1) **Código de Arduino:**

```
/*                TIOA7 Frequency = 20 KHz over pin 3                          */
#include "pendulo.h"
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
#include <Wire.h>

#define PROM_IT 30
#define MAX_FREQ_COUNT 840000
#define ALPHA_COMP_FILTER 0.98
#define TS  0.01

// global variables
Adafruit_MPU6050 IMU;
sensors_event_t a, g, temp;
float theta = 0;
int theta_ref = 0, phi_ref = 0;
float e[2];
float u[2];

void PWM_Waveform_gen_setup(void);

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
  // -------------------------PWM Waveform Gen----------------------------------
  PWM_Waveform_gen_setup();
  // ------------------------ PWM ----------------------------------
  Serial.begin(57600);
  // -------------------Try to initialize IMU ---------------------------------
  if (!IMU.begin()) {
    //Serial.println("Failed to find MPU6050 chip");
    while (1) {
      delay(10);
    }
  }

  Serial.println("MPU6050 Found!");
  IMU.setAccelerometerRange(MPU6050_RANGE_8_G);
  IMU.setGyroRange(MPU6050_RANGE_250_DEG);
  IMU.setFilterBandwidth(MPU6050_BAND_21_HZ);
  delay(50);
  // ------------------------------- IMU ----------------------------------
  // Initial Conditions
  e[0] = 0;
  e[1] = 0;
  u[0] = 0;
```

```
  u[1] = 0;
  //--------------------------------------------------------------------------

}

void TC7_Handler() {
  static uint32_t Count;
  TC2->TC_CHANNEL[1].TC_SR;
  Count++;
  if (Count == 25) {
    digitalWrite(LED_BUILTIN, !digitalRead(LED_BUILTIN));  // Toggle every 1 Hz
    Count = 0;
  }
}

void loop() {
  // ------------------------- VARIABLES
-----------------------------------------------
  float pote_val_send = 0, phi_ref_send = 0,  t1 = 0, t2 = 0;
  float servo_send = 0;
  float theta_ref_send = 0;

  IMU.getEvent(&a, &g, &temp);

  // ------------------------- VARIABLES END
-----------------------------------------
  theta_ref_send = theta_ref * 1.0;

  t1 = micros();
  //---------------------------- CODE
-----------------------------------------------------
  // -------------------- Data adquisition (measurements)
  for(uint8_t i = 0; i < PROM_IT; i++){
    pote_val_send = pote_val_send  + analogRead(A9) * 1.0;
  }
  pote_val_send = pote_val_send / PROM_IT ;
  // Phi reading ready -----------------

  //using complementary filter to get theta IN RAD !!!
  theta = ALPHA_COMP_FILTER * (theta + g.gyro.x * TS) + (1-ALPHA_COMP_FILTER)*
atan2(a.acceleration.y, a.acceleration.z);
  // Theta reading ready ---------------

  // Get phi reference from serial port
  if (Serial.available() > 0) {
    // read the incoming byte:
    phi_ref = Serial.read();
```

```
  }
  //convert reading in negative number if necesary
  if (phi_ref >127)phi_ref = phi_ref - 256;
  phi_ref_send = phi_ref * 1.0;
  // PHI ref Ready ------------

  //-------------------- error calc
  e[1] = theta_ref - theta;

  // ------------------- Controller
  //PD_CONTROL(e, u, TS); //uncomment this line for a PD controller
  //PI_CONTROL(e, u, TS); //uncomment this line for a PI controller
  u[1] = Prop_Control(e[1]); //uncomment this line for a Proportional controller

  // ------------------- Actuator
  servo_send = u[1] * (180 / M_PI);
  //MOVE SERVO (control action) DEGREE !!!!!!
  // This should be the last thing to do
  TC2->TC_CHANNEL[1].TC_RA = SERVO_PWM_REF + deg2servoPWM( (int)servo_send ) +
deg2servoPWM(phi_ref);

  //Send to matlab
  Serial.write("abcd");
  Serial.write((uint8_t *) &pote_val_send,4);//
  Serial.write((uint8_t *) &theta,4);//rad !!
  Serial.write((uint8_t *) &servo_send,4);
  Serial.write((uint8_t *) &phi_ref_send,4);//
  //----------------------------- CODE END
-------------------------------------------------
  t2 = micros() - t1;
  delay(10 - t2/1000);
}

void PWM_Waveform_gen_setup(void){
  // ---------- PWM - Waveform
-------------------------------------------------------------------------------
  PMC->PMC_PCER1 |= PMC_PCER1_PID34;                        // TC7 power ON - Timer
Counter 2 channel 1 IS TC7 - See page 38

  PIOC->PIO_PDR |= PIO_PDR_P28;                             // The pin is no more
driven by GPIO
  PIOC->PIO_ABSR |= PIO_PC28B_TIOA7;                        // Periperal type B  - See
page 859

  TC2->TC_CHANNEL[1].TC_CMR = TC_CMR_TCCLKS_TIMER_CLOCK1  // MCK/2, clk on rising
edge
                            | TC_CMR_WAVE                 // Waveform mode
```

