PERIPUMP

(by Ruben Humar - https://github.com/rubenhumar)

INTRODUCCIÓN

En el siguiente artículo se desarrollara proyecto de hardware de código abierto iniciado por Drmn4ea en http://www.thingiverse.com/thing:454702 el 7 de septiembre de 2014.

Se trata de una bomba peristáltica adaptada del cojinete de engranajes planetarios de Emmett usuario en http://www.thingiverse.com/thing:53451. en Febrero 23 de 2013. Se imprime como una sola pieza, con los rodillos en cautividad y no hay superficies de desgaste giratorios. La única parte no impresa es el tubo.

Este Proyecto nos permitirá obtener una bomba de una gran variedad de prestaciones, pudiendo ser esta configurada tanto para su utilización en laboratorios donde se necesite una gran precisión en las cantidades de reactivos a utilizar, como para un uso más industrial logrando el trasvase de grandes caudales brindando además una característica muy especial como es la de no generar o generar muy poca turbulencia en la sustancia transportada.

El Hardware es desarrollado en software libre en un compilador 3D basado en un lenguaje de descripción textual llamado OpenSCAD, logrando así que el diseño sea totalmente paramétrico lo que permite ajustar la tolerancia si es necesario, cambiar las dimensiones, número de planetas, números de dientes, etc.

El Software es desarrollado en código abierto utilizando un controlador Arduino nano y su respectivo programa de compilación, esto nos permite crear cualquier configuración de funcionamiento deseada, logrando así el control de la velocidad y el sentido giro del motor que impulsa la bomba; también dejando la posibilidad de la colocación de sensores para crear un circuito de lazo cerrado.

MATERIALES

Para la realización del proyecto los materiales utilizados son escasos y de fácil adquisición, siendo estos los detallados a continuación:

- 100g aprox. de filamento “ABS” para la impresión de la bomba
- 1 motor PAP “nmb pm55l-048-hpg9” reciclado de una impresora
- 1 controlador programable “Arduino NANO”
- 1 controlador de motores PAP “Pololu A4988”
- 1 lcd nokia 5110
- 1 kit Mando IR
- 1 placa de prototipos “protoboard”
- 50cm de cable para conexiones
- 50cm de manguera de latex

- 5 resistencias 10k, 2 resistencias 220, 1 resistencia 330

CONSTRUCCIÓN

HARDWARE

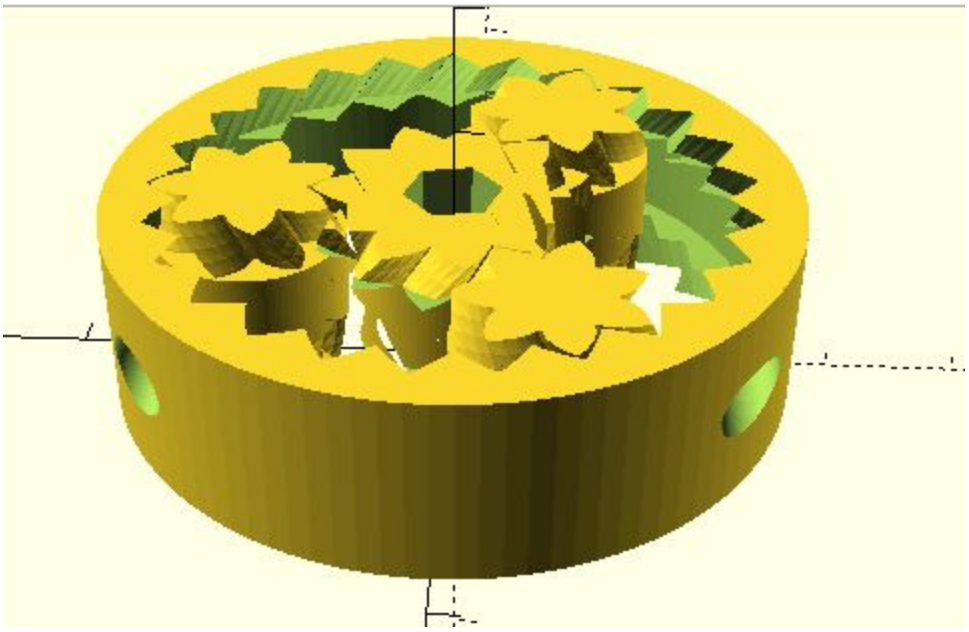
Bomba

Para realizar la bomba utilizamos el programa OpenSCAD y como diseño base la Peristaltic Pump de Drmn4ea, este diseño es bastante bueno y funcional. Debajo se detallan los parámetros configurables que el autor dejó a disposición del fabricante:

```
// --------  Printer-related settings ------------
// Clearance to generate between non-connected parts. If the gears print 'stuck together' or are difficult to separate, try increasing this value. If there is excessive play between them, try lowering it. (default: 0.15mm)
tol=0.15;

// Allowed overhang for overhang removal; between 0 and 0.999 (0 = none, 0.5 = 45 degrees, 1 = infinite)
allowed_overhang = 0.75;

// --------  Details of the tubing used in the pump, in mm ------------

// Outer diameter of your tubing in mm
tubing_od = 4.7625;

// Wall thickness of your tubing
tubing_wall_thickness = 0.79375;
// Amount the tubing should be compressed by the rollers, as a proportion of total thickness (0 = no squish, 1.0 = complete squish)
tubing_squish_ratio = 0.5;

// --------  Part geometry settings ------------

// Approximate outer diameter of ring in mm
D=51.7;

// Thickness i.e. height in mm
T=15;

// Number of planet gears
number_of_planets=3;

// Number of teeth on planet gears
number_of_teeth_on_planets=7;

// Number of teeth on sun gear (approximate)
approximate_number_of_teeth_on_sun=9;
```

```
// pressure angle
P=45;//[30:60]

// number of teeth to twist across
nTwist=1;

// width of hexagonal hole
w=6.7;

DR=0.5*1;// maximum depth ratio of teeth

// ----------------End of customizable values ----------------
```

Logrando así la siguiente pieza:



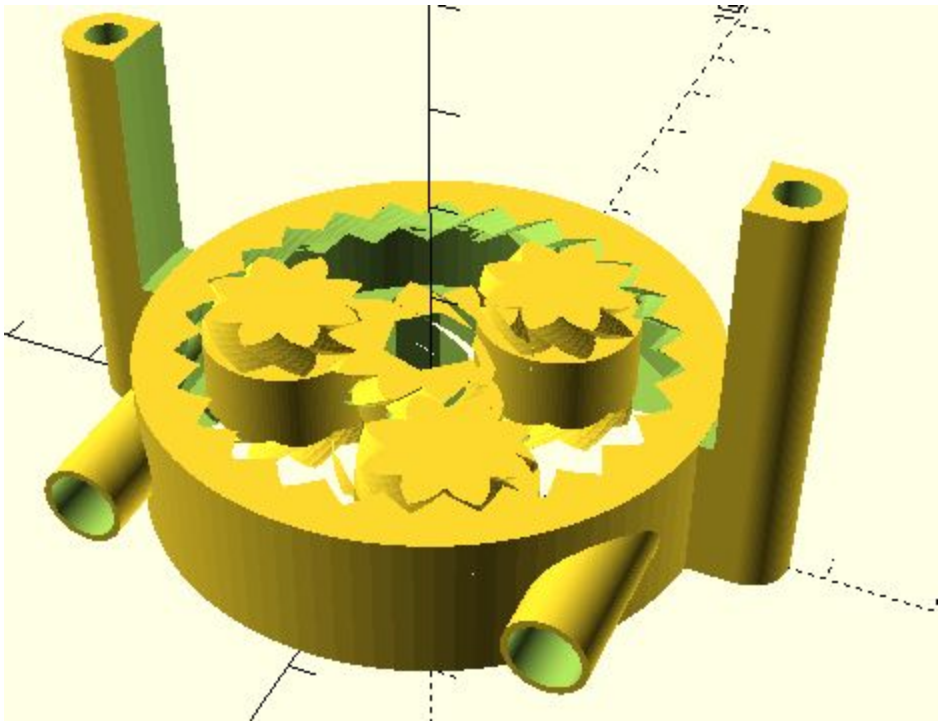
El diseño es completo permitiendo realizar ajustes en relación con la impresión como es una separación para generar entre las partes no conectadas, debido a que una vez impresos son difíciles de separar, así también el valor del voladizo de los engranes fijo; también se ingresan las medidas de la manguera, dimensiones máximas aproximadas y diámetro circunscrito del hexágono del eje central.

Pero para los fines de aplicación es poco práctico debido a que no posee ningún tipo de orificio de fijación al motor, el eje central solo se presenta de forma hexagonal, no permite un buen agarre de la manguera la pieza, y finalmente el idioma utilizado para la descripción de los ajustes es inglés, por lo que se realizó una reversión cambiando el idioma y generando nuevas variables personalizables como se observa a continuación:

```
// --------  Ajustes de geometría de la pieza  ------------
// Orificio de eje:
// Lados de tipo (100 para circulo)
```

```
wh = 6;
// Diametro circunscrito(*0.86 para compatibilidad con alen)
w = 9.2;

// Oreja de soporte:
// Disposicion(no requiere = 0,0)
ejex = 1;
ejey = 0;
// Distancia
dor = 65;
// Diámetro
or = 4;
// Cantidad
cor = 2;
// Sobre Altura (solo hacia +z)(no requiere = 0)
sor = 25;

// tubo exterior de empalme de manguera (no requiere = 0)
te=1;
// Diámetro exterior
ted=8;
// longitud extra al diametro
tel=0;

// Sistema de acople con eje (modo inexacto!!) :
// 1- Medio eje en mitad de altura(no requiere = 0)
me = 0;
// 2- Sobre eje (solo hacia +z)(no requiere = 0)
se = 0;
// Diámetro circunscrito
sed = 7;
// Lados de tipo (100 para circulo)
cse = 6;
// Longitud
led = 10;
// dimensiones de tuerca
espt=1;
ancht=5;
// dimensiones de tornillo
diamt=2;
// ----------------End of customizable values ----------------
```

Logrando así la siguiente pieza:



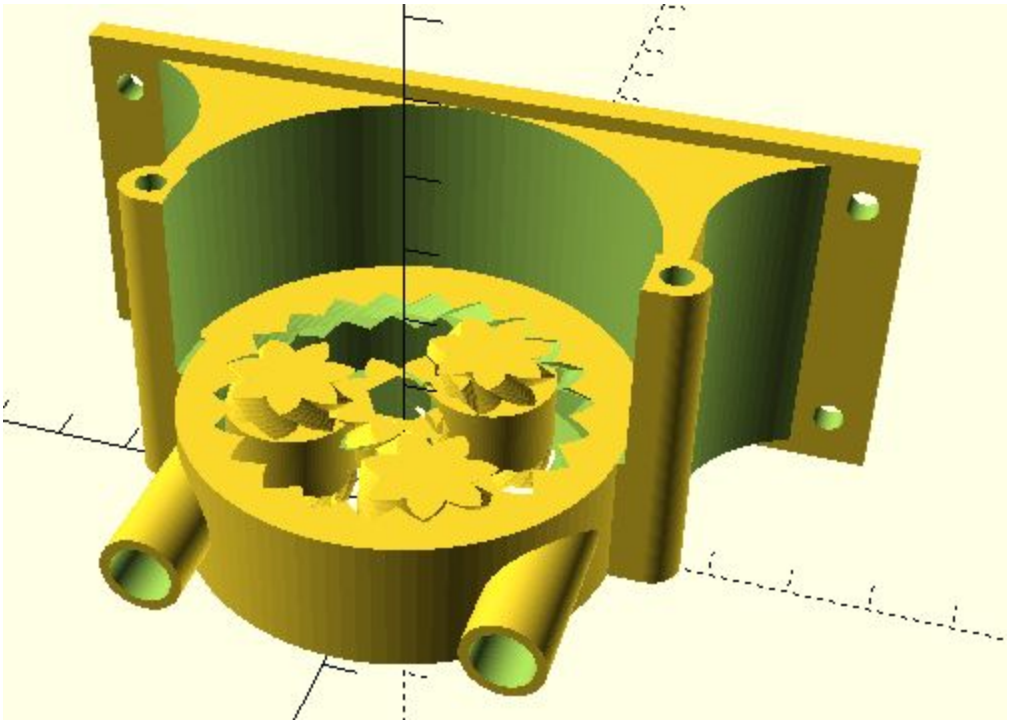
Ahora el diseño permite configurar la fijación a su motor cualquiera sea este, y una correcta retención de la manguera entre otros factores.

