ANEXOS

Anexo 1: COMANDOS AT PARA CONFIGURAR EL MÓDULO BLUETOOTH RN42

COMANDOS AT		CONFIGURACIÓN POR DEFECTO
S7,<1,0>	- 7 bit data mode enable/disable	0= disabled
SA,<1,0>	- Authentication enable/disable	0= disabled
SB,<timer>	- Send BREAK	Not Applicable
SC,<hex word>	- Service Class	0x0000= unknown
SD,<hex word>	- Device Class	0x1F00= undefined
SE,<1,0>	- Encryption enable/disable	0=disabled
SF,1	- Factory Defaults	
SI,<hex word>	- Inquiry Scan window	0x0200
SJ,<hex word>	- Page Scan window	0x0200
SL,<E,O,N>	- Parity	N=None
SM,<0,1,2,3,4,5>	- Mode (0=Slave,1=mstr,2=trig, 3=auto, 4=DTR, 5=ANY)	0=Slave
SN,<text>	- Name	FireFly-xxxx
SO,<text>	- Connect/Disconnect Status String	NULL= no status string
SP,<text>	- Pin Code	1234
SR,<adr>	- Remote Address (SR,Z to remove)	NONE SET
SS,<text>	- Service Name	SPP
ST,<num>	- Config Timer	60 seconds
SU,<rate>	- Baudrate	115K
SW,<hex>	- SNIFF rate	0x0000=disabled
SX,<1,0>	- Bonding	0=disabled
S~,<0-4>	- Profile setting 0=SPP, 1=DCE, 2=DTE, 3=MDM, 4=DUN&SPP	0 = SPP
SZ,<num>	- Raw Baudrate	
S?,<0,1>	- Enable /Disable Role Switch	0=disabled

Anexo 2: CIRCUITO Y MONTAJE DE LOS SENSORES CINEMÁTICOS

ACELERÓMETRO

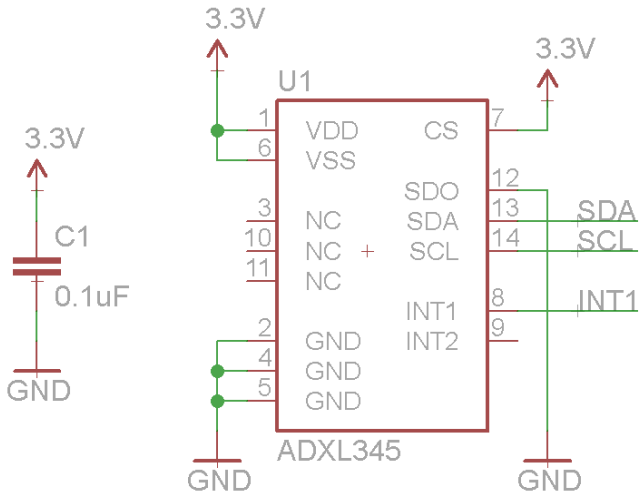


Figura 1: Circuito de conexión acelerómetro ADXL345

Por medio de los pines 13 y 14 del acelerómetro se realiza la comunicación IIC.

GIRÓSCOPO

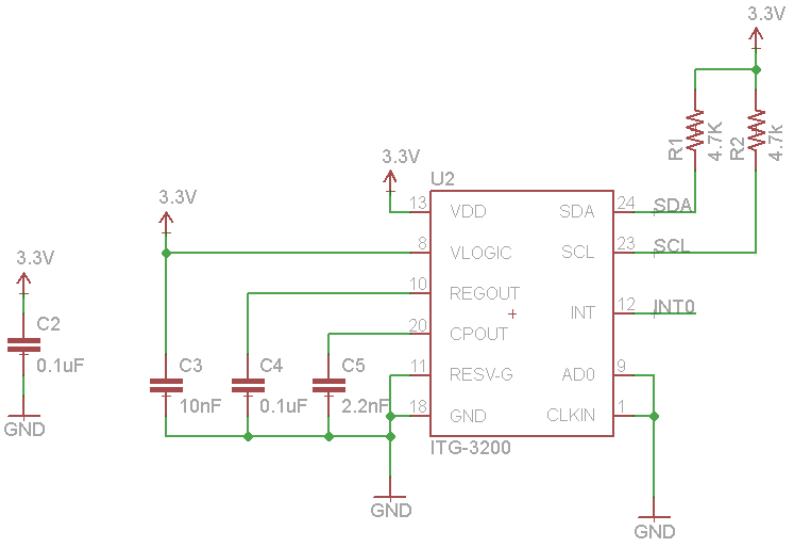


Figura 2: Circuito de conexión Giróscopo ITG3200

El giróscopo empea los pines 23 y 24 para su comunicación IIC con el microcontrolador.

A continuación se presenta el módulo 10121 del acelerómetro ADXL345 y el giróscopo desarrollado por Sparkfun en el programa de diseño EAGLE, con los cuales se trabajó para la obtención de los datos para calcular el ángulo entre los sensores.

