МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Рівненський державний гуманітарний університет

Кафедра інформаційних технологій та моделювання

ІНТЕРНЕТ РЕЧЕЙ

методичні вказівки

до виконання лабораторних робіт

ДЛЯ СТУДЕНТІВ СПЕЦІАЛЬНОСТЕЙ
122 Комп'ютерні науки
113 Прикладна математика
121 Інженерія програмного забезпечення

РІВНЕ – 2024

М 54
УДК 004.77 :681.586 (072)

«Інтернет речей : методичні вказівки до виконання лабораторних робіт» для студентів спеціальностей 122 Комп'ютерні науки, 113 Прикладна математика, 121 Інженерія програмного забезпечення / [уклад. Н.В. Шинкарчук] ; Рівне : РДГУ, 2024. 86 с.

Укладачі: кандидат технічних наук, доцент кафедри інформаційних технологій та моделювання **Шинкарчук Н. В.**

Рецензенти: доктор технічних наук, професор, завідувач кафедри комп'ютерних наук та прикладної математики Національного університету водного господарства та природокористування **Турбал Ю. В.**

кандидат фізико-математичних наук, доцент, завідувач кафедри інформаційних технологій та моделювання Рівненського державного гуманітарного університету **Мороз І. П.**

Методичні вказівки покликані сприяти кращому розумінню і засвоєнню лабораторного матеріалу з дисципліни «Інтернет речей» студентами спеціальностей 122 Комп'ютерні науки, 113 Прикладна математика, 121 Інженерія програмного забезпечення.

Розглянуто та схвалено на засіданні кафедри інформаційних технологій та моделювання, протокол № 3 від 26.03.2024 р.

Розглянуто та рекомендовано до друку навчально-методичною комісією факультету математики та інформатики Рівненського державного гуманітарного університету, протокол № 3 від 27.03.2024 р.

ЗМІСТ

ВСТУП

Методичні вказівки призначені для студентів і викладачів закладів вищої та професійної освіти, для всіх тих, хто зараз у навчанні чи практичній діяльності починає вивчати інноваційну технологію Інтернет речей.

Інтернет речей (Internet of things, IoT) – це концепція мережі, яка складається із взаємозв'язаних фізичних пристроїв, що мають вбудовані датчики, а також програмне забезпечення, яке дозволяє здійснювати передачу даних між навколишнім середовищем і комп'ютерними системами за допомогою використання стандартних протоколів зв'язку. При цьому, датчик – засіб вимірювання, який формує аналоговий або цифровий сигнал у формі зручній для передачі, обробки і зберігання.

Основною властивістю Інтернету речей є можливість підключення до всесвітньої мережі всіляких об'єктів (речей), які людина може використовувати в повсякденному житті, наприклад – годинник. Під терміном «Інтернет речей» розуміється також збір і обмін даними між різними фізичними пристроями на базі мережі, яка обов'язково є підключеною до Інтернету.

Розвиток Інтернету речей найближчим часом буде характеризуватись рядом тенденцій та змін, які вплинуть на різні аспекти повсякденного життя та бізнес. Ось деякі з таких тенденцій: IoT продовжуватиме «розширюватися», включаючи все більше пристроїв; повсюдне впровадження IoT вимагатиме зусиль зі стандартизації та забезпечення сумісності між різними пристроями та системами; із збільшенням числа підключених пристроїв, питання кібербезпеки стануть ще актуальнішими; IoT-пристрої генеруватимуть величезну кількість «сирих» даних; розвиток бездротових технологій передачі даних та низькоспоживаючих пристроїв дозволить зменшити енергоспоживання IoT-пристроїв; IoT буде відігравати ключову роль у розвитку «розумних» міст, включаючи управління транспортом та покращення міської інфраструктури; IoT-технології будуть ще ефективніше використовуватися для моніторингу стану здоров'я пацієнтів; IoT-системи будуть застосовуватися в процесі оптимізації збирання та утилізації відходів, а також управління водопостачанням та електромережами. В підсумку, «процвітання» Інтернету речей продовжуватиметься і робитиме значний вплив на різні сфери людської діяльності.

Методичні вказівки містять дванадцять лабораторних робіт. У яких детально розглянуто електронні компоненти, різноманітні датчики, апаратне та інструментальне забезпечення мікроконтролера Arduino Uno R3 і одноплатного комп'ютера Raspberry Pi 3 Model B.

СТРУКТУРА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Назви змістових модулів і тем	Кількість годин											
	денна форма						*заочна форма*					
	усього	*у тому числі*					усього	*у тому числі*				
		л	п	лаб	інд	с.р.		л	п	лаб	інд	с.р.
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
Змістовий модуль 1. Інноваційна технологія Інтернет речей. Архітектура Інтернету речей.												
Тема 1. Інтернет речей (IoT): технологічний тренд сучасних інформаційних технологій.	10	2				8	10	2				8
Тема 2. Екосистема і безпека IoT. Засоби ідентифікації, вимірювання і передачі даних в IoT-мережах.	10	2		2		6	10	1				9
Тема 3. Проекти Інтернету речей. Сфери використання технології Інтернет речей.	10	2				8	10	1				9
Тема 4. Комунікаційні технології Інтернету речей. Бездротові стандарти передачі даних в IoT-мережах.	10	2		2		6	10					10
Разом за змістовим модулем 1	40	8		4		28	40	4				36
Змістовий модуль 2. Датчик і живлення. Raspberry Pi. GPIO.												
Тема 5. Поняття датчика (сенсора). Цифрові і аналогові датчики.	10	2		2		6	10			1		9
Тема 6. Датчик (сенсор). Інтерфейси підключення і живлення датчиків.	10	2		2		6	10					10
Тема 7. Одноплатний комп'ютер Raspberry Pi.	10	2		4		4	10	1		1		8

Тема 8. Інтерфейс введення-виведення загального призначення (GPIO).	10	2		2		6	10					10
Разом за змістовим модулем 2	40	8		10		22	40	1		2		37
Змістовий модуль 3. Апаратне, програмне та інструментальне забезпечення Raspberry Pi і Arduino.												
Тема 9. Мікроконтролер Arduino.	10	2		6		2	10	1		1		8
Тема 10. Середовище розробки Arduino IDE. Програмний продукт для створення монтажних схем – Fritzing.	10	2		2		6	10			1		9
Тема 11. Операційна система Raspberry Pi OS.	10	2		2		6	10			2		8
Тема 12.Огляд основних конструкцій мови програмування Python.	10	2				8	10					10
Разом за змістовим модулем 3	40	8		10		22	40	1		4		35
Усього годин	**120**	**24**		**24**		**72**	**120**	**6**		**6**		**108**
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**

РОЗПОДІЛ БАЛІВ

1. Поточний контроль знань здійснюється в ході захисту лабораторної роботи.
2. Контроль за виконанням лабораторних робіт забезпечується перевіркою добросовісно оформлених і своєчасно зданих звітів.
3. Підсумковий контроль проводиться у формі складання екзамену.

Поточне тестування та самостійна робота														Сума
Змістовий модуль 1				Змістовий модуль 2				Змістовий модуль 3				Самостійна робота	Екзамен	
Т1	Т2	Т3	Т4	Т5	Т6	Т7	Т8	Т9	Т10	Т11	Т12			
2	4	3	3	3	3	5	3	7	3	3	2	7	40	100
Модульний контроль – 4				Модульний контроль – 4				Модульний контроль – 4						

ЛАБОРАТОРНА РОБОТА №1

***Тема**: Середовище розробки Arduino IDE. Макетна плата. Світлодіод. Кнопка. Потенціометр. Реалізація проектів на базі мікроконтролера Arduino. Fritzing.*

***Мета:** Формування знань та вмінь, щодо: роботи в середовищі розробки Arduino IDE та Fritzing і використання макетної плати, світлодіода, кнопки, потенціометра в проектах Інтернету речей за використання Arduino Uno.*

Arduino – найбільш популярна платформа для розробки як простих, так і досить складних проектів Інтернету речей. Arduino Uno R3 – найкраща плата для початківців. Згідно даних з офіційного сайта, ця плата є найбільш використовуваною і володіє значною кількістю документації з усіх плат сімейства Arduino. Arduino Uno R3 – це плата на основі 8-бітного мікроконтролера ATmega328P (рис. 1.1). По суті, плата Arduino Uno є платою розширення або платою розробника (developer board), серцем якої є ATmega328P.

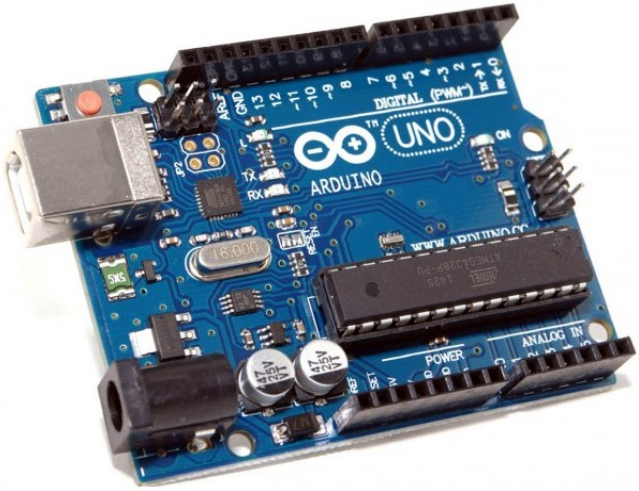


Рис. 1.1. Мікроконтролер Arduino Uno

У Arduino Uno R3 є 14 цифрових входів/виходів, позначених на платі цифрами (з них 6 це піни з підтримкою Pulse-Width Modulation – широтно-імпульсною модуляцією, позначені символом ~), 6 аналогових виходів – A0-A6, кварцовий кристал-резонатором на 16 МГц, наявні виходи для підключення живляння від адаптера на 7-12 В або батарейки-крони на 9 В і кнопка перезавантаження. Окрім цього, плата має виходи заземлення (GND), живлення 5 В і 3.3 В та вбудованим світлодіодом L (D-13).

Для програмування мікроконтролера використовується оболонка Arduino IDE з мовою схожою на C/C++ (рис. 1.2). Вирушаємо на сторінку https://www.arduino.cc/en/main/software, вибираємо версію для операційної системи Windows і скачуємо архівний файл. Він займає трохи більше 80 Мбайт та містить все необхідне, в тому числі і драйвери. Після закінчення завантаження розпаковуємо скачаний файл у зручне для себе місце. Тепер лишається тільки встановити драйвери і підключити Arduino до комп'ютера. На контролері засвітиться живлення у вигляді мигаючих зелених чи червоних світлодіодів.

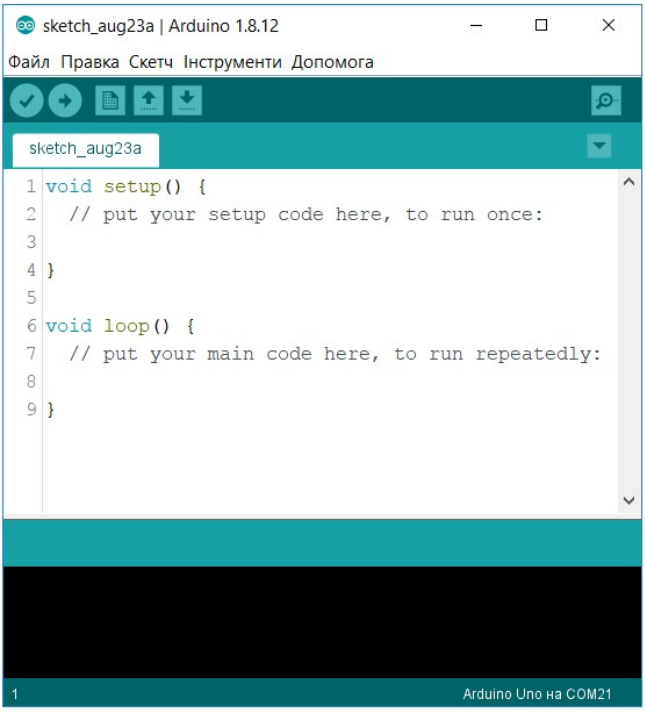


Рис. 1.2. Середовище розробки Arduino IDE

В Arduino IDE за замовчуванням вже містяться приклади різних програм (Sketch) для перевірки працездатності мікроконтролера або швидкого створення своєї програми на базі існуючої, файли з програмою зберігаються з розширенням «.ino». Найпростіша програма складається з двох функцій: setup () – функція викликається одноразово при старті мікроконтролера; loop () – функція викликається після setup () в нескінченному циклі протягом всього часу роботи мікроконтролера. Інтерфейс має просту структуру. У верхній частині розташована панель навігації, нижче кнопки для збереження і завантаження програми в Arduino, слідом текстовий редактор для написання програм, зелена область під текстовим редактором відображає інформацію про завантаження програми на

Arduino, остання частина – це консоль для виведення службової інформації. У нижньому правому куті показується інформація про порт до якого підключена Arduino.

Макетна плата – це універсальна друкована плата для складання та моделювання прототипів електронних пристроїв. Макетні плати поділяються на два типи: для монтажу за допомогою пайки і без неї (рис. 1.3). На багатьох платах для зручності роботи нанесена координатна сітка. Макетні плати можуть бути нарощуваними: на їх бічних гранях розташовані пази для з'єднання декількох плат в одну більшу.

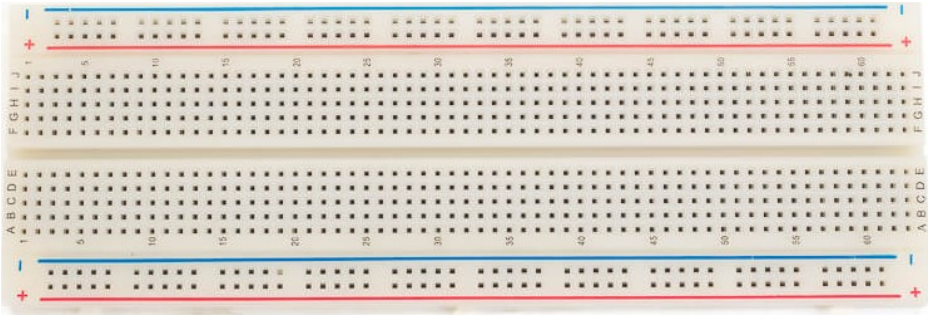


Рис. 1.3. Макетна плата безпаячного монтажу

Є кілька різних типів макетних плат: універсальні – мають виключно металізовані отвори, які потрібно з'єднувати перемичками; для цифрових пристроїв – намічені можливі місця для мікросхем, по всій платі проведено лінії живлення; спеціалізовані – для пристроїв на мікросхемі конкретної моделі. На таких платах є як заздалегідь розведені стандартні лінії, так і матриця отворів та доріжок для нестандартних.

Макетні плати для монтажу в гнізда – у таких макетних платах є тисячі отворів, електрично пов'язані між собою, наприклад, за допомогою металевих смужок. Висновки радіодеталей і мікросхем вставляються в ці отвори, а потім з'єднуються перемичками – шматочками зачищених проводів. Довгі ряди контактів вгорі, посередині і внизу плати – це лінії живлення. Вони служать для з'єднання численних точок схеми з джерелом живлення і «землею». Під кожним отвором розташовані пружні контакти спеціальної форми, зазвичай з нікелевих сплавів для забезпечення високої провідності і довговічності з'єднань. Кожен контакт макетної плати може витримувати більше 10 тисяч використань. Відстань між отворами складає 2,54 мм, що є стандартною відстанню між висновками більшості транзисторів і мікросхем в DIP-корпусах (резистори, конденсатори та інші радіодеталі зазвичай мають гнучкі довгі висновки, які можна встановити з іншим кроком).

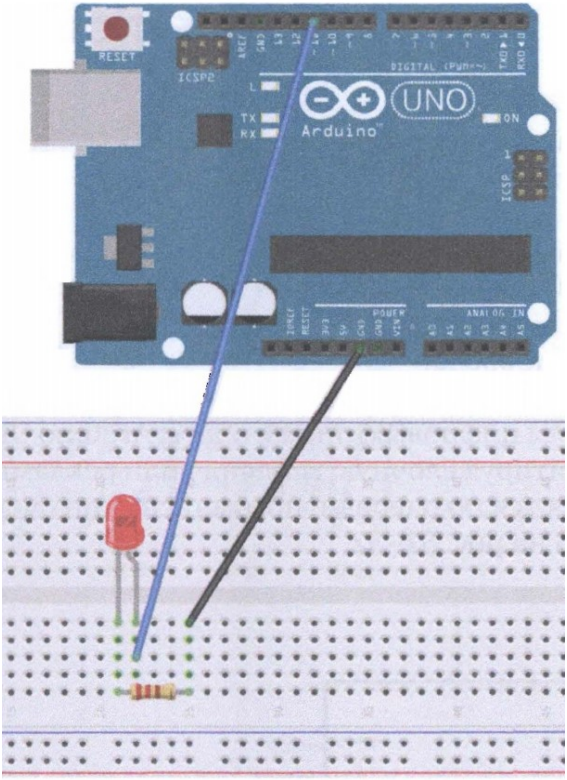
ЗАВДАННННЯ 1 – ПРИКЛАД

СВІТЛОДІОД – це напівпровідниковий прилад, який перетворює електричний струм безпосередньо в світлове випромінювання (рис. 1.4). По-англійськи світлодіод називається light emitting diode (LED). Світлодіод – низьковольтний прилад. Звичайний світло діод споживає від 2 до 4 В постійної напруги при силі струмі до 50 мА. Світлодіод, який використовується для освітлення, споживає таку ж напругу, але струм більший – від декількох сотень мА до 1 А. При підключенні світлодіода необхідно дотримуватись полярності, інакше він може вийти з ладу. Яскравість світлодіода характеризується світловим потоком.

Рис. 1.4. Набір світлодіодів

У світлодіоді, на відміну від лампи розжарювання або люмінесцентної лампи, електричний струм перетвориться безпосередньо в світлове випромінювання і теоретично це можна зробити майже без втрат. Світлодіод мало нагрівається, він механічно міцний і винятково надійний, його термін служби може досягати 100 тисяч годин, що майже в 100 разів більше, ніж у лампочки розжарювання і в 5-10 разів більше ніж у люмінесцентної лампи. Світлодіоди знаходять застосування практично у всіх областях світлотехніки за винятком освітлення виробничих площ та й там можуть використовуватися в аварійному освітленні. Світлодіоди виявляються незамінні в дизайнерському освітленні завдяки їх чистому кольору, а також в світлодинамічних системах. Вигідно їх застосовувати там, де дорого обходиться часте обслуговування, де необхідно жорстко економити електроенергію і де високі вимоги до електробезпеки.

1.1) Схема з'єднання Arduino Uno і макетної плати (в проекті використовується – світлодіод, резистор на 220 Ом і dupont-кабелі).



