NIRMA UNIVERSITY

INSTITUTE OF TECHNOLOGY

NAAC ACCREDITED 'A+' GRADE

ASSIGNMENT REPORT

MICROCONTROLLER AND INTERFACING (2EC701)

TOPIC – WATER FLOW MEASUREMENT

COMPILED BY:-

HARSHVARDHAN SINGH (22BEC120)

URMIT KIKANI (22BEC137)

Water Flow Measurement

Harshvardhan Singh, Urmit Kikani
Department of Electronics & Communication, Institute of Technology
Nirma University, Ahmedabad, Gujarat-382481, India
22bec120@nirmauni.ac.in, 22bec137@nirmauni.ac.in

Introduction:-

If you have ever visited large scale manufacturing companies, the first thing you will notice is that they are all automated. Soft Drink Industries and Chemical industries have to constantly measure and quantify the liquids that they are handling during this automation process, and the most common sensor used to measure the flow of a liquid is a **Flow Sensor**. By using a flow sensor with a microcontroller like Arduino, we can calculate the flow rate, and check the volume of liquid that has passed through a pipe, and control it as required. Apart from manufacturing industries, flow sensors can also be found in the agriculture sector, food processing, water management, mining industry, water recycling, coffee machines, etc.

In this project, we are going to build a **water flow sensor** using Arduino. We will interface the water flow sensor with Arduino and LCD, and program it to display the volume of water, which has passed through the valve. For this particular project, we are going to use the **YF-S201 water flow sensor**, which uses a hall effect to sense the flow rate of the liquid.

YFS201 Water Flow Sensor:-

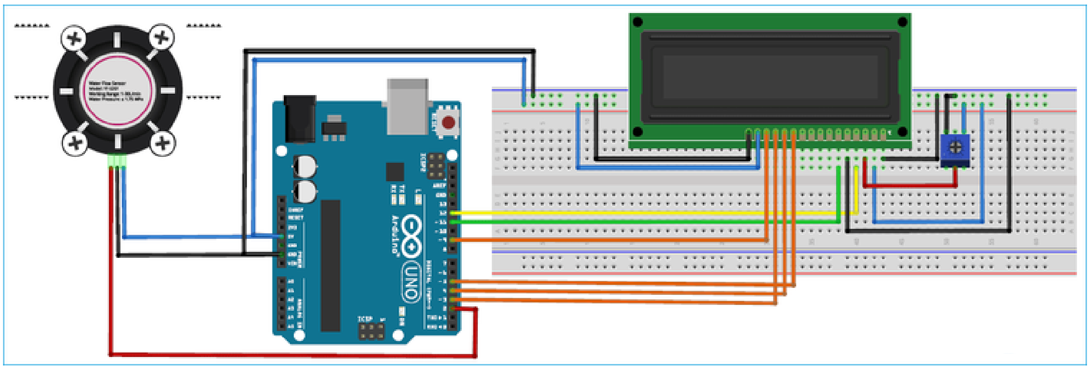
The **working of the YFS201 water flow sensor** is simple to understand. The water flow sensor works on the principle of hall effect. Hall effect is the production of the potential difference across an electric conductor when a magnetic field is applied in the direction perpendicular to that of the flow of current. The water flow sensor is integrated with a magnetic hall effect sensor, which generates an electric pulse with every revolution.

Its design is in such a way that the hall effect sensor is sealed off from the water, and allows the sensor to stay safe and dry. The sensor has 3 wires RED, YELLOW, and BLACK. The red wire is used for supply voltage which ranges from 5V to 18V and the black wire is connected to GND. The yellow wire is used for output(pulses), which can be read by an MCU. The water flow sensor consists of a pinwheel sensor that measures the quantity of liquid that has passed through it.

