AN ARDUINO BASED SYSTEM FOR CONTROLLING AND MONITORING DHW CONSUMPTION

Control and Automation for the efficient use of Energy
2016/2017

João Cordeiro De Sousa
Alvaro Picatoste

Contenido

Objective

The objective of this project is to design an Arduino based application for controlling and monitoring the Domestic Hot Water consumption (DHW) at a domestic level. Using an Arduino UNO board and a flow meter, along with other devices, an application will be designed to measure the water consumption and take different actions based on it. The application will inform the users of the amount of water spent, will warn him or her whenever this consumption is over some limits, which will be possible to be modified by the own user.

Introduction

Water usage at a domestic level is an important sink of waste that can easily corrected with the proper measures. Most people do not know or do not care about the amount of resources that we consume daily. Despite the difficulty and challenging need of affecting the human behavior there is an effect in that awareness changes the common patterns of humans. That is the base for many current systems that they only task is to inform the user about something regarding their surroundings. One of this example is the spread of energy monitoring systems that measure and displays the energy consumption at a domestic level. If the user knows how much energy they are spending it is likely that they will, in greater or lower effect, their consumption.

Water is a today big issue worldwide. Climate change and less precipitation are making this good less available and more expensive, same as other resources produced generate energy. Therefore, it is important for today society to take care of its waste and applying the same concept above explained, letting users knows real-time about their consumption will have a positive effect, it will not be the ultimate solution that will save billions of liters but it will help pushing the changes towards a much more conscientious world. Going further from just monitoring in effecting people behavior, rewarding through gamifications has shown good results in other fields. Good behavior in different ways whilst bad behavior is penalized. Implementing personal good habits regarding water consumption should be rewarded to promote the smart water consumption at a domestic level. As a last option, prohibition is a last measure that might be (and that has been) implemented in some water scarce times, especially in Spain where there are long periods with no precipitation. So is that, that governments have stablished periods of water consumption and forbid its usage out of that hours. Another way of imposing limitations in water consumption is by regulating the amount of water each person can consume.

The project

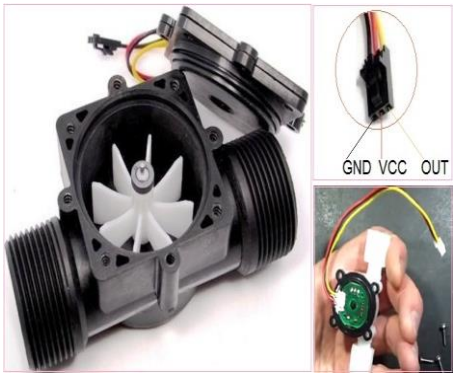
rEWAt is an application that aims to reduce water consumption and energy at a domestic level through monitoring, warning, rewarding regarding the water needed for showering. It is an Arduino based application that will be able to measure the water flow when the users are showering and will perform different task so the users is well aware of the amount of water at any moment, the flow and other information that will likely affect his behavior leading to a smart and better usage of this resource.

Hardware

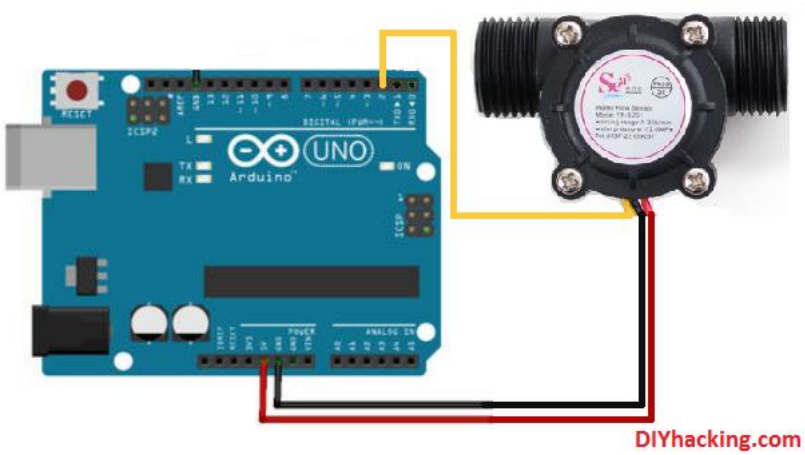
In this first part of the report we will introduce the tools that form the system from a hardware perspective. Most of the devices are included in the material provided but others have been acquired from other sources.