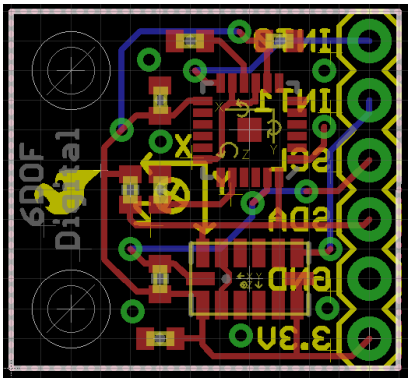


Figura 3: Circuito impreso de sensores cinemáticos

(FUENTE: http://www.sparkfun.com/products/10121**)**

Anexo 3: MONTAJE DEL MÓDULO BLUETOOTH RN 41

El siguiente es el hardware del módulo Bluetooth RN 41 donde se muestra la alimentación a 3.3 voltios y las respectivas conexiones hacia el microcontrolador por medio de los pines UARTTX y UARTRX. También se muestran los leds de conexión y de estado.

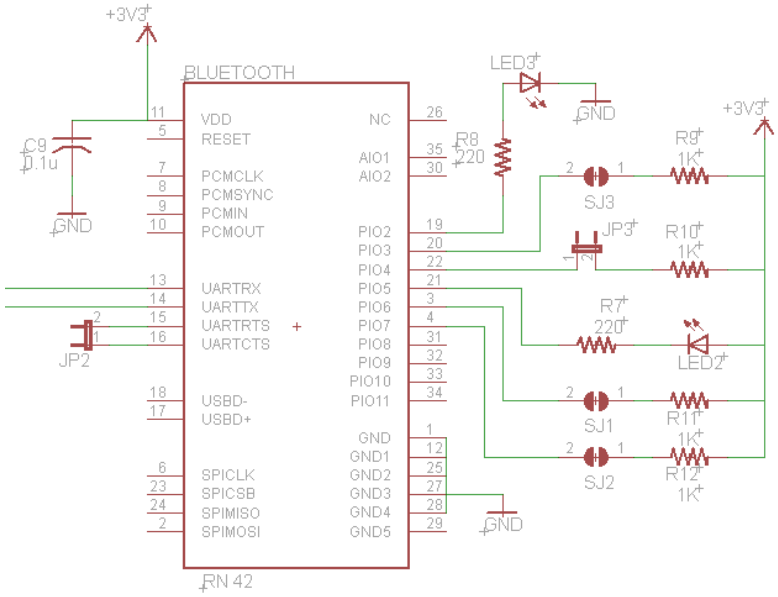


Figura 4: Circuito de conexión Bluetooth RN 42

Anexo 4: BOARD DEL PROTOTIPO DEL GONIÓMETRO

A continuación de muestra la capa frontal de la tarjeta electrónica del prototipo del goniómetro digital.

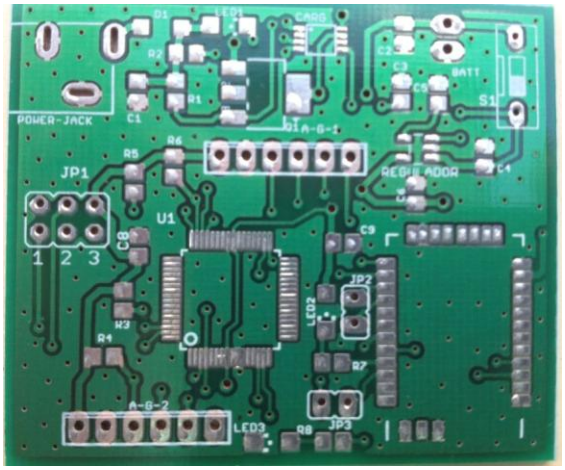


Figura 5: Capa frontal Board física**.**

Esta es la imagen de la parte trasera de la board del goniómetro. Aquí se visualiza claramente la gran capa de ¨tierra¨ con la que disminuimos en gran medida el ruido electromagnético.

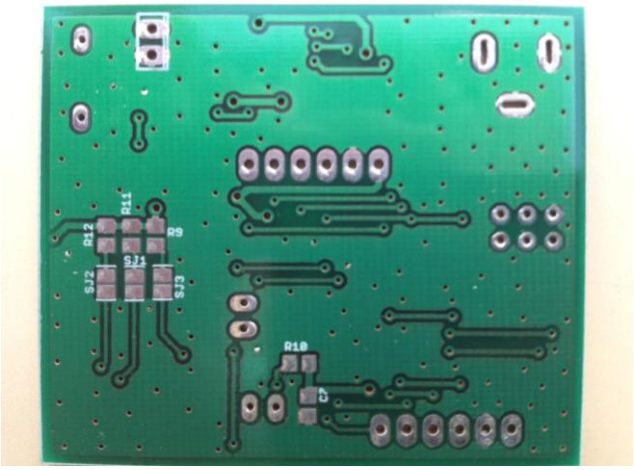


Figura 6: Capa trasera Board física.