1.2) Реалізація проекту «Мигання світлодіода» у Arduino IDE.

```
int ledPin = 11;
void setup ()
{
  pinMode (ledPin, OUTPUT);
}
void loop ()
{
  digitalWrite (ledPin, HIGH);
  delay (1000);
  digitalWrite (ledPin, LOW);
  delay (1000);
}
```

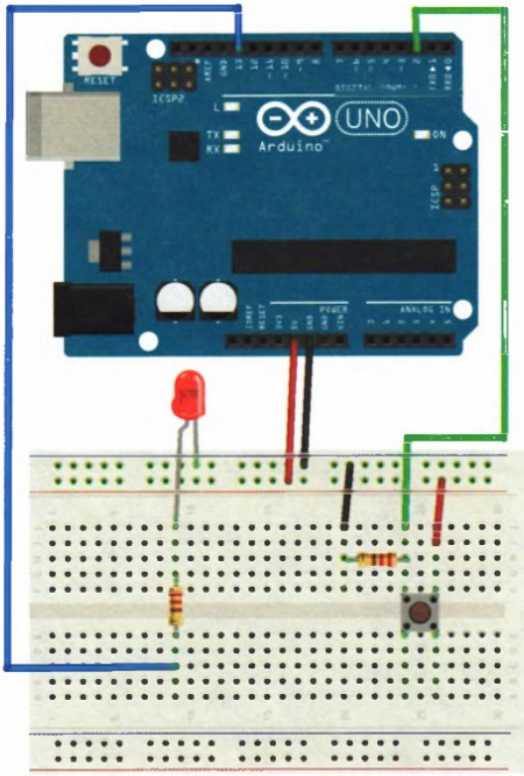
ЗАВДАННННЯ 2 – ПРИКЛАД

КНОПКА – механічний пристрій, який слугує для передачі сигналу (введення інформації). Елементарний фізичний механізм передачі електричного сигналу різних пристроїв шляхом замикання або розмикання двох, або більше контактів (рис. 1.5).

Рис. 1.5. Кнопка

По суті кнопка є датчиком зовнішнього фізичного впливу (натискання). Також це конструктивний елемент, що містить деяку обмежену поверхню, натиснення на яку приводить до вироблення керувального впливу на пов'язаний з нею пристрій. За своєю природою кнопка є первинним перетворювачем зовнішнього фізичного впливу, що передає сигнал у вигляді: переміщення, зусилля, зміни ємності, індуктивності, світлового потоку сполученим з нею пристроям. Два контакти, які комутують сигнальні лінії в процесі натискання кнопки називаються контактною групою. Кнопка може містити як одну, так і декілька контактних груп.

2.1) Схема з'єднання Arduino Uno і макетної плати (в проекті використовується – світлодіод, кнопка, два резистора на 220 Ом та 10 кОм і dupont-кабелі).



2.2) Реалізація проекту «Кнопка і світлодіод» у Arduino IDE.

```
int Button=2;
int LED=13;
int S=0;
void setup()
{
 pinMode(LED, OUTPUT);
 pinMode(Button, INPUT);
}
void loop()
{
  S=digitalRead(Button);
  if (S == HIGH)
    digitalWrite(LED, HIGH);
  else
    digitalWrite(LED, LOW);
}
```

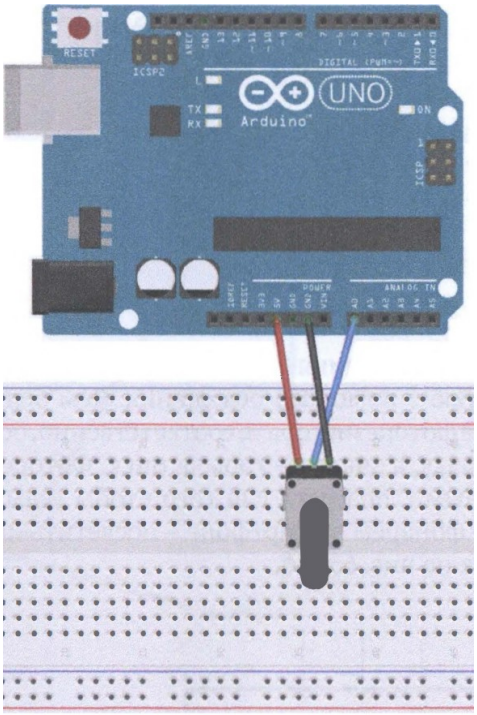
ЗАВДАННННЯ 3 – ПРИКЛАД

ПОТЕНЦІОМЕТР – це назва змінного резистора, включеного як дільник електричної напруги (рис. 1.6). Під потенціометрами, як правило, мають на увазі резистори з рухомим відвідним контактом (движком). Потенціометри використовуються в якості регуляторів параметрів (гучності звуку, потужності, вихідної напруги і.т.п.) та для підстроювання внутрішніх характеристик електричних ланцюгів апаратури. На основі прецизійних потенціометрів побудовано багато типів датчиків кутового або лінійного переміщення. Більшість різновидів змінних резисторів можуть використовуватися як в якості потенціометрів, так і в якості реостатів, різниця в схемах підключення і в призначенні (потенціометр – регулятор напруги, реостат – сили струму).

Рис. 1.6. Потенціометр

Ковзаючий елемент потенціометра може повертатися на один оборот, а точніше, близько 270 градусів. На повний оборот поворот неможливий, так як на решті частини сектора повороту розміщені клеми контактів. Найбільш популярними однооборотні змінні резистори стали в пристроях, які не потребують для регулювання більш одного обороту. З розвитком електронної промисловості крім «класичних» потенціометрів з'явилися також цифрові потенціометри. Такі потенціометри, як правило, представляють собою інтегральні схеми, не мають рухомих частин і дозволяють програмно регулювати власний опір з заданим кроком.

3.1) Схема з'єднання Arduino Uno і макетної плати (в проекті використовується – потенціометр і dupont-кабелі).



3.2) Реалізація проекту «Зчитування аналогового сигналу» у Arduino IDE.

```
int potpin=0;
int ledpin=13;
int val=0;

void setup()
{
 pinMode(ledpin, OUTPUT);
 Serial.begin(9600);
}

void loop()
{
  digitalWrite(ledpin,HIGH);
  delay(50);
  digitalWrite(ledpin, LOW);
  delay (50);
  val=analogRead (potpin);
  Serial.println (val);
}
```

ЗАВДАННННЯ 4 – САМОСТІЙНО

4.1) Використовуючи програму Fritzing: створити власну схему з'єднання (згідно варіанту завдання) – Arduino і макетної плати.

4.2) Написати програму для реалізації поставленої задачі. Варіант обираємо згідно номеру у списку академічної групи.

1) Дано кнопку і світлодіод (жовтий). Реалізувати проект «Натисни і мерехти 1»: натискається кнопка → світлодіод засвідчується 15 разів на 1.5 секунди і гасне на 1 секунду.

2) Дано 3 світлодіода (зелений, жовтий, червоний) і кнопку. Реалізувати проект «Зворотний Світлофор». Затримки засвічення 4-2.5-4 (в секундах) після натиснення кнопки.

3) Дано світлодіод (червоний) і потенціометр. Реалізувати проект «Потенціометр і світлодіод 1»: прокручується потенціометр → змінюється яскравість світлодіода.

4) Дано 3 світлодіода (червоний, жовтий, зелений) і кнопку. Реалізувати проект «Світлофор». Затримки засвічення 5-3-5 (в секундах) після натиснення кнопки.

5) Дано світлодіод (зелений) і потенціометр. Реалізувати проект «Потенціометр і світлодіод 2»: як тільки аналогове значення потенціометра перевищує 512 → вмикається світлодіод (інакше світлодіод – не горить).

6) Дано кнопку і світлодіод (червоний). Реалізувати проект «Натисни і мерехти 2»: натискається кнопка → світлодіод засвідчується на 5 секунди, потім гасне на 3 секунду, а далі безперервно мерехтить з інтервалом у 0.25 секунди.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке потенціометр?
2. Для чого потрібний резистор?
3. Що таке Fritzing?
4. В яких проектах доцільно використовувати світлодіод?
5. Для чого потрібна макетна плата?
6. Які бувають види dupont-кабелю?
7. Що таке резистор?
8. Для чого призначена кнопка?
9. Що таке світлодіод?
10. Мікроконтролер – це…

ЛАБОРАТОРНА РОБОТА №2

Тема: *Фоторезистор. Пасивний зумер. Реалізація проектів на базі мікроконтролера Arduino.*

Мета: *Формування знань та вмінь, щодо: роботи в середовищі розробки Arduino IDE та Fritzing й використання фоторезистора і пасивного зумера в проектах Інтернету речей за використання Arduino Uno.*

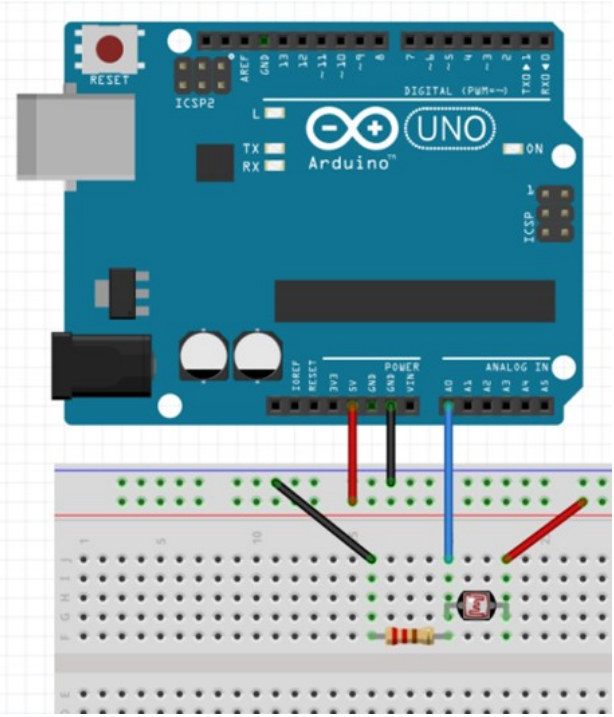
ЗАВДАННННЯ 1 – ПРИКЛАД

ФОТОРЕЗИСТОР – це резистор, електричний опір якого змінюється під впливом світлових променів, які падають на його світлочутливу поверхню і не залежить від прикладеної напруги, як у звичайного резистора (рис. 2.1). Фоторезистор – напівпровідниковий прилад, опір якого змінюються в залежності від того, наскільки сильно освітлена його чутлива поверхня. Конструктивно зустрічаються в різних виконаннях, при цьому для роботи в специфічних умовах можна знайти фоторезистори укладені в металевий корпус з віконцем, через яке потрапляє світло на чутливу поверхню.

Рис. 2.1. Фоторезистор

У промисловості і побутовій електроніці фоторезистори використовуються для вимірювання освітленості, визначення перешкод, ввімкнення вуличного освітлення та іншого. Основне його призначення – переводити кількість світла, що потрапляє на чутливу площу в корисний електричний сигнал. Сигнал надалі може оброблятися як аналоговий чи цифровий – логічною схемою або схемою на базі мікроконтролера.

1.1) Схема з'єднання Arduino Uno і макетної плати (в проекті використовується – фоторезистор, резистор на 220 Ом і dupont-кабелі).



1.2) Реалізація проекту «Фоторезистор» у Arduino IDE.

```
int Light;
void setup()
{
 Serial.begin(9600);
}
void loop()
{
 Light=analogRead(A0);
 Serial.println(Light);
 delay(100);
}
```

ЗАВДАННЯ 2 – ПРИКЛАД

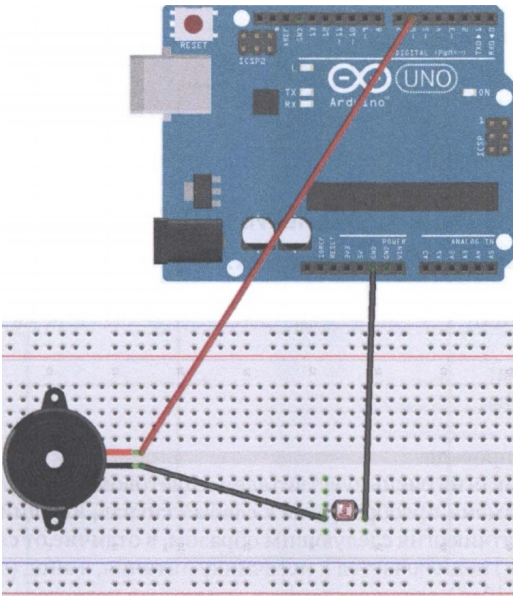
ПАСИВНИЙ ЗУМЕР – це простий модуль для генерації звуків різних частот, елемент невеликої потужності, перетворювач постійного струму у змінний, який завдяки вібрації контакту переривача видає своєрідне дзижчання (рис. 2.2).

Перетворює електричні коливання в звук. Принцип дії – електромагнітний. Широко використовується в телефонах, дозиметрах, радіотехніці, друкарських машинках, пристроях сигналізації, побутовій техніці, іграшках.

Рис. 2.2. Пасивний зумер

«П'єзопіщалка» конструктивно представлена металевою пластиною з нанесеним на неї напиленням з струмопровідної кераміки. Пластина і напилення виступають в ролі контактів. Пристрій полярний, має «+» і «-». Принцип дії зумера заснований на відкритому братами Кюрі в кінці дев'ятнадцятого століття п'єзоелектричного ефекту. Згідно з ним, при подачі електрики на зумер він починає деформуватися. При цьому відбуваються удари об металеву пластинку, яка і виробляє «шум» потрібної частоти.

2.1) Схема з'єднання Arduino Uno і макетної плати (в проекті використовується – фоторезистор, пасивний зумер і dupont-кабелі).



2.2) Реалізація проекту «Фоторезистор і пасивний зумер» у Arduino IDE.

```
int Buzzer=6;
char i;

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

void loop()
{
 while (true)
 {
   for(i=0; i<80; i++)
   {
    digitalWrite(Buzzer, HIGH);
    delay(1);
    digitalWrite(Buzzer, LOW);
    delay(1);
   }
   for(i=0; i <100; i++)
   {
    digitalWrite(Buzzer, HIGH);
    delay(2);
    digitalWrite(Buzzer, LOW);
    delay(2);
   }
 }
}
```

ЗАВДАННЯ 3 – САМОСТІЙНО

3.1) Використовуючи програму Fritzing: створити власну схему з'єднання (згідно варіанту завдання) – Arduino і макетної плати.

3.2) Написати програму для реалізації поставленої задачі. Варіант обираємо згідно номеру у списку академічної групи.

1) Дано фоторезистор і світлодіод (жовтий). Реалізувати проект аналогічний прикладу 2.

2) Дано пасивний зумер і кнопку. Реалізувати проект «Квартирний дзвінок 1»: натискається кнопка → зумер відтворює звук 90 разів із затримка ввімкнення/вимкнення 2 мс.

3) Дано фоторезистор і кнопку. Реалізувати проект «Збираєм дані»: натискається кнопка → і в «монітор порту» виводиться значення фоторезистора.

4) Дано фоторезистор і світлодіод (зелений). Вмикати світіння світлодіода, якщо аналогове значення фоторезистора більше 300.

5) Дано пасивний зумер і кнопку. Реалізувати проект «Зламаний дзвінок»: натискається кнопка → зумер відтворює постійний звук, інакше звук відсутній.

6) Дано пасивний зумер і 2 світлодіода (жовтий і червоний). Реалізувати проект «Звучим і світим»: спочатку зумер відтворює звук 5 секунд, а потім світлодіоди почергово світять ще по 3 секунди.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке фоторезистор?
2. Для чого потрібний пасивний зумер?
3. За яким принципом працює фоторезистор?
4. В яких проектах доцільно використовувати фоторезистор?
5. Для чого потрібні фоторезистори?
6. У яких проектах Інтернету речей доцільно використовувати датчики?
7. Де застосовуються фоторезистори?
8. В яких проектах доцільно використовувати пасивний зумер?
9. Що таке пасивний зумер?
10. Які є види мікроконтролерів Arduino?

ЛАБОРАТОРНА РОБОТА №3

Тема: *Датчик виявлення вогню (фотодіод). Активний зумер. Реалізація проектів на базі мікроконтролера Arduino.*

Мета: *Формування знань та вмінь, щодо: роботи в середовищі розробки Arduino IDE та Fritzing й використання датчика виявлення вогню і активного зумера в проектах Інтернету речей за використання Arduino Uno.*

ЗАВДАННННЯ 1 – ПРИКЛАД

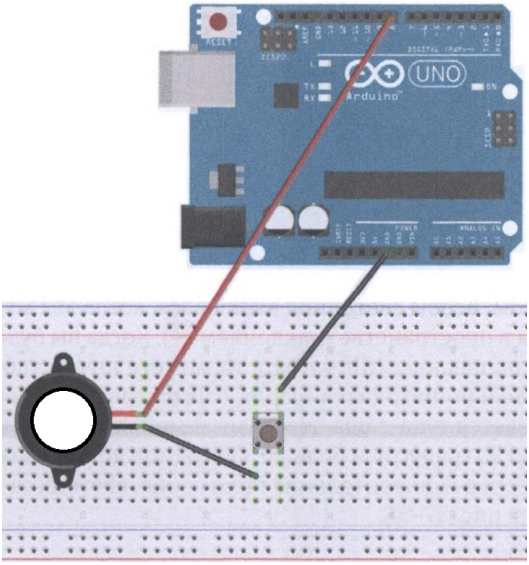
АКТИВНИЙ ЗУМЕР – це сигнальний пристрій, виконаний як електромеханічний, електронний або п'єзоелектричний (рис. 3.1). Активний зумер, іменований також як «пищалка» або active buzzer – найпростіший модуль для отримання звуку частотою близько 2 кГц, що часто може знадобитися при роботі в проектах Інтернету речей.

Рис. 3.1. Активний зумер

Зумер може використовуватися в комп'ютерах, побутовій або автомобільній електроніці, сигнальних пристроях, медицині. Цей пристрій є активним. Це означає, що для його роботи не потрібен зовнішній частотний генератор. Для роботи досить подати напругу. Робоча напруга зумера 5 В. Перед початком роботи з верхньої частини зумера потрібно зняти захисну плівку. Для правильного підключення зумера потрібно дотримуватись полярності.

Принцип роботи. При появі струму в ланцюзі, котушка реле «збуджує» магнітне поле, під дією магнітного поля контакти реле розмикаються. При розмиканні контактів реле, ланцюг розривається, струм перестає текти в ланцюзі, магнітне поле зникає і під дією пружини контакти реле повертаються в початкове положення. Контакти реле в початковому положенні замикають ланцюг, по ланцюзі знову починає текти струм, чутно пищання. Процес розмикання-замикання повторюється до тих пір, поки в ланцюг подається струм. Недоліком таких звуковипромінювачів є низька надійність, викликана зносом механічної частини, ослабленням пружин.

1.1) Схема з'єднання Arduino Uno і макетної плати (в проекті використовується – кнопка, активний зумер і dupont-кабелі).



