Date :

Practical NO: 01

* Aim :- To study Arduino Uno in details.

* Theory :

① Arduino Uno :

Arduino Uno is an open-source micro-controller board based on the ATMega 328 P.

It is widely used for embeded System, robotics, and IOT applications due to its easy to use, affordability & strong community support.

② features of Arduino Uno :

- Microcontroller : ATmega 328 p.
- Operating Voltage : 5 V.
- Input Voltage (re commended) : 7-12 V.
- Digital I/O pins : 14(6 of which can be used as pwm outputs)
- Analog Input pins : 6
- flash memory : 32 KB (with 0.5 KB used by the bootloader)
- SRAM : 2 KB

Date :

- EEPROM : 1KB
- Clock speed : 16 MHz
- USB connection : Used for programming & power supply.
- Power jack : External power supply option.
- Reset Button : Resets the microcontroller

⑤ Pin configuration of Arduino Uno :

①. Power Pin :
Vin : external power supply (7-12 V)
5v : Regulated 5V output
3.3v : Regulated 3.3V output
GND : Ground pins.

2. Digital I/O pins (0-13) : Used for input/output operations.

3. PWM pins (3, 5, 6, 9, 10, 11) : Used for pulse width modulation (PWM).

4. Analog Input Pins (A0-A5) : Used for reading Analog signals.

5. Communication pins:
Tx (1) & Rx (0) : Serial communication

Date :

- SPI Pins (10-13) used for SPI communication
- I2C pins (A4 & A5): Used for I2C communication.

④ Working of Arduino Uno:

The Arduino Uno is programmed using the arduino IDE, which supports C and C++.
The programming process follows those steps:

Steps:

Step ①:- Write Code :- Using Arduino uno programming language.

Step②:- Compile code :- Converts the code into machine language.

step③ :.. Upload code : Sends the compiled code to the microcontroller vice USB.

Step④: Execution : The micro controller execute the instructions.

⑤ Communication Interface :-

- Serial communication (UART): Uses TX(1) & RX(0) for data transmission.

- SPI communication: Use MOSI, MISO, and SCL pins for fast data transfer.
- I2C communication: Uses SDA (A4) and SCL (A5) for connecting multiple devices.

⑥ Applications for Arduino Uno:

- Home Automation
- IOT projects
- Robotics
- Sensor-based Applications
- Embedded System Development.

⑦ Advantages of Arduino Uno:

- Easy to learn & use
- Affordable & open source
- Large community support
- Multiple communication options.
- Compatible with various sensors and shields.

* Result:- Hence, studied to get details of Arduino uno has been successfully.

06/02/25 (A)

Date :

Practical No. 2

* Aim :- To connect and Implement ultrasonic sensor and produce output.

* Theory :-

* Ultrasonic Sensor :

→ An ultrasonic sensor is an instrument that measures the distance to an object using ultrasonic sound waves.

→ In this project we will use HC-SR04 ultrasonic sensor it works by sending sound waves from the transmitter, which thin bounce off of an object and object & return to the receiver.

→ You can determine how far away something is by the time it takes for the sound wave to get back to the sensor.

→ They are widely used in application such as obstacle detection, level sensing, and robotics due to their relatively low cost, easy of interfacing, & ability to work in various environment conditions.

* Connections :-

- Vcc to 5V
- GND to GND

Date :

- Trig to pin 7
- Echo to pin 6.

* Working Principle :

1. Pulse Emission :

The sensor's transmitter emits a brief burst of ultrasonic sound waves (typically of frequencies above 20KHz).

2. Propagation and Reflection :

- These sound waves travel through the air until they hit an object, when they encounter a surface, the waves are reflected back toward the sensor.

3. Echo Reception :

- The sensor's detects the returning sound waves (echo). It then measures the time interval between the emission of pulse & the reception of the echo.

4. Distance calculation:

$$\text{Distance} = \frac{\text{Time} \times \text{speed of sound}}{2}$$

The sensor calculate the distance to the object. The division by 2 accounts for the round trip of second wave.

* Program :

```
const int pingPin = 7 ; // triased pin of U.S.
const int echoPin = 6 ; // Echo Pin of U.S.

void setup () {
 for
 long duration , inch () , cm ;
 pinMode (pingfin , OUTPUT);
 digitalWrite (pingfin , LOW);
 delayMicroseconds (2);

 digitalWrite (pingPin , HIGH);
 delay Microseconds (10);
 digitalWrite (pingPin , LOW );

 PinMode (echoPin , Input );
 duration = pulseIn (echoPin , HIGH);
 inches = microseconds To Inches (duration);
 cm = microseconds To Centimeters (duration);
Serial.print (inches);
Serial.print (" in , ");
Serial.print ( cm );
Serial.print (" cm ");
Serial.print ();

delay (100);

}
```

Date :

```
long microsecondsToInches(long microseconds){
    return microseconds / 74 / 2;
}
long microsecondsToCentimeters(long microseconds){
    return microseconds / 29 / 2;
}
```

* Output :-

11:53 : 10.671 → 19 in, 48 cm
11:53 : 10.771 → 17 in, 44 cm
11:53 : 10.870 → 15 in, 38 cm
11:53 : 10.970 → 17 in, 44 cm
11:53 : 11.102 → 16 in, 42 cm

* Result :- Hence a connected and implemented ultrasonic sensor and produced out.