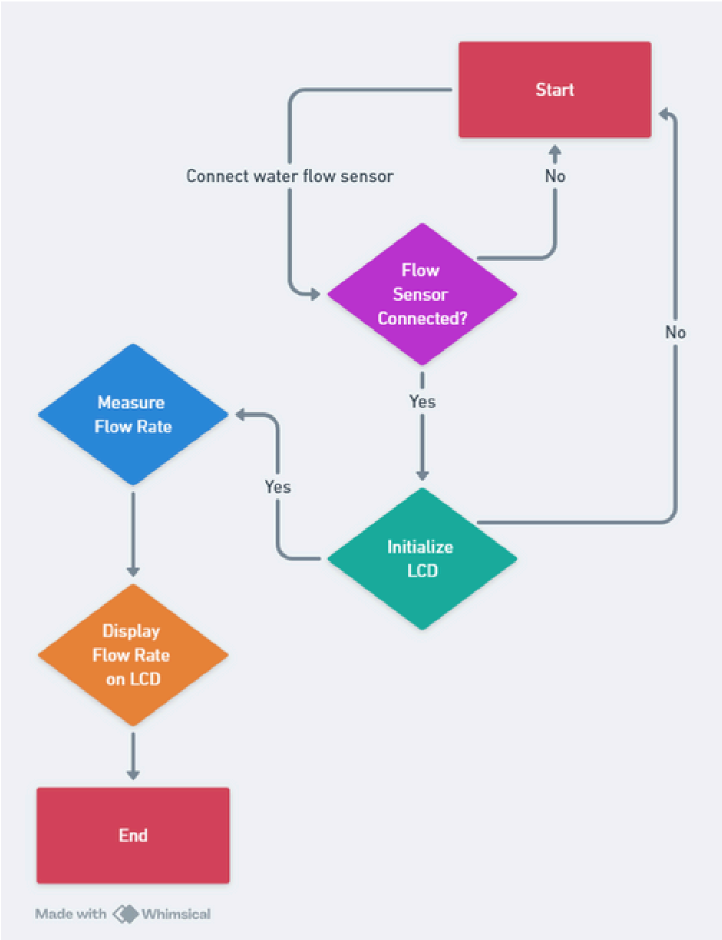
Working:-

The working principle involves the water flow sensor generating pulses proportional to the flow rate of water passing through it. These pulses are then processed by the Arduino Uno, which calculates the flow rate based on the pulse frequency. The calculated flow rate is then displayed on the 16x2 LCD screen for user interface. When the liquid flows through the sensor, it makes contact with the fins of the turbine wheel, which is placed in the path of the flowing liquid. The shaft of the turbine wheel is connected to a hall effect sensor. Due to this, whenever water flows through the valve it generates pulses. Now, all we have to do is to measure the time for the pluses or to count the number of pulses in 1 second and then calculate the flow rates in liter per hour (L/Hr) and then use simple conversion formula to find the volume of the water which had passed through it. To measure the pulses, we are going to use Arduino UNO.

Circuit Diagram:-



Flow Chart:-



Arduino Code:-

```
volatile int flow_frequency; // Measures flow sensor pulses
// Calculated litres/hour
float vol = 0.0,l_minute;
unsigned char flowsensor = 2; // Sensor Input
unsigned long currentTime;
unsigned long cloopTime;
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 9);
void flow () // Interrupt function
{
   flow_frequency++;
}
```

Define Parameters and Input port

```
void setup()
{
  pinMode(flowsensor, INPUT);
  digitalWrite(flowsensor, HIGH); // Optional Internal Pull-Up
  Serial.begin(9600);
  lcd.begin(16, 2);
  attachInterrupt(digitalPinToInterrupt(flowsensor), flow, RISING); // Setup Interrupt
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Water Flow Meter");
  lcd.setCursor(0,1);
  lcd.print("MCI PROJECT :");
  currentTime = millis();
  cloopTime = currentTime;
}
```

Initialization of LCD

```
void loop ()
{
  currentTime = millis();
  // Every second, calculate and print litres/hour
  if(currentTime >= (cloopTime + 1000))
  {
   cloopTime = currentTime; // Updates cloopTime
```

Taking the readings from the sensor

```
  if(flow_frequency != 0){
    // Pulse frequency (Hz) = 7.5Q, Q is flow rate in L/min.
    l_minute = (flow_frequency / 7.5); // (Pulse frequency x 60 min) / 7.5Q = flowrate in L/hour
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Rate: ");
    lcd.print(l_minute);
    lcd.print(" L/M");
    l_minute = l_minute/60;
    lcd.setCursor(0,1);
    vol = vol +l_minute;
    lcd.print("Vol:");
    lcd.print(vol);
    lcd.print(" L");
    flow_frequency = 0; // Reset Counter
    Serial.print(l_minute, DEC); // Print litres/hour
    Serial.println(" L/Sec");
  }
  else {
    Serial.println(" Water flow rate = 0 ");
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Rate: ");
    lcd.print( flow_frequency );
    lcd.print(" L/M");
    lcd.setCursor(0,1);
    lcd.print("Vol:");
    lcd.print(vol);
    lcd.print(" L");
  }
 }
}
```