No conformes con esto surgió la necesidad de realizar una fijación de todo el conjunto a una base por lo que se agregó este fragmento de código para lograrlo:

```
      difference(){
      translate([0,-D/3.1,T/2])
   cube([D*1.25,D/2,T],center=true);
  translate([0,0,-0.1])
  cylinder(r=D/2,T+sor+1,$fn=100);};

  difference(){
      translate([0,-D/3.1,(T+sor)/2])
   cube([D*1.25,D/2,T+sor],center=true);
      translate([0,0,-0.1])
  cylinder(r=D/1.75,T+sor+1,$fn=100);};

  difference(){
   translate([0,-D/2-6,(T+sor)/2])
   cube([D*2.1,4,T+sor+4],center=true);

for (x=[1,-1]){
      translate([50*x,.1,5])
      rotate([90,0,0])
  cylinder(r=or/2,100,$fn=100);
        translate([50*x,.1,38])
      rotate([90,0,0])
  cylinder(r=or/2,100,$fn=100);};};
```

```
        difference(){
      translate([D/1.7,-30.5,0])
   cube([20,20,T+sor],center=false);
  translate([D,-12,-0.1])
  cylinder(r=20,T+sor+1,$fn=100);};

           difference(){
      translate([-D/1.7-20,-30.5,0])
   cube([20,20,T+sor],center=false);
  translate([-D,-12,-0.1])
  cylinder(r=20,T+sor+1,$fn=100);};
```

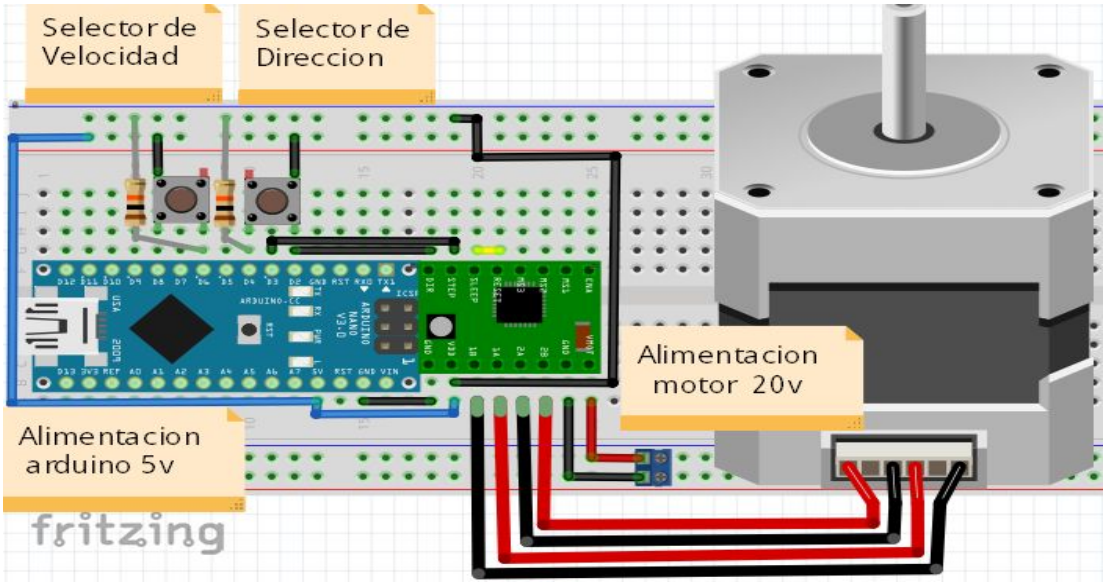
Logrando así la siguiente pieza:



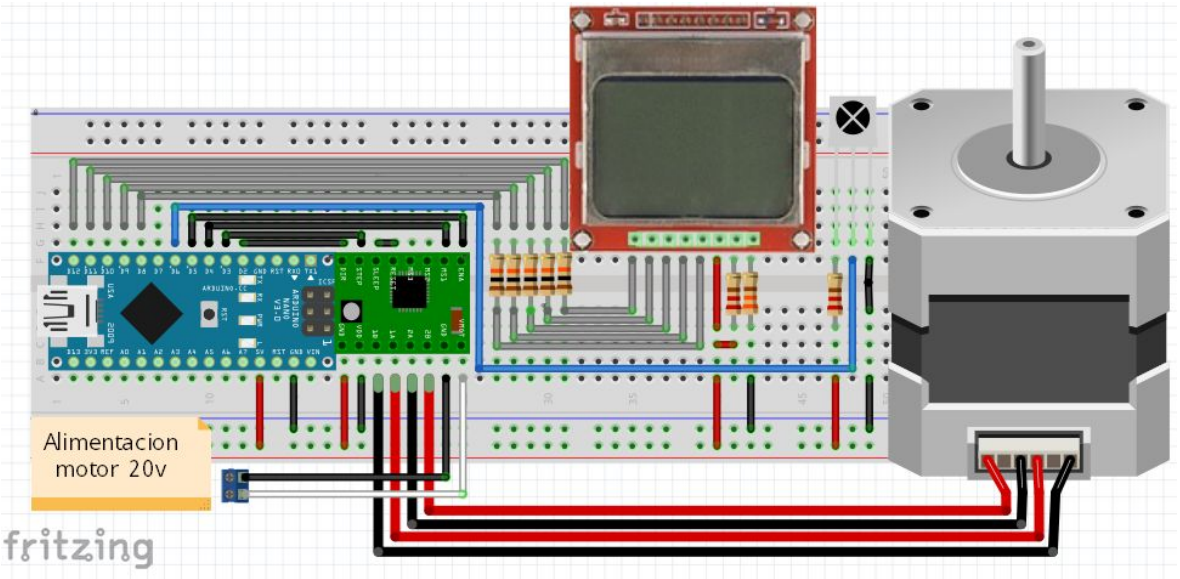
Una vez hechos estos ajustes se obtuvo la pieza final con la que se realizara el prototipo.

Circuitos

El circuito de funcionamiento y control de la bomba se realizó con un motor PAP nmb pm55l-048-hpg9 reciclado de una impresora, Controlado por un Pololu A4988 quien a su vez es controlado por un Arduino NANO el cual recibe las ordenes de encendido, variación de velocidad y de dirección de dos pulsadores conectados a sus pines quienes al dar la un pulso generan que los pines salgan de su estado de bajo generados por las resistencias de pull-down para pasar a un estado alto y asi lograr la variación del parámetro deseado, para un fácil armado del circuito se decidió montarlo en un protoboard siendo todo esto esquematizado aquí:



Luego de esta placa prototipo se realizo remplazando los selectores de velociadad y sentido por un display lcd Nokia 5110 y un modulo de mando a distancia IR logrando lo siguiente:



SOFTWARE

Para realizar el código de operación de Arduino nano se utilizó el programa oficial de arduino con el cual se genero para la primer placa prototipo el siguiente código:

```
int der = 4; //Pulsador derecha
int izq = 6; //Pulsador izquierda
int a = 0;  //gen counter
int b = 8;  //gen counter
int stp = 3; //paso
int dir = 2; //direccion

void setup()
{
  pinMode(der, INPUT);
  pinMode(izq, INPUT);
  pinMode(stp, OUTPUT);
  pinMode(dir, OUTPUT);
}

void loop()
{
  if (digitalRead(der) == HIGH)
  {
    delay(500);
    if ((a) == 0)
    {
      digitalWrite(dir, LOW);
      a++;
    }
    else {
      digitalWrite(dir, HIGH);
      a--;
    }
  }
  if (digitalRead(izq) == HIGH)
  {
    if (b <= 8)
    {
      b--;
      delay(500);
      if (b < 0) {
        b = 8;
        delay(500);
      }
    }
  }

  digitalWrite(stp, HIGH);
  delay(1 + pow(2, b));
  digitalWrite(stp, LOW);
  delay(1 + pow(2, b));
}
```

Este código permite obtener la recepción de datos a través de las pines , 4 y 6 información con la que dependiendo del estado de estas entradas podremos seleccionar la velocidad de funcionamiento deseada a partir de 9 velocidades preestablecidas, y también seleccionar el sentido de giro del motor.