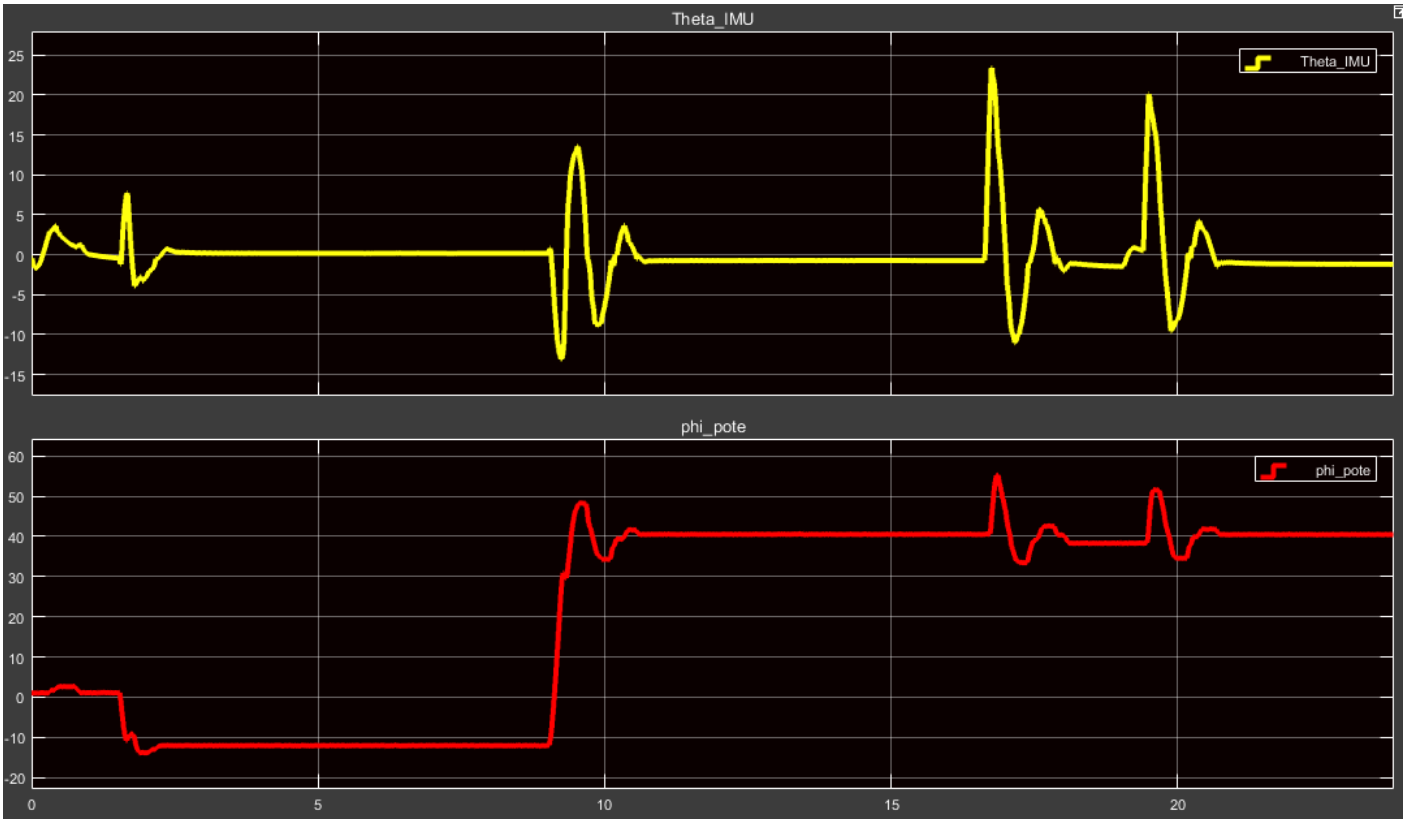
```
                              | TC_CMR_WAVSEL_UP_RC        // UP mode with
automatic trigger on RC Compare
                              | TC_CMR_ACPA_CLEAR         // Clear TIOA7 on RA
compare match  -- See page 883
                              | TC_CMR_ACPC_SET;          // Set TIOA7 on RC
compare match (Counter clear on RC compare match)

  TC2->TC_CHANNEL[1].TC_RC = MAX_FREQ_COUNT;  //<******************** for a 50Hz
Frequency = (Mck/2) / 50 Hz =  TC_RC , Mck = 84e6
  TC2->TC_CHANNEL[1].TC_RA = SERVO_PWM_REF;  //<********************   Duty cycle =
(TC_RA/TC_RC) * 100  % this is my ZERO deg

  TC2->TC_CHANNEL[1].TC_IER = TC_IER_CPCS;                  // Interrupt on RC
compare match
  NVIC_EnableIRQ(TC7_IRQn);

  TC2->TC_CHANNEL[1].TC_CCR = TC_CCR_SWTRG | TC_CCR_CLKEN; // Software trigger TC7
counter and enable
  // ---------- PWM - Waveform
--------------------------------------------------------------------------------
```

2) Respuestas del sistema real



En amarillo esta el angulo theta medido con la IMU, mientras que en rojo se muestra el angulo phi medido con el potenciometro. Se muestra como el controlador sigue llevando el ángulo del péndulo a cero aun cuando el ángulo del brazo cambia de valor (primero a -10 grados y luego a 40 grados)