The main device: Flow meter

This application is all based on the flow meter YF-S201. It was first ordered online through a well-known website (www.dealextreme.com) but after some weeks it was decided to invest more and buy it on a physical store in Madrid. Once acquired, the first step was to connect it to Arduino in order to check its performance and evaluate if its error is under the manufacturer's range. They way the flow meter work is based in pulses of a rotameter in its interior, shown in the following picture:



As the water flows through the sensor, the wheel starts turning and the device send a pulse that is then converted into a flow measure. As it is stated in the device datasheet each pulse corresponds to a flow of approximately 2.25 m. To program this in the Arduino IDE an *Interrupt* function will be used as is the most appropriate way to keep track of the number of pulses, and therefore, the total amount of water that has gone though the device. More details about the coding will be explained later. The flow meter has three cables that must be connected according to the following picture:



Other Devices

Together with the flow meter, main actor of the whole system, there will be needed other devices to warn the user about the water usage, whether by visual or sounds ways.

LEDs: LEDs of different color will be used to visual warn the user of his water consumption.

Speaker: It will be installed in order to let the user knows by the ears that he might be using too much water. It is easy that visual means fails on its purpose of notifying the user.

Along with these devices some other common electronic components have been implemented in the circuit: resistances, button, cables, etc. The final circuit will be shown in the later section “Results”.

Software

Once the components have been assembled the software that controls it is explained in this section. The first step in the project was the design on paper of the main features that needed to be implemented. As every software project the plan was designed in small steps to check that the features worked per what as planned.

Step 1: Measuring the flow.

The most crucial part of the project is measuring the flow properly. For that purpose the first step was to write the code to receive the information from the flow meter. A simple first draft of the code was written to check if the pulses that the device emits when liquid goes through it. As it is the main function of the application we have used an

Interrupt function so the part of the code dedicated to measure the flow is always running, indecently of the rest of the lines of de code.

First attempts:

In order to calibrate the device, the first step was to connect and try to receive data from the sensor into the Arduino interface. Therefore a simple code was written to see if the pulses were received by Arduino. As it has been mentioned before the way to process in this regard was implementing an Interrupt function. We have used this kind of function because it

The results of the first try are satisfactory as the program records every pulse that the flowmeter is sending to the Arduino. Our next Step to move on with the coding is to convert this pulses into a known quantity of liquid (water in our case). For that purpose and following the instructions of the datasheet we will do a simple math:

Amount of liquid (L) = pulseCount * 2.25 / 1000

Implementing this equation in the code and comparing the measured amount with a known quantity of water we can check the error of the measure. It is expected to have a high error as the device is not of high quality. The next test has been performed in order to tackle this issue:

In this first step, the Serial displayed the following values:

The Number of pulses is 220 Total amount of liquid (L)495.00
The Number of pulses is 220 Total amount of liquid (L)497.25
The Number of pulses is 222 Total amount of liquid (L)499.50
The Number of pulses is 223 Total amount of liquid (L)501.75
The Number of pulses is 223 Total amount of liquid (L)504.00

The Number of pulses is 224 Total amount of liquid (L)504.00
The Number of pulses is 224 Total amount of liquid (L)504.00
The Number of pulses is 224 Total amount of liquid (L)504.00

In the test, we used a jar with exactly 500 mL of water and we poured it through the sensor. In this example shown the measure is accurate but other tries did not achieve such good results but under the error range stated by the manufacturer. As in this first design of the system the accuracy is not that import it is considered that the flow meter works properly and no adjustment will be made with this regard.