La salida de datos de este controlador está a cargo de los pines 2 y 3 quienes son responsables de en primer lugar tomar un estado alto para la rotación en sentido horario o un estado bajo para la rotación en sentido anti horario, y transmitir el tren de pulsos que hará que el rotor del motor varié un paso con una frecuencia seleccionada por el usuario.

Siendo modificado luego para la segunda placa prototipo por este:

```
#include "IRremote.h"
#include "LCD5110_Basic.h"

LCD5110 myGLCD(12,11,10,8,9); //LCD5110 myGLCD(CLK, DIN, DC, RST, CE)
IRrecv irrecv(6);           // create instance of 'irrecv'
decode_results results;      // create instance of 'decode_results'

int dir=2;  //direccion
int stp=3;  //paso
int en=5;  //paso
int ms1=4;  //paso

extern uint8_t arduino_logo[];
extern uint8_t utn_logo[];
extern uint8_t lab_logo[];
extern uint8_t SmallFont[];
extern uint8_t MediumNumbers[];
extern uint8_t BigNumbers[];

String main_0 = "PERIPUMP";
String main_1 = "TRASVASAR";
String main_2 = "AJUSTES";

String menu_1 = "Trasvasar";
String menu_2 = "Ajustes";

String menu_1_1 = "Por Volumen";
String menu_1_2 = "Por Tiempo";
String menu_1_3 = "Serial";
String menu_1_4 = "Libre";

String menu_2_1 = "Purgar";
String menu_2_2 = "Calibrar";
String menu_2_3 = "Prueba";

String menu_volver = "Volver";

float vol=0;
float vold=0;
int flu=0;
int sent=1;
int tiempo=0;
int tiempot=0;
int tiempoa=0;
int por=0;
int cant=1;
int s=0;
int se=0;
int t=0;
int p=0;

int v=0;
int vl=0;

int ti=0;
int ml=0;
long pa=0;
int del=0;
int Ms1=0;

float paml=0;
float tml=0;

int i=0;

int ini=0;
int e=0;
int o=0;
int x=0;
int l=0;
int menu=0;

void setup()
{

 pinMode(dir , OUTPUT);//direccion
 pinMode(stp , OUTPUT);//paso
 pinMode(en , OUTPUT);//enable
 pinMode(ms1 , OUTPUT);//enable

 digitalWrite(ms1, LOW);
 digitalWrite(en, HIGH);

 Serial.begin(9600);
 Serial.println("PERIPUMP START");
 irrecv.enableIRIn();

 myGLCD.InitLCD();

 myGLCD.setContrast(60);
 myGLCD.clrScr();
 myGLCD.drawBitmap(0, 0, arduino_logo, 84, 48);
 delay(2000);

 myGLCD.clrScr();
 myGLCD.setFont(SmallFont); //Declara el tipo de fuente a usar
 myGLCD.print("iniciando", CENTER, 0);
 myGLCD.print("|          |", CENTER, 16);
 myGLCD.print("por favor", CENTER, 32);
 myGLCD.print("espere", CENTER, 40);
 delay(250);
  for (int a=0; a<12; a++)
 {
  myGLCD.print("\\", 6+(a*6), 16);
  delay(230);
 }

 myGLCD.clrScr();
 myGLCD.drawBitmap(0, 0, utn_logo, 84, 48);
 delay(5000);

 myGLCD.clrScr();
 myGLCD.drawBitmap(0, 0, lab_logo, 84, 48);
 delay(8000);

 myGLCD.clrScr();
 Menu();
}


void loop()
{
 switch(ini){

 case 0:
 if (irrecv.decode(&results)){ // se recive seÃ±al IR?
   IR();
   irrecv.resume(); // Recivir siguiente valor
   Menu();
   Menu();//translateIR();   // Serial.println(results.value, HEX);  UN Comment to see raw values
 }
   break;

 case 1:
 if (irrecv.decode(&results)){ // se recive seÃ±al IR?
   ValorIR();
   irrecv.resume(); // Recivir siguiente valor
   Menu();
```

```
   Menu();//translateIR();   //
Serial.println(results.value, HEX);  UN Comment
to see raw values
  }
    break;

  case 2:
  for (int si=0; si<=1500; si++)
  {
     myGLCD.print("FINALIZADO!", CENTER, 0);
     delay(1000);
     myGLCD.invertText(true);
     myGLCD.print("FINALIZADO!", CENTER, 0);
     myGLCD.invertText(false);
     delay(1000);

   if (irrecv.decode(&results)){ // se recive
seÃ±al IR?
   IR();
   irrecv.resume(); // Recivir siguiente valor
   };
   if (o==1){
   si=1500;
   o=0;
   ini=0;
   menu=0;
   myGLCD.clrRow(0);
   };
  };
   ini=0;
   menu=0;
   myGLCD.clrRow(0);
   s=0;
   break;
};
};

void IR()
{
  switch(results.value){
  case 0xFFA25D:
  e++;
     Serial.println("+");
    break;
  case 0xFF629D:
  o=1;
     Serial.println("ok");
    break;
  case 0xFFE21D:
  e--;
     Serial.println("-");
    break;
  default:
    break;
  }
};
void TrasIR()
{
   if (irrecv.decode(&results)){ // se recive seÃ±al
IR?
   irrecv.resume(); // Recivir siguiente valor
switch(results.value){
case 0xFF906F:
 if(Ms1==0){
   digitalWrite(ms1, HIGH);
   Ms1=1;
 }else{
   digitalWrite(ms1, LOW);
   Ms1=0;
 }
  Serial.println("EQ");
  break;

case 0xFFC23D:
  p=1;
  Serial.println(">||");
  break;
 }
};
};

void ValorIR()
{
switch(results.value){ // toma las acciones de las
seÃ±aes IR}

 case 0xFF629D:
    Serial.println("ok");
  o=0;
  ini=0;
  vl=0;
  v=0;
  x=0;
  i=0;
  e++;
  break;

 case 0xFF6897:
  Serial.println("0");
  v=0;
  i++;
  break;

 case 0xFF9867:
  Serial.println("100+");
  break;

 case 0xFFB04F:
  Serial.println("200+");
  break;

 case 0xFF30CF:
  Serial.println("1");
  v=1;
  i++;
  break;

 case 0xFF18E7:
  Serial.println("2");
  v=2;
  i++;
  break;

case 0xFF7A85:
  Serial.println("3");
  v=3;
  i++;
  break;

case 0xFF10EF:
  Serial.println("4");
  v=4;
  i++;
  break;

case 0xFF38C7:
  Serial.println("5");
  v=5;
  i++;
  break;

case 0xFF5AA5:
  Serial.println("6");
  v=6;
  i++;
  break;

case 0xFF42BD:
  Serial.println("7");
  v=7;
  i++;
  break;

case 0xFF4AB5:
  Serial.println("8");
  v=8;
  i++;
  break;

case 0xFF52AD:
  Serial.println("9");
  v=9;
  i++;
  break;
 };
 delay(200);
 if ((x+1)==i){
  vl=v;
  i=1;
 }else{
  vl=vl*10+v;
 }
};



void Menu()
{
   switch(menu)  {
```

```
   case 0:
   if(e<=-1){
     e=1;
   };
   if(e>=2){
     e=0;
   };
  myGLCD.clrScr();
  myGLCD.setFont(SmallFont);
  myGLCD.print(main_0, CENTER, 0);
  myGLCD.print(menu_1, CENTER, 16);
  myGLCD.print(menu_2, CENTER, 24);
  myGLCD.print("LAB FD 2016", CENTER, 40);

   switch(e)  {
//------------------------------------------------------------
-
  case 0:
  myGLCD.invertText(true);
  myGLCD.print(menu_1, CENTER, 16);
  myGLCD.invertText(false);
    if (o==1){
   o=0;
   e=0;
   menu=1;
   };
  break;
  case 1:
  myGLCD.print(menu_1, CENTER, 16);
  myGLCD.invertText(true);
  myGLCD.print(menu_2, CENTER, 24);
  myGLCD.invertText(false);
    if (o==1){
   o=0;
   e=0;
   menu=2;
   };
     break;
};
   break;
//------------------------------------------------------------
-
    case 1:
   if(e<=-1){
     e=4;
   };
   if(e>=5){
     e=0;
   };
  myGLCD.clrScr();
  myGLCD.setFont(SmallFont);
  myGLCD.print(main_1, CENTER, 0);