Anexo 5: DIAGRAMA DE FLUJO DEL SOFTWARE

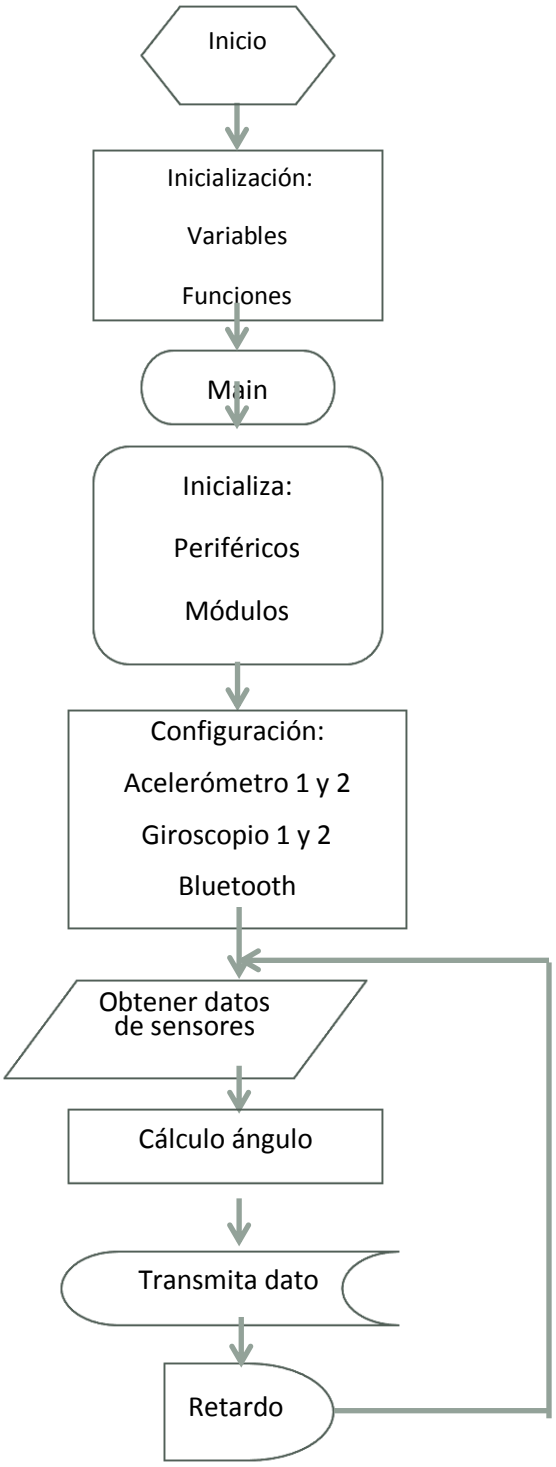
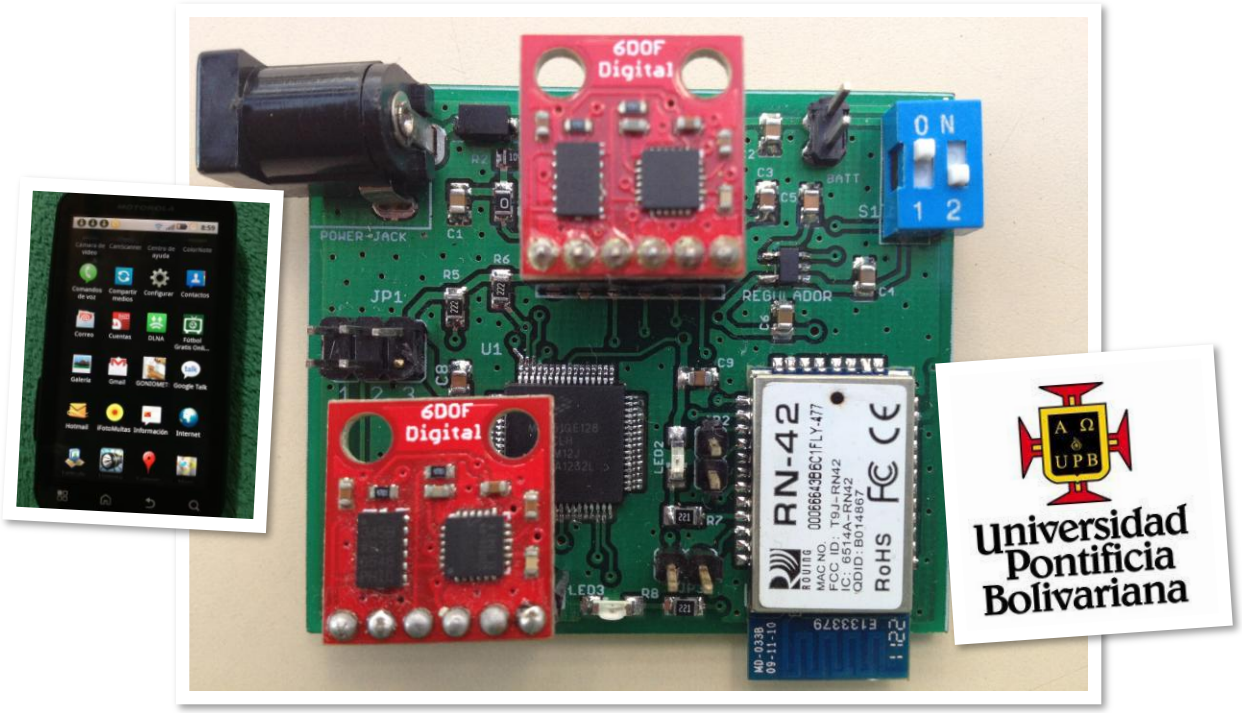


Figura 7: Diagrama de flujo del Software

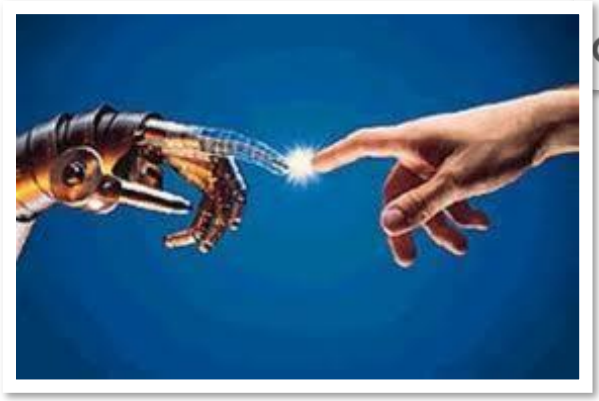
Anexo 6: MANUAL DE USUARIO DE LA APLICACIÓN PARA DISPOSITIVOS ANDROID



Goniómetro digital

basado en sensores cinemáticos con comunicación inalámbrica en tiempo real

A continuación se presentan las instrucciones para el correcto funcionamiento del prototipo del "GONIÓMETRO DIGITAL BASADO EN SENSORES CINEMÁTICOS CON COMUNICACIÓN INALÁMBRICA EN TIEMPO REAL" desarrollado como proyecto de grado en el grupo de bioingeniería de la Universidad Pontificia Bolivariana.