Step 2: Visual notification through LEDs

After confirmation of the proper measuring function the next step is to implement the visual notification system. For that purpose, three LEDs have been connected to different pins of the Arduino systems and a code was written to turn them on when the water consumption (Total Volume) has reached different values. In the safe status, when the water consumption is low, the green LED is ON, letting the user know that he is behaving good. If it keeps consuming water for a certain period, the Total Volume will exceed the safety threshold and the Yellow LED Will be turned on as a *Caution* notification, making de user aware of his or her high-water consumption to hurry up and end the shower as soon as possible. Finally, if the user ignores this notification and the water keeps flowing the Total Volume will reach the *Alert* and the Red LED will be turned on. This final LED will tell the user that he is absolutely consuming too much water and he should immediately get out of the shower. In the following lines, a sample of the code is shown in the moment the LEDs are turned on. For this first draft of the feature the threshold have been set to be 250 mL for turning on the Yellow LED and 350 mL to turn on the Red Light.

Normal state, Green Indicator is on:

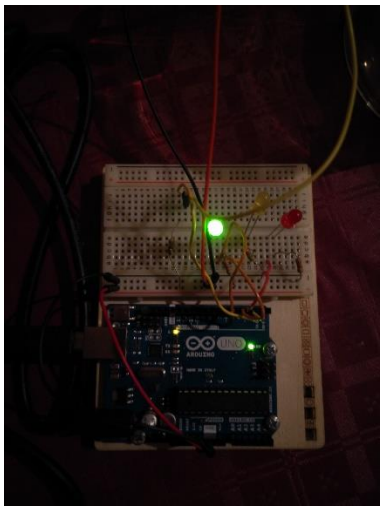
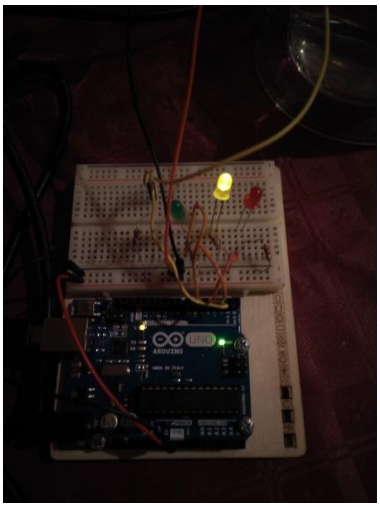
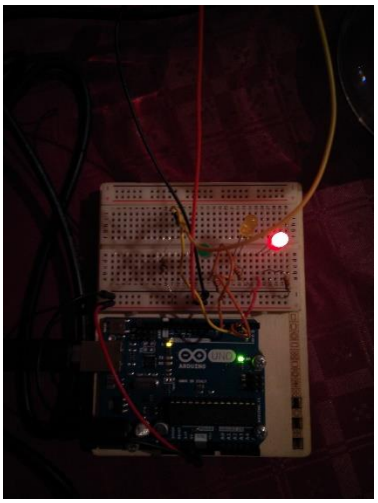
ms: 24454. Pulse no. 62. The Flow is: 14.37 L/min. Total Volume139.50 mL
ms: 24966. Pulse no. 63. The Flow is: 14.37 L/min. Total Volume141.75 mL
ms: 25478. Pulse no. 64. The Flow is: 14.37 L/min. Total Volume144.00 Ml

Caution state, Yellow indicator turns on:

ms: 57221. Pulse no. 111. The Flow is: 28.74 L/min. Total Volume249.75 mL
ms: 57734. Pulse no. 112. The Flow is: 14.37 L/min. Total Volume252.00 mL

Alert state, Red indicator turns on:

