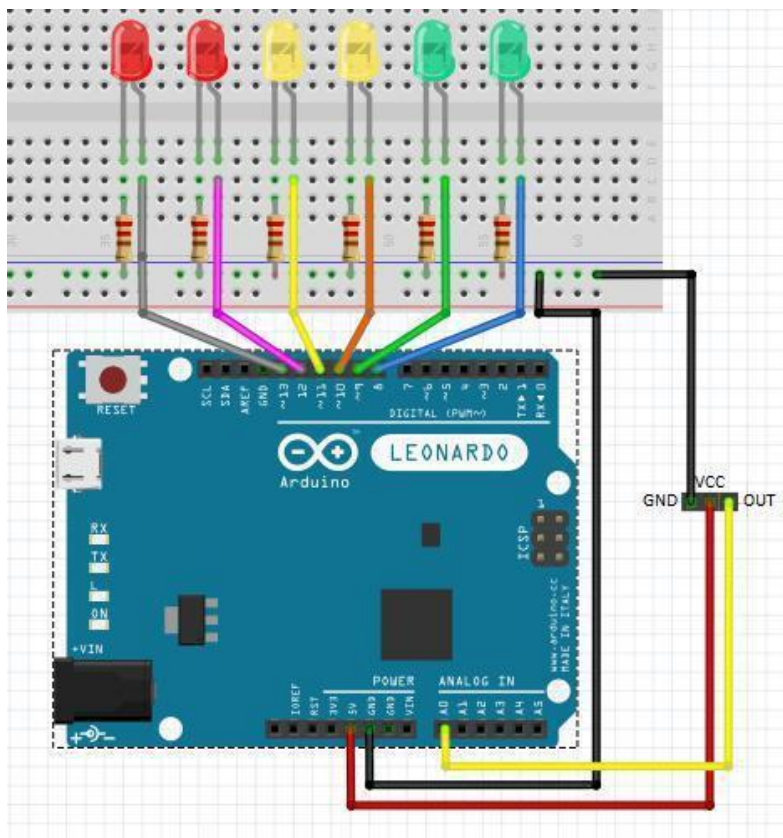


Рис. 11 - Схема лабораторної установки.

Підготуємо всі необхідні нам деталі.

Алгоритм виконання роботи:

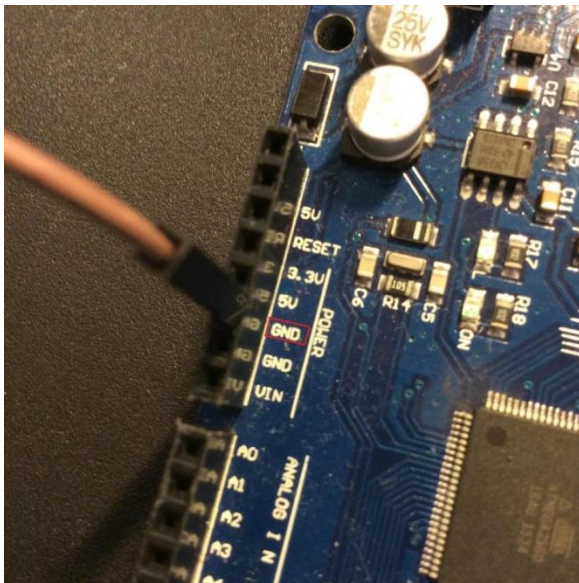
1. Підключаємо світлодіоди:

а) катод кожного з світлодіодів підключаємо до землі через резистор 220 Ом;

б) аноди з'єднуємо з одним з цифрових виводів Ардуїно. В нашому випадку це 8, 9, 10, 11, 12 і 13

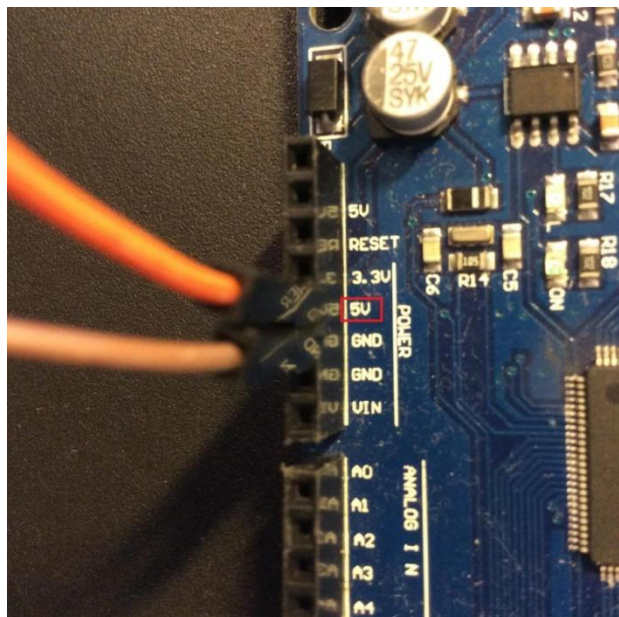
2. Підключаємо датчик:

