ARDUINO UNO DEVELOPMENT KIT PROGRAM

Arduino is an open-source, board that has a Microchip ATmega328P microcontroller on it. This microcontroller has a set of Digital & Analog input and output pins. The operating voltage of the board is 5V. It has 14 digital I/O pins & 6 Analog input pins. The clock frequency of the microcontroller is 16 MHz.

List of Experiments

1. Blinking of an LED using Arduino Uno
2. Fading of an LED using Arduino Uno
3. Interfacing a water-level sensor with Arduino Uno
4. Interfacing an ultrasonic sensor with Arduino Uno
5. MQ-6 gas sensor interfacing with Arduino Uno
6. Interfacing a buzzer with Arduino Uno
7. LED chaser with Arduino Uno

Experiment 1: Blinking of an LED using Arduino Uno

Aim: Blinking an LED is an introductory Arduino project in which we control an LED using Arduino. LED blinking refers to the process of continuously turning an LED (Light Emitting Diode) and off in a repetitive pattern.

Components Required:

- 1 X LED
- Breadboard
- Arduino UNO
- Jumper wires
- Arduino IDE Software , Select Arduino Uno Board, COM port.

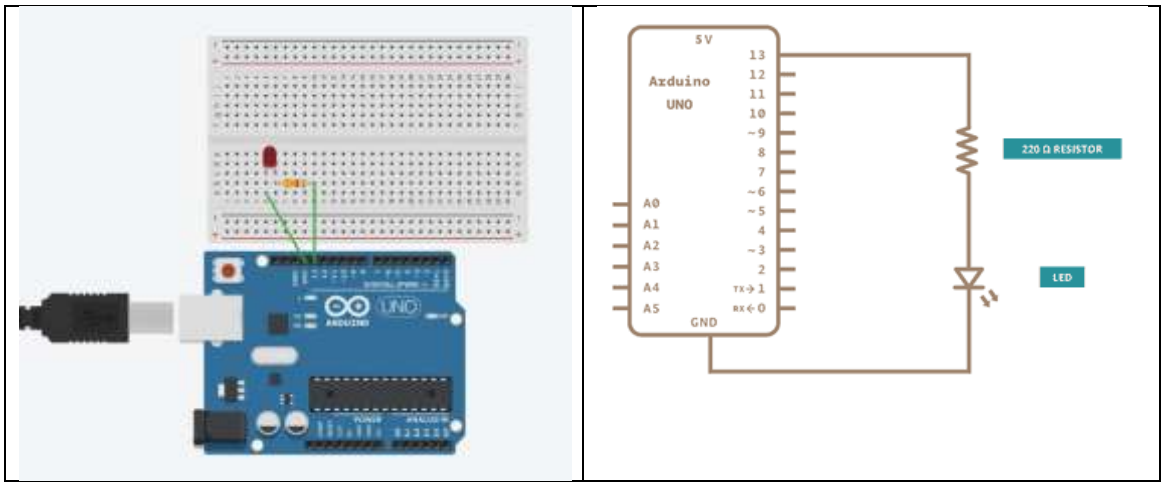
Working Procedure:

- setup() and loop() are two fundamental Arduino functions for controlling the behavior of Arduino, which form the foundation of any Arduino program.
- The setup() function is only called once when the Arduino board boots up or is reset. Its goal is to set pin modes, initialize variables, and execute any other necessary setup tasks before the main loop begins. This function can be used to configure settings that should only be changed once over the board's lifespan.
- The loop() function is the heart of an Arduino program. After the setup() function is executed, the loop() function starts running repeatedly until the Arduino is powered

off or reset. It contains the main code that performs the desired tasks, controls the board, user input.

- In the code, we have declared two integers, LEDpin and delayT. LEDpin represents the pin number of the Arduino where LEDs need to be connected, and delayT is an integer variable for the delay() function. The delay() function accepts values in milliseconds.

Circuit:



Code:

```
int LEDpin = 13;

int delayT = 1000;

void setup() {

  // put your setup code here, to run once:

pinMode(LEDpin, OUTPUT);

}

void loop() {

  // put your main code here, to run repeatedly:

digitalWrite(LEDpin, HIGH);

delay(delayT);

digitalWrite(LEDpin, LOW);

delay(delayT);

}
```

Result: Hence, the blinking of LED using Arduino Uno is executed and the output is verified successfully.

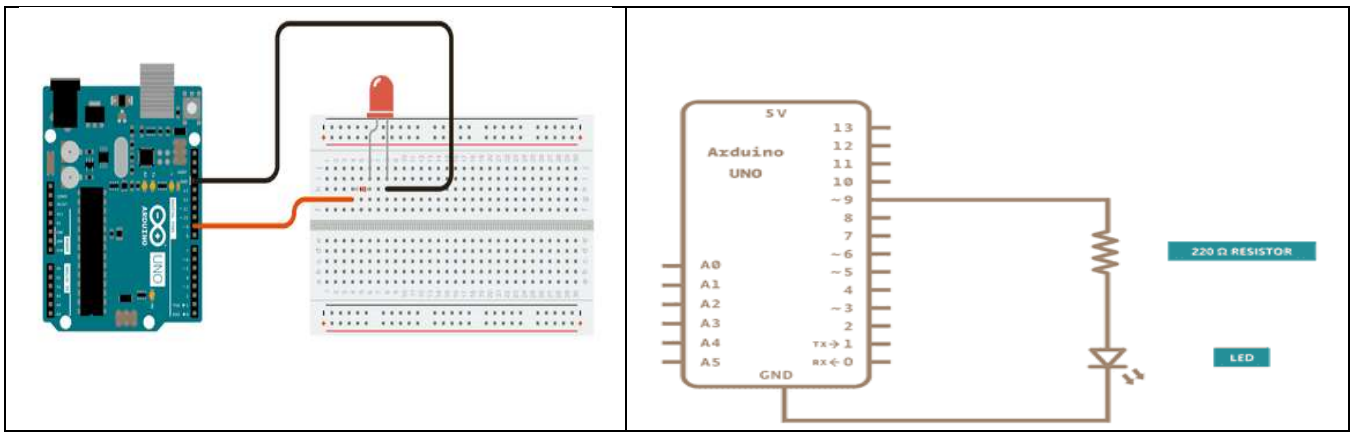
Experiment 2: Fading of an LED using Arduino Uno

Aim: This experiment demonstrates the use of the analogWrite() function in fading an LED off and on. AnalogWrite uses pulse width modulation (PWM), turning a digital pin on and off very quickly with different ratio between on and off, to create a fading effect.

