#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

int solenoidValve = 12;

int solenoidValve1 = 11;// relay at pin 12

byte sensorInterrupt = 0; // 0 = digital pin 2

byte sensorPin = 2;

byte sensorInterrupt1 = 1; // 1 = digital pin 3

byte sensorPin1 = 3;

byte sensorInterrupt2 = 0; //

byte sensorPin2 = 4;

byte sensorInterrupt3 = 1; //

byte sensorPin3 = 5;

float calibrationFactor = 4.5;

volatile byte pulseCount;

volatile byte pulseCount1;

volatile byte pulseCount2;

volatile byte pulseCount3;

float flowRate;

unsigned int flowMilliLitres;

unsigned long totalMilliLitres;

float flowRate1;

unsigned int flowMilliLitres1;

unsigned long totalMilliLitres1;

float flowRate2;

unsigned int flowMilliLitres2;

unsigned long totalMilliLitres2;

float flowRate3;

unsigned int flowMilliLitres3;

unsigned long totalMilliLitres3;

float difference;

float difference1;

unsigned long oldTime;

unsigned long oldTime1;

unsigned long oldTime2;

unsigned long oldTime3;

LiquidCrystal\_I2C lcd(0x27,20,4);

void setup()

{

lcd.init(); // initialize the lcd

lcd.init();

// Print a message to the LCD.

lcd.backlight();

Serial.begin(9600);

pinMode(sensorPin, INPUT);

digitalWrite(sensorPin, HIGH);

pinMode(sensorPin1, INPUT);

digitalWrite(sensorPin1, HIGH);

pinMode(sensorPin2, INPUT);

digitalWrite(sensorPin2, HIGH);

pinMode(sensorPin3, INPUT);

digitalWrite(sensorPin3, HIGH);

pinMode(solenoidValve , OUTPUT); // setting pinmode for relay

digitalWrite(solenoidValve ,LOW);

pinMode(solenoidValve1 , OUTPUT); // setting pinmode for relay

digitalWrite(solenoidValve1 ,LOW);

pulseCount = 0;

flowRate = 0.0;

flowMilliLitres = 0;

oldTime = 0;

pulseCount1 = 0;

flowRate1 = 0.0;

flowMilliLitres1 = 0;

oldTime1 = 0;

pulseCount2 = 0;

flowRate2 = 0.0;

flowMilliLitres2 = 0;

oldTime2 = 0;

pulseCount3 = 0;

flowRate3 = 0.0;

flowMilliLitres3 = 0;

oldTime3 = 0;

difference = 0;

difference1 = 0;

lcd.begin(16, 2);

lcd.clear();

lcd.setCursor(1,0);

lcd.print("Water FlowMeter");

lcd.setCursor(1,1);

lcd.print(" sensing... ");

delay(500);

attachInterrupt(sensorInterrupt, pulseCounter, FALLING);

attachInterrupt(sensorInterrupt1, pulseCounter1, FALLING);

attachInterrupt(sensorInterrupt2, pulseCounter2, FALLING);

attachInterrupt(sensorInterrupt3, pulseCounter3, FALLING);

}

void loop()

{

delay(100);

if((millis() - oldTime) > 1000) // Only process counters once per second

{

detachInterrupt(sensorInterrupt);

detachInterrupt(sensorInterrupt1);

detachInterrupt(sensorInterrupt2);

detachInterrupt(sensorInterrupt3);

flowRate = ((1000.0 / (millis() - oldTime)) \* pulseCount) / calibrationFactor;

flowRate1 = ((1000.0 / (millis() - oldTime)) \* pulseCount1) / calibrationFactor;

flowRate2 = ((1000.0 / (millis() - oldTime)) \* pulseCount2) / calibrationFactor;

flowRate3 = ((1000.0 / (millis() - oldTime)) \* pulseCount3) / calibrationFactor;

difference = flowRate - flowRate1;

difference1= flowRate2 - flowRate3;

oldTime = millis();

flowMilliLitres = (flowRate / 60) \* 1000;

totalMilliLitres += flowMilliLitres;

unsigned int frac;

lcd.clear();

lcd.setCursor(1,0);

lcd.print("Flowrate1:");

lcd.print(int(flowRate)); // Print the integer part of the variable

lcd.print("."); // Print the decimal point

// Determine the fractional part. The 10 multiplier gives us 1 decimal place.