а)



б)





B)



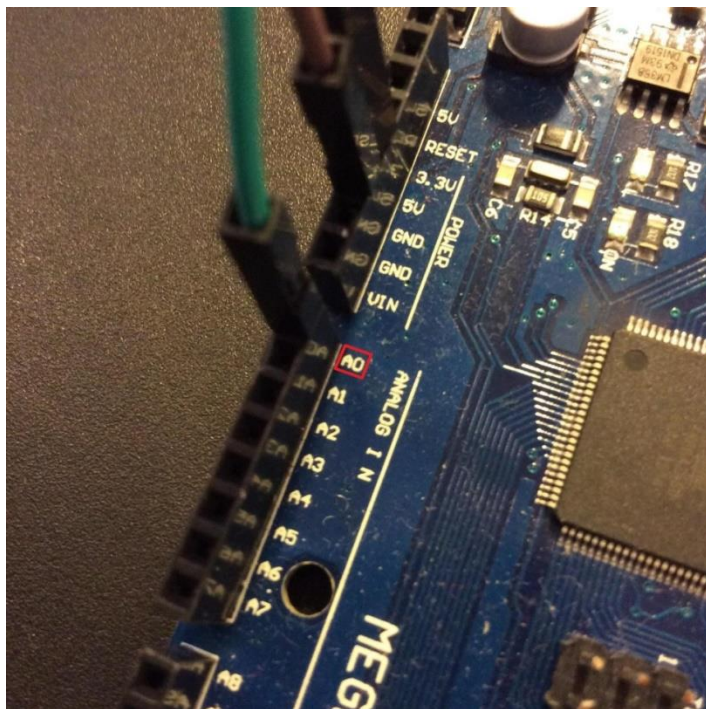


Рис.12 – Підключення а) з'єднаємо вихід Gnd з землею, б) Vcc (+) – з живленням 5v, в) підключаємо висновок Out до аналогового порту "A0" Arduino.

Робота в середовищі програмування Arduino IDE

Визначаємо пін датчика звуку і граничне значення.

```
#define SOUND_SENSOR A0
#define THRESHOLD_VALUE 50
```

Оголосимо масив пинов для світлодіодів

```
int ledMass[6] = {8, 9, 10, 11, 12, 13};
```

У функції setup () встановлюємо Піни для світлодіодів на вихід і пін датчика на вхід.


```

void setup()
{
    for (int i=0; i<6; i++)
        pinMode(ledMass[i], OUTPUT);
    pinMode(SOUND_SENSOR, INPUT);
}

```

Перед тим як перейти до функції loop () розберемо 2 додаткові функції: turnOnLED (int sensorValue) і turnOffLED (). turnOnLED включає світлодіоди і приймає значення сигналу з датчика. turnOffLED вимикає світлодіоди.

Переходимо до функції turnOnLED.

```

void turnOnLED(int sensorValue)
{
    // ...
}

```

Визначимо рівень шуму.

```

int leds = sensorValue / 50;

```

Оскільки у нас 6 світлодіодів, потрібно обмежити значення.

```

leds = constrain(leds, 0, 6);

```

Запалюємо потрібне значення світлодіодів.

```

for (int i = 0; i < leds; i++)
    digitalWrite(ledMass[i], HIGH);

```

Внаслідок отримуємо наступну функцію:

```

void turnOnLED(int sensorValue)
{
    int leds = sensorValue / 50;
    leds = constrain(leds, 0, 6);
    for (int i = 0; i < leds; i++)
        digitalWrite(ledMass[i], HIGH);
}

```

Розберемо функцію turnOffLED. У циклі вимикаємо все світлодіоди.


```
void turnOffLED()
{
    for (int i = 0; i < 6; i++)
        digitalWrite(ledMass[i], LOW);
}
```

Переходимо до функції loop ()

```
void loop()
{
    }
```

Прочитуємо сигнал з датчика.

```
int sensorValue = analogRead(SOUND_SENSOR);
```

Якщо значення сигналу більше, ніж порогове, то викликаємо функцію запалювання світлодіодів turnOnLED

і чекаємо 100 мс.

```
if(sensorValue > THRESHOLD_VALUE)
{
    turnOnLED(sensorValue);
    delay(100);
};
```