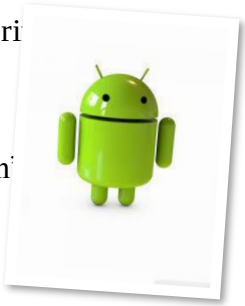
CONTENIDO

Manual del usuario

Goniómetro Digital

Configuración del dispositivo móvil

La aplicación del goniómetro digital está desarrollada para dispositivos móviles con sistema operativo ANDROID.



Pulse el botón de inicio para ir a la pantalla pri
teléfono.

Pulse el botón "Menú", luego "Configuración
continuación, "Aplicaciones".

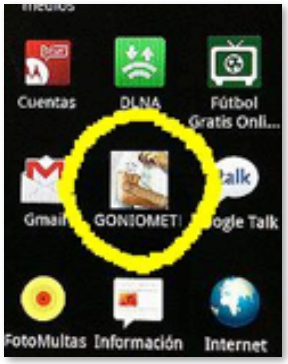
Activar la opción de "Fuentes desconocidas".

- ✓ Luego toque la opción "Desarrollo".
- ✓ Active las opciones de "Depuración de USB" y "Permanecer actvo".

Descargar la aplicación

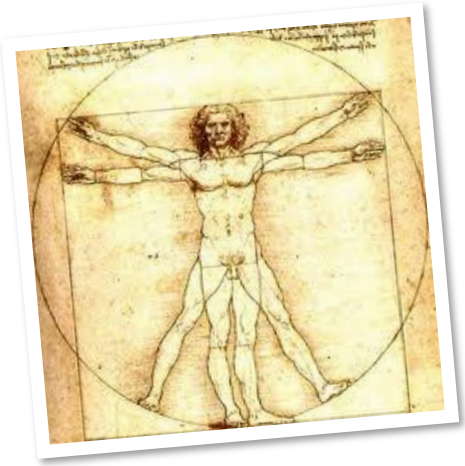
Se recomienda al usuario tener instalada una aplicación para lectura de códigos de barras.

- ✓ Ejecutar la aplicación de lectura de códigos.
- ✓ Escanear el siguiente código QR.
- ✓ Dar click en "Permitir descarga". La aplicación tiene el siguiente ícono.
- ✓ "Abrir" aplicación.



Alimentación y encendido

- ✓ Conecte la batería del goniómetro en su lugar. El dispositivo cuenta con un sistema de carga para la batería de Litio. En caso de que ésta se encuentre descargada, conecte un regulador de voltaje de 12 voltios al dispositivo.
- ✓ Ubique el dispositivo a un lado de la articulación a medir. El sensor 2 ubíquelo al otro lado.



Encienda el goniómetro desde el interruptor 1 azul de la parte superior derecha.

- ✓ Encienda el módulo Bluetooth de
- ✓ Ejecute la aplicación.
- ✓ En la pantalla principal de la aplicación, presione “Conectar BT”.

Nota: A partir del momento en que se enciende el goniómetro digital, se cuenta con un minuto para emparejar el dispositivo con el teléfono. De lo contrario, hay que apagar y encender nuevamente el goniómetro.

Uso de la aplicación

Pantalla Principal

Allí se presentan las siguientes opciones:

- ✓ Articulaciones:
- ✓ Terapias:
- ✓ Instrucciones:
- ✓ Conectar BT: Para conectar el teléfono con el dispositivo.
- ✓ Desconectar BT: Para terminar la conexión.
- ✓ Salir: Presiona para cerrar la aplicación.

Pantalla Articulaciones

Presenta los valores de los ángulos máximos y mínimos alcanzados por las principales articulaciones del cuerpo humano.

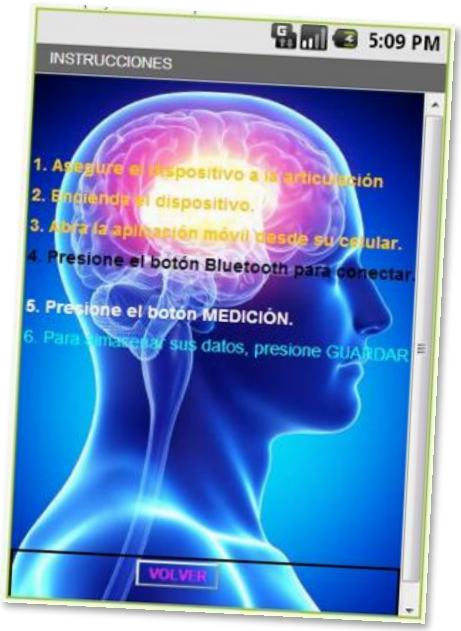
Desde aquí puede ir a "MEDICIÓN" para iniciar la medida de los ángulos.

