Universidad del Valle de Guatemala
Departamento de Ingeniería Mecatrónica y Electrónica
Digital 2

Diego Mencos
Carné 18300

“Pseudo-Código PIC”

```
//Universidad del Valle de Guatemala
//Digital 2
//Diego Mencoss 18300


#define _XTAL_FREQ 4000000

#include <xc.h>
#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
#include "MPU6050.h"
#include "I2C.h"
#include "USART.h"

//Definimos variables
char Ay, Az;
char Ax;
uint8_t contador;
uint8_t valorRX;


// BEGIN CONFIG
#pragma config FOSC = INTRC_NOCLKOUT  // Oscillator Selection bits (HS oscillator)
#pragma config WDTE = OFF // Watchdog Timer Enable bit (WDT enabled)
#pragma config PWRTE = OFF // Power-up Timer Enable bit (PWRT disabled)
#pragma config BOREN = ON // Brown-out Reset Enable bit (BOR enabled)
#pragma config LVP = OFF // Low-Voltage (Single-Supply) In-Circuit Serial Programming
Enabl e bit (RB3 is digital I/O, HV on MCLR must be used for programming)
#pragma config CPD = OFF // Data EEPROM Memory Code Protection bit (Data
EEPROM code protection off)
#pragma config WRT = OFF // Flash Program Memory Write Enable bits (Write protection
off; all program memory may be written to by EECON control)
#pragma config CP = OFF // Flash Program Memory Code Protection bit (Code protection
off)
//END CONFIG



//Prototipos de funciones
```

```
void setup(void);
void __interrupt() isr(void);



//Configuramos interrupciones
//Interrupciones

void __interrupt() isr(void) {
    if (PIR1bits.RCIF == 1) {
        valorRX = UART_get_char(); //Aqui recibimos el dato de la recepcion
        PIR1bits.RCIF = 0;
        // PORTD = valorRX;
    }
}

void main() {
    setup();
    MPU6050_init();
    UART_config();

    while (1) {

        if (valorRX == 0b11110010) {//LED ROJO
            PORTCbits.RC2 = 0;
            RCREG = 0; //Ponemos en 0 para que no vuelva a entrar al if
            valorRX = 0;
        } else if (valorRX == 0b11111110) { //LED ROJO
            PORTCbits.RC2 = 1;
            RCREG = 0; //Ponemos en 0 para que no vuelva a entrar al if
            valorRX = 0;
        } else if (valorRX == 0b11110111) {//LED VERDE
            PORTCbits.RC1 = 0;
            RCREG = 0; //Ponemos en 0 para que no vuelva a entrar al if
            valorRX = 0;
        } else if (valorRX == 0b11111011) { //LUZ VERDE
            PORTCbits.RC1 = 1;
            RCREG = 0; //Ponemos en 0 para que no vuelva a entrar al if
            valorRX = 0;
        }

        UART_send_char(0);
        __delay_ms(200);

        Ax = MPU6050_get_Ax(); // Acelerometro eje x
        PORTD=Ax;
        UART_send_char('a');
```

```
        __delay_ms(200);
        UART_send_char(Ax);
        __delay_ms(200);

        Ay = MPU6050_get_Ay(); // Acelerometro eje y
        UART_send_char('b');
        __delay_ms(200);
        UART_send_char(Ay);
        __delay_ms(200);

        Az = MPU6050_get_Az(); // Acelerometro eje z
        UART_send_char('c');
        __delay_ms(200);
        UART_send_char(Az);
        __delay_ms(200);




        //  Gx = MPU6050_get_Gx(); // Giroscopio eje x
        //  Gy = MPU6050_get_Gy(); // Giroscopio eje y
        //  Gz = MPU6050_get_Gz(); // Giroscopio eje z


    }
}

void setup(void) {
    ANSEL = 0;
    ANSELH = 0;
    TRISA = 0;
    TRISA = 0;
    PORTA = 0;
    TRISB = 0;
    PORTB = 0;
    TRISC = 0;
    PORTC = 0;
    TRISD = 0;
    PORTD = 0;
    TRISE = 0;
    PORTE = 0;
    contador = 0;
    I2C_Master_Init(100000);
    INTCONbits.GIE = 1; //Habilitamos las interrupciones
    INTCONbits.PEIE = 1; //Habilitamos las interrupciones perifericas
    PIR1bits.RCIF = 0; //Apagamos la bandera del RX
}
```