Date :

Practical No. 3

* **Aim :-** To connect and implement servo motor produced output.

* **Theory :-**

* Servo Motor *

→ A servo motor is an electromechanical device designed to provide precise control of angular position, speed and acceleration. Unlike a standard DC motor, which rotates continuously when powered, a server motor is capable of stopping at a specific angle as directed by a control signal.

* Working Principle of a servo motor *

- PWM control :
- The servo motor receives a pulse width modulated signal from a microcontroller. The duration of the pulse determines the desired angular position.

- Internal feedback :
- Inside the servo, a potentiometer continuously monitor the actual position of the motor shaft.

- Closed - loop system :
- The servo's control circuitry compare the * signal (desired position)

Date :

with the potential but output (actual position) and adjust the motor's movement accordingly.

- Output :
  - The motor rotates to the specified angle and maintain that position until a new PWM signal is received.

* Program :

```
#include <Servo.h>

// Declare the servo pin
int servoPin = 9

// Create a servo object

Servo Servo1;
void setup()
{

  // We need to attach the servo to the
     used pin number.
  Servo1.attach(servoPin);

}

void loop() {
```

Date :

```
// make servo go to 0 degrees.
Servo1.write(0);
delay(1000);
// make servo go to 90 degrees
Servo1.write(90);
delay(1000);
// Make servo go to 180 degrees
servo1.write(180);
delay(1000);

}
```

* Result :- Hence a connected and implemented servo motor and produced output.

Date :

Practical NO: 04

* Aim :- To connect and implement LED to Arduino UNO and produce output.

* Theory :-

An LED (Light emitting diode) is a semiconductor device that emits light when an electric current flows through it. Arduino UNO can control the LED by providing an ON/off signal via a digital pin.

- A simple LED circuit consist of
  - An LED
  - A current-Limiting resistor (typically 220Ω or 330Ω)
  - Arduino UNO.

- Circuit connections:

① Connect the positive leg (anode) of the LED to digital pin 13 of Arduino.

② Connect the negative leg (cathode) to one end of a 220 Ω resistor.

③ Connect the other end of the resistor of GND of Arduino.

Date :

* Program :-

```
const int ledpin = 4;

Void setup () {
    pinMode (Ledpin, OUTPUT);
     4 as an output
   }

Void loop () {
    digitalwrite (led pin, HIGH);
LED ON
    delay (1000);
    digitalWrite (LedPin, LOW);
LED off
    delay (1000);
}
```

* Output :

The LED will blink continuously, turning ON for 1 seconds and off for 1 second in a loop.

* Result :-

The LED was successfully connected to the Arduino UNO & blinked at 1- second intervals as expected.

Date :

Practical No. 5

* Aim :- To connect and implement 2 LED to Arduino UNO and produced output.

* Theory :-

In this practical, we connect two LED's to an Arduino UNO and control them using digital output pins, By programming the Arduino, we can turn the LEDS on and OFF in a sequence, creating a blinking pattern.

This is useful for applications like indicator lights, alternating signals, or simple traffic light systems.

- Components Required :

1. Arduino UNO.
2. Two LEDs (any color)
3. Two 220 Ω Resistors
4. Jumper wires
5. Breadboard (optional)

* Circuit Connections :

1. LED 1 (Red) :

- Anode (+) → digital pin 9.
- Cathode (−) → one end of a 220 Ω resistor
- Other end of the resistor → GND

Date :

② LED 2 (Green):

- Anode (+) → Digital pin 10
- Cathode (-) → One end of a 220 Ω resistor.
- other end of the resistor → GND.

* Advantges:

① Multi-LED control.

② Easy to implement.

③ low power consumption.

* Disadvantages:

① Limited control

② more components Needed.

* Program:

```
const int Led 1 = 9;
const int Led 2 = 10;

void setup () {
    Pinmode (Led 1, output);
    Pinmode (Led 2, output);

}
```

Date :

```
void loop () {
    digital write ( LED 1 , HIGH );
1 ON
    digitalWrite ( LED 2, low );
2 off
    digitalWrite ( Led 2, low ) delay (1000 );

    digital write ( Led 1, High );
1 off
    digital Write (led 2, High );
2 ON
    delay (1000);
}
```

* Output :

- Let LED 1 turn ON while LED 2 turns off for 1 second.
- Let LED 1 turn off while LED 2 turns on for 1 seconds.
- The cycle repeats continuously

* Result :

Both LED were successfully control using the Arduino blinking alternately at 1 second intervals.

