East West University

Department of Computer Science & Engineering

Lab Report

CSE406

Internet of Things

Submitted By

Md Ashik

2020-3-60-032

**Water Level Detection Using Arduino**

**Water Level Detection Using Arduino**

**Lab Report**

**How It Works**

This system functions as an electronic water level indicator. A water sensor is connected to an analog input pin on the Arduino to measure the amount of water it is submerged in. The sensor's resistance changes based on its submersion, which results in a varying analog voltage reading.

The Arduino code continuously reads this analog value and, based on predefined thresholds, determines if the water level is "Low," "Medium," or "High." It then activates one of three LEDs (Green, Yellow, or Red) to provide a clear visual indication of the current water level.

**Hardware Components and Pin Connections**

|  |  |  |
| --- | --- | --- |
| Component | Arduino Pin | Connection Details |
| Water Level Sensor | A0 | Sensor signal output to analog pin |
| Red LED | 2 | Anode to pin 2 (via resistor) |
| Yellow LED | 3 | Anode to pin 3 (via resistor) |
| Green LED | 4 | Anode to pin 4 (via resistor) |
| All LEDs | GND | Cathode to ground |
| Sensor | 5V, GND | Power and ground as required |

**Pin Connection Diagram:**

Water Level Sensor  
 Signal ────── A0 (Arduino)  
 VCC ────── 5V  
 GND ────── GND  
  
Red LED Anode ─ Pin 2 Cathode ─ GND (via resistor)  
Yellow LED Anode ─ Pin 3 Cathode ─ GND (via resistor)  
Green LED Anode ─ Pin 4 Cathode ─ GND (via resistor)

**Arduino Code Analysis**

* const int sensorPin = A0;  
  const int redLED = 2;  
  const int yellowLED = 3;  
  const int greenLED = 4;  
    
  void setup() {  
   pinMode(redLED, OUTPUT);  
   pinMode(yellowLED, OUTPUT);  
   pinMode(greenLED, OUTPUT);  
   Serial.begin(9600);  
  }  
    
  void loop() {  
   int sensorValue = analogRead(sensorPin);  
   Serial.println(sensorValue); // For calibration  
    
   // Adjust these thresholds based on your sensor's readings  
   if (sensorValue < 300) {  
   // Low water  
   digitalWrite(redLED, LOW);  
   digitalWrite(yellowLED, LOW);  
   digitalWrite(greenLED, HIGH);  
   } else if (sensorValue < 600) {  
   // Medium water  
   digitalWrite(redLED, LOW);  
   digitalWrite(yellowLED, HIGH);  
   digitalWrite(greenLED, LOW);  
   } else {  
   // High water  
   digitalWrite(redLED, HIGH);  
   digitalWrite(yellowLED, LOW);  
   digitalWrite(greenLED, LOW);  
   }  
    
   delay(500);  
  }

Analysis

* **Global Variables:**
* sensorPin: Defines A0 as the analog input pin for the water sensor.
* redLED, yellowLED, greenLED: Define digital pins 2, 3, and 4 as outputs for the respective LEDs.
* **setup() function:**
* This function runs once at the start.
* pinMode(): Configures the LED pins as OUTPUT.
* Serial.begin(9600): Initializes serial communication at a 9600 baud rate to allow the Arduino to send data to the Serial Monitor for debugging and calibration.
* **loop() function:**
* This function runs repeatedly.
* analogRead(sensorPin): Reads the voltage from the water sensor and converts it into a digital value between 0 and 1023. This value is stored in sensorValue.
* Serial.println(sensorValue): Prints the current sensor value to the Serial Monitor.
* **Conditional Logic (if-else if-else):**
* If sensorValue is less than 300 (low water), the **Green LED** turns on.
* If sensorValue is between 300 and 599 (medium water), the **Yellow LED** turns on.
* If sensorValue is 600 or greater (high water), the **Red LED** turns on.
* delay(500): Pauses the loop for 500 milliseconds to prevent the readings from fluctuating too rapidly.

**Key Features:**

* **Initialization:** Sets up three LEDs as outputs and starts serial communication.
* **Sensor Reading:** Continuously reads analog value from the sensor.
* **Threshold Logic:**
  + *Below 300*: Green LED ON (Low water)
  + *300–599*: Yellow LED ON (Medium water)
  + *600 and above*: Red LED ON (High water)
* **Serial Output:** Prints sensor value for real-time monitoring and calibration.
* **Delay:** 500 ms delay for readability and stability.

**How It Works**

1. **Sensor Measurement:**  
   The analog water level sensor outputs a voltage proportional to the water level, which is read by Arduino’s analog pin (A0).
2. **Decision Logic:**  
   The code compares the sensor value to two thresholds (300 and 600), determining the current water level category.
3. **LED Indication:**
   * **Low Water:** Only the green LED lights up, signaling the tank or vessel is nearly empty.
   * **Medium Water:** Only the yellow LED lights up, indicating a moderate water level.
   * **High Water:** Only the red LED lights up, warning that the water level is high or the tank is full.
4. **Serial Monitoring:**  
   The sensor value is printed to the Serial Monitor, allowing for calibration and verification of system performance.