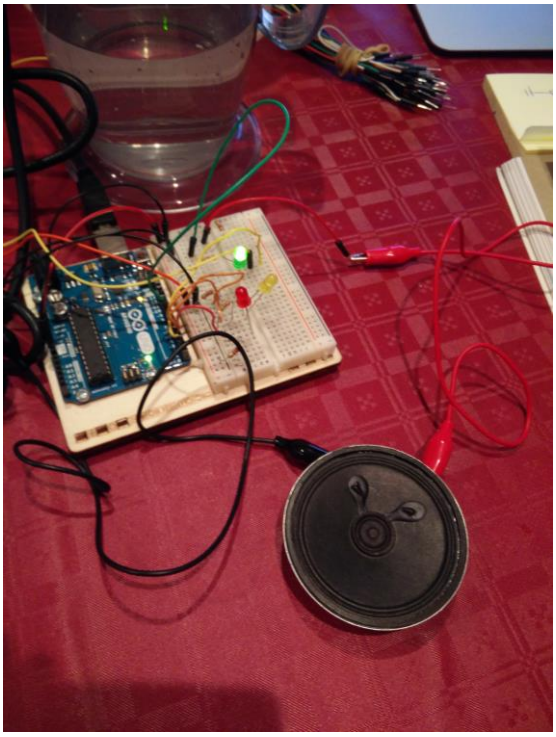
ms: 125899. Pulse no. 155. The Flow is: 93.41 L/min. Total Volume348.75 mL
ms: 126414. Pulse no. 159. The Flow is: 43.11 L/min. Total Volume357.75 mL

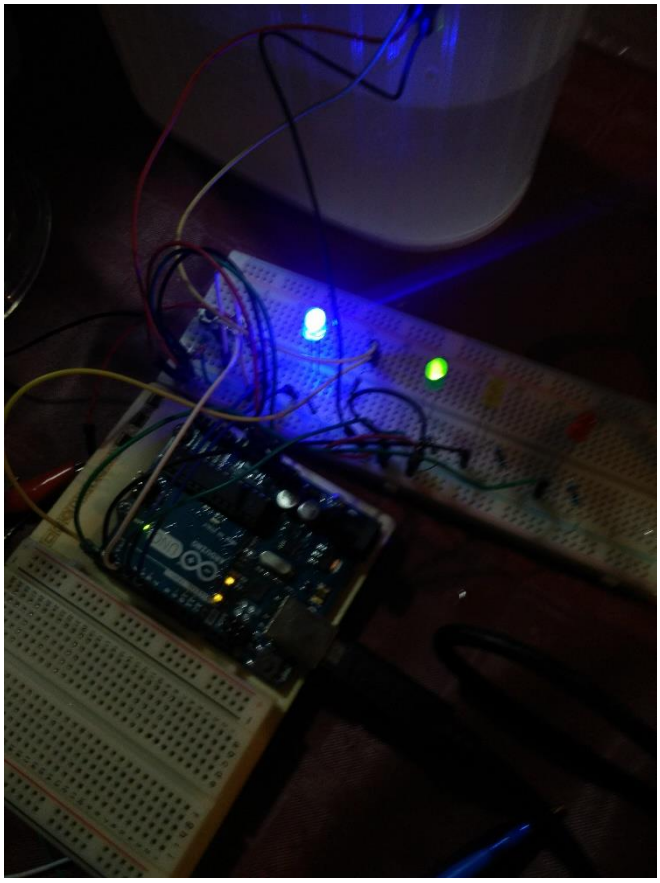
The previous pictures show the system in the different states of the system, from the normal status on the left, to the alert status in the left, being the Caution mode represented in the middle.

Step 3: Sound notification

As users could miss the visual notification and with the objective of increasing the effectiveness of the system in its savings purposes, along with the visual notification, the system will have a sound system to notify the user of the water consumption. When the green LED is on, the user is under the *caution* threshold limit, there will be not sound notification. When the first threshold is reach and the system has left the *normal* status, at the same time than the yellow light is turned on, a sound coming from the speaker will let the user know about the situation. Finally, when the the red light turns on, another louder and uncomfortable sounds will announce the exceed of water consumption and the entrances to the *alarm* mode. The sound notifications just stop once the user has closed the sink and the water flow is zero.