“Pseudo-Código Esp32”

```
#include <TFT_eSPI.h>
#include <SPI.h>
#include <Wire.h>
#include <WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"


#define WLAN_SSID       "TURBONETT_F895FB"
#define WLAN_PASS       "795C90366B"
#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883
#define AIO_USERNAME    "men18300"
#define AIO_KEY         "aio_jdTp83i1XFRIGRVXa8wFhdnx1bGO"

#define BUTTON1   35
#define BUTTON2   0
#define LED_GREEN 19
#define LED_RED   21
#define RXD2 16
#define TXD2 17

#define FF17 &FreeSans9pt7b
#define FF21 &FreeSansBold9pt7b
#define ROW1 0,16
#define ROW2 0,38
#define ROW3 0,60
#define ROW4 0,82
#define ROW5 0,104
#define ROW6 0,126


WiFiClient client;

// Setup the MQTT client class by passing in the WiFi client and MQTT server and login
details.
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT,
AIO_USERNAME, AIO_KEY);
Adafruit_MQTT_Subscribe ledControl = Adafruit_MQTT_Subscribe(&mqtt,
AIO_USERNAME "/feeds/fucks");
Adafruit_MQTT_Subscribe ledVerde = Adafruit_MQTT_Subscribe(&mqtt,
AIO_USERNAME "/feeds/ledverde");
Adafruit_MQTT_Publish temperature = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME
"/feeds/temperature");
```

```
////////////////////////////////////////////////////////////////////////////////////////
Adafruit_MQTT_Publish AxADAFRUIT = Adafruit_MQTT_Publish(&mqtt,
AIO_USERNAME "/feeds/Ax");
Adafruit_MQTT_Publish AyADAFRUIT = Adafruit_MQTT_Publish(&mqtt,
AIO_USERNAME "/feeds/Ay");
Adafruit_MQTT_Publish AzADAFRUIT = Adafruit_MQTT_Publish(&mqtt,
AIO_USERNAME "/feeds/Az");
////////////////////////////////////////////////////////////////////////////////////////

//HardwareSerial Serial2(2);
void setup()
{
  pinMode(BUTTON1, INPUT_PULLUP);
  pinMode(BUTTON2, INPUT_PULLUP);
  pinMode(LED_GREEN, OUTPUT);
  pinMode(LED_RED, OUTPUT);

  Serial2.begin(9600, SERIAL_8N1, RXD2, TXD2);
  Serial.begin(115200);


  while (!Serial);
  while (!Serial2);

  // Connect to WiFi access point.
  Serial.println(); Serial.println();
  Serial.print("Connecting to ");
  Serial.println(WLAN_SSID);


  WiFi.begin(WLAN_SSID, WLAN_PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println();
  Serial.println("WiFi connected");
  Serial.println("IP address: "); Serial.println(WiFi.localIP());



  mqtt.subscribe(&ledControl);
  mqtt.subscribe(&ledVerde);
}
```

```
float ambientCelsius = 0.0;
float objectCelsius = 0.0;

void loop()
{


  MQTT_connect();

  Adafruit_MQTT_Subscribe *subscription;

  while ((subscription = mqtt.readSubscription(5000))) {
    if (subscription == &ledControl) {
      Serial.print(F("Got: "));
      Serial.println((char *)ledControl.lastread);

      if (!strcmp((char*) ledControl.lastread, "ON")) {
        digitalWrite(LED_GREEN, HIGH);
        Serial2.write(0x0D);

      }
      else {
        digitalWrite(LED_GREEN, LOW);
        Serial2.write(0x0C);
      }
    }
    else if (subscription == &ledVerde) {
      Serial.print(F("Got: "));
      Serial.println((char *)ledVerde.lastread);

      if (!strcmp((char*) ledVerde.lastread, "ON")) {
        digitalWrite(LED_RED, HIGH);
        Serial2.write(57);

      }
      else {
        digitalWrite(LED_RED, LOW);
        Serial2.write(58);
      }
    }



  }
```

```
Serial.print(F("\nSending Aceloremeter value "));
int conteo=Serial2.read();
Serial.print(conteo);
delay (200);
Serial.print("…");

//////////////////////////////////////////////////////////

if (conteo == 6) {
  int Ax = Serial2.read();
  if (!AxADAFRUIT.publish(Ax)) {
    Serial.println(F("Failed sending Ax"));
  }
  else {
    Serial.println(F("OK! sending Ax "));
    Serial.print(Ax);
  }



}
else if (conteo == 24) {
  int Ay = Serial2.read();
  if (!AyADAFRUIT.publish(Ay)) {
    Serial.println(F("Failed sending Ay" ));
  }
  else {
    Serial.println(F("OK! sending Ay"));
    Serial.print(Ay);
  }




}
else if (conteo == 30) {
  int Az = Serial2.read();
  if (!AzADAFRUIT.publish(Az)) {
    Serial.println(F("Failed sending Az"));

  }
  else {
    Serial.println(F("OK! sending Az"));
    Serial.print(Az);
```

```
    }



  }



void MQTT_connect()
{
  int8_t ret;

  // Stop if already connected.
  if (mqtt.connected()) {
    return;
  }

  Serial.print("Connecting to MQTT… ");

  uint8_t retries = 3;

  while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected
    Serial.println(mqtt.connectErrorString(ret));
    Serial.println("Retrying MQTT connection in 5 seconds…");
    mqtt.disconnect();

    delay(5000); // wait 5 seconds

    -retries;
    if (retries == 0) {
      // basically die and wait for WDT to reset me
      while (1);
    }
  }

  Serial.println("MQTT Connected!");
}
```