  myGLCD.print(menu_1_1, CENTER, 8);
  myGLCD.print(menu_1_2, CENTER, 16);
  myGLCD.print(menu_1_3, CENTER, 24);
  myGLCD.print(menu_1_4, CENTER, 32);
  myGLCD.print(menu_volver, CENTER, 40);

   switch(e)  {
  case 0:
  myGLCD.invertText(true);
  myGLCD.print(menu_1_1, CENTER, 8);
  myGLCD.invertText(false);
    if (o==1){
  o=0;
  myGLCD.clrScr();
  myGLCD.setFont(SmallFont);
  myGLCD.print(menu_1_1, CENTER, 0);
  menu=3;
  t=0;
  s=0;
  e=6;
  l=0;
  };
    break;

case 1:
myGLCD.print(menu_1_1, CENTER, 8);
myGLCD.invertText(true);
myGLCD.print(menu_1_2, CENTER, 16);
myGLCD.invertText(false);
    if (o==1){
  o=0;
  myGLCD.clrScr();
  myGLCD.setFont(SmallFont);
  myGLCD.print(menu_1_2, CENTER, 0);
  menu=3;
  t=1;
  s=0;
  l=0;
  };
    break;

case 2:
myGLCD.print(menu_1_2, CENTER, 16);
myGLCD.invertText(true);
myGLCD.print(menu_1_3, CENTER, 24);
myGLCD.invertText(false);
    if (o==1){
  myGLCD.clrScr();
  myGLCD.setFont(SmallFont);
  myGLCD.print(menu_1_3, CENTER, 0);
  o=0;
  menu=3;
  t=0;
  s=1;
  l=0;
  e=0;
  };
    break;
case 3:
myGLCD.print(menu_1_3, CENTER, 24);
myGLCD.invertText(true);
myGLCD.print(menu_1_4, CENTER, 32);
myGLCD.invertText(false);
    if (o==1){
  myGLCD.clrScr();
  myGLCD.setFont(SmallFont);
  myGLCD.print(menu_1_4, CENTER, 0);
  o=0;
  menu=3;
  t=0;
  s=0;
  l=1;
  e=6;
  };
    break;
  case 4:
  myGLCD.print(menu_1_4, CENTER, 32);
  myGLCD.invertText(true);
  myGLCD.print(menu_volver, CENTER, 40);
  myGLCD.invertText(false);
   if (o==1){
   o=0;
   menu=0;
   };
    break;
};
    break;
//------------------------------------------------------
    case 2:
   if(e<=-1){
     e=3;
   };
   if(e>=4){
     e=0;
   };
  myGLCD.clrScr();
  myGLCD.setFont(SmallFont);
  myGLCD.print(main_2, CENTER, 0);
  myGLCD.print(menu_2_1, CENTER, 16);
  myGLCD.print(menu_2_3, CENTER, 24);
  myGLCD.print(menu_2_2, CENTER, 32);
  myGLCD.print(menu_volver, CENTER, 40);

   switch(e)  {
  case 0:
  myGLCD.invertText(true);
  myGLCD.print(menu_2_1, CENTER, 16);
  myGLCD.invertText(false);
   if (o==1){
   o=0;
   e=0;
   menu=0;
   Purgar();
   };
    break;

  case 1:
  myGLCD.print(menu_2_1, CENTER, 16);
  myGLCD.invertText(true);
  myGLCD.print(menu_2_3, CENTER, 24);
  myGLCD.invertText(false);
   if (o==1){
   o=0;
   e=0;
   menu=5;
   };
    break;
```

```
  case 2:
  myGLCD.print(menu_2_3, CENTER, 24);
  myGLCD.invertText(true);
  myGLCD.print(menu_2_2, CENTER, 32);
  myGLCD.invertText(false);
      if (o==1){
   myGLCD.clrScr();
   myGLCD.setFont(SmallFont);
   myGLCD.print(menu_2_3, CENTER, 0);
   o=0;
   e=0;
   menu=4;
   };
      break;
  case 3:
  myGLCD.print(menu_2_2, CENTER, 32);
  myGLCD.invertText(true);
  myGLCD.print(menu_volver, CENTER, 40);
  myGLCD.invertText(false);
   if (o==1){
   o=0;
   e=0;
   menu=0;
  };
     break;
};
     break;
//-------------------------------------------------------
    case 3:
    if(s==1){
      if(e<=-1){
      e=5;
      };
    }else{
      if(l==1){
      if(e<=1){
      e=5;
      };
      };
      if(e<=0){
      e=5;
      };
    };
    if(e>=6){
      if(l==1){
      e=2;
      }else{
      e=1;
      };
      if(s==1){
      e=0;
      };
    };
  myGLCD.setFont(SmallFont);
  myGLCD.print(menu_volver, RIGHT, 40);
  myGLCD.print("Iniciar", LEFT, 40);
  myGLCD.print("Sent.:", LEFT, 32);
  if (se==0){
  myGLCD.print("Ahorario", RIGHT, 32);
  }else{
  myGLCD.print(" Horario", RIGHT, 32);
  }
  myGLCD.print("Flu. :", LEFT, 24);
  myGLCD.printNumI(flu , 52, 24, 2, '0');
  myGLCD.print("mls", RIGHT, 24);
  if (t==1){
  myGLCD.print("Tiempo:", LEFT, 16);
  myGLCD.printNumI(tiempo, 50, 16, 4, '0');
  myGLCD.print("s", RIGHT, 16);
  }else{
  if (l==0){
  myGLCD.print("Vol. :", LEFT, 16);
  myGLCD.printNumI(vol, 46, 16, 4, '0');
  myGLCD.print("ml", RIGHT, 16);
  }
  }
  if (s==1){
  myGLCD.print("Cant.:", LEFT, 8);
  myGLCD.printNumI(cant, 46, 8, 3, '0');
  myGLCD.print("und", RIGHT, 8);
  }
   switch(e)  {

  case 0:
  myGLCD.invertText(true);
  myGLCD.print("Cant.:", LEFT, 8);
  myGLCD.invertText(false);
    if (o==1){
  ini=1;
  x=3;
  cant=vl;
  myGLCD.invertText(true);
  myGLCD.printNumI(cant, 46, 8, 3, '0');
  myGLCD.invertText(false);
  };
     break;

  case 1:
  myGLCD.invertText(true);
  if (t==1){
  myGLCD.print("Tiempo:", LEFT, 16);
  myGLCD.invertText(false);
     if (o==1){
  ini=1;
  x=4;
  tiempo=vl;
  myGLCD.invertText(true);
  myGLCD.printNumI(tiempo, 50, 16, 4, '0');
  myGLCD.invertText(false);
  };
  }else{
  myGLCD.print("Vol. :", LEFT, 16);
  myGLCD.invertText(false);
      if (o==1){
  ini=1;
  x=4;
  vol=vl;
  myGLCD.invertText(true);
  myGLCD.printNumI(vol, 46, 16, 4, '0');
  myGLCD.invertText(false);
  };
  };
     break;

  case 2:
  myGLCD.invertText(true);
  myGLCD.print("Flu. :", LEFT, 24);
  myGLCD.invertText(false);
  if (o==1){
  ini=1;
  x=2;
  flu=vl;
  myGLCD.invertText(true);
  myGLCD.printNumI(flu , 52, 24, 2, '0');
  myGLCD.invertText(false);
  };
     break;

  case 3:
  myGLCD.invertText(true);
  myGLCD.print("Sent.:", LEFT, 32);
  myGLCD.invertText(false);
  if (o==1){
    if (se==0){
     se=1;
    }else{
     se=0;
    };
  myGLCD.invertText(true);
  if (se==1){
  myGLCD.print("Ahorario", RIGHT, 32);
  }else{
  myGLCD.print(" Horario", RIGHT, 32);
  }
  myGLCD.invertText(false);
   delay(500);
   o=0;
  };
      break;

  case 4:
  myGLCD.invertText(true);
  myGLCD.print("Iniciar", LEFT, 40);
  myGLCD.invertText(false);
      if (o==1){
   o=0;
   menu=0;
   e=0;
   Tras();
   };
     break;

  case 5:
  myGLCD.invertText(true);
  myGLCD.print(menu_volver, RIGHT, 40);
  myGLCD.invertText(false);
      if (o==1){
   o=0;
   menu=1;
   e=0;
```

```
  };
    break;

};
    break;
//-------------------------------------------------------
  case 4:
  if(e<=-1){
    e=4;
  };
  if(e>=5){
    e=0;
  };

 myGLCD.print("del:", LEFT, 8);
 myGLCD.print("ml:", LEFT, 16);
 myGLCD.print("ti:", LEFT, 24);
 myGLCD.print("pa:", LEFT, 32);
 myGLCD.print(menu_volver, CENTER, 40);

 myGLCD.printNumI(del, 46, 8, 4, '0');
 myGLCD.printNumI(ml, 46, 16, 4, '0');
 myGLCD.printNumI(ti, 46, 24, 4, '0');
 myGLCD.printNumI(pa, 46, 32, 6, '0');

  switch(e)  {