Pantalla Terapias

Muestra las mediciones de las últimas cinco sesiones. Tanto su ángulo mayor, como menor.

Seleccione la sesión que desee guardar.

Al dar "REINICIAR" se borran los datos guardados.



Pantalla Instrucciones

Muestra al usuario una guía rápida para su operación.

Al presionar "VOLVER", la aplicación regresa al inicio.

Pantalla Medición

En la parte superior se visualiza el ángulo que el dispositivo está calculando en ese momento.

En la parte inferior se observan las mediciones máximas y mínimas tomadas durante esa sesión.

Al presionar "GUARDAR DATOS", la aplicación muestra las terapias donde sarán almacendos los datos obtenidos.

Finalizar la aplicación

- ✓ Vaya a la pantalla principal
- ✓ Presione "DESCONECTAR BT".
- ✓ Pulse "SALIR".

Anexo 7: FUNCIONES DEL PROGRAMA PARA LA COMUNICACIÓN BLUETOOTH

```
FUNCION INICIALIZACIÓN MÓDULO SCI */
void ini_sci(){
          SCI1C1=0x00;
       SCI1C2=0x0C;
       SCI1BD=0x0D;
}

/* FUNCION CONFIGURACIÓN BLUETOOTH  */
void config_bt(void){
          SCI1D=0x53;                  //S    **<S7,1>**
    while(SCI1S1_TC==0);
          SCI1D=0x37;                  //7
    while(SCI1S1_TC==0);
          SCI1D=0x2C;                  //,
    while(SCI1S1_TC==0);
          SCI1D=0x31;                  //1
        while(SCI1S1_TC==0);

          SCI1D=0x53;                  //S    **<SB,37>**
    while(SCI1S1_TC==0);
          SCI1D=0x42;                  //B
    while(SCI1S1_TC==0);
          SCI1D=0x2C;                  //,
    while(SCI1S1_TC==0);
          SCI1D=0x33;                  //3   37 ms
    while(SCI1S1_TC==0);
          SCI1D=0x37;                  //7
    while(SCI1S1_TC==0);

          SCI1D=0x53;                  //S    **<SC,0002>**
    while(SCI1S1_TC==0);
          SCI1D=0x43;                  //C
    while(SCI1S1_TC==0);
          SCI1D=0x2C;                  //,
    while(SCI1S1_TC==0);
          SCI1D=0x30;                  //0
    while(SCI1S1_TC==0);
          SCI1D=0x30;                  //0
    while(SCI1S1_TC==0);
          SCI1D=0x30;                  //0
    while(SCI1S1_TC==0);
          SCI1D=0x32;                  //2
    while(SCI1S1_TC==0);

          SCI1D=0x53;                  //S     **<SM,0>**
    while(SCI1S1_TC==0);
          SCI1D=0x4D;                  //M
    while(SCI1S1_TC==0);
          SCI1D==0x2C;                 //,
```

```
while(SCI1S1_TC==0);
      SCI1D=0x30;                 //0        esclavo
    while(SCI1S1_TC==0);

      SCI1D=0x53;                 //S     **<SN,GONIOMETRO>**
while(SCI1S1_TC==0);
      SCI1D=0x4E;                 //N
while(SCI1S1_TC==0);
      SCI1D=0x2C;                 //,
while(SCI1S1_TC==0);
      SCI1D=0x71;                 //G
    while(SCI1S1_TC==0);
      SCI1D=0x4F;                 //O
    while(SCI1S1_TC==0);
      SCI1D=0x4E;                 //N
    while(SCI1S1_TC==0);
      SCI1D=0x49;                 //I
    while(SCI1S1_TC==0);
      SCI1D=0x4F;                 //O
    while(SCI1S1_TC==0);
      SCI1D=0x4D;                 //M
    while(SCI1S1_TC==0);
      SCI1D=0x45;                 //E
    while(SCI1S1_TC==0);
      SCI1D=0x54;                 //T
    while(SCI1S1_TC==0);
      SCI1D=0x52;                 //R
    while(SCI1S1_TC==0);
      SCI1D=0x4F;                 //O
    while(SCI1S1_TC==0);

      SCI1D=0x53;                 //S    **<SO,CONNECT>**
while(SCI1S1_TC==0);
      SCI1D=0x4F;                 //O
while(SCI1S1_TC==0);
      SCI1D=0x2C;                 //,
while(SCI1S1_TC==0);
      SCI1D=0x43;                 //C
    while(SCI1S1_TC==0);
      SCI1D=0x4F;                 //O
while(SCI1S1_TC==0);
      SCI1D=0x4E;                 //N
    while(SCI1S1_TC==0);
      SCI1D=0x4E;                 //N
while(SCI1S1_TC==0);
      SCI1D=0x45;                 //E
while(SCI1S1_TC==0);
      SCI1D=0x43;                 //C
    while(SCI1S1_TC==0);
```

```
      SCI1D=0x54;                 //T
   while(SCI1S1_TC==0);
      SCI1D=0x53;                 //S    **<SR,1>**
while(SCI1S1_TC==0);
      SCI1D=0x52;                 //R
while(SCI1S1_TC==0);
      SCI1D=0x2C;                 //,
while(SCI1S1_TC==0);
      SCI1D=0x31;                 //1    write last add
    while(SCI1S1_TC==0);

      SCI1D=0x53;                 //S    **<ST,60>**
while(SCI1S1_TC==0);
      SCI1D=0x54;                 //T
while(SCI1S1_TC==0);
      SCI1D=0x2C;                 //,
while(SCI1S1_TC==0);
      SCI1D=0x36;                 //6
    while(SCI1S1_TC==0);
      SCI1D=0x30;                 //0
while(SCI1S1_TC==0);
      SCI1D=0x53;                 //S    **<SU,96>**
while(SCI1S1_TC==0);
      SCI1D=0x55;                 //U
while(SCI1S1_TC==0);
      SCI1D=0x2C;                 //,
while(SCI1S1_TC==0);
      SCI1D=0x39;                 //9
while(SCI1S1_TC==0);
      SCI1D=0x36;                 //6    9600 baud
while(SCI1S1_TC==0);
      SCI1D=0x53;                 //S    **<SW,0000>**
while(SCI1S1_TC==0);
      SCI1D=0x57;                 /W
while(SCI1S1_TC==0);
      SCI1D=0x2C;                 //,
while(SCI1S1_TC==0);
      SCI1D=0x30;                 //0
while(SCI1S1_TC==0);
      SCI1D=0x30;                 //0
while(SCI1S1_TC==0);
      SCI1D=0x30;                 //0
while(SCI1S1_TC==0);
      SCI1D=0x30;                 //0    disable
while(SCI1S1_TC==0);              }
```