**Output Analysis**

The attached Serial Monitor output demonstrates the system’s response to varying water levels:

0  
0  
...  
50  
318  
393  
391  
405  
425  
428  
435  
590  
610  
619  
661  
685  
693  
684  
696  
668

* **0–299:** Indicates the sensor is dry or at a very low water level (green LED ON).
* **300–599:** Sensor detects a medium water level (yellow LED ON).
* **600 or above:** Sensor detects a high water level (red LED ON).

**Transitions** in the readings show the sensor’s ability to detect gradual changes in water level. The system is responsive and the thresholds are effective for distinguishing between low, medium, and high water levels.

**Output Screenshot**

Red Led turned On for High Water level (600 or above)

A person holding a circuit board

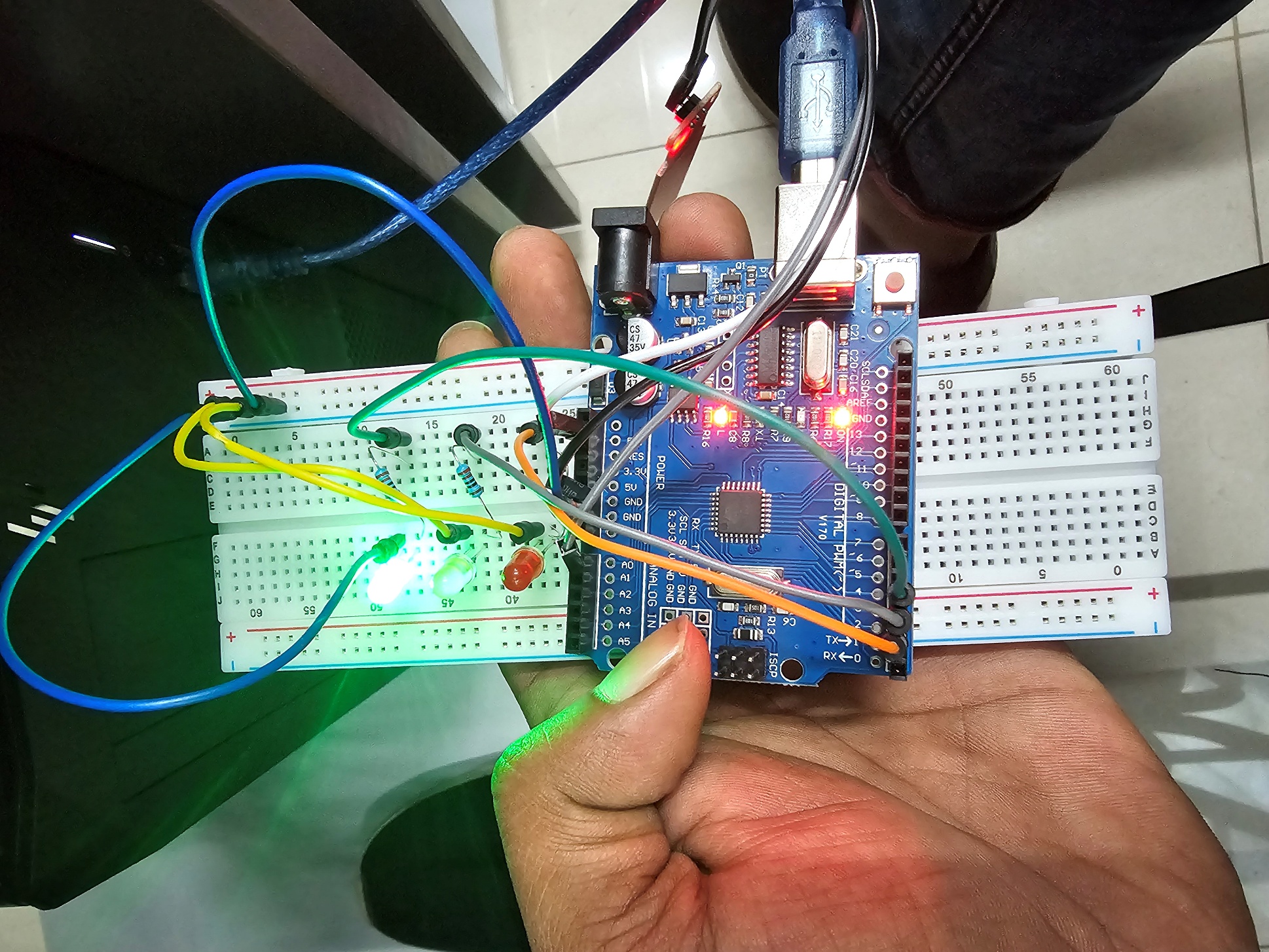
AI-generated content may be incorrect.

Yellow Led turned On for mediam Water level (300-599)

A hand holding a circuit board with wires

AI-generated content may be incorrect.

Green Led turned On for low (0-299) Water level



**Conclusion**

This Arduino-based water level detection system accurately monitors and displays water levels using simple analog sensing and threshold logic. The LED indicators provide clear, immediate feedback, and the serial output enables easy calibration and troubleshooting. The system is robust and suitable for basic water level monitoring applications.