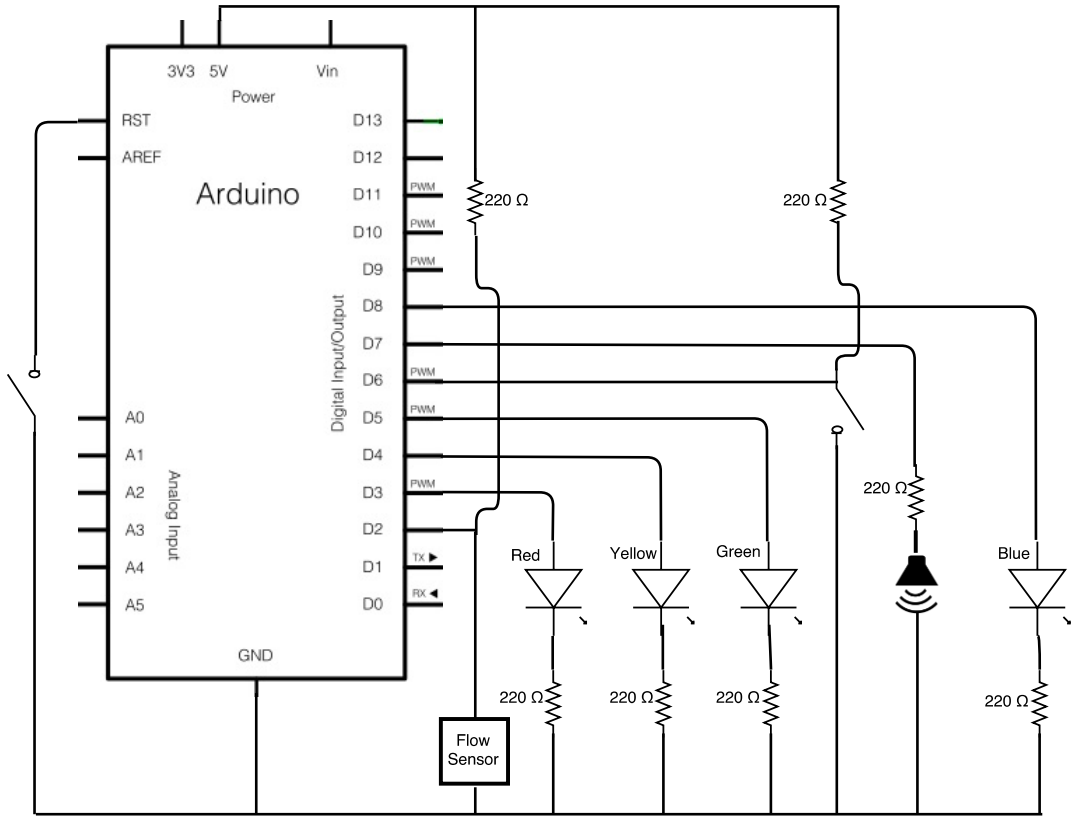
Step 4: Allowing Level of control: Comfort shower vs ECO-shower.



Step 5: Resetting

The results

The final code is shown in the ANNEX 1. With all the features included the system is a fully integrated tool to awake users' awareness about water consumption through monitoring and notification. It measures and displays the amount of water consumed in real-time and notifies, through visual and sounding means when the consumptions exceeds some limits. The expected results haven been tried in real deployments, but it is expect to achieve a water consumption reduction at a domestic level.



Future work

The first step would be to install the device in real pilots. Implementing the system in houses to check how the users behave when receiving the different notifications and measure the reduction of water consumption potential that the rEWAt actually has. Once the first results are satisfactory there is room to implement different improvements.

The system as it is designed nowadays cannot actually “act” over the amount of water consumption of the users but in following versions an actuator together with a control system could set the framework to impose a water flow limitations according to the time or amount of water already consumed. These restrictions in a first stage would come directly from the user consciences but if the scarcity of water in the future is a big issue, limitations from the government could be implemented using this kind of devices.

The current system can be further improved by developing another feature on top of what has been already designed. This version of rEWAt focuses on the amount of water consumed, disregarding its temperature, and therefore the energy consumed for warming it for user’s comfort. Adding a temperature sensor and an actuator over the hot water flow or the boiler d reduce the energy consumed in this regard. For example, exceeding on hot water consumption could be shorted by limiting the amount of if that the user can enjoy after some time/amount.

There other companies working in similar ideas, such as HYDRAO, that presented last November during the Utility their colored LED shower, in which they have implemented microturbines to power the LEDs as well as WiFi communications to send and store the data in the cellphone to enhance gamification and competition to compare your consumption with other users. The user also can change the sequence of colors from

the application. It would be interesting to implement also these features under the rEWAt system so we are not left behind our most direct competitor.