1.2) Реалізація проекту «Активний зумер і кнопка» у Arduino IDE.

```
int Buzzer=8;
void setup()
{
    pinMode(Buzzer, OUTPUT);
}
void loop()
{
 unsigned char i;
 while (true)
 {
   for(i=0; i <100; i++)
   {
    digitalWrite(Buzzer, HIGH);
    delay(2);
    digitalWrite(Buzzer, LOW);
    delay(2);
   }
 }
}
```

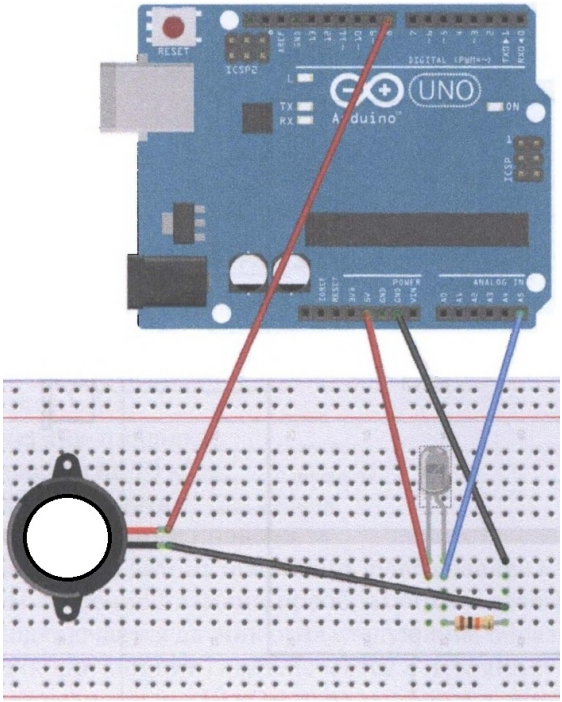
ЗАВДАННННЯ 2 – ПРИКЛАД

ДАТЧИК ВИЯВЛЕННЯ ВОГНЮ (ФОТОДІОД). Особливе місце в електротехніці займають фотодіоди, які застосовуються в різних пристроях і приладах. Фотодіоди це напівпровідникові елементи, які володіють світлочутливістю (рис. 3.2). Їх основна функція – трансформація світлового потоку в електросигнал. Такі напівпровідники застосовуються в складі різних приладів, функціонування яких базується на використанні світлових потоків. Найчастіше в якості фотодіода застосовують напівпровідникові елементи з p-n переходом. Фотодіод, в залежності від його матеріалу, призначений для реєстрації світлового потоку в інфрачервоному, оптичному і ультрафіолетовому діапазоні довжин хвиль.

Рис. 3.2. Датчик виявлення вогню (фотодіод)

Фотодіоди широко використовуються в системах управління, метрології, робототехніці та інших областях. Також вони використовуються у складі інших компонентів, наприклад в оптореле. Ті що адаптовані до роботи з мікроконтролером, знаходять застосування в якості різних датчиків, наприклад, датчиків освітленості, відстані, пульсу.

2.1) Схема з'єднання Arduino Uno і макетної плати (в проекті використовується – фотодіод, активний зумер, резистор на 10 кОм і dupont-кабелі).



2.2) Реалізація проекту «Фотодіод» у Arduino IDE.

```
int FotoDiod=A5;
int Buzzer=8;
int S=0;
void setup()
{
Serial.begin(9600);
pinMode(Buzzer, OUTPUT);
pinMode(FotoDiod, INPUT);
}
void loop()
{
S=analogRead(FotoDiod);
Serial.println(S);
 if (S>900)
   digitalWrite(Buzzer, HIGH);
 else
   digitalWrite(Buzzer, LOW);
}
```

ЗАВДАННННЯ 3 – САМОСТІЙНО

3.1) Використовуючи програму Fritzing: створити власну схему з'єднання (згідно варіанту завдання) – Arduino і макетної плати.

3.2) Написати програму для реалізації поставленої задачі. Варіант обираємо згідно номеру у списку академічної групи.

1) Дано фотодіод і світлодіод (червоний). Реалізувати проект «Фотодіод 1»: якщо аналогове значення фотодіода перевищує 500, то увімкнути світлодіод.

2) Дано активний зумер і кнопку. Реалізувати проект «Квартирний дзвінок 1»: натискається кнопка → зумер відтворює звук 20 разів з інтервалом звучання в 2 секунди та затримкою в 1 секунду.

3) Дано фотодіод і активний зумер та світлодіод (жовтий). Реалізувати проект «Фотодіод 2»: якщо аналогове значення фотодіода менше 500, то зумер відтворює постійний безперервний звук і загоряється світлодіод.

4) Дано потенціометр і активний зумер. Реалізувати проект «Все залежить від потенціометра»: перетягування потенціометра підвищує або ж понижує гучність звуку, що генерує активний зумер.

5) Дано фотодіод і кнопку. Реалізувати проект «Фотодіод 4». Після натиснення кнопки на екран монітору виводиться поточне аналогове значення фотодіода.

6) Дано активний зумер і кнопку. Реалізувати проект «Квартирний дзвінок 2»: натискається кнопка → зумер відтворює постійний безперервний звук.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке фотодіод?
2. Яка різниця між датчиком і сенсором?
3. Який принцип роботи фотодіоду?
4. В яких проектах доцільно використовувати фотодіоди?
5. Що таке діод?
6. Чим різняться між собою dupont-кабелі?
7. Що таке активний зумер?
8. В яких проектах доцільно використовувати датчик виявлення вогню?
9. Для чого призначений активний зумер?
10. Що в програмі визначає запис: Serial.begin(9600)?

ЛАБОРАТОРНА РОБОТА №4

Тема: *Аналоговий датчик рівня води T1592. Клавіатурна матриця 4*4. Реалізація проектів на базі мікроконтролера Arduino.*

Мета: *Формування знань та вмінь, щодо: роботи в середовищі розробки Arduino IDE та Fritzing й використання датчика рівня води T1592 і клавіатурної матриці в проектах Інтернету речей за використання Arduino Uno.*

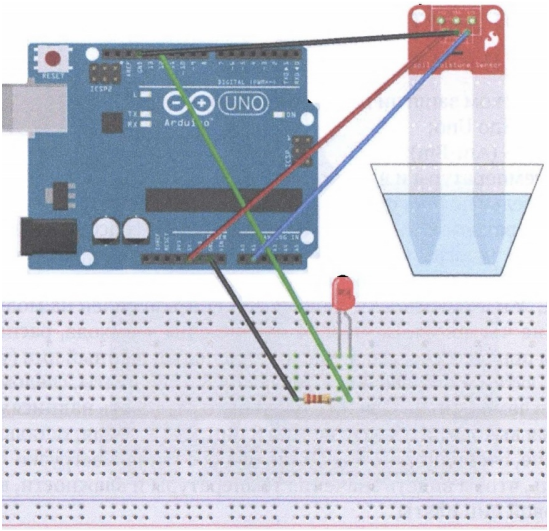
ЗАВДАННННЯ 1 – ПРИКЛАД

АНАЛОГОВИЙ ДАТЧИК РІВНЯ ВОДИ T1592. Для визначення кількості води в ємності використовують спеціальні пристрої – датчики рівня. Компактні вимірювачі широко застосовуються в автоматизації систем водопостачання, а також в промисловому і побутовому обладнанні. З їхньою допомогою можна відстежувати рівень будь-яких рідин, існують варіанти, розроблені для сипучих матеріалів. Конструктивно датчики рівня води розрізняються залежно від області застосування і вимог до точності вимірювання (рис. 4.1).

Рис. 4.1. Датчик рівня води T1592

Даний датчик можна використовувати для вимірювання рівня води, він містить ряд з десяти відкритих мідних доріжок, п’ять з яких є живильними, а п’ять – чутливими. Ці доріжки чергуються так, що між кожними двома доріжками живлення є одна чутлива доріжка. Зазвичай ці доріжки не з’єднані між собою, але при зануренні вони з’єднуються водою. На платі розташований індикатор живлення, який спалахує при подачі на плату напруги. Ряд відкритих паралельних провідників разом діють як змінний резистор (потенціометр), опір якого змінюється в залежності від рівня води. Зміна опору відповідає відстані від верхівки датчика до поверхні води. Опір обернено пропорційний висоті води: чим більше води в яку занурений датчик, тим краще провідність і тим менший опір; чим менше води, в яку занурений датчик, тим гірше провідність і тим більший опір. Датчик відповідно до опору видає вихідну напругу за величиною якої визначається рівень води.

1.1) Схема з'єднання Arduino Uno і макетної плати (в проекті використовується – датчик рівня води T1592, резистор на 220 Ом, світлодіод і dupont-кабелі).



1.2) Реалізація проекту «Датчик рівня води T1592» у Arduino IDE.

```
int Pin=1;
int LED=12;
int S=0;
void setup()
{
 Serial.begin(9600);
 pinMode(LED, OUTPUT);
}
void loop()
{
 S=analogRead(Pin);
 if (S>550)  // Determine the variable S, 700 - MAX
    {
     digitalWrite(LED, HIGH);
    }
 else
    {
     digitalWrite(LED, LOW);
    }
Serial.println(S);
delay(1000); }
```

ЗАВДАННННЯ 2 – ПРИКЛАД

КЛАВІАТУРНА МАТРИЦЯ 4*4. Ми вже створювали на Arduino проекти, де є потреба у користувацькому введенні інформації. Для цього дуже часто використовуються кнопки, причому для підключення кожної кнопки потрібно один порт Arduino. Якщо потрібно використовувати багато кнопок, то це велика розкіш, а іноді це просто фізично неможливо реалізувати. Один з варіантів економії входів при підключенні кнопок – використання матричних клавіатур, наприклад матричної клавіатури 4x4 (рис. 4.2).

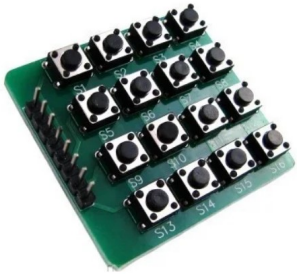
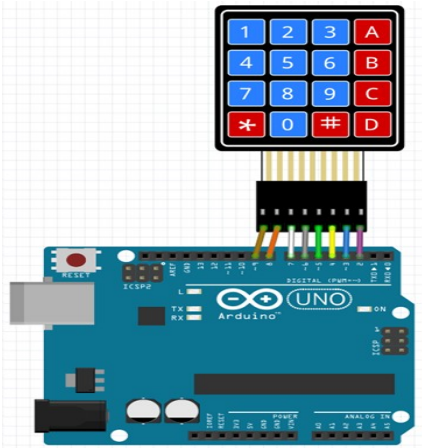


*Рис. 4.2. Клавіатурна матриця 4*4*

На платі матричної клавіатури знаходиться 16 кнопок, які розташовані в матриці 4x4. Принцип роботи клавіатури наступний. Поперемінно подаємо сигнал низького рівня на один з контактів 1-4. Якщо кнопка натиснута, то за наявністю сигналу низького рівня на контактах 5-8 визначаємо натиснуту кнопку з ряду 1-4. Наприклад, подаємо LOW на вхід 2 (на 1,3,4 – сигнал HIGH). Якщо на контакті 6 сигнал LOW, значить натиснута друга кнопка з другого ряду – "5".

2.1) Схема з'єднання Arduino Uno і макетної плати (в проекті використовується – клавіатурна матриця 4*4 і dupont-кабелі).



2.2) Реалізація проекту «Клавіатурна матриця 4*4» у Arduino IDE.

```
#include <Keypad.h>
const byte ROWS=4;
const byte COLS=4;
char keys [ROWS][COLS]={
{'1','2','3','A'},
{'4','5','6','B'},
{'7','8','9','C'},
{'*','0','#','D'}
};

byte rowPins [ROWS]={2,3,4,5};
byte colPins [COLS]={6,7,8,9};

Keypad keypad=Keypad(makeKeymap (keys), rowPins,
colPins, ROWS, COLS);


void setup ()
{
 Serial.begin (9600);
}

void loop()
{
 char key=keypad.getKey();
 if (key!= NO_KEY)
    {
     Serial.println(key);
     }
}
```

ЗАВДАННЯ 3 – САМОСТІЙНО

3.1) Використовуючи програму Fritzing: створити власну схему з'єднання (згідно варіанту завдання) – Arduino і макетної плати.

3.2) Написати програму для реалізації поставленої задачі. Варіант обираємо згідно номеру у списку академічної групи.

1) Дано датчик рівня води і 2 світлодіода (червоний, зелений).

Реалізувати проект «Рівень води 1»: якщо аналогове значення датчика більше 600 – загоряється і постійного горить червоний світлодіод, інакше постійно горить зелений.

2) Дано клавіатурну матрицю 4*4 і активний зумер. Реалізувати проект «Клавіатура гуде»: натискається кнопка «3» → вмикається звуковий сигнал активного зумера, якщо натискається інша кнопка → на консоль видається повідомлення «Hi, my friend».

3) Дано датчик рівня води і пасивний зумер. Реалізувати проект «Рівень води 2»: якщо аналогове значення датчика менше 300 – пасивний зумер починає видавати безперервний звуковий сигнал з частотою 99 МГц.

4) Дано клавіатурну матрицю 4*4 і два світлодіода (зелений, жовтий). Реалізувати проект «Сяюча клавіатура»: натискається кнопка «1» → вмикається зелений світлодіод, натискається кнопка «9» → вмикається жовтий світлодіод.

5) Дано датчик рівня води і активний зумер. Реалізувати проект «Рівень води 3»: якщо аналогове значення датчика більше 400 – активний зумер починає видавати переривчатий звук з інтервалами в одну секунду.

6) Дано клавіатурну матрицю 4*4 і 3 світлодіода (жовтий,червоний, зелений). Реалізувати проект «Світлофор»: натискається кнопка «5» → вмикається почергово червоний, жовтий, зелений світлодіод, затримки засвічення 6-4-6 секунд.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке датчик?
2. Чим аналоговий датчик відрізняється від цифрового?
3. В яких проектах доцільно використовувати клавіатурну матрицю?
4. За яким принципом працює клавіатурна матриця?
5. Чим відрізняються датчики рівня води?
6. Що таке клавіатурна матриця?
7. Де застосовуються датчики рівня води?
8. Яке призначення мідних доріжок на датчику рівня води?
9. Що в програмі визначає запис: if (key!= NO_KEY)?
10. У яких проектах Інтернету речей доцільно використовувати датчик рівня води?

ЛАБОРАТОРНА РОБОТА №5

Тема: *Датчик виявлення звуку LM393. Модуль управління KY-023. Реалізація проектів на базі мікроконтролера Arduino.*

Мета: *Формування знань та вмінь, щодо: роботи в середовищі розробки Arduino IDE та Fritzing й використання датчика виявлення звуку LM393 і модуля управління KY-023 в проектах Інтернету речей за використання Arduino Uno.*

ЗАВДАННННЯ 1 – ПРИКЛАД

ДАТЧИК ВИЯВЛЕННЯ ЗВУКУ LM393. Датчик звуку – дозволяє отримати аналогове чи цифрове значення, що відповідає рівню гучності звуку. Модуль має високу чутливість. Сигнал на виході модуля не повторює форму звукового сигналу, а відповідає рівню його гучності в будь-який проміжок часу. Датчик являє собою невелику плату з встановленим на ньому мікрофоном, мікрофонним підсилювачем, регулятором чутливості у вигляді змінного резистора. Мікрофон перетворює звукові коливання в коливання електричного струму. Сигнал з мікрофона необхідно посилити за допомогою компаратора LM393. Датчик має вихід з логічним рівнем. Спрацював датчик – на виході з'явився логічний 0 (рис. 5.1).

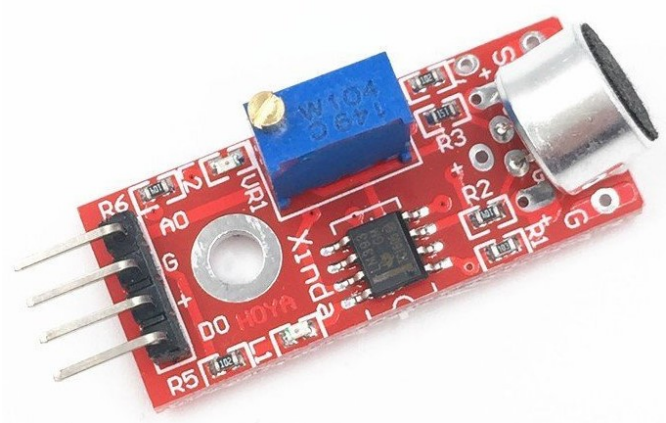
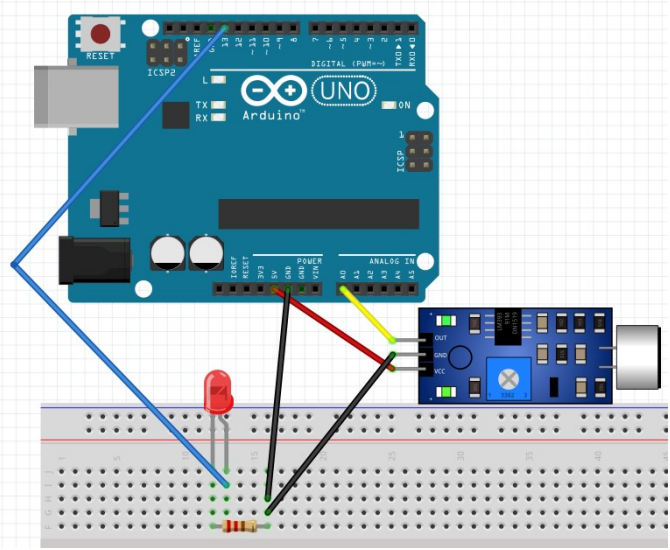


Рис. 5.1. Датчик виявлення звуку LM393

Регулятором чутливості можна змінювати поріг від якого рівня звуку буде спрацьовувати датчик від слабкого, гучного або дуже гучного звуку. Датчик має світлодіод, що сигналізують про наявність низького рівня на виході. Робоча напруга живлення 5 В. Можна застосовувати для управління розумним будинком за допомогою ударів долонями.

1.1) Схема з'єднання Arduino Uno і макетної плати (в проекті використовується – датчик виявлення звуку LM393, світлодіод, резистор на 220 Ом і dupont-кабелі).



