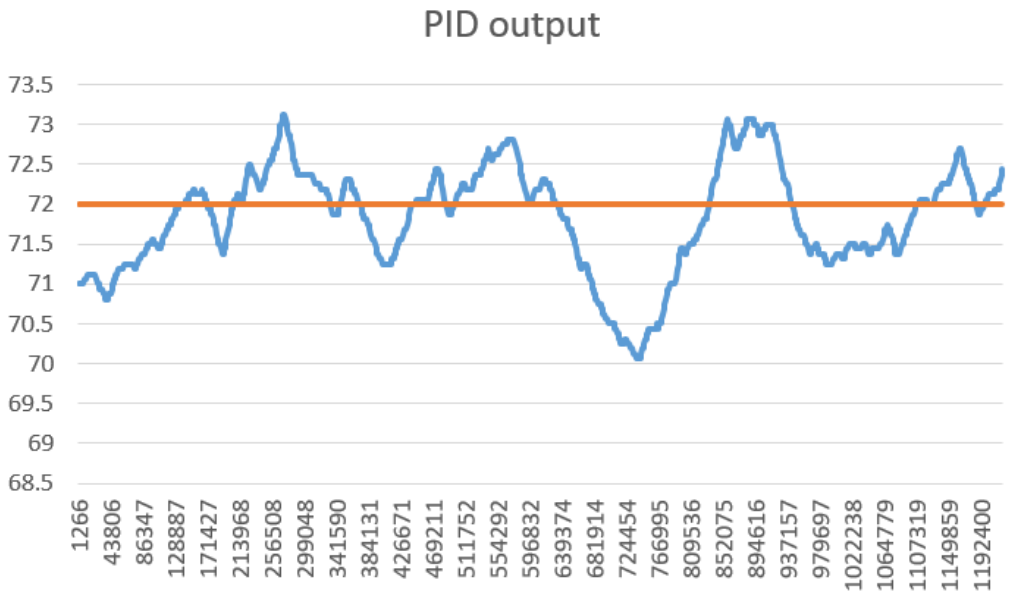
Montesclaros, Ray Mart 26 November 2019

Tirona, Victor Lorenzo ELC151.2

With our parameters, a mean square difference of **0.470469** was obtained.



Appendix

```
#include <OneWire.h>
#include <DallasTemperature.h>
// constants for DS18B20 temp sensor
#define ds18Sig 3
// Setup a oneWire instance to communicate with any OneWire devices
// (not just Maxim/Dallas temperature ICs)
OneWire oneWire(ds18Sig);
// Pass our oneWire reference to Dallas Temperature.
DallasTemperature tempDS18(&oneWire);
int relay = 9; //PWM pin
int ds18Vcc = 4;
int ds18Gnd = 5;
//PID constants
double kp = 70;
double ki = 0.001;
double kd = 3;
long sampleTime = 1000; // in milliseconds
float targetTemp = 72;
float lowError = -1;
float highError = 1;

unsigned long currentTime, previousTime = 0;
float currTemp;
double elapsedTime;
double lastError;
double PIDout;
double proError, intError, derError;
```

```
void setup() {
  Serial.begin (9600);
  pinMode (relay, OUTPUT);
  pinMode (ds18Vcc, OUTPUT);
  pinMode (ds18Gnd, OUTPUT);
  pinMode (ds18Sig, INPUT);
  digitalWrite(ds18Vcc, HIGH);
  digitalWrite(ds18Gnd, LOW);
}

void loop() {
  currTemp = readTempDS18(); //read temp
  currentTime = millis(); //get current time
  if ((currentTime - previousTime) > sampleTime) {
    PIDout = computePID(currTemp);
    // digitalWrite(relay, HIGH);
    double PIDout_map = map(PIDout, 0, 255, 140, 190); //control the PWM output based on PID value
    analogWrite (relay, PIDout_map);

    //Serial.print(",");
//    Serial.print("Target Temp: ");
    Serial.print(targetTemp);
    Serial.print(",");
//    Serial.print("Current Temp: ");
    Serial.print(currTemp);
    Serial.print(",");
    Serial.print(PIDout);
    Serial.print(",");
    Serial.print(PIDout_map);
```

```
    Serial.print(",");
    Serial.println(currentTime);
//    Serial.println("----------");
  }
}
double computePID(double input) {
  // currentTime = millis(); //get current time
  elapsedTime = (double)(currentTime - previousTime); //compute time elapsed from previous computation
  proError = targetTemp - input; // determine error
  if (lowError < proError && proError < highError)
    intError += proError * elapsedTime; // compute integral
  derError = (proError - lastError) / elapsedTime; // compute derivative
//  Serial.println("----------");
//  Serial.print("kp*proError: ");
//  Serial.print(kp * proError);
//  Serial.print(", ");
//  Serial.print("ki*intError: ");
//  Serial.print(ki * intError);
//  Serial.print(", ");
//  Serial.print("kd*derError: ");
//  Serial.println(ki * derError);
//  Serial.println("----------");
  double out = kp * proError + ki * intError + kd * derError; //PID output
  lastError = proError; //remember current error
  previousTime = currentTime; //remember current time
  if (out < 0) out = 0;
  else if (out > 255) out = 255;
  return out; //have function return the PID output
```

```
}
float readTempDS18() {
  tempDS18.requestTemperatures();
  delay(150);
  return tempDS18.getTempCByIndex(0);
}
```