Anexo 8: CÓDIGO DEL PROGRAMA DEL GONIÓMETRO DIGITAL

```
#include <hidef.h>      /* for EnableInterrupts macro */
#include <math.h>
#include "derivative.h"  /* include peripheral declarations */


///VARIABLES   ///
byte bandera1=0;
byte bandera2=0;
byte rx_dato=0;
unsigned long retardo=0;

int accel_xh, accel2_xh;              //REG 0x32    DATAX0
int accel_xl, accel2_xl;              //REG 0x33    DATAX1
int accel_yh, accel2_yh;              //REG 0x34    DATAY0
int accel_yl, accel2_yl;              //REG 0x35    DATAY1
int accel_zh, accel2_zh;              //REG 0x36    DATAZ0
int accel_zl, accel2_zl;              //REG 0x37    DATAZ1

int giro_xh, giro2_xh;                //REG 0x1D    GYRO_XOUT_H
int giro_xl, giro2_xl;                //REG 0x1E    GYRO_XOUT_L
int giro_yh, giro2_yh;                //REG 0x1F    GYRO_YOUT_H
int giro_yl, giro2_yl;                //REG 0x20    GYRO_YOUT_L
int giro_zh, giro2_zh;                //REG 0x21    GYRO_ZOUT_H
int giro_zl, giro2_zl;                //REG 0x22    GYRO_ZOUT_L

int accelx=0, accel2x=0;
int accely=0, accel2y=0;
int accelz=0, accel2z=0;
int girox=0, giro2x=0;
int giroy=0, giro2y=0;
int giroz=0, giro2z=0;
int tmp=0;
int r1=0, r2=0;
int f1=0, f2=0;
int t1=0, t2=0;
int AnguloFinal=0;

///  DECLARACION DE FUNCIONES  ///
void ini_clock();
void ini_perifericos();
void ini_cop();
void ini_sci();
void ini_iic1();
void ini_iic2();
void config_bt();

void enviar_dato_i2c(byte dato);          //funcion escribir un dato
void delay(unsigned long retardo);
void leer_acel1_i2c(byte direccion);      //lee acelerometro 1
```

```
void leer_gyro1_i2c(byte direccion);          //lee giroscopio 1
void escribir_acel1_i2c(byte direccion,byte dato);
void escribir_gyro1_i2c(byte direccion,byte dato);
void leer_acel2_i2c(byte direccion);          //lee acelerometro 2
void leer_gyro2_i2c(byte direccion);          //lee giroscopio 2
void escribir_acel2_i2c(byte direccion,byte dato);
void escribir_gyro2_i2c(byte direccion,byte dato);
int angulo();
int ro1();
int fi1();
int teta1();
int ro2();
int fi2();
int teta2();


void main(void) {

  EnableInterrupts;          /* enable interrupts */

    ini_clock();
    ini_perifericos();
    ini_sci();
    ini_iic1();
    ini_iic2();
    config_bt();

                    //***CONFIG ACELEROMETROS***//
    escribir_acel1_i2c(0x2E,0x80);    //Int_Enable       data-ready
    escribir_acel1_i2c(0x31,0x08);    //Data_Format      Full-resol
    escribir_acel2_i2c(0x2E,0x80);    //Int_Enable       data-ready
    escribir_acel2_i2c(0x31,0x08);    //Data_Format      Full-resol

                //***CONFIG GIROSCOPIOS***//
    escribir_gyro1_i2c(0x16,0x03);    //DLPF Full scale
    escribir_gyro1_i2c(0x17,0x01);    //INT_CONFIG       Raw_rdy_en
    escribir_gyro2_i2c(0x16,0x03);    //DLPF Full scale
    escribir_gyro2_i2c(0x17,0x01);    //INT_CONFIG       Raw_rdy_en

    for(;;) {

      __RESET_WATCHDOG();          /* feeds the dog */

        leer_acel1_i2c(0x32);   //Obtengo datos acelerometro 1
        accel_xh=tmp;
        leer_acel1_i2c(0x33);
        accel_xl=tmp;
        leer_acel1_i2c(0x34);
        accel_yh=tmp;
```

```