1.2) Реалізація проекту «Датчик виявлення звуку LM393» у Arduino IDE.

```
int Pin=A0;
int LED=13;
int S=0;

void setup()
{
 Serial.begin(9600);
 pinMode(LED, OUTPUT);
}

void loop()
{
 S=analogRead(Pin);
 Serial.print("S=");
 Serial.println(S);
 digitalWrite(LED, HIGH);
 delay(S);
 digitalWrite(LED, LOW);
 delay(S);
}
```

ЗАВДАННННЯ 2 – ПРИКЛАД

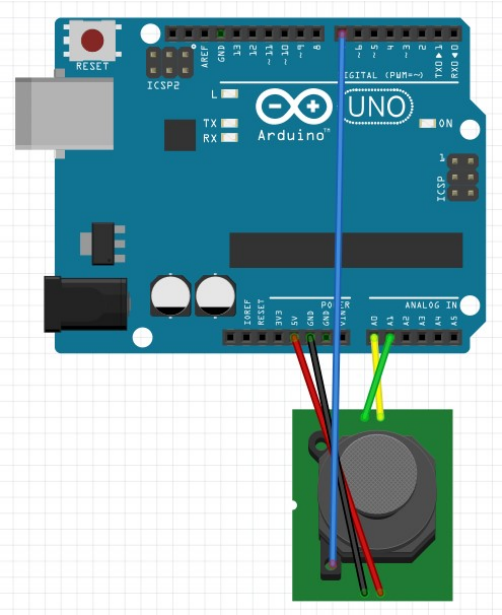
МОДУЛЬ УПРАВЛІННЯ KY-023 (ДЖОЙСТИК). Джойстик є одним з пристроїв для зручної передачі інформації від людини до комп'ютера або мікроконтролера (рис. 5.2). Джойстики використовуються для управління рухом роботів, мобільних платформ і інших механізмів. Аналоговий джойстик являє собою ручку, яка кріпиться на шарнірі з двома потенціометрами, які визначають положення джойстика по осі X і Y та містить тактову кнопку Z.

Рис. 5.2. Модуль управління KY-023

На модулі джойстика є п'ять пінів: Vcc, Ground, X, Y, Key. При нахилі ручки обертаються рухливі контакти кожного з двох потенціометрів номіналом 10 кОм, які визначають положення по осях X і Y. Середній контакт кожного потенціометра виведений на контакти VRX і VRY роз'єму, а крайні підключені до живлення і землі.

Також джойстик оснащений тактовою кнопкою, яка спрацьовує при вертикальному натисканні на ручку, дані знімаються з контакту Key. При відпусканні ручки джойстика, вона плавно повертається в центральне (нульове) положення. Дані з потенціометрів по осях X і Y можуть набувати значень від 0 до 1023.

2.1) Схема з'єднання Arduino Uno і макетної плати (в проекті використовується – джойстик KY-023 і dupont-кабелі).



2.2) Реалізація проекту «Джойстик KY-023» у Arduino IDE.

```
int Pin=7;
int S;

void setup()
{
 pinMode(Pin, INPUT);
 Serial.begin(9600);
}
void loop()
{
 S=analogRead(A0);
 Serial.print("X:");
 Serial.print(S, DEC);
 S=analogRead(A1);
 Serial.print(" | Y:");
 Serial.print(S, DEC);
 S=digitalRead(Pin);
 Serial.print(" | Z:");
 Serial.println(S, DEC);
 delay(100);
}
```

ЗАВДАННННЯ 3 – САМОСТІЙНО

3.1) Використовуючи програму Fritzing: створити власну схему з'єднання (згідно варіанту завдання) – Arduino і макетної плати.

3.2) Написати програму для реалізації поставленої задачі. Варіант обираємо згідно номеру у списку академічної групи.

1) Дано джойстик KY-023 і світлодіод (червоний). Реалізувати проект «X»: вмикати світлодіод, якщо координата по X більша 512.

2) Дано датчик виявлення звуку LM393 і активний зумер. Реалізувати проект «Сигналізація»: вмикається безперервний звуковий сигнал після виявлення звуку, згідно налаштованого порогу.

3) Дано датчик виявлення звуку LM393 і 3 світлодіода (червоний, зелений, жовтий). Реалізувати проект «Тебе виявлено»: всі світлодіоди починають синхронно мигати, якщо виявлено звук, згідно налаштованого порогу.

4) Дано джойстик KY-023 і пасивний зумер. Реалізувати проект «Y»: вмикати мелодійне пищання зумера на частоті 300 МГц, якщо координата по Y менша 512.

5) Дано джойстик KY-023 і активний зумер. Реалізувати проект «Z»: вмикати переривчасте пищання зумера, якщо координата по Z дорівнює 0 (нулю).

6) Дано датчик виявлення звуку LM393 і фотодіод. Реалізувати проект «Чую»: на консоль видаються дані з фоторезистора після виявлення звуку, згідно налаштованого порогу.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке модуль управління?
2. Для чого потрібний датчики виявлення звуку?
3. Яке призначення мікрофону на датчику виявлення звуку?
4. В яких проектах доцільно використовувати модуль управління?
5. Для чого потрібний модуль управління KY-023?
6. За яким принципом працює модуль управління KY-023?
7. В яких проектах доцільно використовувати датчики виявлення звуку?
8. Які бувають різновиди датчиків звуку?
9. В яких проектах Інтернету речей доцільно використовувати аналогові датчики?
10. Що в програмі визначає запис: S=analogRead(A0)?

ЛАБОРАТОРНА РОБОТА №6

Тема: *Інфрачервоний приймач VS1838B і пульт дистанційного керування. Реалізація проектів на базі мікроконтролера Arduino.*

Мета: *Формування знань та вмінь, щодо: роботи в середовищі розробки Arduino IDE та Fritzing і використання інфрачервоного приймача VS1838B і пульта дистанційного керування в проектах Інтернету речей за використання Arduino Uno.*

ЗАВДАННННЯ 1 – ПРИКЛАД

ІНФРАЧЕРВОНИЙ ПРИЙМАЧ VS1838B І ПУЛЬТ ДИСТАНЦІЙНОГО КЕРУВАННЯ. Модуль інфрачервоного (ІЧ) передавача призначений для організації зв'язку, який широко використовується для приведення в дію телевізійного чи іншого пристрою з короткої дистанції за прямої видимості (рис. 6.1). Так як інфрачервоне дистанційне керування використовує світло, воно вимагає прямої видимості для ввімкнення пристрою. Сигнал може бути відбитим так само як і будь-який інший світловий промінь. Інфрачервоні приймачі мають обмежений робочий кут, який головним чином залежить від оптичних характеристик фотодіода, проте він може бути збільшеним за допомогою спеціальних модулів.

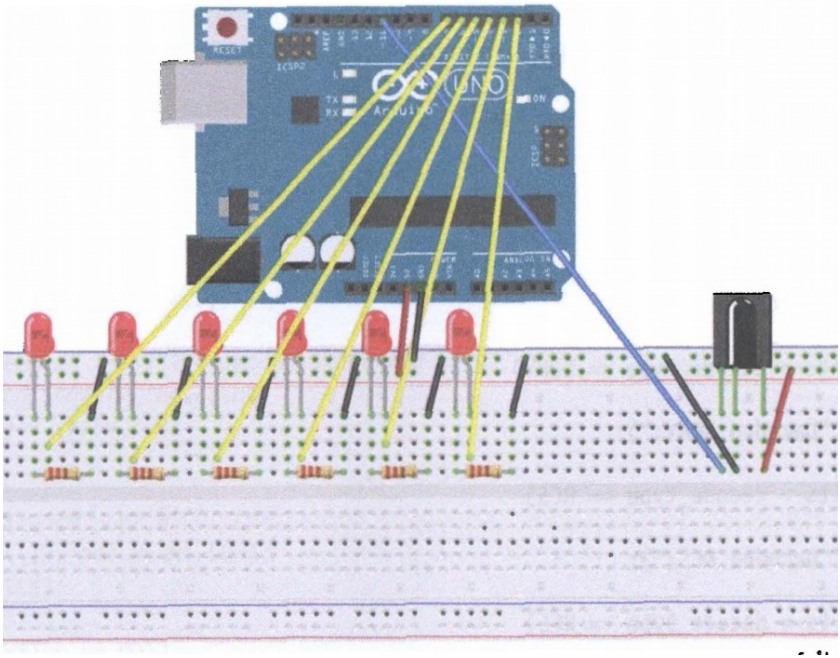
Рис. 6.1. Інфрачервоний приймач VS1838B і пульт дистанційного керування

Інфрачервоний зв'язок – це поширена, недорога і проста у використанні технологія бездротового зв'язку. Світло ІЧ-діапазону дуже схоже на видиме світло, за винятком того, що воно має дещо більшу довжину хвилі. Це означає, що ІЧ-випромінювання є невидимим для людського ока. ІЧ-випромінювання – ідеальне рішення для розгортання бездротового зв'язку.

Тепер трішки теорії. Спочатку коротко розглянемо внутрішню роботу загальних протоколів ІЧ-зв'язку. А потім перейдемо до прикладів, які дозволяють

передавати і приймати ІЧ-дані за допомогою Arduino. Для того щоб дізнатися, який протокол використовується в пульті дистанційного управління, потрібно зібрати схемку з ІЧ-приймачем, зняти сигнал і порівняти його з даними відомих протоколів. Правда є один момент, ІЧ-приймач розрахований на роботу з однією несучою частотою, а в деяких протоколах значення несучих частот відрізняються – одні використовують 38 кГц (наприклад, протокол фірми NEC), інші 36 кГц або 40 кГц. Тому одна і та ж схема не зможе працювати з усіма без винятку ІЧ-пультами дистанційного керування. Одним з найпоширеніших протоколів роботи є протокол NEC, яких розглянемо детальніше. Для передачі даних використовується несуча частота 38 кГц. «Посилка» складається з стартового імпульсу і чотирьох байтів даних – адреса, інвертоване значення адреси, команда, інвертоване значення команди. Адреса та команда передаються двічі для підвищення надійності. Кожен біт починається з пачки імпульсів несучої частоти. Тривалість пачки рівна 562 мкс. Шляхом зміни тривалості інтервалу між пачками імпульсів здійснюється кодування нулів і одиниць. При передачі логічної одиниці інтервал від початку поточної до початку наступної пачки імпульсів становить 2.25 мс, а при передачі логічного нуля – 1.125 мс.

1.1) Схема з'єднання Arduino Uno і макетної плати (в проекті використовується – інфрачервоний приймач VS1838B, пульт дистанційного керування, резистори на 220 Ом, світлодіоди і dupont-кабелі).



1.2) Реалізація проекту «Інфрачервоний приймач VS1838B» у Arduino IDE.

```
#include <IRremote.h>

int RECV_PIN=11;
int LED1=2;
int LED2=3;
int LED3=4;
int LED4=5;
int LED5=6;
int LED6=7;
long on1=0x00FFA25D;
long off1=0x00FFE01F;
long on2=0x00FF629D;
long off2=0x00FFA857;
long on3=0x00FFE21D;
long off3=0x00FF906F;
long on4=0x00FF22DD;
long off4=0x00FF6897;
long on5=0x00FF02FD;
long off5=0x00FF9867;
long on6=0x00FFC23D;
long off6=0x00FFB04F;

IRrecv irrecv(RECV_PIN);

decode_results results;

void dump(decode_results * results)
{
int count=results-> rawlen;

  if(results-> decode_type==UNKNOWN)
  {
   Serial.println("Could not decode message");
  }
  else
  {
  if(results-> decode_type==NEC)
```

```
  {
   Serial.print("Decoded NEC:");
  }
  else if(results-> decode_type==SONY)
  {
   Serial.print("Decoded SONY:");
  }
  else if(results-> decode_type==RC5)
  {
   Serial.print("Decoded RC5:");
  }
  else if(results-> decode_type==RC6)
  {
   Serial.print("Decoded RC6:");
  }
   Serial.print(results-> value, HEX);
   Serial.print("(");
   Serial.print(results-> bits, DEC);
   Serial.println("bits)");
  }
   Serial.print("Raw(");
   Serial.print(count, DEC);
   Serial.print("):");
   for(int i=0; i <count; i++)
   {
  if((i%2)==1)
   {
  Serial.print(results-> rawbuf [i] * USECPERTICK, DEC);
   }
   else
   {
  Serial.print(-(int) results-> rawbuf [i] * USECPERTICK,
DEC);
   }
   Serial.print("");
   }
   Serial.println("");
}
void setup()
```

```
{
 pinMode(RECV_PIN, INPUT);
 pinMode(LED1, OUTPUT);
 pinMode(LED2, OUTPUT);
 pinMode(LED3, OUTPUT);
 pinMode(LED4, OUTPUT);
 pinMode(LED5, OUTPUT);
 pinMode(LED6, OUTPUT);
 pinMode(13, OUTPUT);
 Serial.begin(9600);
 irrecv.enableIRIn(); // Start the receiver
}

int on=0;
unsigned long last=millis();
void loop()
{
  if(irrecv.decode(& results))
  {
  if(millis() - last> 250)
  {
  on =! on;
  digitalWrite(13, on? HIGH: LOW);
  dump(& results);
  }
  if(results.value==on1)
  digitalWrite(LED1, HIGH);
  if(results.value==off1)
  digitalWrite(LED1, LOW);
  if(results.value==on2)
  digitalWrite(LED2, HIGH);
  if(results.value==off2)
  digitalWrite(LED2, LOW);
  if(results.value==on3)
  digitalWrite(LED3, HIGH);
  if(results.value==off3)
  digitalWrite(LED3, LOW);
  if(results.value==on4)
  digitalWrite(LED4, HIGH);
```

```
  if(results.value==off4)
  digitalWrite(LED4, LOW);
  if(results.value==on5)
  digitalWrite(LED5, HIGH);
  if(results.value==off5)
  digitalWrite(LED5, LOW);
  if(results.value==on6)
  digitalWrite(LED6, HIGH);
  if(results.value==off6)
  digitalWrite(LED6, LOW);
  last=millis();
  irrecv.resume();
  }
}
```

ЗАВДАННЯ 2 – САМОСТІЙНО

2.1) Використовуючи програму Fritzing: створити власну схему з'єднання (згідно варіанту завдання) – Arduino і макетної плати.

2.2) Написати програму для реалізації поставленої задачі. Варіант обираємо згідно номеру у списку академічної групи.

1) Дано інфрачервоний приймач VS1838B і 3 світлодіода (червоний, жовтий, зелений). Взявши за основу приклад1 реалізувати проект «Інфрачервоний 1» – вмикаються лише 3 світлодіода на задані 3 кнопки.

2) Дано інфрачервоний приймач VS1838B і 3 активних зумера. Взявши за основу приклад1 реалізувати проект «Інфрачервоний 2» – вмикаються 3 активні зумери на 3 задані кнопки.

3) Дано інфрачервоний приймач VS1838B і 5 світлодіодів (червоний, жовтий, зелений, червоний, жовтий). Взявши за основу приклад1 реалізувати проект «Інфрачервоний 3» – вмикаються лише 5 світлодіодів на задані 5 кнопок.

4) Дано інфрачервоний приймач VS1838B, активних зумер і світлодіод. Взявши за основу приклад1 реалізувати проект «Інфрачервоний 4» – вмикаються активний зумер і світлодіод на задані 2 кнопки.

5) Дано інфрачервоний приймач VS1838B, пасивний зумер і світлодіод. Взявши за основу приклад1 реалізувати проект «Інфрачервоний 5» – вмикаються пасивний зумер і світлодіод на задані 2 кнопки.

6) Дано інфрачервоний приймач VS1838B і 3 пасивних зумера. Взявши за основу приклад1 реалізувати проект «Інфрачервоний 6» – вмикаються 3 пасивні зумери на задані кнопки.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке інфрачервоний приймач?
2. Для чого потрібний пульт дистанційного керування?
3. Дайте визначення поняттю – несуча частота.
4. У яких проектах Інтернету речей доцільно застосовувати інфрачервоний приймач?
5. Для чого потрібний інфрачервоний приймач?
6. Що в програмі визначає запис: #include <IRremote.h>?
7. Для чого потрібний пульт дистанційного керування?
8. Що таке протокол з точки зору інфрачервоного зв'язку?
9. Що в програмі визначає запис: Serial.print("Decoded SONY:")?
10. У яких приладах і техніці застосовується інфрачервоний приймач?

ЛАБОРАТОРНА РОБОТА №7

Тема: *Одноплатний комп'ютер Raspberry Pi 3 Model B. Методи встановлення операційної системи Raspberry Pi OS. Платформа InitialState. GPIO. Реалізація проектів на базі Raspberry Pi. Бібліотека RPi.GPIO і ISStreamer.*

Мета: *Формування знань та вмінь, щодо: роботи з операційною системою Raspberry Pi OS та Fritzing і використання платформи InitialState, контактів GPIO, бібліотеки RPi.GPIO і ISStreamer, а також макетної плати, світлодіода, кнопки, пасивного зумера в проектах Інтернету речей за використання Raspberry Pi.*

RASPBERRY PI – це одноплатний комп'ютер, який несподівано для його розробників набув великої популярності (рис. 7.1). Випускається у декількох версіях, ціна $25 – $80 (залежно від моделі). Незважаючи на досить маленькі розміри – це справжній комп'ютер, на який встановлюється операційна система і який працює майже як звичайний комп'ютер.

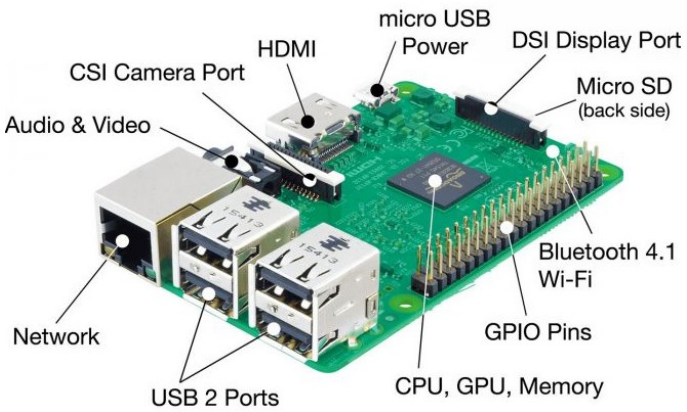


Рис. 7.1. Одноплатний комп'ютер Raspberry Pi 3 Model B

У 2011 році Raspberry Pi був сконструйований і представлений. На початках призначався для навчання комп'ютерним наукам учнів. Він повинен був стати бюджетним інструментом, але виявився популярний і за межами шкільних кабінетів. Популярності в інших сферах посприяли робототехніки та інженери, а також перемога на виставці ARM TechCon в тому ж 2011 році. І сьогодні мікрокомп'ютер Raspberry Pi продовжує активно розвиватися і використовуватися. З цим одноплатним комп'ютером можна вчитися не тільки конструювати девайси, але і програмувати. Плату можна підключити до пристроїв вводу-виводу, щоб почати працювати з власним кодом. По всьому світу програмісти й інженери експериментують з Raspberry Pi, діляться своїми досягненнями та утворюють мультинаціональне технічне співтовариство.

ВСТАНОВЛЕННЯ ОПЕРАЦІЙНОЇ СИСТЕМИ НА RASPBERRY PI.