 case 0:
 myGLCD.invertText(true);
 myGLCD.print("del:", LEFT, 8);
 myGLCD.invertText(false);
  if (o==1){
 ini=1;
 x=4;
 del=vl;
 myGLCD.invertText(true);
 myGLCD.printNumI(del, 46, 8, 4, '0');
 myGLCD.invertText(false);
 };
     break;

 case 1:
 myGLCD.invertText(true);
 myGLCD.print("ml:", LEFT, 16);
 myGLCD.invertText(false);
  if (o==1){
 ini=1;
 x=4;
 ml=vl;
 myGLCD.invertText(true);
 myGLCD.printNumI(ml, 46, 16, 4, '0');
 myGLCD.invertText(false);
 };
    break;

  case 2:
 myGLCD.invertText(true);
 myGLCD.print("ti:", LEFT, 24);
 myGLCD.invertText(false);
  if (o==1){
 ini=1;
 x=4;
 ti=vl;
 myGLCD.invertText(true);
 myGLCD.printNumI(ti, 46, 24, 4, '0');
 myGLCD.invertText(false);
 };
     break;

  case 3:
 myGLCD.invertText(true);
 myGLCD.print("pa:", LEFT, 32);
 myGLCD.invertText(false);
  if (o==1){
 ini=1;
 x=6;
 pa=vl;
 myGLCD.invertText(true);
 myGLCD.printNumI(pa, 46, 32, 6, '0');
 myGLCD.invertText(false);
 };
      break;

  case 4:
 myGLCD.invertText(true);
 myGLCD.print(menu_volver, CENTER, 40);
 myGLCD.invertText(false);
      if (o==1){
   o=0;
   menu=2;
   e=0;
   };
     break;
 };
    break;
//-------------------------------------------------------
   case 5:
 myGLCD.clrScr();
 myGLCD.invertText(true);
 myGLCD.print(menu_2_3, CENTER, 0);
 myGLCD.invertText(false);

    myGLCD.print("volumen", CENTER, 8);
//tiempo transcurrido
    myGLCD.print("conocido a", CENTER, 16);
//tiempo transcurrido
    myGLCD.print("trasvasar:", CENTER, 24);
//tiempo transcurrido

   if(e<=-1){
     e=1;
   };
   if(e>=2){
     e=0;
   };

 myGLCD.print("ml:", LEFT, 32);
 myGLCD.print("ml", RIGHT, 32);
 myGLCD.print("COMENZAR", CENTER, 40);
 myGLCD.printNumI(ml, 46, 32, 4, '0');

  switch(e)  {
 case 0:
 myGLCD.invertText(true);
 myGLCD.print("ml:", LEFT, 32);
 myGLCD.invertText(false);
  if (o==1){
 ini=1;
 x=4;
 ml=vl;
 myGLCD.invertText(true);
 myGLCD.printNumI(ml, 46, 32, 4, '0');
 myGLCD.invertText(false);
 };
      break;

 case 1:
 myGLCD.invertText(true);
 myGLCD.print("COMENZAR", CENTER, 40);
 myGLCD.invertText(false);
  if (o==1){
 Prueba();
 o=0;
 };
    break;
  };
 break;
};
};

void Tras()
{
 paml=pa/ml;
 float dela=1/(flu*paml)*1000;
 int pasostotal=paml*vol;

 tiempoa=millis()/1000;

 digitalWrite(en, LOW);
 if (se==0){
 digitalWrite(dir, LOW);
 }else{
 digitalWrite(dir, HIGH);
 };

 myGLCD.clrRow(1);
 myGLCD.clrRow(2);
 myGLCD.clrRow(3);
 myGLCD.clrRow(4);
 myGLCD.clrRow(5);

 myGLCD.setFont(SmallFont);
 myGLCD.print("vol", LEFT, 8);
 myGLCD.print("desp", LEFT, 16);
 myGLCD.print("ml", RIGHT, 16);

if (t==0){
 myGLCD.print("de:", 15, 24);
 myGLCD.print("ml",56, 24);
 }else{
```

```
 myGLCD.print("en:", 15, 24);
 myGLCD.print("s",56, 24);
 };

 myGLCD.print("t:", LEFT, 32);
 myGLCD.print("s",34, 32);

 myGLCD.print("p:", 48, 32);
 myGLCD.print("%", RIGHT, 32);

if (s==0){
 myGLCD.print("s:",LEFT, 40);
 if (se==0){
 myGLCD.print("Ahor", 10, 40);
 }else{
 myGLCD.print(" Hor", 10, 40);
 };
 myGLCD.print("v:", 44, 40);
 myGLCD.print("mls",65, 40);
  }else{
 myGLCD.print("cant=", LEFT, 40);
 myGLCD.print("      und", RIGHT, 40);
 myGLCD.print("/", 43 , 40);
  };

if (l==0){
    if(t==0){
//volumen
 for (int c=1; c<=cant; c++)
 {
//catidad
   TrasIR();
   if (p==0){
 for (int pasos=0; pasos<=pasostotal; pasos++)
 {
   TrasIR();
   if (p==0){
   tiempot=millis()/1000-tiempoa;
   vold=pasos/paml;
   myGLCD.setFont(MediumNumbers);
   myGLCD.printNumI(vold, 24, 8, 4, '0');
//volumen desplazado
   myGLCD.setFont(SmallFont);
   por=100*vold/vol;
   myGLCD.printNumI(vol, CENTER, 24, 4,'0');
//volumen total
   myGLCD.printNumI(tiempot, 9, 32, 4,'0');
//tiempo transcurrido

  if (s==1){
   myGLCD.printNumI(c, 25, 40, 3,'0');
//cantidad
   myGLCD.printNumI(cant, 48, 40, 3,'0');
//cantidad
  }else{
   myGLCD.printNumI(flu, 53, 40, 2,'0'); //caudal
  };

   myGLCD.printNumI(por, 60, 32, 3,'0');
//progreso

            if(pasos%2){
            digitalWrite(stp, LOW);
            }else{
            digitalWrite(stp, HIGH);
            };
            delay(dela);
   }else{
    pasos=pasostotal;
   };
   };
   if (cant>1){
   delay(10000);
   };
   }else{
    c=10000;
   };
   };

   }else{
//tiempo
     tiempot=millis()/1000-tiempoa;
   for (int pasos=0; tiempot<=tiempo; pasos++)
   {
   TrasIR();
   if (p==0){
   tiempot=millis()/1000-tiempoa;
   vold=pasos/paml;
   por=100*tiempot/tiempo;
   myGLCD.setFont(MediumNumbers);
   myGLCD.printNumI(vold, 24, 8, 4, '0');
//volumen desplazado
   myGLCD.setFont(SmallFont);
   myGLCD.printNumI(tiempot, CENTER, 24,
4,'0'); //tiempo transcurrido
   myGLCD.printNumI(tiempo, 9, 32, 4,'0');
//tiempo
   myGLCD.printNumI(flu, 53, 40, 2,'0'); //caudal
   myGLCD.printNumI(por, 60, 32, 3,'0');
//progreso

            if(pasos%2){
            digitalWrite(stp, LOW);
            }else{
            digitalWrite(stp, HIGH);
            };
            delay(dela);

              }else{
    tiempot=tiempo;
   };
   };
   };



   }else{
//libre
      myGLCD.clrRow(2);
      myGLCD.clrRow(3);
      myGLCD.clrRow(4);
      myGLCD.clrRow(5);
   for (int  pasos=0; pasos<=999999999;
pasos++)
   {
   TrasIR();
   if (p==0){
   tiempot=millis()/1000-tiempoa;
   myGLCD.setFont(SmallFont);
   vold=pasos/paml;
   myGLCD.print("vol desplazado:", CENTER, 8);
   myGLCD.print("tiempo:", LEFT, 40);

   myGLCD.print("ml", RIGHT, 24);
   myGLCD.print("s ", RIGHT, 40);

   myGLCD.printNumI(tiempot, 45, 40, 4, '0');
//tiempo transcurrido

   myGLCD.setFont(BigNumbers);
   myGLCD.printNumI(vold, CENTER, 16, 4,'0');
//volumen desplazado

            if(pasos%2){
            digitalWrite(stp, LOW);
            }else{
            digitalWrite(stp, HIGH);
            };
            delay(dela);
   }else{
    pasos=999999999;
   };
};
};
 digitalWrite(ms1, LOW);
 Ms1=0;
 digitalWrite(en, HIGH);
 ini=2;
 p=0;
 s=0;
 t=0;
 l=0;
 o=0;
 myGLCD.clrRow(0);
 loop();

 };

void Purgar()
{
 digitalWrite(en, LOW);
 myGLCD.clrScr();
 myGLCD.invertText(true);
 myGLCD.print(menu_2_1, CENTER, 0);
 myGLCD.invertText(false);
 myGLCD.print("PURGANDO...", CENTER, 24);
 myGLCD.print("fin = >||", CENTER, 40);
  for (int j=0; j<=10000; j++)
   {
   TrasIR();
  if (p==0){
```

