int level1=0,level2=0;

int rw=0,rc=0,rn=0,t=0,lm=0;

const int mv=6,sv=10,led=12,buz=11,wl1=A4,wl2=A3;

int fsd=0,bh=0,button=A2,br=0;

const int rwm=8,rwt=7;

int rwtl=A0;

int rwtd=0,irr=0;

long f=0;

int of=A5,ur=A1;

const int buz2=13;

int ofd=0,b2=0,svt=0,buzs=0,urd=0;

long int ofc=0,ftt=0;

const int rain=4,rv=9;

const int motor=5;

int sensorInterrupt1 = 1; // interrupt 1

int sensorInterrupt2 = 0;

const int sensorPin1 = 3; //Digital Pin 2

const int sensorPin2 = 2;

unsigned long SetPoint = 5000; //400 milileter

/\*The hall-effect flow sensor outputs pulses per second per litre/minute of flow.\*/

float calibrationFactor = 4.5; //You can change according to your datasheet

volatile byte pulseCount1 =0;

volatile byte pulseCount2 =0;

float flowRate1 = 0.0;

float flowRate2 = 0.0;

unsigned int flowMilliLitres1 =0;

unsigned long totalMilliLitres1 = 0;

unsigned long oldTime = 0;

unsigned int flowMilliLitres2 =0;

unsigned long totalMilliLitres2 = 0;

unsigned long oldTime1 = 0;

unsigned long oldTime3 = 0,raintime=0,urtime=0;

void setup()

{

Serial.begin(9600);

pinMode(motor,OUTPUT);

pinMode(rain,INPUT);

pinMode(rv,OUTPUT);

pinMode(mv,OUTPUT);

pinMode(sv,OUTPUT);

pinMode(led,OUTPUT);

pinMode(buz,OUTPUT);

pinMode(rwm,OUTPUT);

pinMode(rwt,OUTPUT);

pinMode(buz2,OUTPUT);

digitalWrite(sv,HIGH);

digitalWrite(mv,HIGH);

delay(2000);

Serial.begin(9600);

pinMode(sensorPin1, INPUT);

digitalWrite(sensorPin1, HIGH);

pinMode(sensorPin2, INPUT);

digitalWrite(sensorPin2, HIGH);

/\*The Hall-effect sensor is connected to pin 2 which uses interrupt 0. Configured to trigger on a FALLING state change (transition from HIGH (state to LOW state)\*/

attachInterrupt(sensorInterrupt1, pulseCounter1, FALLING); //you can use Rising or Falling

attachInterrupt(sensorInterrupt2, pulseCounter2, FALLING);

}

void loop()

{

level1=analogRead(wl1);

level2=analogRead(wl2);

rw=digitalRead(rain);

t=millis();

br=analogRead(button);

ofd=analogRead(of);

urd=analogRead(ur);

if(ofd<500)

{

ofc=ofc+1;

delay(5);

}

//tank level maintaining

if(level1>500)

{

digitalWrite(motor,HIGH);

}

if(level2<500)

{

digitalWrite(motor,LOW);

}

//Rain water harvesting

if((rw==HIGH)&&(rn==0))

{

rc=rc+1;

lm=0;

if(rc==3000)

{

delay(5000);

rn=1;

digitalWrite(rv,HIGH);

}

}

if((rw==LOW)&&(lm==0))

{

if(rc>0)

rc=rc-1;

if((rc<=0))

{

delay(5000);

digitalWrite(rv,LOW);

lm=1;

}

}

if((rw==HIGH)&&(rn==1))

{

rc=rc+1;

lm=0;

if(rc>=10000)

{

digitalWrite(rv,HIGH);

}

}

//Water usage and leakage

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

{

// Disable the interrupt while calculating flow rate and sending the value to the host // Because this loop may not complete in exactly 1 second intervals we calculate the number of milliseconds that have passed since the last execution and use that to scale the output. We also apply the calibrationFactor to scale the output based on the number of pulses per second per units of measure (litres/minute in this case) coming from the sensor.

flowRate1 = ((1000.0 / (millis() - oldTime)) \* pulseCount1) / calibrationFactor; // Note the time this processing pass was executed. Note that because we've

// disabled interrupts the millis() function won't actually be incrementing right

// at this point, but it will still return the value it was set to just before

// interrupts went away.

oldTime = millis(); // Divide the flow rate in litres/minute by 60 to determine how many litres have

// passed through the sensor in this 1 second interval, then multiply by 1000 to

// convert to millilitres.

flowMilliLitres1 = (flowRate1 / 60) \* 1000; // Add the millilitres passed in this second to the cumulative total

totalMilliLitres1 += flowMilliLitres1;

unsigned int frac; // Print the flow rate for this second inlitres / minute

Serial.print(totalMilliLitres1,DEC);

Serial.println("mL");

Serial.print("\t");

}

if((millis() - oldTime1) > 1000) // Only process counters once per second

{

flowRate2 = ((1000.0 / (millis() - oldTime1)) \* pulseCount2) / calibrationFactor; // Note the time this processing pass was executed. Note that because we've

// interrupts went away.

oldTime1 = millis(); // Divide the flow rate in litres/minute by 60 to determine how many litres have

// passed through the sensor in this 1 second interval, then multiply by 1000 to

// convert to millilitres.

flowMilliLitres2 = (flowRate2 / 60) \* 1000; // Add the millilitres passed in this second to the cumulative total

totalMilliLitres2 += flowMilliLitres2;

unsigned int frac;

}

Serial.print(totalMilliLitres2,DEC);

Serial.println("mL");

Serial.print("\t");

if ((flowRate1==0)&&(flowRate2>0)&&(svt==0))

{ digitalWrite(sv,LOW);

digitalWrite(led,HIGH);// Reset the pulse counter so we can start incrementing again

pulseCount2 = 0;

pulseCount1=0;//

svt=1;

}

if((flowRate2>0)&&(svt==1))

{

digitalWrite(sv,HIGH);

digitalWrite(mv,LOW);

digitalWrite(buz,HIGH);

}

br=analogRead(button);

if((br>100))

{

digitalWrite(sv,HIGH);

digitalWrite(led,LOW);

digitalWrite(mv,HIGH);

digitalWrite(buz,LOW);

svt=0;

totalMilliLitres1=totalMilliLitres2;

}

// Irrigation

t=millis();

if((rn==0)&&(irr==0))

{

raintime=millis();

rwtd=analogRead(rwtl);

if((rwtd<500)&&(irr==0))

{

digitalWrite(rwt,HIGH);

irr=1;

}

if((rwtd>500)&&(irr==0))

{

digitalWrite(rwm,HIGH);

irr=1;

}

}

if(((millis()-raintime)>5000)&&(irr=1));

{

digitalWrite(rwt,LOW);

digitalWrite(rwm,LOW);

raintime=0;

}

t=millis();

f=t/1000;

if(f>=300000)

{

irr=0;

}

if(f>=60000)

{

rn=0;

}

if((ofc>1000)&&(buzs==0)&&(b2==0))

{

digitalWrite(buz2,HIGH);

b2=1;

}

urd=analogRead(ur);

if((urd>500)&&(b2==1))

{

digitalWrite(buz2,LOW);

ofc=0;

buzs=1;

b2=0;

urtime=millis();

}

if((ofc>3000)&&(buzs==0)&&((millis()-urtime)>5000))

{

digitalWrite(sv,LOW);

ofc=0;

}

}

void pulseCounter1()

{

// Increment the pulse counter

pulseCount1++; }

void pulseCounter2()

{

// Increment the pulse counter

pulseCount2++; }