Далі вимикаємо світлодіоди.

```
turnOffLED();
```

В результаті отримуюмо наступний код:

```
#define SOUND_SENSOR A0
#define THRESHOLD_VALUE 50
int ledMass[6] = {8, 9, 10, 11, 12, 13};

void setup()
{
    for (int i=0; i<6; i++)
        pinMode(ledMass[i], OUTPUT);
    pinMode(SOUND_SENSOR, INPUT);
}

void loop()
{
    int sensorValue = analogRead(SOUND_SENSOR);
    if(sensorValue > THRESHOLD_VALUE)
    {
        turnOnLED(sensorValue);
        delay(100);
    };
    turnOffLED();
}

void turnOnLED(int sensorValue)
{
    int leds = sensorValue / 50;
    leds = constrain(leds, 0, 6);
    for (int i = 0; i < leds; i++)
        digitalWrite(ledMass[i], HIGH);
}

void turnOffLED()
{
    for (int i = 0; i < 6; i++)
        digitalWrite(ledMass[i], LOW);
}
```

Завдання: зробити шумометр, в якому чим вище рівень шуму, тим менше світлодіодів буде загорятися. Як зміниться код програми?

Контрольні питання:

В яких одиницях вимірюється шум?

Які способи уникнення шуму Ви знаєте?

Які норми зашумленості?

Назвіть пороги чутливості шуму для людини, собаки та поясніть чому вони різні.

3.7 Тема 7. Керування шаговим мотором.

Мета роботи: навчання Arduino

Завдання: навчитися керувати кроковим двигуном

Збірка лабораторної установки

Особлива примітка! У наборах є крокові двигуни, які не відповідають технічним нормам ДСТУ, в одній і тій-же серії двигуна - різні передавальні числа редукторів. Замість одного обороту, двигун, може робити 4.

Для досягнення поставленої задачі, підготуємо всі необхідні нам деталі.

Компоненти лабораторної установки:

1. Контролер Arduino UNO
2. Кроковий двигун.
3. Драйвер для крокової двигуна.
4. Джерело живлення.
5. Дріт для підключення.

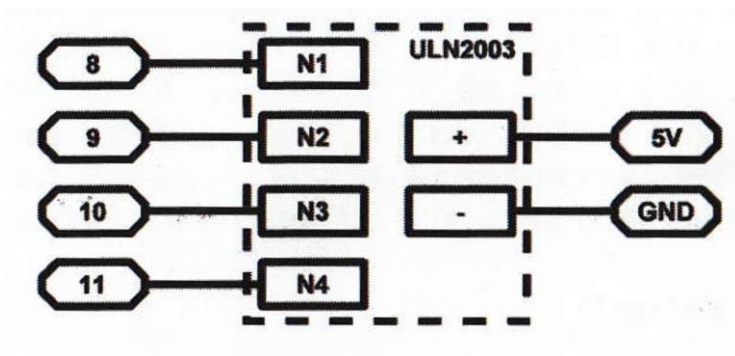


Рис. 13 - Схема лабораторної установки.

Алгоритм виконання роботи:

1. Підключаємо піни від драйвера крокової двигуна до мікроконтролеру з 8 по 11 (сірий, фіолетовий, синій, зелений).
2. Підключаємо кроковий двигун до драйверу.
3. Для живлення рекомендується використовувати джерело зовнішнього живлення. (В даній лабораторній роботі використовуємо контролер як джерело живлення).
4. Підключаємо мікроконтролер до мережі.

Робота в середовищі програмування Arduino IDE

Код програми:

```
#include <Stepper.h> // Підключення бібліотеки для роботи з кроковим двигуном
```

```
const int STEPS_MOTOR = 32; // Число кроків для повного обороту двигуна
```

const int STEPS_CIRCLE = 2048; // Число кроків для
одного повного обороту валу редуктора двигуна

// Згідно характеристикам 28BYJ-48
кут повороту на один крок $5,625^\circ$

// Виходить, що таких кроків буде $360^\circ / 5,625^\circ = 64$

// $2048 = 32 * 64$

Stepper stepper (STEPS_MOTOR, 8, 10, 9, 11); //
Створюємо об'єкт Stepper з відповідними пинами і числом
кроків для мотора

void setup () {}

void loop ()

{

stepper.setSpeed (100); // Задамо швидкість

stepper.step (45); // Виконати 45 кроків за годинниковою
стрілкою

delay (1000); // затримка на 1 секунду

stepper.setSpeed (300); // Встановити швидкість швидше

stepper.step (-45); // Виконати 45 кроків проти
годинникової стрілки

delay (1000); // Затримка на 1 секунду

```
stepper.step (STEPS_CIRCLE); // Виконати повний  
оборот за годинниковою стрілкою
```

```
delay (1500); // Затримка 1,5 секунди
```

```
}
```

Важливо знати! Номери пинов записуємо саме в цьому порядку Stepper stepper (STEPS_MOTOR, 8, 10, 9, 11).

Важливо знати! Не ставити швидкість крокової двигуна понад 1000, якщо не використовувати зовнішнє джерело живлення. Мотор починає гудіти і не крутиться, через брак потужності, це може привести до виходу з ладу лабораторне обладнання.