ANNEX I: The code

```
const int sensorPin=2;

volatile int pulseCount;

const int sensorInterrupt=0; //For Interrupt function we use pin 2 of arduino (interrupt pin 0 for this purpose)

float totalliquid=0;

float totalliquidprev=0;

float flow=0;

float flowLmin=0;

int oldTime=0;

int pinTono=7;

int margin=0; //ml

int baselineFLOW=0; // baseline value to compare with


//pushbuton

int inPin = 6;        // the button is connected here

int outPin = 8;       // the LED stating the Status goes to this pin

int state = HIGH;     // by default we set the ECO state

int reading;          // for reading the status

int previous = LOW;   // the previous reading

long time = 0;        // the last time the output pin was toggled

long debounce = 200;  // the debounce time for avoiding noise

//


void setup() {

  pinMode(inPin, INPUT);

  pinMode(outPin, OUTPUT);

  pinMode(sensorPin, INPUT);

  Serial.begin(9600);

  attachInterrupt(0, counter, RISING);
```

```
  pulseCount=0;

    // the for() loop saves some extra coding
  for (int pinNumber = 3; pinNumber < 7; pinNumber++) {
    if (pinNumber == inPin){
      continue;
    }
    pinMode(pinNumber, OUTPUT);
    digitalWrite(pinNumber, LOW);

  }
}

void loop() {

  amoroso();
  if(state == HIGH){
    margin = 250;
    baselineFLOW = 300;
  }
  else {
    margin=1000;
    baselineFLOW=400;
  }
  int dif = (millis()-oldTime);
  int difV= (totalliquid-totalliquidprev);
  if( dif > 500){

    flow=(float)difV/(float)dif;
    flowLmin=flow*3600;
```

```
  Serial.print("ms: ");
  Serial.print(millis());
  Serial.print(". Pulse no.  ");
  Serial.print(pulseCount);
  Serial.print(". The Flow is:  ");
  Serial.print(flowLmin);
  Serial.print(" L/min");
  Serial.print(". Total Volume");
  Serial.print(totalliquid);
  Serial.println(" mL");
  oldTime = millis();
  totalliquidprev = totalliquid;
  }

  // if the current expent water is lower than the baseline
// turn off all LEDs
if (totalliquid < baselineFLOW) {
  digitalWrite(5, HIGH);
  digitalWrite(4, LOW);
  digitalWrite(3, LOW);
} // if the water consumption increases, turn a LED on
else if (totalliquid >= baselineFLOW  && totalliquid < baselineFLOW+margin) {
  digitalWrite(5, LOW);
  digitalWrite(4, HIGH);
  digitalWrite(3, LOW);
    for (int i=30;i<400;i=i+50){
      tone(7,i,500);
      delay(100);
    }
   if(flow==0){
      exit(0);
```

```
    }
  } // if the temperature rises 4-6 degrees, turn a second LED on
  else if (totalliquid >= baselineFLOW + margin) {
    digitalWrite(5, LOW);
    digitalWrite(4, LOW);
    digitalWrite(3, HIGH);
    for (int i=30;i<2000;i=i+50){
      tone(7,i,100);
      delay(10);
     }
  if(flow==0){
    exit(0);
  }
  }

}

void counter()
{
   pulseCount++; //function to increment "count" by 1
   totalliquid=pulseCount*2.25;
}

void amoroso()
{
  reading = digitalRead(inPin);
  if (reading == HIGH && previous == LOW && millis() - time > debounce) {
    if (state == HIGH){
      state = LOW;
         }
    else
```