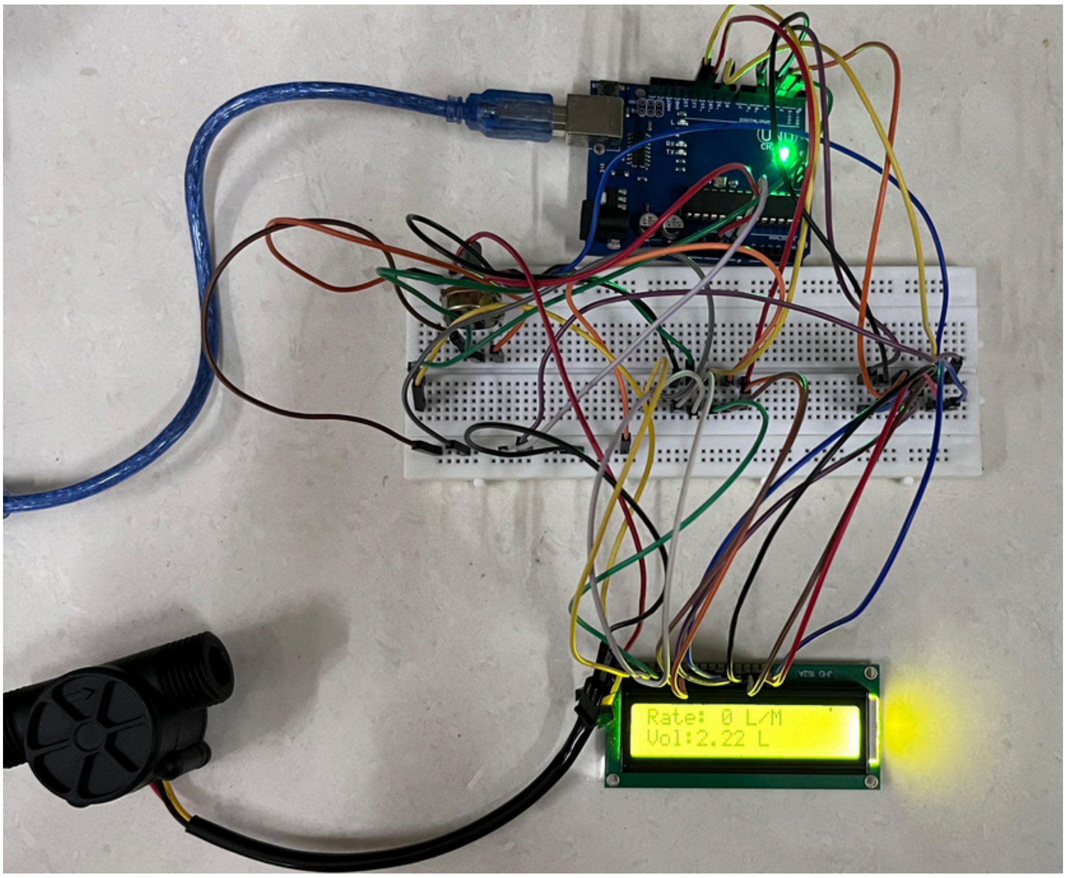
Display the Output on LCD

Assembly Language Program

```
ORG 0000H

// INITIALIZE TIMER 0 AS COUNTER AND TIMER 1 AS TIMER (1 SECOND DELAY)
RPT: MOV TMOD, #15H      // Set Timer 0/1 to 16-bit mode (counter/timer)
     SETB P3.4           //Set P3.4 as input pin (external pulses)
     MOV TL0, #00
     MOV TH0, #00
     SETB TR0
     MOV R0, #280         // Set delay count for 1 second (assuming crystal frequency of 12 MHz)

AGAIN: MOV TL1, #00H
       MOV TH1, #00H
       SETB TR1
BACK:  JNB TF1, BACK
       CLR TF1
       CLR TR1

       DJNZ R0, AGAIN


// CONVERT PULSE COUNT (IN T0) TO ASCII

       MOV A, TL0
       ADD A, #'0'          // Add offset to convert to ASCII for digits 0-9
       JC OVERFLOW          // If overflow (count > 99), handle separately
       MOV P2, A            // Move ASCII digit to P2 port
       SJMP NEXT_DIGIT      // Jump to display high byte (if applicable)

OVERFLOW: MOV A, #'0' + 10 // Set ASCII for '1' (hundreds digit)
       MOV P2, A            // Move ASCII digit to P2 port

NEXT_DIGIT: MOV A, TH0
       ADD A, #'0'
       MOV P1, A


// INTERFACE THE LCD WITH THE AT89C51
    MOV A,#38H
    ACALL COMAND
    MOV A,#0FH
    ACALL COMAND
    MOV A,#01H
    ACALL COMAND
    MOV A,#80H
    ACALL COMAND
    MOV R3,#0
    MOV DPTR,#STRN

    LP:CLR A
    MOVC A,@A+DPTR
    ACALL DATAA
    INC DPTR
    LOOP:JNZ LP
    SJMP LL
    LL:MOV A,#10H
    ACALL COMAND
     MOV A,#0CH
    ACALL COMAND
    ACALL SCROLL
    HER :SJMP HER
```