Додаткова бібліотека для роботи з кроковим двигуном – AccelStepper
(https://www.pjrc.com/teensy/td_libs_AccelStepper.html).

Контрольні питання:

1. Скільки кроків необхідно задати двигуну для повного обороту валу редуктора?
2. Виберіть правильну відповідь:
Функція setSpeed (n)
 - а) Змушує двигун обертатися
 - б) Встановлює швидкість обертання і змушує двигун обертатися
 - в) Тільки встановлює швидкість обертання
3. У чому вимірюється швидкість двигуна? (оберти за хвилину)
4. Чи є функція step () блокуючою?

5. При вказівці в дужках в функції `step()` позитивного числа, двигун буде обертатися
- а) за годинниковою стрілкою
 - б) проти годинникової стрілки
 - в) не обертатися
6. З чого складається кроковий мотор?
7. Як працює кроковий мотор?
8. Чи можна підключати живлення безпосередньо до мікроконтролера?

СПИСОК ЛІТЕРАТУРИ

1. Основи мехатроніки : навч. посіб. / уклад.: С. М. Пересада, М. В. Пушкар. Київ : КПІ ім. Ігоря Сікорського, 2020. 137 с.
2. Шликов В. В. Мікропроцесорна техніка. Практикум : навч. посіб. Київ : КПІ ім. Ігоря Сікорського, 2018. 145 с.
3. Основи мехатроніки : навч. посіб. / О. М. Артюх, О. В. Дударенко, В. В. Кузьмін та ін. Запоріжжя : НУ «Запорізька політехніка», 2021. 372 с.
4. Трет'як А. В., Кльон А. М. Основи робототехніки : навч. посіб. Полтава : Національний університет «Полтавська політехніка імені Юрія Кондратюка», 2024. 135 с.
5. Михайлов Є. П., Лінгур В. М. Маніпулятори та промислові роботи : навч. посіб. Одеса : ОНПУ, 2019. 233 с.
6. Пелевін Л. Є., Почка К. І., Гаркавенко О. М., Мішук Д. О., Русан І. В. Синтез робототехнічних систем в машинобудуванні : підручник. Київ : ТОВ «НВП «Інтерсервіс»», 2016. 258 с.
7. Banzi M., Shiloh M. Getting Started with Arduino. 4th ed. Sebastopol : Maker Media, 2022. 240 p.
8. Monk S. Programming Arduino: Getting Started with Sketches. 3rd ed. New York : McGraw-Hill Education, 2022. 192 p.
9. Boxall J. Arduino Workshop: A Hands-on Introduction with 65 Projects. 2nd ed. San Francisco : No Starch Press, 2021. 432 p.
10. Margolis M., Jepsen B., Weldin N. R. Arduino Cookbook. 3rd ed. Sebastopol : O'Reilly Media, 2020. 796 p.
11. Bolton W. Mechatronics: Electronic Control Systems in Mechanical and Electrical Engineering. 7th ed. Harlow : Pearson Education, 2021. 728 p.

12. Alciatore D. G., Histanal M. B. Introduction to Mechatronics and Measurement Systems. 6th ed. New York : McGraw-Hill Education, 2021. 624 p.

13. Craig J. J. Introduction to Robotics: Mechanics and Control. 4th ed. Harlow : Pearson Education, 2018. 408 p.

14. Arduino Documentation. Language Reference. URL: <https://docs.arduino.cc/language-reference/> (дата звернення: 12.05.2025).

15. Arduino Documentation. Servo Library. URL: <https://docs.arduino.cc/libraries/servo/> (дата звернення: 12.06.2025).

Навчальне видання

МЕТОДИЧНІ ВКАЗІВКИ
до виконання лабораторних робіт з дисципліни
**«Мехатронні та робототехнічні системи: базові
принципи і практика»**

*Для здобувачів вищої освіти спеціальності G7 –
«Автоматизація, комп'ютерно-інтегровані технології
та робототехніка»
усіх форм навчання
(електронне видання)*

Укладачі СОНІКОВА Тетяна Геннадіївна,
АСМАНКІНА Анастасія Анатоліївна

Оригінал-макет Асманкіна А.А.

Підписано до друку _____
Формат 60x84¹/₁₆. Папір типограф. Гарнітура Times.
Друк офсетний. Умов. друк. арк. №. Облік. вид. арк.
Тираж ____ екз. Вид. № _____. Замов. № _____. Ціна
договірна.

Видавництво Східноукраїнського національного
університету
імені Володимира Даля

Адреса видавництва: вул. Іоанна Павла II, 17
Тел.+38(050)218 04 78, факс (06452) 4 03 42
e-mail: vidavnictvoSNU.ua@gmail.com