leer_acel1_i2c(0x35);
accel_yl=tmp;
leer_acel1_i2c(0x36);
accel_zh=tmp;
leer_acel1_i2c(0x37);
accel_zl=tmp;

leer_gyro1_i2c(0x1D);      //Obtengo datos giroscopio 1
giro_xh=tmp;
leer_gyro1_i2c(0x1E);
giro_xl=tmp;
leer_gyro1_i2c(0x1F);
giro_yh=tmp;
leer_gyro1_i2c(0x20);
giro_yl=tmp;
leer_gyro1_i2c(0x21);
giro_zh=tmp;
leer_gyro1_i2c(0x22);
giro_zl=tmp;

accelx=(accel_xh<<8)|accel_xl;  //Valores en x, y, z   1
accely=(accel_yh<<8)|accel_yl;
accelz=(accel_zh<<8)|accel_zl;
girox=(giro_xh<<8)|giro_xl;
giroy=(giro_yh<<8)|giro_yl;
giroz=(giro_zh<<8)|giro_zl;

leer_acel2_i2c(0x32);    //Obtengo datos acelerometro 2
accel2_xh=tmp;
leer_acel2_i2c(0x33);
accel2_xl=tmp;
leer_acel2_i2c(0x34);
accel2_yh=tmp;
leer_acel2_i2c(0x35);
accel2_yl=tmp;
leer_acel2_i2c(0x36);
accel2_zh=tmp;
leer_acel2_i2c(0x37);
accel2_zl=tmp;

leer_gyro2_i2c(0x1D);    //Obtengo datos giroscopio 2
giro2_xh=tmp;
leer_gyro2_i2c(0x1E);
giro2_xl=tmp;
leer_gyro2_i2c(0x1F);
giro2_yh=tmp;
leer_gyro2_i2c(0x20);
giro2_yl=tmp;
leer_gyro2_i2c(0x21);
```

```
        giro2_zh=tmp;
        leer_gyro2_i2c(0x22);
        giro2_zl=tmp;

        accel2x=(accel2_xh<<8)|accel2_xl;//Valores en x, y, z  2
        accel2y=(accel2_yh<<8)|accel2_yl;
        accel2z=(accel2_zh<<8)|accel2_zl;
        giro2x=(giro2_xh<<8)|giro2_xl;
        giro2y=(giro2_yh<<8)|giro2_yl;
        giro2z=(giro2_zh<<8)|giro2_zl;

                ro1();              //Cálculos
                fi1();
                teta1();
                ro2();
                fi2();
                teta2();

            SCI1D=AnguloFinal;                  //TX Angulo
            SCI1C2_TCIE = 1;                    //habilite int por TXD 1
          do{                                   //espere a que se complete la TX
            }while(bandera1 == 0);
          bandera1 = 0;

          delay(1000);


    } /* loop forever */

}

///***   FUNCIONES   ***///
/* FUNCIÓN INICIALIZACIÓN DEL RELOJ */
void ini_clock(void){
        ICSC1_CLKS=0b00;              //CLOCK
        ICSC1_IREFS=1;
}


/* FUNCIÓN PARA ACTIVAR TODOS LOS PERIFÉRICOS */
void ini_perifericos(void){
        SCGC1=0xFF;
        SCGC2=0xFF;
}


/* FUNCIÓN PARA DESHABILITAR EL COP */
void ini_cop(void){
        SOPT1_COPE=0;         //WACTH DOG
```

```
}

/*  FUNCIÓN RETARDO */
void delay(unsigned long retardo){
                    while(retardo>0){               //llego a cero?
                    retardo--;                      //no --> decrementa
          }
}
/*  FUNCIONES ÁNGULOS DE COSENOS DIRECTORES  */
int ro1(){
        int r1=0;
    r1=(accelx/(sqrt((accely*accely)+( accelz*accelz))));
  return r1;
}

int ro2(){
        int r2=0;
    r2=(accel2x/(sqrt((accel2y*accel2y)+( accel2z*accel2z))));
  return r2;
}

int fi1(){
        int f1=0;
    f1=(accely/(sqrt((accelx*accelx)+( accelz*accelz))));
  return f1;
}

int fi2(){
        int f2=0;
    f2=(accel2y/(sqrt((accel2x*accel2x)+( accel2z*accel2z))));
  return f2;
}

int teta1(){
        int t1=0;
    t1=((sqrt((accelz*accelx)+( accely*accely)))/ accelz);
  return t1;
}

int teta2(){
        int t2=0;
    t2=((sqrt((accel2z*accel2x)+( accel2y*accel2y)))/ accel2z);
  return t2;
}
```