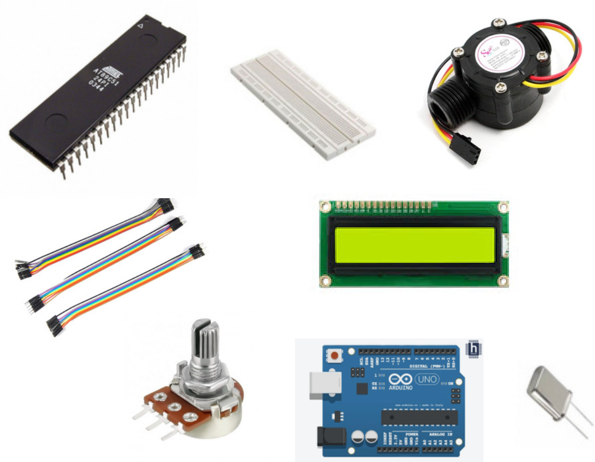
```
SCROLL :
  CLR A
  MOV R6,A
  MOV R6,#16
  BANE:MOV A,#1CH//
  ACALL COMAND
  DJNZ R6,BANE
  RET

    COMAND:
    ACALL READY
    MOV P1,A
    CLR P2.0// COMAND ON
    CLR P2.1 // WRITE DATA
    SETB P2.2
    ACALL DELAY //LATCH DATA
    CLR P2.2
    RET
DATAA:  ACALL READY
        MOV P1,A
    ACALL DELAY //LATCH DATA
    CLR P2.2
    RET
DATAA:  ACALL READY
        MOV P1,A
        SETB P2.0// DATA ON
        CLR P2.1//WRITE DATA
        SETB P2.2
        ACALL DELAY //LATCH
        CLR P2.2
        RET
DELAY:
        MOV R3,#20
        HERA:MOV R4,#252
        HERE:DJNZ R4,HERE
        DJNZ R3,HERA
        RET
READY:
        SETB P1.7
        CLR P2.0 // SELECTS COMAND REGESTER
        SETB P2.1 // FOR READING
        BACK1 : CLR P2.2
        ACALL DELAY
        SETB P2.2
        JB P1.7,BACK1
        RET

ORG 0600H
    STRN: DB "FLOW MEASUREMENT",0

END
```

Output:-



List of Materials:-

1. Arduino Uno
2. AT89S51 (optional)
3. Crystal Oscillator (optional)
4. IC Bed (optional)
5. Water flow sensor
6. 16x2 LCD display
7. Jumper wires
8. Breadboard
9. Potentiometer



.

Applications:-

1. **Industrial Processes**: Enables efficient water usage and machinery operation in manufacturing, cooling, and chemical processes.
2. **Environmental Monitoring**: Facilitates the study of water dynamics in natural environments like rivers and helps manage water resources sustainably. Assists in optimising irrigation, identifying leaks, and conserving water in agricultural practices..
3. **Home Automation**: Contributes to smart home setups by monitoring water usage, detecting leaks, and promoting water conservation.
4. **Aquaculture and Hydroponics**: Ensures proper water flow for the health and growth of aquatic organisms or plants in controlled environments.
5. **Water Distribution Networks**: Aids municipalities and utilities in managing water supply networks, detecting leaks, and optimizing water distribution.

Summary:-

The water flow measurement system using Arduino provides a simple yet effective solution for real-time flow rate monitoring. By integrating a water flow sensor and an LCD display with Arduino, accurate measurements can be obtained and displayed for different applications.

Set of Questions:-

1. How does the water flow sensor generate pulses?
2. Can the Arduino Uno handle multiple sensors for simultaneous flow measurement?
3. What factors can affect the accuracy of flow rate measurement in this system?
4. How can this system be expanded for data logging and analysis?
5. What are the advantages of using Arduino for water flow measurement compared to traditional methods?