На сьогодні є багато різних дистрибутивів доступних для Raspberry Pi. Raspberry Pi OS (раніше відомий як Raspbian) – найбільш підтримуваний і стабільний дистрибутив на базі Debian, оптимізований для даного одноплатного комп'ютера. Raspberry Pi OS є офіційно підтримуваною операційною системою Raspberry Pi Foundation.

Є три способи встановлення операційної системи на Raspberry Pi:

1) Через офіційний сайт, детальніше:
https://www.raspberrypi.org/downloads/raspberry-pi-os/

2) Використати інсталятор NOOBS (New Out Of Box Software):
https://www.raspberrypi.org/downloads/noobs/

3) Використати утиліту Raspberry Pi Imager:
https://www.raspberrypi.org/downloads/

INITIAL STATE

Офіційний сайт: – https://www.initialstate.com/

Документація: – https://support.initialstate.com/hc/en-us/articles/360003857212

Компанія базується в м. Нешвілл (США, штат Теннессі), була заснована в 2012 році компанією Tektronix. Initial State – це IoT-платформа потокового передавання та візуалізації даних (рис. 7.2). Полегшує кожному, від початківців до підприємств, надсилати дані з підключених до Інтернету пристроїв, датчиків та програм у хмару, де ці дані можна отримати в будь-який час і миттєво перетворити на щось чудове – інтерактивні панелі інструментів у реальному часі, діаграми, статистику, сповіщення тощо.

Рис. 7.2. Платформа Інтернету речей Initial State

Стек технологій був розроблений для забезпечення послідовного, безпечного та корисного досвіду для користувачів у великих масштабах. API

доступ до якого здійснюється через HTTPS/TLS, забезпечує простий спосіб для користувачів надсилати та отримувати дані з високонадійної бази (сховища) даних в режимі реального часу. Набір інтерактивних візуалізацій даних у реальному часі був створений з нуля, щоб забезпечити унікальний та потужний спосіб візуалізації даних. Візуалізація доступна через інтерфейс веб-браузера.

СХЕМА GPIO RASPBERRY PI. Інтерфейс введення/виведення загального призначення (General-purpose input/output, GPIO) – це інтерфейс для зв'язку між компонентами комп'ютерної системи, наприклад, між мікропроцесором і різними периферійними пристроями. Контакти GPIO можуть діяти як входи, так і як виходи. Це, як правило, підлягає налаштуванню. До GPIO можна підключати різноманітні виконавчі пристрої, датчики, дисплеї, контролери, різні модулі і різну периферію (рис. 7.3).

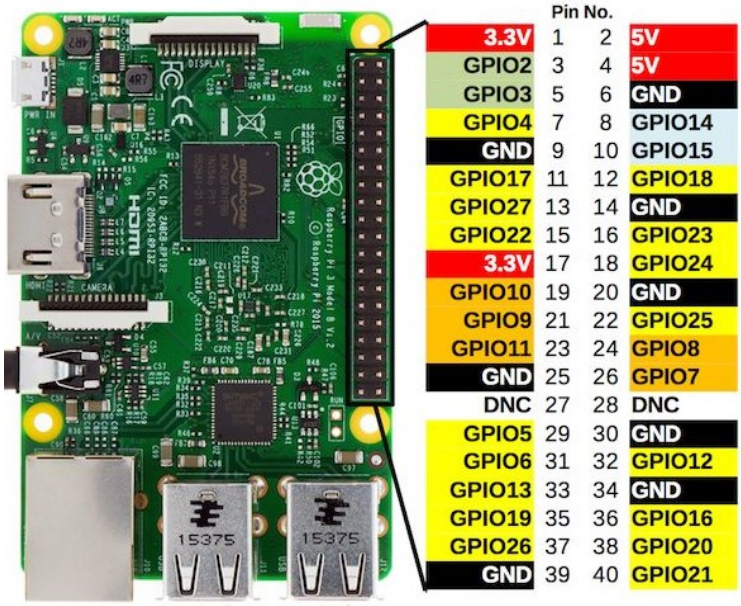


Рис. 7.3. Схема і розпіновка GPIO-контактів Raspberry Pi

GPIO-контакти часто групуються в порти. Ідея GPIO полягає в тому, що іноді системному інтегратору для побудови повної системи, яка використовує чіп, може виявитися корисним мати кілька додаткових ліній цифрового управління. З них можна організувати додаткові схеми, які інакше довелося б створювати з «нуля». Одноплатні комп'ютери (наприклад, Raspberry Pi) використовують GPIO зокрема для читання інформації з різних датчиків навколишнього середовища (сенсорів).

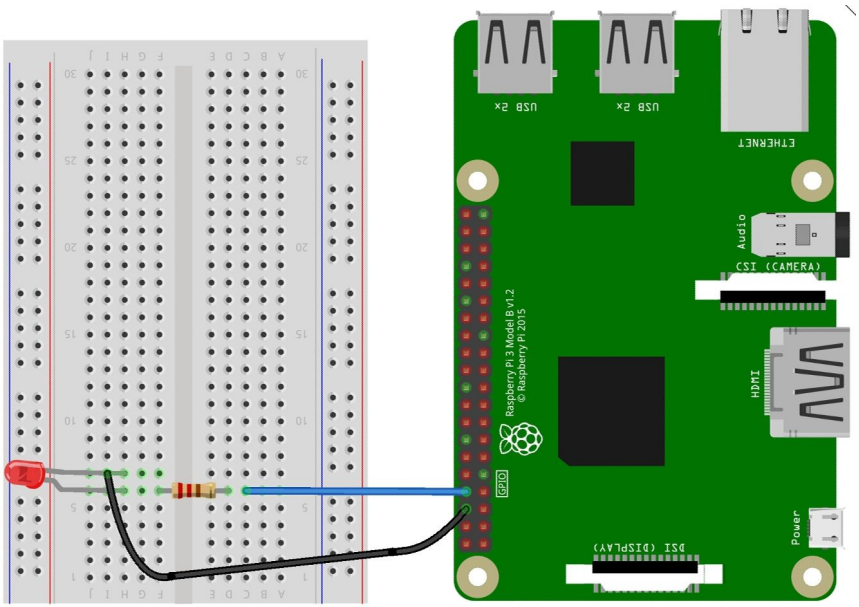
Цікаво те, що у Raspberry Pi за вхід і вихід GPIO можуть відповідати одні і ті ж контакти. Те, як вони себе повинні вести, визначається написаною програмою. Загальна кількість контактів дорівнює 40, вони утворюють два ряди по 20 пінів, нумеруються так: зліва – непарні, справа – парні.

Як було сказано раніше, всі вони поділяються на три групи. До першої відносяться живлення (маркуються як Power). При цьому у різних контактів Power різна напруга. Це обов'язково слід враховувати при підключенні датчиків чи сенсорів. До другої – заземлюючі контакти (маркуються як GND або Ground). Вони потрібні, щоб відводити електричний струм, тим самим забезпечуючи безпечне використання датчиків чи сенсорів. До третьої – контакти Output/Input (маркуються, наприклад як GPIO14). Саме через них можна налаштувати прийом або надсилання сигналу, наприклад до кнопки чи світлодіода.

ЗАВДАННННЯ 1 – ПРИКЛАД

СВІТЛОДІОД – це напівпровідниковий прилад, який перетворює електричний струм безпосередньо в світлове випромінювання.

1.1) Схема з'єднання Raspberry Pi і макетної плати (в проекті використовується – світлодіод, резистор на 220 Ом і dupont-кабелі).



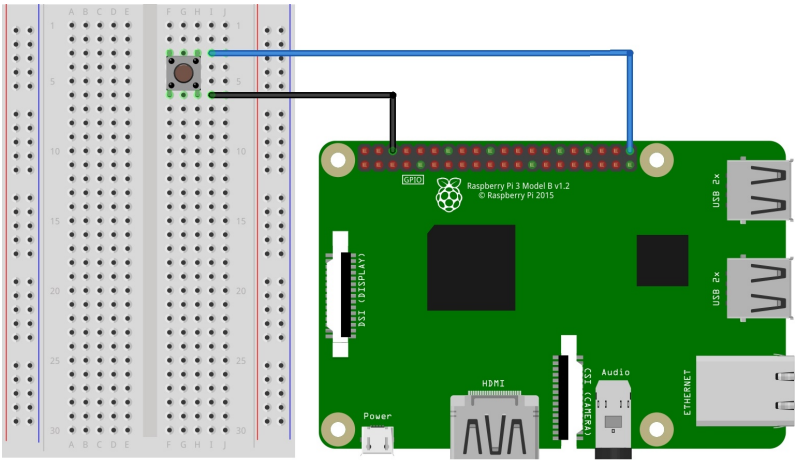
1.2) Реалізація проекту «Мигання світлодіода» у Raspberry Pi (щоб почати працювати з GPIO у Python потрібно встановити бібліотеку RPi.GPIO, деталі встановлення за посиланням https://robotclass.ru/articles/raspberry-pi-gpio-python/, а також потрібно встановити бібліотеку ISStreamer для роботи з Initial State).

```
from RPi import GPIO
from time import sleep
from random import random
from ISStreamer.Streamer import Streamer

ACCESS_KEY  = 'ist_jOqcwksNtmshhFL76SwAOoQB2M6vYGCY'
BUCKET_KEY  = 'PH3JCB9A9X48'
BUCKET_NAME = 'Naz'

# create a Streamer instance
streamer = Streamer(bucket_name=BUCKET_NAME, bucket_key=BUCKET_KEY,
access_key=ACCESS_KEY)

led=14

GPIO.setmode(GPIO.BCM)
GPIO.setup(led, GPIO.OUT)

for i in range(10):
   print('БЛІК -> %s' % (i+1))
   GPIO.output(led, True)
   sleep(random())
   GPIO.output(led, False)
   sleep(random())

# send some data
streamer.log('myNumber', 4)
streamer.log('myMessage', 'Привіт світ IoT')

streamer.flush()
streamer.close()

GPIO.cleanup()
```

ЗАВДАННННЯ 2 – ПРИКЛАД

КНОПКА – механічний пристрій, який слугує для передачі сигналу (введення інформації). Елементарний фізичний механізм передачі електричного сигналу різних пристроїв шляхом замикання або розмикання двох, або більше контактів.

2.1)Схема з'єднання Raspberry Pi і макетної плати (в проекті використовується – кнопка і dupont-кабелі).



2.2) Реалізація проекту «Кнопка» у Raspberry Pi.

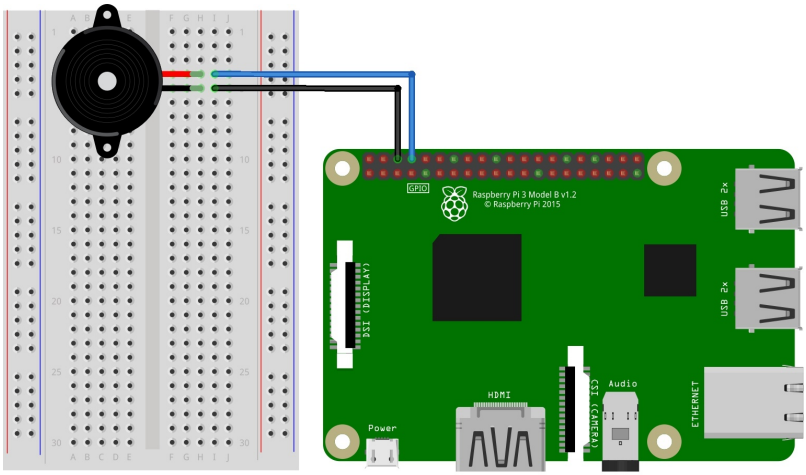
```
from RPi import GPIO
from time import sleep
from ISStreamer.Streamer import Streamer

ACCESS_KEY  = 'ist_jOqcwksNtmshhFL76SwAOoQB2M6vYGCY'
BUCKET_KEY  = 'PH3JCB9A9X48'
BUCKET_NAME = 'Naz'

# create a Streamer instance
streamer = Streamer(bucket_name=BUCKET_NAME, bucket_key=BUCKET_KEY,
access_key=ACCESS_KEY)

button=21
GPIO.setmode(GPIO.BCM)
```

```
GPIO.setup(button,GPIO.IN,GPIO.PUD_UP)

print('НАТИСНІТЬ КНОПКУ... \n')
sleep(1)
if GPIO.input(button)==True:
    print('Щось не так')
if GPIO.input(button)==False:
    print('Успіх! Кнопка натиснута')

# send some data
streamer.log('myNumber', 4)
streamer.log('myMessage', 'Привіт світ IoT')

streamer.flush()
streamer.close()

GPIO.cleanup()
```

ЗАВДАННЯ 3 – ПРИКЛАД

ПАСИВНИЙ ЗУМЕР це простий модуль для отримання звуків різних частот, елемент невеликої потужності, перетворювач постійного струму у змінний, який завдяки вібрації контакту переривача видає своєрідне дзижчання.

3.1) Схема з'єднання Raspberry Pi і макетної плати (у проекті використовується – пасивний зумер і dupont-кабелі).



3.2) Реалізація проекту «Пасивний зумер» у Raspberry Pi.

```
import RPi.GPIO as GPIO
from time import sleep

Buzzer=14

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(Buzzer, GPIO.OUT)

global Buzz    # Глобальна змінна для GPIO.PWM
Buzz = GPIO.PWM(Buzzer, 440)     # Початкова частота 440MHz
Buzz.start(50)      # Start Buzzer pin with 50% duty ration

CL=[0, 131, 147, 165, 175, 196, 211, 248] # Частоти звучання нот в герцах (мала октава)
CM=[0, 262, 294, 330, 350, 393, 441, 495] # Частоти звучання нот в герцах (1 октава)
CH=[0, 525, 589, 661, 700, 786, 882, 990] # Частоти звучання нот в герцах (2 октава)

song=[   CM[1], CM[1], CM[1], CL[5], CM[3], CM[3], CM[3], CM[1], # Звукові ноти (30 шт)
         CM[1], CM[3], CM[5], CM[5], CM[4], CM[3], CM[2], CM[2],
         CM[3], CM[4], CM[4], CM[3], CM[2], CM[3], CM[1], CM[1],
         CM[3], CM[2], CL[5], CL[7], CM[2], CM[1] ]

beat=[   1, 1, 2, 2, 1, 1, 2, 2, # Звукові біти (30 шт)
         1, 1, 2, 2, 1, 1, 3, 1,
         1, 2, 2, 1, 1, 2, 2, 1,
         1, 2, 2, 1, 1, 3 ]

print('ПІСЕНЬКА ... ')
for i in range(len(song)):
   Buzz.ChangeFrequency(song[i])
   sleep(beat[i]*0.5)

print('\n\nКІНЕЦЬ !!! ')
```

```
Buzz.stop() # СТОП для Buzzer
GPIO.output(Buzzer, True) # Buzzer перемикаєм в High
GPIO.cleanup()
```

ЗАВДАННЯ 4 – САМОСТІЙНО

4.1) Використовуючи програму Fritzing: створити власну схему з'єднання (згідно варіанту завдання) – Raspberry Pi і макетної плати.

4.2) Написати програму для реалізації поставленої задачі. Варіант обираємо згідно номеру у списку академічної групи.

Частоти звучання нот в герцах беремо тут → http://caitik.ru/notes.html

Умовні позначення 1:

Д → До,	ДД → До-дієз,	Р → Ре,	РД → Ре-дієз,
М → Мі,	Ф → Фа,	ФД → Фа-дієз,	СО → Соль,
СОД → Соль-дієз,	Л → Ля,	С → Сі,	СБ → Сі-бемоль

Умовні позначення 2:

В → Велика октава, М → Мала октава, 01 → 1 октава, 02 → 2 октава

1) Дано 5 світлодіодів (червоний, жовтий, зелений, червоний, жовтий). Реалізувати проект «Світлодіодні вогні 1»: перший і п'ятий світлодіод горять постійно, а інші три мигають із затримками засвічення 5-3-5 секунд, паузи 0.25-1-0.5 секунд. Відобразити текстову і числову інформацію про виконання програми в IoT-платформі InitialState.

2) Дано 1 пасивний зумер, 7 нот (Д – ДД – Р – РД – М – Ф – СО) і значення їхніх частот в герцах (М – 01 – 02). Реалізувати проект «Звукові хвилі 1». Відобразити текстову і числову інформацію про виконання програми в IoT-платформі InitialState.

3) Дано 2 кнопки і 2 світлодіода (червоний, зелений). Реалізувати проект «Натисни і мерехти»: натискаємо першу кнопку → світлодіод мерехтить 9 разів по 2 секунди (затримка 1 с), натискаємо другу кнопку → світлодіод мерехтить 5 разів по 3 секунди (затримка 0.75 с). Відобразити текстову і числову інформацію про виконання програми в

IoT-платформі InitialState.

4) Дано 4 світлодіода (зелений, жовтий, жовтий, зелений). Реалізувати проект «Світлодіодні вогні 3»: всі світлодіоди мигають із затримками засвічення 4-3-2-1 секунди, паузи 1-0.75-0.5-0.25 секунд. Відобразити текстову і числову інформацію про виконання програми в IoT-платформі InitialState.

5) Дано 1 пасивний зумер, 7 нот (С – СБ – Р – М – Ф – ДД – Д) і значення їхніх частот в герцах (В – 01 – М). Реалізувати проект «Звукові хвилі 2». Відобразити текстову і числову інформацію про виконання програми в IoT-платформі InitialState.

6) Дано 4 світлодіода (червоний, жовтий, жовтий, червоний). Реалізувати проект «Світлодіодні вогні 2»: другий і четвертий світлодіод горять постійно, а інші два мигають із затримками засвічення 15-10 секунд, паузи 1.25-1.5 секунд. Відобразити текстову і числову інформацію про виконання програми в IoT-платформі InitialState.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке одноплатний комп'ютер?
2. Що таке Raspberry Pi?
3. Для чого потрібний GPIO?
4. Чим корисна бібліотека ISStreamer?
5. В яких проектах доцільно використовувати платформу InitialState?
6. Для чого потрібна бібліотека RPi.GPIO?
7. Що таке system on a Chip?
8. Які є методи встановлення операційної системи на Raspberry Pi?
9. Що таке командна оболонка операційної системи?
10. Що таке Linux?

ЛАБОРАТОРНА РОБОТА №8

Тема: *Датчик температури та вологості DHT11. Світлодіодний модуль RGB KY-016. Реалізація проектів на базі одноплатного комп'ютера Raspberry Pi. Бібліотека Adafruit_DHT.*

Мета: *Формування знань та вмінь, щодо: роботи з Raspberry Pi OS та Fritzing й використання платформи InitialState, контактів GPIO, бібліотеки RPi.GPIO і Adafruit_DHT, а також датчика температури та вологості DHT11 й модуля RGB KY-016 в проектах Інтернету речей за використання Raspberry Pi.*

ЗАВДАННЯ 1 – ПРИКЛАД

ДАТЧИК ТЕМПЕРАТУРИ І ВОЛОГОСТІ DHT11. Датчики, виконані на основі електронної техніки називаються електронними датчиками. Окремий датчик може вимірювати одну або одночасно декілька фізичних величин. До складу датчика входять чутливі і перетворюючі елементи. Основними характеристиками електронних датчиків є чутливість і похибка. Датчики широко використовуються в наукових дослідженнях і експерементах, контролі якості, телеметрії, системах автоматизованого управління та в інших областях діяльності, де потрібне отримання вимірювальної інформації.