Date :

Practical NO. 6

* Aim :- To connect and Implement IR sensor and produce output.

* Theory :-

An Infrared (IR) sensor detects obstacles by emitting infrared light and measuring the reflected signal.

It is commonly used in line following robots, object detection, and security systems.

* Components Required :

① Arduino UNO

② IR Sensor Module

③ 1 LED (optional, for testing)

④ Jumper Wires

* Circuit connections :

① IR sensor VCC → 5V (Arduino)

② IR sensor GND → GND (Arduino)

③ IR Sensor Out → Digital pin 2 (Arduino)

④ (optional) LED → Digital Ping (for o/p).

Date :

* Advantages :-

① Non - contact Detection :

② Low power consumption.

③ Versatile Application.

* Disadvantages :

① Limited range.

② Affects performance in sunlight.

* Code :

```
const int irsensor = 2;
const int led = 9;

void setup () {
    pinMode (irsensor, INPUT);
    PinMode (Led, OUTPUT);
    serial.begin (9600);
}
void loop () {
    int sensorstate =
    digitalRead (irsensor);

if (sensorstate == low) {
    digitalWrite (Led, HIGH);
```

Date :

object detected, turn LED ON
serial.println("object Detected");
} else {
digitalWrite(Led, LOW);
object, turn LED off
serial.println("NO object
Detected");
}
delay(500);
}

* Outputs:

- If an object is detected, LED turns ON, and "Object Detected" prints in serial monitor.
- If no object is detected, LED stays offs, and "No objected Detected" prints in Serial Monitor.

* Results:

The IR sensor was successfully connected and detected objects, triggering the LED accordingly.

Date :

Practical NO. 7

* Aim :- To connect & implement 2 Servo motor to Arduino UNO and produce output.

* Theory :-

A servo motor is a precise motor used for robotics arm, RC cars, and automation system. It rotates within a specific angular ranges (60° to 180°) based on PWM signals from the Arduino.

* Components Required :

① Arduino UNO

② Two servo motors

③ External power source (if needed)

④ Jumper wires

* Circuit connections :

① Servo 1 :

- VCC → 5V (Arduino)
- GND → GND (Arduino)
- Signal → Digital Pin 10

Date :

* Advantage:

① Precise Angular control

② Low Power Usage

③ Common in Robotics.

* Disadvantages:

① Limited Rotation Range.

② Requires PWM signal

③ Weak under Heavy load.

* Code:

```
#include <servo.h>

servo servo1;
Servo servo2;

void setup () {
    servo1.attach (9);
to pin 9
    servo 2. attach (10);
 to pin 10
}
```

Date :

```
void loop(){
    servo1.write(0);
  0 degrees
    servo2.write(180);
 180 degrees
    delay(1000);

    servo1.write(180);
 180 degrees
    servo2.write(0);
  0 degrees
    delay(1000);
}
```

* Output:-

servo 1 rotates from 0° to 180° while servo 2 rotates from 180° to 0°, then they swap positions every second.

* Results-

Both servo were successfully controlled using Arduino, rotating to specified angles at 1-second intervals.

Date :

Practical No. 8

* Aim :- To connect and interface RFID with Arduino UNO and produce output.

* Theory :-

The MFRC522 is a popular RFID (Radio frequency Identification) Module used for reading and writing data to RFID tags. This module is widely used in projects involving access control, identification systems, & data storage.

- The MFRC522 RFID module has a total of 8 pins, which are as follows.

1. VCC
2. GND
3. IRQ
4. MSO (Master In slave Out).
5. MOSI (Master out slave In).
6. RST
7. SCK (Serial clock).
8. SDA (Slave Data Address).

The components required for this project are:

- RFID reader
- RFID tag
- Arduino
- Piezo buzzer
- Breadboard
- Jumper wire

Date :

Setup:-

- 3.3 v is given to 3.3v pin of Arduino.
- RST is given to the digital pin 9.
- GND is given to ground pin of arduino.
- IRQ is not connected.
- MISO is connected to digital pin 12.
- MOSI is connected to digital pin 11.