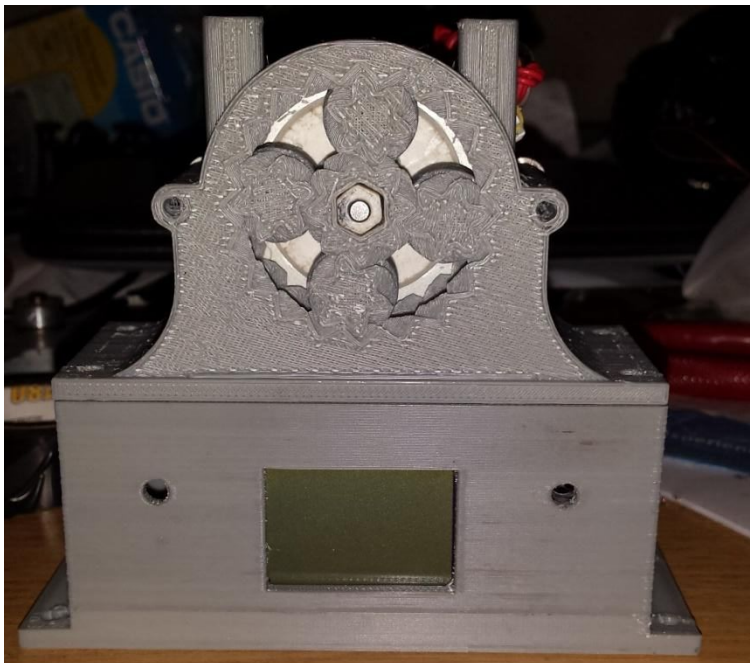
```
   digitalWrite(stp, HIGH);
   delay(10);
   digitalWrite(stp, LOW);
   delay(10);
  }else{
    j=10000;
  };
 };
 digitalWrite(ms1, LOW);
 Ms1=0;
 myGLCD.setFont(SmallFont);
 digitalWrite(en, HIGH);
 ini=2;
 p=0;
 s=0;
 t=0;
 l=0;
 myGLCD.clrRow(0);
 loop();
};

void Prueba()
{
 pa=0;
 ti=0;
 if(del==0){
  del=50;
 };
 digitalWrite(en, LOW);
 myGLCD.clrScr();
 myGLCD.invertText(true);
 myGLCD.print(menu_2_3, CENTER, 0);
 myGLCD.invertText(false);
    myGLCD.print("tiempo=", LEFT, 40);
    myGLCD.print("presione >||", CENTER, 8);
//tiempo transcurrido
    myGLCD.print("cuando se", CENTER, 16);
//tiempo transcurrido
    myGLCD.print("trasvasen", CENTER, 24);
//tiempo transcurrido
    myGLCD.printNumI(ml, 30, 32, 4, '0');
    myGLCD.print("ml", 60, 32); //tiempo
transcurrido

 tiempoa=millis()/1000;
  for (int j=0; j<=10000; j++)
   {
   TrasIR();
  if (p==0){
   digitalWrite(stp, HIGH);
   pa++;
   delay(del);
   digitalWrite(stp, LOW);
   pa++;
   delay(del);
   ti=millis()/1000-tiempoa;
   myGLCD.printNumI(ti, RIGHT, 40, 4, '0');
//tiempo transcurrido
   }else{
    j=10000;
   myGLCD.printNumI(ti, RIGHT, 40, 4, '0');
//tiempo transcurrido
   };
};
 digitalWrite(ms1, LOW);
 Ms1=0;
 digitalWrite(en, HIGH);
 ini=2;
 p=0;
 s=0;
 t=0;
 l=0;
 o=0;
 myGLCD.clrRow(0);
 loop();
};
```