Датчик DHT11 – це цифровий датчик температури і вологості, який дозволяє калібрувати цифровий сигнал на виході (рис. 8.1). Складається з ємнісного датчика вологості і температури, включає в собі АЦП для перетворення аналогових сигналів значень вологості і температури в цифрові. DHT11 є популярним і його часто використовують в проектах моніторингу навколишнього середовища і кліматичних вимірюваннях.

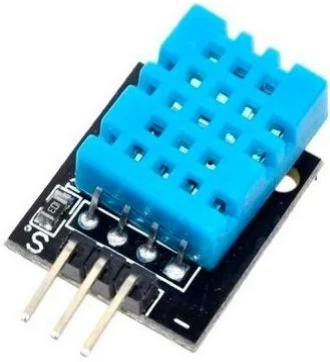
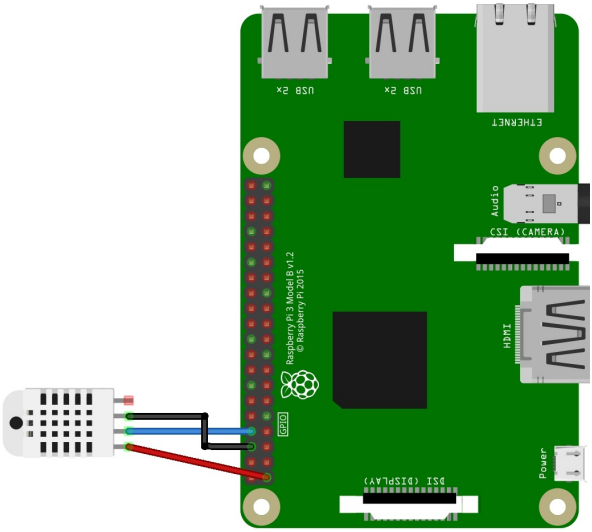


Рис. 8.1. Датчик температури і вологості DHT11

Датчик має достатню точність, простий у використанні, доступний за невелику ціну. Датчик складається з двох частин, перша використовується для

вимірювання температури, друга – для вологості повітря. DHT11 доступний в двох варіаціях як окремий датчик у вигляді пластикового корпусу з металевими контактами або як готовий модуль з датчиком і припаяними елементами обв'язки. Другий варіант набагато простіше використовувати в реальних проектах і рекомендується для початківців.

1.1)Схема з'єднання Raspberry Pi і макетної плати (в проекті використовується – датчик температури та вологості DHT11 і dupont-кабелі).



1.2) Реалізація проекту «Датчик DHT11» у Raspberry Pi (щоб почати працювати з датчиком DHT11 у Python потрібно встановити спеціальну бібліотеку Adafruit_DHT. Деталі встановлення описано в наступному посиланні http://www.circuitbasics.com/how-to-set-up-the-dht11-humidity-sensor-on-the-raspberry-pi).

```
import sys
import time
import Adafruit_DHT
from ISStreamer.Streamer import Streamer

ACCESS_KEY  = 'ist_jOqcwksNtmshhFL76SwAOoQB2M6vYGCY'
BUCKET_KEY  = 'PH3JCB9A9X48'
BUCKET_NAME = 'Naz'

# create a Streamer instance
streamer = Streamer(bucket_name=BUCKET_NAME, bucket_key=BUCKET_KEY,
access_key=ACCESS_KEY)
```

```
DHT_SENSOR=Adafruit_DHT.DHT11
DHT_PIN=14

print('\nДАТЧИК DHT11 -- ТЕМПЕРАТУРА & ВОЛОГІСТЬ')
print('-------------------------------------')

Humidity, Temperature = Adafruit_DHT.read_retry(DHT_SENSOR,DHT_PIN)

if (Temperature is not None) and (Humidity is not None):
  print('\nТемпература => {0:0.1f}*C \n\nВологість =>
{1:0.1f}%\n'.format(Temperature, Humidity))
  streamer.log('myNumber', Temperature)
  sleep(2)
  streamer.log('myNumber', Humidity)
else:
  print('Щось пішло не ТАК!')

streamer.flush()
streamer.close()
```

ЗАВДАННННЯ 2 – ПРИКЛАД

СВІТЛОДІОДНИЙ МОДУЛЬ RGB KY-016. Модуль триколірного RGB світлодіода може використовуватися в проектах на мікроконтролерах для світлової індикації будь-якого процесу, наприклад: подачі живлення, замикання реле, передача і прийом даних та іншого (рис. 8.2). Для використання модуля триколірного RGB світлодіода потрібно підключити до нього живлення і керуючий сигнал від контролера або іншого керуючого мікропроцесорного пристрою.

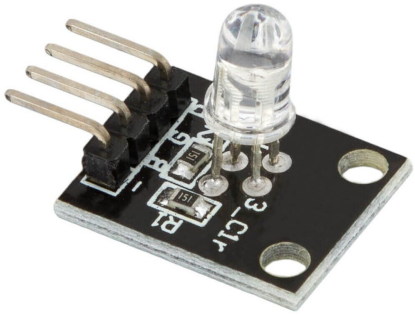
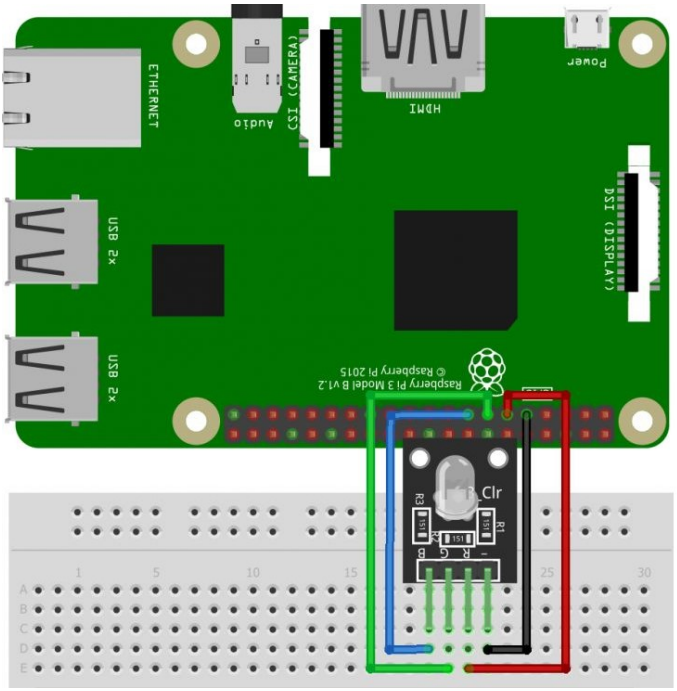


Рис. 8.2. Світлодіодний модуль RGB KY-016

Вбудований в модуль світлодіод може світитися синім, зеленим, червоним кольорами, також кольори світіння можна включати попарно або всі одночасно. Модуль може працювати як під управлінням мікроконтролера так і від іншого керуючого мікропроцесорного пристрою. RGB має загальний контакт від'ємної полярності. За допомогою керуючого сигналу відбувається формування кольору світіння. Модуль триколірного RGB світлодіода має один 4-х контактний роз'єм для підключення заземлення і керуючих сигналів. Живлення здійснюється або від керуючого пристрою або від зовнішніх джерел (наприклад, батарейки). Напруга живлення RGB становить 3.3-5 В. Наявна можливість подавати різне значення ШІМ-сигналу в діапазоні від 0 до 255, що дозволить отримати практично будь-який колір з 16000000 можливих.

2.1)Схема з'єднання Raspberry Pi і макетної плати (в проекті використовується – світлодіодний модуль RGB KY-016 і dupont-кабелі).



2.2) Реалізація проекту «Світлодіодний модуль RGB KY-016» у Raspberry Pi.

```
from RPi import GPIO
from time import sleep
```

```
GPIO.setmode(GPIO.BCM)

RUNNING = True
blue=22
green=27
red=17

GPIO.setup(red, GPIO.OUT)
GPIO.setup(green, GPIO.OUT)
GPIO.setup(blue, GPIO.OUT)

Freq = 100 # Частота ШІМ

RED = GPIO.PWM(red, Freq)
GREEN = GPIO.PWM(green, Freq)
BLUE = GPIO.PWM(blue, Freq)

print('\nНАТИСНІТЬ CTRL+C ЩОБ ВИЙТИ...\n')
try:
        while RUNNING:
                RED.start(100) # Lighting up the pins. 100 means giving 100% to
the pin
                GREEN.start(1)
                BLUE.start(1)

        # Changing the width of PWM, this command is used
                for x in range(1,101):
                        GREEN.ChangeDutyCycle(x)
                        sleep(0.05)

                for x in range(1,101):
                        RED.ChangeDutyCycle(101-x)
                        sleep(0.025)

                for x in range(1,101):
                        GREEN.ChangeDutyCycle(101-x)
                        BLUE.ChangeDutyCycle(x)
                        sleep(0.025)
```

```
                    for x in range(1,101):
                                BLUE.ChangeDutyCycle(x)
                                sleep(0.025)
except KeyboardInterrupt:

        RUNNING = False
        RED.stop()
        GREEN.stop()
        BLUE.stop()

        GPIO.cleanup()
```

ЗАВДАННЯ 3 – САМОСТІЙНО

3.1) Використовуючи програму Fritzing: створити власну схему з'єднання (згідно варіанту завдання) – Raspberry Pi і макетної плати.

3.2) Написати програму для реалізації поставленої задачі. Варіант обираємо згідно номеру у списку академічної групи.

1) Дано датчик температури і вологості DHT11. Реалізувати проект «Збір великих даних 1» → визначити динаміку (незмінність, зростання чи спадання) зміни температури і вологості за останні 5 хвилин, дані відбираються з інтервалом в 1 хвилина. Відобразити текстову і числову інформацію про виконання програми в IoT-платформі InitialState.

2) Дано датчик температури і вологості DHT11. Реалізувати проект «Збір великих даних 2» → визначити середнє арифметичне значення температури і вологості за 5 хвилин, дані відбираються з інтервалом в 10 секунд. Відобразити текстову і числову інформацію про виконання програми в IoT-платформі InitialState.

3) Дано датчик температури і вологості DHT11. Реалізувати проект «Збір великих даних 3» → визначити максимальне і мінімальне значення температури за останні 5 хвилин, дані відбираються з інтервалом в 10 секунд. Відобразити текстову і числову інформацію про виконання програми в IoT-платформі InitialState.

4) Дано світлодіодний модуль RGB KY-016 і кнопку. Реалізувати проект «Світлодіодний модуль і кнопка»: після натиснення кнопки, модуль KY-016 засвічує почергово зелений, синій і червоний колір зі затримки 6-6-6 секунд. Відобразити текстову і числову інформацію про виконання програми в IoT-платформі InitialState.

5) Дано світлодіодний модуль RGB KY-016 і активний зумер. Реалізувати проект «Світлодіодний модуль і активний зумер»: одночасно із загорянням фіолетового, жовтого чи білого кольору модуля KY-016 вмикається і активний зумер. Затримки для KY-016 наступні: 7-5-6 секунд, паузи 1.25 секунд. Відобразити текстову і числову інформацію про виконання програми в IoT-платформі InitialState.

6) Дано датчик температури і вологості DHT11. Реалізувати проект «Збір великих даних 4» → визначити максимальне і середнє арифметичне значення вологості за останні 5 хвилин, дані відбираються з інтервалом в 5 секунд. Відобразити текстову і числову інформацію про виконання програми в IoT-платформі InitialState.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Де застосовується датчик температури та вологості?
2. У яких проектах Інтернету речей доцільно використовувати датчик DHT11?
3. В чому відмінність між датчиком DHT11 і DHT22?
4. Для чого потрібний світлодіодний модуль?
5. Що в програмі визначає запис: DHT_PIN=14?
6. Чим відрізняються датчики температури?
7. Для чого в програмі Python потрібен запис GPIO.setwarnings?
8. Чому потрібно встановлювати спеціальну бібліотеку для роботи з датчиком DHT11?
9. При роботі з датчиком DHT11 необхідний резистор?
10. Чи можна регулювати рівень чутливості датчика DHT11?

ЛАБОРАТОРНА РОБОТА №9

Тема: *Рідкокристалічний дисплей LCD 1602A (16*2, синє підсвічення). Реалізація проектів на базі одноплатного комп'ютера Raspberry Pi. Бібліотека Adafruit_CharLCD.*

Мета: *Формування знань та вмінь, щодо: роботи з Raspberry Pi OS та Fritzing й використання платформи InitialState, контактів GPIO, бібліотеки RPi.GPIO і Adafruit_CharLCD, а також рідкокристалічного дисплея LCD 1602A в проектах Інтернету речей за використання Raspberry Pi.*

ЗАВДАННННЯ 1 – ПРИКЛАД

РІДКОКРИСТАЛІЧНИЙ ДИСПЛЕЙ LCD 1602A. Бажаєте, щоб у ваших проектах Arduino відображались користувацькі повідомлення або показники датчиків? Тоді для цих цілей можна рекомендувати дисплей 1602A. Подібні дисплеї надзвичайно поширені і представляють собою швидкий спосіб додати текстовий інтерфейс в будь-який проект Інтернету речей.

Рідкокристалічний дисплей (Liquid Crystal Display) скорочено LCD побудований на технології рідких кристалів. LCD 1602A це електронний модуль заснований на драйвері HD44780 від Hitachi (рис. 9.1). LCD1602 має 16 контактів і може працювати в 4-бітному режимі або 8-бітному режимі. Для підключення є доступні плати з послідовним інтерфейсом I2C. Даний дисплей ідеально підходить для відображення тексту і символів. Має світлодіодну підсвітку і може відображати 32 символа в кодуванні ASCII в двох рядах по 16 символів в кожному з них. Якщо пригдянутись до дисплея уважно, можна побачити маленькі пік селі – прямокутники 5×8.

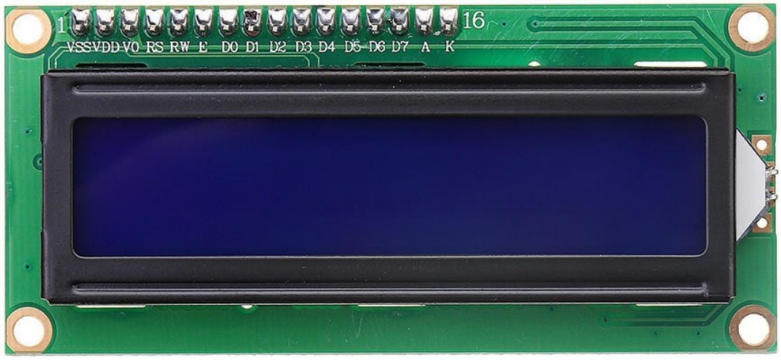
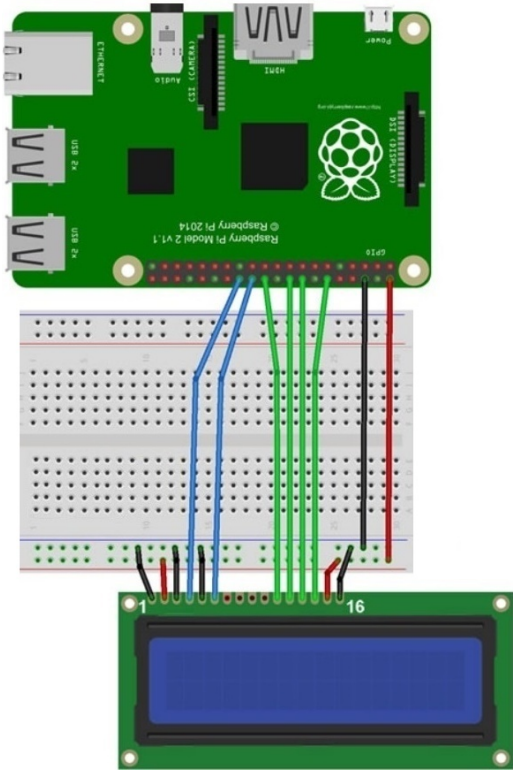


Рис. 9.1. LCD 1602A

На зворотній частині модуля розташовано два чіпа в «краплинному» виконанні (ST7066U і ST7065S) і електрична обв'язка. Є можливість змінювати підсвічування LCD дисплея за допомогою ШІМ. Рідкокристалічний дисплей LCD 1602A є хорошим вибором, коштує недорого і включає різні виробничі модифікації.

1.1)Схема з'єднання Raspberry Pi і макетної плати (в проекті використовується – рідкокристалічний дисплей LCD 1602A і dupont-кабелі).



1.2) Реалізація проекту «Рідкокристалічний дисплей LCD 1602A» у Raspberry Pi (щоб працювати з LCD 1602A у Python потрібно встановити спеціальну бібліотеку Adafruit_CharLCD).

```
from time import sleep
from Adafruit_CharLCD import Adafruit_CharLCD
from ISStreamer.Streamer import Streamer

ACCESS_KEY  = 'ist_jOqcwksNtmshhFL76SwAOoQB2M6vYGCY'
```

```
BUCKET_KEY  = 'PH3JCB9A9X48'
BUCKET_NAME = 'Naz'

streamer = Streamer(bucket_name=BUCKET_NAME, bucket_key=BUCKET_KEY,
access_key=ACCESS_KEY)

lcd=Adafruit_CharLCD(rs=7, en=8, d4=25, d5=24, d6=23, d7=18, cols=16, lines=2)

lcd.clear()
lcd.message('First print\n in LCD display')
sleep(1)

for x in range(0,16):
   lcd.move_right()
   sleep(0.1)

sleep(3)

for x in range(0,16):
   lcd.move_left()
   sleep(0.1)

sleep(1)
lcd.clear()
lcd.message('This is\n       GOOD')

streamer.log('myNumber', 1602)
streamer.log('myMessage', 'Привіт LCD 1602')

streamer.flush()
streamer.close()
```

ЗАВДАННЯ 2 – САМОСТІЙНО

2.1) Використовуючи програму Fritzing: створити власну схему з'єднання (згідно варіанту завдання) – Raspberry Pi і макетної плати.

2.2) Написати програму для реалізації поставленої задачі. Варіант обираємо згідно номеру у списку академічної групи.

Примітка. Інформацію про функції бібліотеки Adafruit_CharLCD вичерпати з освітніх контентів мережі Інтернет.

1) Дано рідкокристалічний дисплей LCD 1602A. Реалізувати проект

«Рух» → продемонструвати рух тексту вліво і вправо на LCD 1602A. Відобразити текстову інформацію про виконання програми в IoT-платформі InitialState.