Components Required:

- 1 X LED
- 220 ohm resistor
- Breadboard
- Arduino UNO
- Jumper wires
- Arduino IDE Software , Select Arduino Uno Board, COM port.

Curcuit :



Working Procedure:

- Connect the anode (the longer, positive leg) of your LED to digital output pin 9 on your board through a 220 ohm resistor.
- Connect the cathode (the shorter, negative leg) directly to ground.
- After declaring pin 9 to be your ledPin, there is nothing to do in the setup() function of your code.
- The analogWrite() function that you will be using in the main loop of your code requires two arguments: One telling the function which pin to write to, and one indicating what PWM value to write.
- In order to fade your LED off and on, gradually increase your PWM value from 0 (all the way off) to 255 (all the way on), and then back to 0 once again to complete the cycle. In the sketch below, the PWM value is set using a variable called brightness. Each time through the loop, it increases by the value of the variable fadeAmount.
- If brightness is at either extreme of its value (either 0 or 255), then fadeAmount is changed to its negative. In other words, if fadeAmount is 5, then it is set to -5. If it's -

5, then it's set to 5. The next time through the loop, this change causes brightness to change direction as well.

- analogWrite() can change the PWM value very fast, so the delay at the end of the sketch controls the speed of the fade. Try changing the value of the delay and see how it changes the fading effect.

Code:

```
int led = 9;         // the PWM pin the LED is attached to
int brightness = 0;  // how bright the LED is
int fadeAmount = 5;  // how many points to fade the LED by

// the setup routine runs once when you press reset:
void setup() {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

Result: Hence, the fading of LED using Arduino Uno is executed and the output is verified successfully.

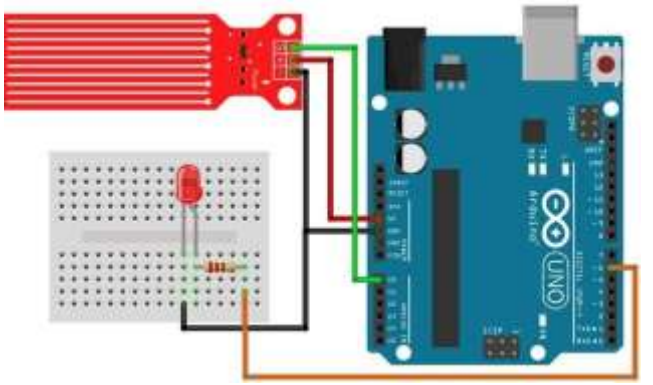
Experiment 3: Interfacing a water-level sensor with Arduino Uno

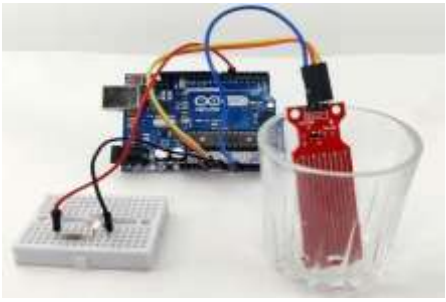
Aim: This experiment is to interface a water level sensor with Arduino to measure the water level.

Components Required:

- 1 X Water
- 220 ohm resistor
- Breadboard

- Arduino UNO
- Jumper wires
- Arduino IDE Software , Select Arduino Uno Board, COM port.





Code

```
// Sensor pins pin D6 LED output, pin A0 analog Input
#define ledPin 6
#define sensorPin A0
void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, LOW);
}
void loop() {
  int sensorValue = analogRead(sensorPin);
  if (sensorValue > 570)  {
    int outputValue = map(sensorValue, 570, 800, 0, 255);
    Serial.println(outputValue);
    analogWrite(ledPin, outputValue); // generate PWM signal
  }
}
```

Reference: https://circuitdigest.com/microcontroller-projects/interfacing-water-level-sensor-with-arduino

---

FOLLOW THE ABOVE PROCEDURES FOR THE FOLLOWING EXPERIMENTS ALSO

Experiment 4: Interfacing an ultrasonic sensor with Arduino Uno

Reference: https://www.instructables.com/Interfacing-Ultrasonic-Sensor-With-Arduino-1/

https://lastminuteengineers.com/arduino-sr04-ultrasonic-sensor-tutorial/

Experiment 5: Interfacing an ultrasonic sensor with Arduino Uno

Reference: https://techatronic.com/lpg-gas-sensor-interface-with-arduino-mq6/

https://www.watelectronics.com/mq6-gas-sensor/

Experiment 6: Interfacing a buzzer with Arduino Uno

Reference: https://projecthub.arduino.cc/SURYATEJA/use-a-buzzer-module-piezo-speaker-using-arduino-uno-cf4191

Experiment 7: Interfacing LED chaser with Arduino Uno

Reference: https://projecthub.arduino.cc/Sparkbuzzer/multiple-rgb-led-chaser-using-arduino-uno-1ff98c

---