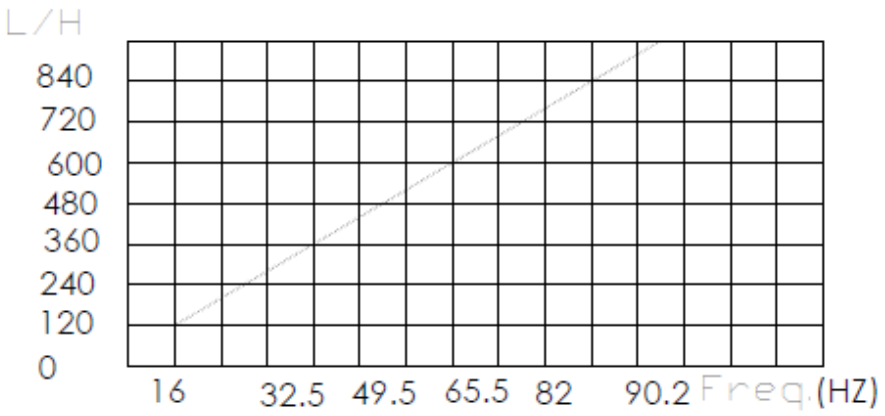
```
      state = HIGH;
    time = millis();
  }
  digitalWrite(outPin, state);
  previous = reading;
}
```

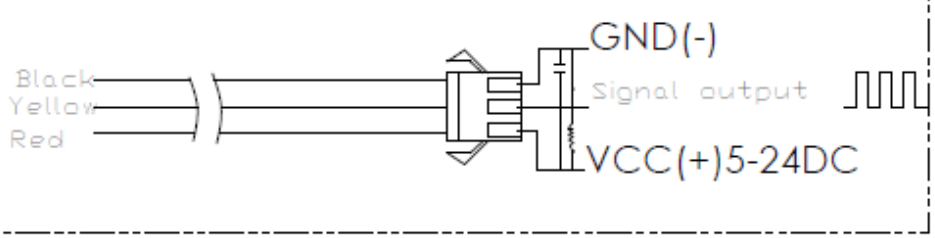
ANNEX II: Flow Meter Datasheet

Flow-Pulse	Flow-Curve	Connection method

一.Flow Range:100L/H-/1800H-L/H

Flow (L/H)	Frezq.(HZ)	Erro range
120L/H	16	±10
240L/H	32.5	
360L/H	49.3	
480L/H	65.5	
600L/H	82	
720L/H	90.2	





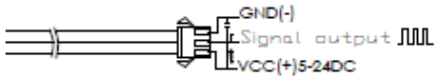
YIFA the plastics Ltd
Prodcut Introduction

1.Modle:YF-21

2.Product Name:Hall sensor

3.Flow Range: 1-30L/MIN

4.(1)Connection Method



(2)Voltage Range 3.5-24VDC, Pluse Characteristic:F=7Q(L/MIN).

(3)Extent of error:±5%.

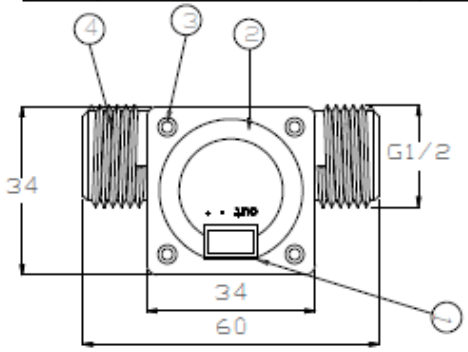
(4)Flow-Pulse

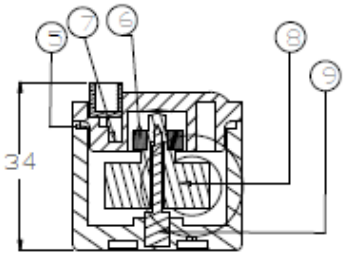
2L/MIN=16HZ 4L/MIN=32.5HZ 6L/MIN=49.3HZ

8L/MIN=65.5HZ 10L/MIN=82HZ

5.Bom

No.	Item	Material	Qty.
1	Connection wire		1
2	Bonnet	PA	1
3	Screw		4
4	Valve body	PA	1
5	Leak press valve		1
6	Magnet		1
7	Hall		1
8	Impeller	POM	1
9	Rustless steel axis	SUS304	1
10			
11			





ANNEX III: Status of the system

In the following picture the system as it is today it is shown in the following picture. All the devices and connections can be checked according to the Circuit shown in Results.