frac = (flowRate - int(flowRate)) \* 10;

lcd.print(frac, DEC) ; // Print the fractional part of the variable

lcd.print("L/min");

lcd.print("\t");

lcd.setCursor(1,1);

lcd.print("Flowrate2:");

lcd.print(int(flowRate1));

lcd.print(".");

frac = (flowRate1 - int(flowRate1)) \* 10;

lcd.print(frac, DEC) ; // Print the fractional part of the variable

lcd.print("L/min");

delay(2000);

lcd.clear();

lcd.setCursor(1,0);

lcd.print("Flowrate3:");

lcd.print(int(flowRate2)); // Print the integer part of the variable

lcd.print("."); // Print the decimal point

// Determine the fractional part. The 10 multiplier gives us 1 decimal place.

frac = (flowRate2 - int(flowRate2)) \* 10;

lcd.print(frac, DEC) ; // Print the fractional part of the variable

lcd.print("L/min");

lcd.print("\t");

lcd.setCursor(1,1);

lcd.print("Flowrate4:");

lcd.print(int(flowRate3));

lcd.print(".");

frac = (flowRate3 - int(flowRate3)) \* 10;

lcd.print(frac, DEC) ; // Print the fractional part of the variable

lcd.print("L/min");

delay(2000);

if(difference==0)

{

SetSolinoidValveOFF();

lcd.clear();

lcd.setCursor(1,0);

lcd.print("No Leakage detected");

lcd.setCursor(1,1);

lcd.print("in pipe 1");

}

delay(2000);

if(difference1==0)

{

SetSolinoidValveOFF();

lcd.clear();

lcd.setCursor(1,0);

lcd.print("No Leakage detected in pipe 2");

lcd.setCursor(1,1);

lcd.print("in pipe 2");

}

delay(2000);

if(difference1>0)

{

SetSolinoidValveOFF();

lcd.clear();

lcd.setCursor(1,0);

lcd.print("No Leakage detected");

lcd.setCursor(1,1);

lcd.print("in pipe 1");

}

delay(2000);

if(difference1>0)

{

SetSolinoidValveOFF();

lcd.clear();

lcd.setCursor(1,0);

lcd.print("No Leakage detected

");

lcd.setCursor(1,1);

lcd.print("in pipe 2");

}

delay(2000);

// Reset the pulse counter so we can start incrementing again

pulseCount = 0;

pulseCount1 = 0;

pulseCount2 = 0;

pulseCount3 = 0;

// Enable the interrupt again now that we've finished sending output

attachInterrupt(sensorInterrupt, pulseCounter, FALLING);

attachInterrupt(sensorInterrupt1, pulseCounter1, FALLING);

attachInterrupt(sensorInterrupt2, pulseCounter2, FALLING);

attachInterrupt(sensorInterrupt3, pulseCounter3, FALLING);

}

}

/\*

Insterrupt Service Routine

\*/

void pulseCounter()

{

// Increment the pulse counter

pulseCount++;

}

void pulseCounter1()

{

// Increment the pulse counter

pulseCount1++;

}

void pulseCounter2()

{

// Increment the pulse counter

pulseCount2++;

}

void pulseCounter3()

{

// Increment the pulse counter

pulseCount3++;

}

/\* Function for solenoid valve open and close\*/

void SetSolinoidValveOFF()

{

digitalWrite(solenoidValve, HIGH);

}

void SetSolinoidValveOFF1()

{

digitalWrite(solenoidValve1, HIGH);

}