2) Дано рідкокристалічний дисплей LCD 1602A. Реалізувати проект «Повідомлення» → продемонструвати відображення тексту на LCD 1602A. Відобразити текстову інформацію про виконання програми в IoT-платформі InitialState.

3) Дано рідкокристалічний дисплей LCD 1602A. Реалізувати проект «Очищення» → продемонструвати процес очищення тексту на LCD 1602A. Відобразити текстову інформацію про виконання програми в IoT-платформі InitialState.

4) Дано рідкокристалічний дисплей LCD 1602A. Реалізувати проект «Блимання» → продемонструвати процес блимання тексту на LCD 1602A. Відобразити текстову інформацію про виконання програми в IoT-платформі InitialState.

5) Дано рідкокристалічний дисплей LCD 1602A. Реалізувати проект «Курсор» → продемонструвати процес відображення і зникання курсору на LCD 1602A. Відобразити текстову інформацію про виконання програми в IoT-платформі InitialState.

6) Дано рідкокристалічний дисплей LCD 1602A. Реалізувати проект «Підсвітка» → продемонструвати процес підсвітки тексту на LCD 1602A. Відобразити текстову інформацію про виконання програми в IoT-платформі InitialState.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Для чого потрібна бібліотека Adafruit_CharLCD?
2. В яких проектах доцільно використовувати LCD 1602A?
3. При роботі з LCD 1602A необхідний резистор?
4. Які два модулі розташовані на платі LCD 1602A?
5. Чи можна працювати з LCD 1602A без використання бібліотеки Adafruit_CharLCD?
6. Що таке рідкокристалічний дисплей?
7. Що в програмі визначає запис: lcd.clear()?
8. Для чого потрібний LCD 1602A?
9. Скільки контактів має LCD 1602A?
10. На які категорії поділяються GPIO-контакти?

ЛАБОРАТОРНА РОБОТА №10

Тема: *Радіочастотна ідентифікація. RFID-модуль RC522 з карткою доступу і брелком. Реалізація проектів на базі одноплатного комп'ютера Raspberry Pi. Бібліотека mfrc522.*

Мета: *Формування знань та вмінь, щодо: роботи з Raspberry Pi OS та Fritzing й використання платформи InitialState, контактів GPIO, бібліотеки RPi.GPIO і mfrc522, а також RFID-модуля RC522 в проектах Інтернету речей за використання Raspberry Pi.*

ЗАВДАННННЯ 1 – ПРИКЛАД

РАДІОЧАСТОТНА ІДЕНТИФІКАЦІЯ, RFID-МОДУЛЬ RC522.

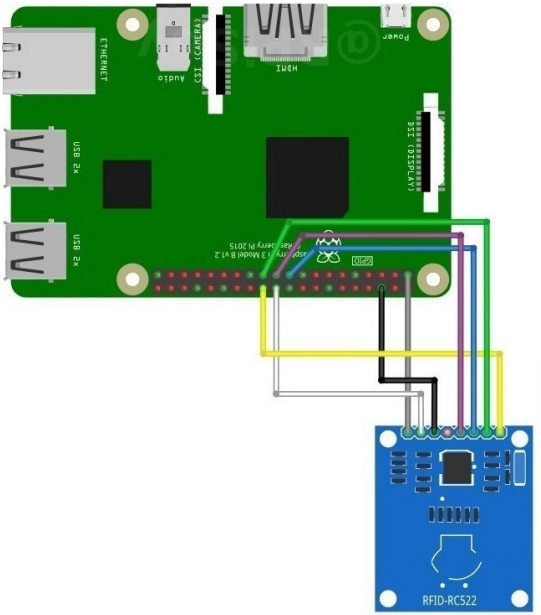
Радіочастотна ідентифікація (RFID) – це технологія автоматичної безконтактної ідентифікації об'єктів за допомогою радіочастотного каналу зв'язку. Базова система RFID складається з: радіочастотної мітки, зчитувача інформації (рідера), прилада для обробки інформації. Ідентифікація об'єктів проводиться за унікальним цифровим кодом, який зчитує з пам'яті електронної мітки, що прикріплюється до об'єкта ідентифікації. Зчитувач містить в своєму складі передавач і антену за допомогою яких випромінюється електромагнітне поле певної частоти. Потрапивши в зону дії пристрою, який зчитує поля радіочастотні мітки, він «відповідає» власним сигналом (ідентифікаційний номер товару, призначені для користувача дані та інше). Сигнал вловлюється антеною зчитувача, інформація розшифровується і передається в мікроконтролер для опрацювання. Переважна більшість сучасних систем контролю доступу використовує в якості засобів доступу ідентифікатори, які працюють на частоті 125 кГц. Захист від копіювання та підробки забезпечують ідентифікатори в чіпах яких реалізований криптографічний захист. Це безконтактні смарт-карти, які працюють на частоті 13,56 МГц, найбільш поширеними з них є карти компанії Mifare. У картах цих стандартів криптозахист організований на високому рівні і підробка таких карт практично неможлива.

RFID-RC522 – це модуль з SPI-інтерфейсом для роботи з RFID-мітками. Робоча частота – 13.56 МГц. У комплекті з модулем наявні дві RFID-мітки: у вигляді карти і брелка (рис. 10.1). Модуль RFID RC522 відноситься до обладнання радіочастотної ідентифікації, виконує функцію розпізнавання свій-чужий. Зчитувач RFID-RC522 або MF-RC522 це стаціонарно встановлюваний пристрій, який спрацьовує при піднесенні на близьку відстань радіочастотної мітки. Основа модуля мікросхема MFRC522 фірми NXP. В якості мітки застосовуються пластикові карти, брелки та інші предмети зі спеціальною вмонтованою

мікросхемою і антеною. Прилад здатний обробляти інформацію одночасно від декількох міток. Для зчитування інформації з мітки їй потрібно досить ненадовго потрапити в зону реєстрації. Мітка працює без джерела живлення за рахунок енергії радіохвиль. Широко застосовується в електронних замках дверей.

Рис. 10.1. RFID-модуль RC522

1.1) Схема з'єднання Raspberry Pi і макетної плати (в проекті використовується – RFID-модуль RC522 і dupont-кабелі).



1.2) Реалізація проекту «RFID-модуль RC522 ЧАСТИНА 1» у Raspberry Pi (щоб працювати з RC522 у Python потрібно встановити спеціальну бібліотеку mfrc522).

Примітка. Перед запуском програм-прикладів треба налаштувати SPI і дві бібліотеки.

```
from RPi import GPIO
from mfrc522 import SimpleMFRC522

reader = SimpleMFRC522()

try:
        text = input('Введіть імя: ')
        print('Тепер піднесіть картку')
        reader.write(text)
        print('OK! ЗаписанО')
finally:
        GPIO.cleanup()
```

1.3) Реалізація проекту «RFID-модуль RC522 ЧАСТИНА 2» у Raspberry Pi.

```
from RPi import GPIO
from mfrc522 import SimpleMFRC522

reader = SimpleMFRC522()

print('Зчитуємо дані...')
try:
        id, text = reader.read()
        print(id)
        print(text)
finally:
        GPIO.cleanup()
```

ЗАВДАННННЯ 2 – САМОСТІЙНО

2.1) Використовуючи програму Fritzing: створити власну схему з'єднання (згідно варіанту завдання) – Raspberry Pi і макетної плати.

2.2) Написати програму для реалізації поставленої задачі. Варіант обираємо згідно номеру у списку академічної групи.

1) Дано RFID-модуль RC522. Реалізувати проект «RFID-модуль і InitialState 1»: на основі прикладу 1.2, реалізувати запис облікових даних на пластикову карту і IoT-платформу InitialState.

2) Дано RFID-модуль RC522. Реалізувати проект «RFID-модуль і InitialState 2»: на основі прикладу 1.3, реалізувати зчитування облікових даних з брелка і їхній запис IoT-платформу InitialState.

3) Дано RFID-модуль RC522. Реалізувати проект «RFID-модуль і InitialState 3»: на основі прикладу 1.2, реалізувати запис облікових даних на брелок і IoT-платформу InitialState.

4) Дано RFID-модуль RC522. Реалізувати проект «RFID-модуль і InitialState 4»: на основі прикладу 1.3, реалізувати зчитування облікових даних з пластикової картки і їхній запис IoT-платформу InitialState.

5) Дано RFID-модуль RC522. Реалізувати проект «RFID-модуль і InitialState 5»: на основі прикладу 1.2 і 1.3, реалізувати відображення графічної інформації (емоджі) на IoT-платформу InitialState при зчитуванні даних з пластикової картки і брелка.

6) Дано RFID-модуль RC522. Реалізувати проект «RFID-модуль і InitialState 6»: на основі прикладу 1.2 і 1.3, реалізувати відображення графічної інформації (емоджі) на IoT-платформу InitialState при записі даних на пластикову картку і брелок.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке радіочастотна ідентифікація?
2. При проектуванні «Розумного будинку» доцільно використовувати RC522?
3. Бібліотека mfrc522 платна чи безкоштовна для використання?
4. У яких сферах використовують радіочастотну ідентифікацію?
5. Чи потрібна бібліотека для роботи з RC522?
6. Що означає запис from mfrc522 import SimpleMFRC522?
7. Який принцип роботи радіочастотної ідентифікації?
8. Яка основна функція картки доступу і брелка?
9. Для чого в програмі потрібно використовувати функцію GPIO.cleanup()?
10. Яке функціональне призначення RFID-модуля RC522?

ЛАБОРАТОРНА РОБОТА №11

Тема: *Кроковий двигун 28BYJ-48. Сервопривод Tower Pro 9g SG90. Реалізація проектів на базі одноплатного комп'ютера Raspberry Pi.*

Мета: *Формування знань та вмінь, щодо: роботи з Raspberry Pi OS та Fritzing й використання платформи InitialState, контактів GPIO, бібліотеки RPi.GPIO, а також крокового двигуна 28BYJ-48 і сервопривода Tower Pro 9g SG90 в проектах Інтернету речей за використання Raspberry Pi.*

ЗАВДАННННЯ 1 – ПРИКЛАД

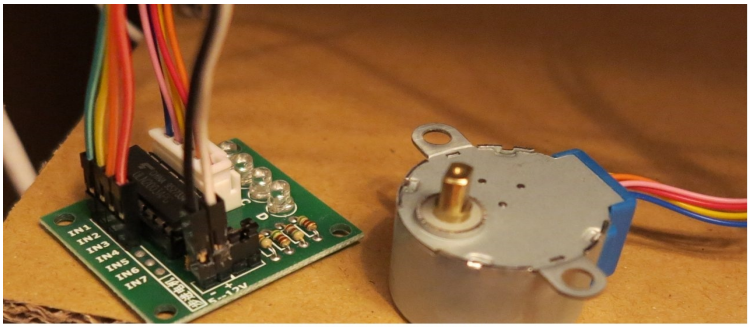
КРОКОВИЙ ДВИГУН 28BYJ-48. Крокові двигуни розроблені для використання в механізмах, де деталі повертаються точно на потрібний кут. Обертання вала крокового двигуна складається з малих переміщень – кроків. Як для свого маленького розміру він досить потужний завдяки тому, що у ньому стоїть редуктор. 28BYJ-48 комплектується разом з драйвером двигуна на мікросхемі ULN2003. Чотирьохфазний кроковий двигун 28BYJ-48 – це безколекторний двигун, обертання валу здійснюється кроками (дискретне переміщення). На роторі, розташований магніт, а навколо нього розташовані котушки, якщо по черзі подавати струм на ці котушки, створюється магнітне поле, яке відштовхує або притягає магнітний вал, тим самим змушуючи двигун обертатися. Така конструкція дозволяє з великою точністю керувати валом. Центральні відводи котушок підключені разом і служать для живлення двигуна. На валу 28BYJ-48 є 8 магнітів, які чергуються полюсами, тобто чотири магніти з двома полюсами (рис. 11.1).

Рис. 11.1. Кроковий двигун 28BYJ-48

Одна з можливий сфер застосувань крокового двигуна – це аматорська робототехніка. Використовуючи 28BYJ-48 легко отримати модель електроприводу робота, що відноситься до класу мотор-колесо, це дозволяє збирати роботів здатних розвернутися на місці і що володіють точним позиціонуванням в просторі завдяки цифровому управлінню двигуном. Кроковий двигун може точно

переміщатися на мінімально можливий кут. Можна вважати, що кроковий двигун дещо схожий на сервопривод. Однак, сервоприводи обмежені кутом повороту в діапазоні від 0 до 180°, кроковий же двигун може обертатися безперервно, подібно двигуну постійного струму. Перевагою крокових двигунів є те, що можна досягти набагато більшого ступеня контролю над рухом об'єкта. До недоліком крокових двигунів можна віднести більш складне управління, ніж у випадках з сервоприводами або моторами постійного струму.

1.1)Схема з'єднання Raspberry Pi і макетної плати (в проекті використовується – кроковий двигун 28BYJ-48 і dupont-кабелі).



ПІДКЛЮЧЕННЯ НА RPI

ПРИКЛАД НА ARDUINO

1.2) Реалізація проекту «Кроковий двигун 28BYJ-48» у Raspberry Pi.

```
from RPi import GPIO
from time import sleep
import os, random
```

```
from ISStreamer.Streamer import Streamer

ACCESS_KEY  = 'ist_jOqcwksNtmshhFL76SwAOoQB2M6vYGCY'
BUCKET_KEY  = 'PH3JCB9A9X48'
BUCKET_NAME = 'Naz'

# create a Streamer instance
streamer = Streamer(bucket_name=BUCKET_NAME, bucket_key=BUCKET_KEY,
access_key=ACCESS_KEY)

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

class stepmotor:

    def __init__(self, IN1, IN2, IN3, IN4):

        self.IN1 = IN1 # IN1
        self.IN2 = IN2 # IN2
        self.IN3 = IN3 # IN3
        self.IN4 = IN4 # IN4

        self.time = 0.001

        GPIO.setup(IN1,GPIO.OUT)
        GPIO.setup(IN2,GPIO.OUT)
        GPIO.setup(IN3,GPIO.OUT)
        GPIO.setup(IN4,GPIO.OUT)

        GPIO.output(IN1, False)
        GPIO.output(IN2, False)
        GPIO.output(IN3, False)
        GPIO.output(IN4, False)

    def Step1(self):
        GPIO.output(self.IN4, True)
        sleep (self.time)
        GPIO.output(self.IN4, False)
```

```
def Step2(self):
    GPIO.output(self.IN4, True)
    GPIO.output(self.IN3, True)
    sleep (self.time)
    GPIO.output(self.IN4, False)
    GPIO.output(self.IN3, False)

def Step3(self):
    GPIO.output(self.IN3, True)
    sleep (self.time)
    GPIO.output(self.IN3, False)

def Step4(self):
    GPIO.output(self.IN2, True)
    GPIO.output(self.IN3, True)
    sleep (self.time)
    GPIO.output(self.IN2, False)
    GPIO.output(self.IN3, False)

def Step5(self):
    GPIO.output(self.IN2, True)
    sleep (self.time)
    GPIO.output(self.IN2, False)

def Step6(self):
    GPIO.output(self.IN1, True)
    GPIO.output(self.IN2, True)
    sleep (self.time)
    GPIO.output(self.IN1, False)
    GPIO.output(self.IN2, False)

def Step7(self):
    GPIO.output(self.IN1, True)
    sleep (self.time)
    GPIO.output(self.IN1, False)

def Step8(self):
    GPIO.output(self.IN4, True)
    GPIO.output(self.IN1, True)
```

```
        sleep (self.time)
        GPIO.output(self.IN4, False)
        GPIO.output(self.IN1, False)

    def left(self, step):
        for i in range(step):
            self.Step1()
            self.Step2()
            self.Step3()
            self.Step4()
            self.Step5()
            self.Step6()
            self.Step7()
            self.Step8()
            print('Step left: ',i)

    def right(self, step):
        for i in range(step):
            self.Step8()
            self.Step7()
            self.Step6()
            self.Step5()
            self.Step4()
            self.Step3()
            self.Step2()
            self.Step1()
            print('Step right: ',i)

half = int(512/2)
quarter = int(512/4)
full = int(512)
twice = int(512*2)

motor1 = stepmotor(4,17,20,21)

for i in range(1):
    motor1.right(twice)

for i in range(1):
```

```
    motor1.left(twice)
# send some data
streamer.log('myNumber', twice)
streamer.log('myMessage', 'Кроковий двигун 28BYJ-48')

streamer.flush()
streamer.close()

GPIO.cleanup()
```

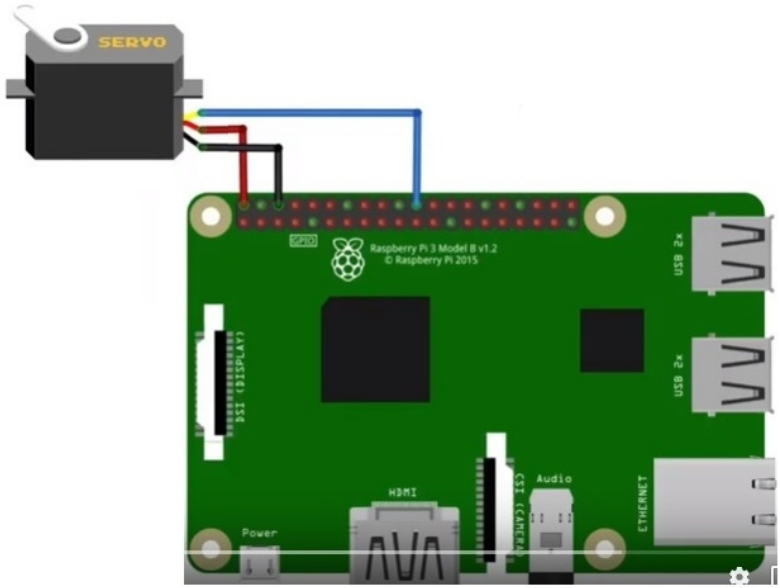
ЗАВДАННННЯ 2 – ПРИКЛАД

СЕРВОПРИВОД TOWER PRO 9G SG90. Сервоприводом є будь-який тип механічного приводу, що має в складі датчик (положення, швидкості, зусилля) і блок управління приводом (електронну схему), автоматично підтримує необхідні параметри на датчику відповідно до заданого зовнішньому значенням. Основні компоненти сервоприводу – це привід, датчик, блок керування, конвертер. Сервоприводом називається такий привід, точне управління яким здійснюється через негативний зворотний зв'язок і дозволяє таким чином домогтися необхідних параметрів руху робочого пристрою. Механізми цього типу мають датчик, які відслідковує конкретний параметр, наприклад швидкість, положення або зусилля, а також блок управління, завдання якого – підтримувати в автоматичному режимі необхідний параметр в процесі роботи пристрою в залежності від сигналу з датчика в кожен момент часу. Чимало підсилювачів і регуляторів з негативним зворотним зв'язком можуть бути віднесені до сервоприводу. Наприклад, до сервоприводів відносяться гальмівна система і рульове керування в автомобілях, де підсилювач ручного приводу обов'язково має негативний зворотний зв'язок по положенню (рис. 11.2).