* Program:

```
# include <SPI.h>
# include <RFID.h>

byte cared[5] = {107, 135, 241, 197, 216};

void setup(){
 serial.begin(9600);
 SPI.begin();
 rfid.init();
 pinMode(buzzer, OUTPUT);
 Serial.println("RFID System ready.");
}
 Serial.println();
 boolean match = true;
 for (int i=0; i<5; i++){
 if (rfid.serNum[i] != cared[i]){
   match = false;
   break;
  }
}
```

Date

```
if (match) {
  Serial.println("☑ Access granted: Authorized card.");
}
  else {
  Serial.println("☒ Access denied: Unauthorized card.");
  digitalWrite(buzzer, HIGH);
  delay(300);
  digitalWrite(buzzer, LOW);
  delay(50);

  for (int i=0; i<2; i++) {
   digitalWrite(buzzer, HIGH);
   delay(50);
   digitalWrite(buzzer, LOW);
   delay(50);
  }
 }
   delay(2000)
   rfid.halt();
 }
}
```

* Output:-

When authorized: ☑ Access granted:
Authorized card.

When unauthorized: ☒ Access denied:
Unauthorized card.

* Result:-

Hence, the RFID connected & interfaced with Arduino UNO successfully.

Date :

Practical No. 9

* Aim :- To connect and interface LDR module with Arduino UNO and produce output.

* Theory :-

The LDR module (Light Dependent Resistor module) is a widely used sensor for detecting & measuring light intensity. This module is commonly employed in projects for light detection, & security applications.

The LDR module typically has 4 pins :

1. VCC.
2. GND.
3. DO (Digital output).
4. AO (Analog output).

The working principle of a Light Dependent Resistor sensor is based on change in its resistance in response to varying light intensities. An LDR primarily consist of a photoresistor, a component whose resistance varies with amount of light it is exposed.

* Setup :-

Being by connecting the LDR sensor's VCC pin to the 5V output on Arduino for power.

Date :

Then, connect GND pin of LDR to one of the GND pins on the Arduino. The finally connect the Analog output pin (A0) to A0 of Arduino.

for indication, we are using LED connected to digital pin D2 of Arduino via 220-Ohm resistor.

* Program :-

```
const int ledPin = 2;
const int IdrPin = A0;

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

void loop()
{
int IdrValue = analogRead (IdrPin);
Serial.println (IdrValue);

if(IdrValue > 600)
{
 digitalWrite(ledPin, HIGH);

}
```

Date :

```
else
{
 digitalWrite(LedPin, LOW);
}
 delay(500);
}
```

* Output :

Bighter Room: Serial Monitor:

234
210
256
278
....
LED: OFF

Darker Room: Serial Monitor:

710
604
723
....
LED: ON

* Result :

Hence, the RFID connected & interfaced with Arduino UNO Successfully.

Date :

Practical No. 10

* Aim :- To connect and interface 16×2 LCD Display with Arduino UNO and produce output.

* Theory :-

The 16×2 LCD Display module is a widely used component for presenting text & symbo in electronic projects. This module is commonly utilized in embedded systems, microcontroller applications, & IOT setups requiring textual information display.

The 16×2 LCD Display module typically has 16 pins.

1. VSS.
2. VDD.
3. VO.
4. RS (Register select).
5. RW (Read / Write).
6. E (Enable).
7. A (Anode).
8. K (cathode).

* Setup :-

Components list :

1. Arduino (any model like UNO, Nano, Meg

Date :

2. 16×2 LCD Display
3. 10k Potentiometer
4. Breadboard.
5. Jumper Wires
6. 220Ω Resistor

* Control Pins :

- Connect LCD RS to Arduino pin 13.
- Connect LCD RW to Arduino pin 12.
- Connect LCD E to Arduino pin 11.

* Data pin (8-bit mode):

- Connect LCD D0 to Arduino Pin 10
- Connect LCD D1 to Arduino pin 9
- Connect LCD D2 to Arduino Pin 8
- Connect LCD D3 to Arduino pin 7
- Connect LCD D4 to Arduino pin 6
- Connect LCD D5 to Arduino pin 5
- Connect LCD D6 to Arduino pin 4
- Connect LCD D7 to Arduino pin 3.

* Programs.

```
#include <LiquidCrystal.h>

LiquidCrystal Lcd(13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3);

unsigned char Character1[8] = {0x04, 0x1F, 0x11, 0x11, 0x1F,
    0x1F, 0x1F, 0x1F};
```

Date :

```
unsigned char character2[8] = {0x01, 0x03, 0x07,
        0x1F, 0x1F, 0x07, 0x03, 0x01};

void setup(){
 lcd.begin(16,2);
 lcd.clear();
 lcd.createChar(0, character1);
 lcd.createChar(1, character2);
}
void loop(){
lcd.setCursor(0,0);
lcd.print("Hello!!!");
lcd.setCursor(0,1);
lcd.write(byte(0));
lcd.write(1);
}
```

* Result:-

Hence, the 16 x 2 LCD Display connected & interfaced with Arduino UNO successfully.