Anexo 9: FUNCIONES DEL PROGRAMA PARA LA CONFIGURACIÓN IIC

```
/* FUNCION INICIALIZA IIC1 */

void ini_iic1(){
        IIC1F=0x7F;
        IIC1C1=0xE0;
        IIC1S=0x10;
}

/* FUNCION INICIALIZA IIC2 */
void ini_iic2(){
        IIC2F=0x7F;
        IIC2C1=0xE0;
        IIC2S=0x10;
}

/* FUNCIÓN PARA ENVIAR DATO IIC */
void enviar_dato_i2c(byte dato){
            bandera2=0;                 //inicie bandera de acceso
            IIC1D=dato;                 //transmita dato
            while(bandera2 == 0);       //espere que se transmita
}

/* FUNCIÓN ESCRIBIR ACELERÓMETRO 1 */
void escribir_acel1_i2c(byte direccion,byte dato){
            IIC1C1_TX=1;                //Modo TX por el maestro
            IIC1C1_TXAK=1;               //ACK
            IIC1C1_MST=1;               //START
            enviar_dato_i2c(0xA0);       //dirección WRITE=A0
            enviar_dato_i2c(direccion); //dirección a accesar
            enviar_dato_i2c(dato);      //dato a la celda
            IIC1C1_MST=0;               //STOP
}

/* FUNCIÓN ESCRIBIR GIROSCOPIO 1 */
void escribir_gyro1_i2c(byte direccion,byte dato){
            IIC1C1_TX=1;                //Modo TX por el maestro
            IIC1C1_TXAK=1;              //ACK
            IIC1C1_MST=1;               //START
            enviar_dato_i2c(0b1101000);         //dirección  WRITE=A0
            enviar_dato_i2c(direccion);     //dirección a accesar
            enviar_dato_i2c(dato);      //dato a la celda
            IIC1C1_MST=0;               //STOP
}


/* FUNCIÓN LEER ACELERÓMETRO 1 */
void leer_acel1_i2c(byte direccion){
            IIC1C1_TX=1;                //TX por el maestro
            IIC1C1_TXAK=0;              //no esperan los ACK
```

```
                IIC1C1_MST=1;               //START
                enviar_dato_i2c(0xA0);      //dir accel para escritura
                enviar_dato_i2c(direccion); //dir a accesar
                IIC1C1_RSTA=1;              //START
                enviar_dato_i2c(0xA1);      //dir accel para lectura
                bandera2=0;                 //aclare bandera
                IIC1C1_TXAK=1;              //ACK
                IIC1C1_TX=0;                //Modo de RX del maestro
                tmp=IIC1D;                  //toma dato dummy
                IIC1C1_MST=0;               //se conmuta a modo esclavo
                tmp=IIC1D;
}


/* FUNCIÓN LEER GIROSCOPIO 1 */
void leer_gyro1_i2c(byte direccion){
                IIC1C1_TX=1;                //TX por el maestro
                IIC1C1_TXAK=0;              //no esperan los ACK
                IIC1C1_MST=1;               //START
                enviar_dato_i2c(0b1101000); //dir gyro para escritura
                enviar_dato_i2c(direccion); //dir a accesar
                IIC1C1_RSTA=1;              //START
                enviar_dato_i2c(0b1101001); //dir gyro para lectura
                bandera2=0;                 //aclare bandera
                IIC1C1_TXAK=1;              //ACK
                IIC1C1_TX=0;                //Modo de RX del maestro
                tmp=IIC1D;                  //toma dato dummy
                IIC1C1_MST=0;               //Modo esclavo
                tmp=IIC1D;
}


/* FUNCIÓN ESCRIBIR ACELERÓMETRO 2 */
void escribir_acel2_i2c(byte direccion,byte dato){
                IIC2C1_TX=1;                    //Modo TX
                IIC2C1_TXAK=1;                  //ACK
                IIC2C1_MST=1;                   //START
                enviar_dato_i2c(0xA0);          //dir WRITE=A0
                enviar_dato_i2c(direccion);     //dir a accesar
                enviar_dato_i2c(dato);          //dato a la celda
                IIC2C1_MST=0;                   //STOP
}



/* FUNCIÓN ESCRIBIR GIROSCOPIO 2 */
void escribir_gyro2_i2c(byte direccion,byte dato){
```

```
                IIC2C1_TX=1;                    //Modo TX por el maestro
                IIC2C1_TXAK=1;                  //ACK
                IIC2C1_MST=1;                   //START
                enviar_dato_i2c(0b1101000);     //dir WRITE=A0
                enviar_dato_i2c(direccion);     //dir a accesar
                enviar_dato_i2c(dato);          //dato a la celda
                IIC2C1_MST=0;                   //STOP
}


/* FUNCIÓN LEER ACELERÓMETRO 2 */
void leer_acel2_i2c(byte direccion){
                IIC2C1_TX=1;                    //TX por el maestro
                IIC2C1_TXAK=0;                  //no esperan los ACK
                IIC2C1_MST=1;                   //START
                enviar_dato_i2c(0xA0);          //dir accel para escritura
                enviar_dato_i2c(direccion);     //dir a accesar
                IIC2C1_RSTA=1;                  //START
                enviar_dato_i2c(0xA1);          //dir accel para lectura
                bandera2=0;                     //aclare bandera
                IIC2C1_TXAK=1;                  //ACK
                IIC2C1_TX=0;                    //Modo de RX del maestro
                tmp=IIC1D;                      //toma dato dummy
                IIC2C1_MST=0;                   //se conmuta a modo esclavo
                tmp=IIC1D;
}

/* FUNCIÓN LEER GIROSCOPIO 2 */
void leer_gyro2_i2c(byte direccion){
                IIC2C1_TX=1;                    //TX por el maestro
                IIC2C1_TXAK=0;                  //no esperan los ACK
                IIC2C1_MST=1;                   //START
                enviar_dato_i2c(0b1101000); //dir gyro para escritura
                enviar_dato_i2c(direccion);     //dir a accesar
                IIC2C1_RSTA=1;                  //START
                enviar_dato_i2c(0b1101001); //dir gyro para lectura
                bandera2=0;                     //aclare bandera
                IIC2C1_TXAK=1;                  //ACK
                IIC2C1_TX=0;                    //Modo de RX del maestro
                tmp=IIC1D;                      //toma dato dummy
                IIC2C1_MST=0;                   //se conmuta a modo esclavo
                tmp=IIC1D;
}
```