Рис. 11.2. Сервопривод Tower Pro 9g SG90

Сервопривод це пристрій, який забезпечує перетворення сигналу в відповідне переміщення чи поворот. Являє собою прямокутну коробку з мотором, схемою і редуктором всередині та вихідним валом, який може повертатися на фіксований кут, який визначається вхідним сигналом. Існує багато видів сервоприводів, які розрізняються габаритами, матеріалом шестерень (пластмаса, метал), способом керування (аналогові і цифрові), швидкістю обертання валу, крутним моментом, діапазоном повороту (120°, 180°, безперервне обертання). Сервоприводи володіють хорошим крутним моментом до 13 кг/см, металевими шестернями і тому часто використовуються в авіамоделюванні та робототехніці (наприклад, для повороту голови або руки робота). В якості приводу може використовуватися наприклад пневмоциліндр зі штоком або електродвигун з редуктором. Датчиком зворотного зв'язку може бути енкодер (датчик кута повороту) або датчик Холла.

2.1)Схема з'єднання Raspberry Pi і макетної плати (в проекті використовується – сервопривод Tower Pro 9g SG90 і dupont-кабелі).



2.2) Реалізація проекту «Сервопривод Tower Pro 9g SG90» у Raspberry Pi.

```
from RPi import GPIO
from time import sleep
from ISStreamer.Streamer import Streamer
```

```
ACCESS_KEY  = 'ist_jOqcwksNtmshhFL76SwAOoQB2M6vYGCY'
BUCKET_KEY  = 'PH3JCB9A9X48'
BUCKET_NAME = 'Naz'

# create a Streamer instance
streamer = Streamer(bucket_name=BUCKET_NAME, bucket_key=BUCKET_KEY,
access_key=ACCESS_KEY)

servuk=14 # На малюнку-схемі інший контакт GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(servuk,GPIO.OUT)

p=GPIO.PWM(servuk,50)
p.start(7.5)

# send some data
streamer.log('myNumber', servuk)
streamer.log('myMessage', 'Servuk')

try:
    while True:
        p.ChangeDutyCycle(7.5)
        print('Вліво')
        sleep(1)
        p.ChangeDutyCycle(12.5)
        print('Центр')
        sleep(1)
        p.ChangeDutyCycle(2.5)
        print('Вправо\n')
        sleep(1)
except KeyboardInterrupt:
    p.stop()
    GPIO.cleanup()
```

ЗАВДАННЯ 3 – САМОСТІЙНО

3.1) Використовуючи програму Fritzing: створити власну схему з'єднання (згідно варіанту завдання) – Raspberry Pi і макетної плати.

3.2) Написати програму для реалізації поставленої задачі. Варіант обираємо згідно номеру у списку академічної групи.

1) Дано сервопривод Tower Pro. Реалізувати проект «Сервопривод 1»: виконати поворот сервопривода на 180 градусів за часовою стрілкою, час здійснення повороту і кут повороту записати в InitialState.

2) Дано кроковий двигун. Реалізувати проект «Кроком руш 1»: виконати один повний поворот за годинниковою стрілкою, час здійснення повороту і кут повороту записати в InitialState.

3) Дано сервопривод Tower Pro. Реалізувати проект «Сервопривод 2»: виконати поворот сервопривода на 90 градусів проти часової стрілки, час здійснення повороту і кут повороту записати в InitialState.

4) Дано кроковий двигун 28BYJ-48. Реалізувати проект «Кроком руш 2»: виконати один повний поворот проти годинникової стрілки, час здійснення повороту і кут повороту записати в InitialState.

5) Дано кроковий двигун 28BYJ-48. Реалізувати проект «Кроком руш 3»: виконати один повний поворот за годинниковою стрілкою і один проти годинникової стрілки, час закінчення повороту і сумарний кут повороту записати в InitialState.

6) Дано сервопривод Tower Pro. Реалізувати проект «Сервопривод 3»: виконати поворот сервопривода на 270 градусів за часовою стрілкою, час здійснення повороту і кут повороту записати в InitialState.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке кроковий двигун?
2. Чим відрізняються крокові двигуни?
3. Що таке сервопривод?
4. В яких проектах Інтернету речей доцільно використовувати сервоприводи?
5. Чи потрібна бібліотека для роботи з сервоприводом?
6. За яким принципом працює кроковий двигун?
7. Чи має обмеження в куті повороту кроковий двигун 28BYJ-48?
8. В яких проектах Інтернету речей доцільно використовувати кроковий двигун?
9. Де застосовуються крокові двигуни?
10. Що в програмі визначає запис: import os, random?

ЛАБОРАТОРНА РОБОТА №12

Тема: *Датчик вібрації і нахилу SW-520D. Клавіатурна матриця 4*4. Реалізація проектів на базі одноплатного комп'ютера Raspberry Pi. Клас Keypad.*

Мета: *Формування знань та вмінь, щодо: роботи з Raspberry Pi OS та Fritzing й використання платформи InitialState, контактів GPIO, бібліотеки RPi.GPIO, класу Keypad, а також датчика вібрації і нахилу SW-520D і клавіатурної матриці 4*4 в проектах Інтернету речей за використання Raspberry Pi.*

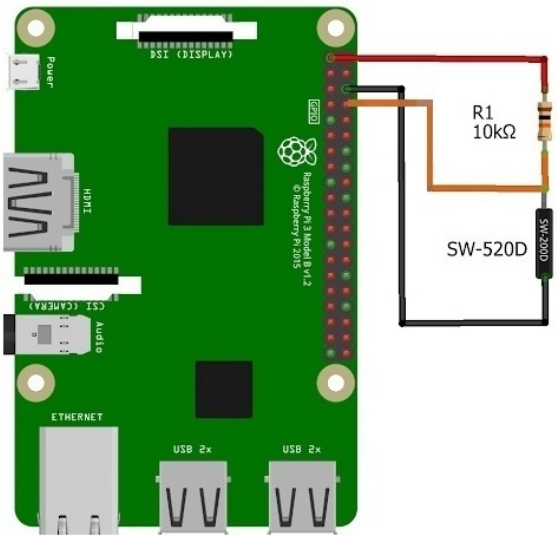
ЗАВДАННННЯ 1 – ПРИКЛАД

ДАТЧИК ВІБРАЦІЇ І НАХИЛУ SW-520D. На сьогодні, датчики нахилу є важливими компонентами систем охоронної сигналізації. Сенсорні датчики нахилу визначають кут нахилу або рух. Такі датчики можуть бути реалізовані з використанням технології ртуті і роликових кульок та можуть бути встановлені з використанням механічного різьблення, магнітів або клеїльних засобів залежно від того на який тип поверхні вони встановлюються. Датчики вібрації (їх ще іноді називають датчиками сигналізації) застосовуються для виявлення зловмисних дій вібраційного характеру і широко використовуються в автомобільних системах, різних охоронних сигналізаціях, дозволяють навіть виявити незначні вібрації при зародковому землетрусі.

Датчик вібрації та нахилу серії SW-520D працює від 5 В і споживає до 5 мА (рис. 12.1). Максимальна напруга, яка може без наслідків витримати датчик, складає 20 В, а максимальний струм – 0.3 А. Якщо виводи спрямовані вниз, то при нахилі більше 15° відбудеться спрацювання. Коли виводи спрямовані вгору, спрацювання буде трохи ускладнене. Він може зафіксувати два положення в просторі (вертикальне чи горизонтальне), а також виявити вібрацію. Всередині датчика вібрації розташовується металева кулька з електродами, поміщеними в корпус. Коли кулька скочується до електродів, ланцюг замикається. Модуль сумісний з будь-якими мікроконтролерами, включаючи Arduino. Підключається до плати через аналоговий і цифровий вихід, другий контакт підключається до GND.

Рис. 12.1. Датчик вібрації і нахилу SW-520D

1.1)Схема з'єднання Raspberry Pi і макетної плати (в проекті використовується – датчик вібрації і нахилу SW-520D і dupont-кабелі).



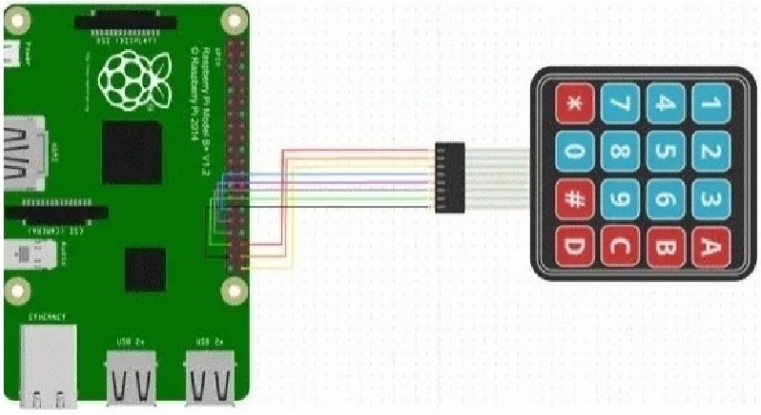
1.2) Реалізація проекту «Датчик вібрації і нахилу SW-520D» у Raspberry Pi.

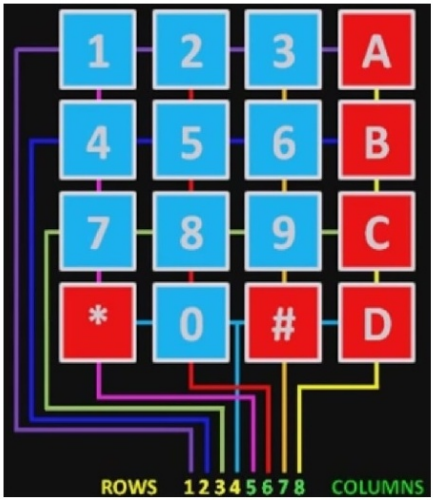
```
from RPi import GPIO
SW=14

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(SW, GPIO.IN)

print('Завмираєм в очікуванні...\n')

try:
    while True:
        if GPIO.wait_for_edge(SW, GPIO.BOTH):
            print('A-A-A...Нахил і Відбрація')
except KeyboardInterrupt:
    print('На все добре')
    GPIO.cleanup()
```

ЗАВДАННЯ 2 – ПРИКЛАД

КЛАВІАТУРНА МАТРИЦЯ 4*4.

2.1)Схема з’єднання Raspberry Pi і макетної плати (в проекті використовується – клавіатурна матриця і dupont-кабелі). *Примітка. В програмі-прикладі інша схема з’єднань.*





2.2) Реалізація проекту «Клавіатурна матриця 4*4» у Raspberry Pi.

```
from RPi import GPIO

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

MATRIX=[ [1, 2, 3, 4],
```

```
        [5, 6, 7, 8],
        [9, 10,11,12],
        [13,14,15,16] ]

ROW = [14,15,18,23]
COL = [6,13,19,26]

for j in range(4):
    GPIO.setup(COL[j], GPIO.OUT)
    GPIO.output(COL[j], 1)

for i in range(4):
    GPIO.setup(ROW[i], GPIO.IN, pull_up_down=GPIO.PUD_UP)

print('\nПІСЛЯ ЗАВЕРШЕННЯ ВВОДУ...НАТИСНІТЬ CTRL+C \n')
try:
  while(True):
      for j in range (4):
          GPIO.output(COL[j], 0)

          for i in range(4):
              if GPIO.input(ROW[i])==0:
                  print(MATRIX[i][j])
                  while(GPIO.input(ROW[i])==0):
                      pass
          GPIO.output(COL[j], 1)

except KeyboardInterrupt:
    GPIO.cleanup()
```

ЗАВДАННЯ 3 – САМОСТІЙНО

3.1) Використовуючи програму Fritzing: створити власну схему з'єднання (згідно варіанту завдання) – Raspberry Pi і макетної плати.

3.2) Написати програму для реалізації поставленої задачі. Варіант обираємо згідно номеру у списку академічної групи.

1) Дано клавіатурну матрицю 4*4 і активний зумер. Реалізувати проект «Клавіатура і зумер»: якщо натискається кнопка «8» → вмикається звуковий сигнал активного зумера, якщо натискається інша кнопка → її

число відображається на InitialState.

2) Дано датчик вібрації і нахилу SW-520D та світлодіод (червоний). Реалізувати проект «Датчик вібрації і нахилу 1»: при детекції нахилу чи вібрації світлодіод починає мигати з довільним інтервалом і затримкою. На InitialState пересилається день тижня і число виконання роботи.

3) Дано клавіатурну матрицю 4*4 і два світлодіода (жовтий і червоний). Реалізувати проект «Клавіатура і світлодіод 1»: якщо натискається кнопка «5» → вмикається і постійно горить червоний світлодіод, якщо натискається інша кнопка → вмикається і постійно горить жовтий світлодіод. На InitialState пересилається вітальне повідомлення.

4) Дано датчик вібрації і нахилу SW-520D та активний зумер. Реалізувати проект «Датчик вібрації і нахилу 2»: при детекції нахилу чи вібрації активний зумер починає пищати з довільним інтервалом і затримкою. На InitialState пересилається день тижня і число виконання роботи.

5) Дано клавіатурну матрицю 4*4 і світлодіод (зелений). Реалізувати проект «Клавіатура і світлодіод 2»: якщо натискається кнопка «5» → вмикається і постійно горить світлодіод, якщо натискається інша кнопка → її число помножене на 5 відображається на InitialState.

6) Дано датчик вібрації і нахилу SW-520D та пасивний зумер. Реалізувати проект «Датчик вібрації і нахилу 3»: при детекції нахилу чи вібрації пасивний зумер починає пищати з довільним інтервалом і затримкою. На InitialState пересилається день тижня і число виконання роботи.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що в програмі визначає запис: pass?
2. Яку силу струму споживає датчик вібрації та нахилу серії SW-520D?
3. Яке призначення кнопки в електроніці?
4. Для чого потрібно знати GPIO схему?
5. Що таке аналогового-цифровий перетворювач?
6. Для чого в програмі потрібно вносити запис sleep()?
7. За яким принципом працює датчик нахилу?
8. У яких приладах використовується клавіатурна матриця?
9. У яких проектах доцільно використовувати SW-520D?
10. Чи потрібна бібліотека для роботи з датчиком вібрації і нахилу?

ВИМОГИ ДО ВИКОНАННЯ ЛАБОРАТОРНИХ РОБІТ

1. Загальні вимоги до виконання лабораторних робіт:

   а) опрацювати лекційний матеріал із даної теми;

   б) виконати всі завдання лабораторної роботи за період визначений навчальним планом для відповідної академічної групи;

   в) підготувати звіт на підставі виконаної лабораторної роботи у друкованому або електронному вигляді.

2. Звіт до кожної лабораторної роботи повинен містити:
   - титульну сторінку;
   - номер і тему відповідної роботи;
   - мету та постановку завдання;
   - хід виконання роботи;
   - відповіді на контрольні запитання;
   - лаконічні висновки.

3. Вимоги щодо оформлення звіту:
   - звіт оформляється у текстовому процесорі (Word, Google Docs, LibreOffice, SmartOffice, Apache OpenOffice.org тощо);
   - тип шрифту Times New Roman;
   - кегель – 14;
   - міжрядковий інтервал – 1.15;
   - абзацний відступ – 1.25 см;
   - параметри сторінки: зверху та знизу – 2 см, зліва – 3 см, справа – 1.5 см;
   - вирівнювання основного тексту – по ширині;
   - виділення, підкреслення та нахил тексу на вибір виконавця.

В підсумку, оцінювання лабораторної роботи проводиться на підставі перевірки виконаних завдань та оформленого згідно вимог звіту.

СПИСОК ПИТАНЬ ВИНЕСЕНИХ НА САМОСТІЙНУ РОБОТУ

1. Інтернет речей та міжмашинна взаємодія.
2. Туманні і граничні обчислення.
3. Основи цифрових і вбудованих систем.
4. Стандартизація Інтернету речей.
5. Радіочастотна інтерференція.
6. Напрямки практичного застосування Інтернету речей.
7. Комунікації малого радіусу дії.
8. Програмно-визначена мережа (SDN).
9. Топологія та архітектура мережі 4G LTE.
10. Фізичний рівень LoRa.
11. Рівень MAC LoRaWAN.
12. Топологія LoRaWAN.
13. Використання технології Інтернет речей у різних сферах діяльності.
14. Програмна мережева взаємодія.
15. Радіоспектр.
16. Zigbee.
17. Проблеми впровадження Інтернету речей.
18. Інтернет-маршрутизація.
19. Програмні системи моделювання вбудованих систем.
20. Інтернет сільськогосподарських речей.
21. Фізична безпека Інтернету речей.
22. Вузькосмуговий та широкосмуговий зв’язок.
23. Технологія міжмашинної взаємодії (M2M).
24. Індустрія 4.0.
25. Цифрова трансформація.
26. Інтелектуальне місто.
27. Промисловий Інтернет речей.
28. Топологія 6LoWPAN.
29. Безпека IEEE 802.11.
30. Технології доступу стільникового зв’язку.
31. Протокол MQTT.
32. Широтно-імпульсна модуляція.
33. Годинник реального часу DS1307.
34. Архітектура і організація вбудованих систем.
35. Робототехнічні системи.
36. Стандарти 802.15.
37. Протокол CoAP.
38. Архітектура та топологія Thread.
39. Хмарна архітектура OpenStack.
40. Апаратна безпека Інтернету речей.

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. Семюел Грінгард Інтернет речей. Харків : «КСД», 2018. 176 с.
2. Мачей Кранц. Інтернет речей. Нова технологічна революція. Київ : Ексмо, 2018. 336 с.
3. Семюел Грінгард Інтернет речей. Майбутнє вже тут. Харків : «КСД», 2017. 224 с.
4. Баранов А.А. Інтернет речей: теоретико-методологічні основи правового регулювання. Том І. Сфери застосування, ризики і бар'єри, проблеми правового регулювання. Харків : Право, 2018. 344 с.
5. Девід Роуз Дивовижні технології. Дизайн та Інтернет речей. Харків : «КСД», 2018. 336 с.
6. Chin S., Weaver J. Raspberry Pi with Java: Programming the Internet of Things. Oracle Press, 2016. 261 c.

Навчально-методичне видання

«ІНТЕРНЕТ РЕЧЕЙ»

методичні вказівки до виконання лабораторних робіт

для студентів спеціальностей
122 Комп'ютерні науки
113 Прикладна математика
121 Інженерія програмного забезпечення

Друкується в авторській редакції

Підписано до друку 27.03.2024 р.
Формат 60×90/16.
Ум. друк. арк. 3,9. Тираж 50 прим.
Замовлення № 690/2

Відділ мережевого та інформаційного забезпечення
Рівненського державного гуманітарного університету.
33028, м. Рівне, вул. С. Бандери, 12.