Con el ultimo código mencionado se obtiene una recepción de datos atravez del pines 6 siendo esta información recibida atravez del módulo de mando IR.

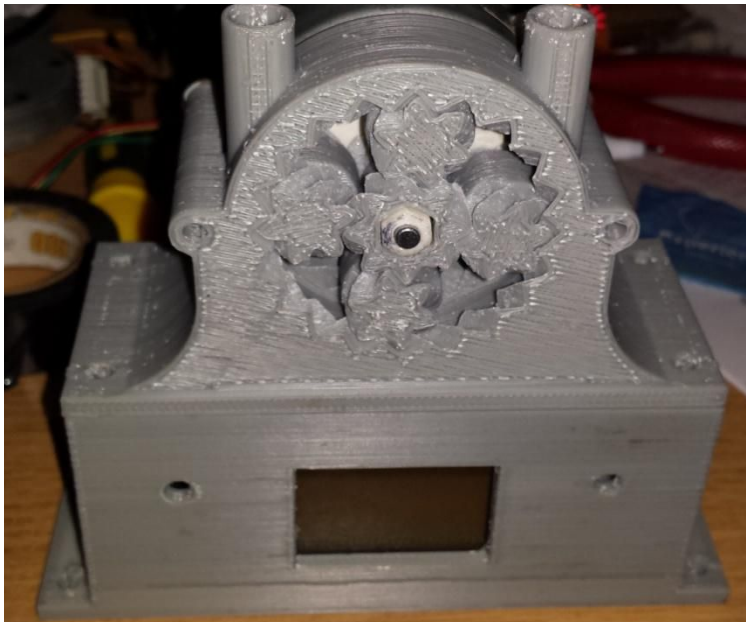
La salida de datos de este controlador está a cargo de los pines 2,3 y 4,5 quienes en primer caso informan sobre velocidad y sentido y en segundo caso activan o desactivan los modos de espera y de micro paso.

Y finalmente la salida de datos hacia la pantalla lcd es enviada desde los pines del 9 al 12, generando estos los gráficos y el menú en pantalla.

Imágenes de pripump terminada




LAB FD
Ruben Humar 2016



Referencias

Drmn4ea http://www.thingiverse.com/thing:454702

Emmett http://www.thingiverse.com/thing:53451

https://es.wikipedia.org/wiki/OpenSCAD