Apéndice B

Código completo utilizado para el control del acelerómetro del Nunchuk

```
#include <Wire.h>

uint8_t outbuf[6];           // arreglo para almacenar las
variables de salida del Arduino
int cnt = 0;
//int ledPin = 13;

int pulseWidth = 0;    // Ancho de pulso
long lastPulse = 0;    // tiempo en milisegundos del último
pulso
int refreshTime = 20;  // tiempo en milisegundos necesitado
entre pulsos
int minPulse = 700;    // ancho de pulso minimo
int pulseWidth2 = 0;

#define pwbuffsize 4
int pwbuff[pwbuffsize]; // buffer para la variable pulseWidths
int pwbuffpos = 0;      // posición en pwbuff
int pwbuff2[pwbuffsize];
int pwbuffpos2 = 0;

int t = 0;  // cuando el contador llega a 25 se lee el Nunchuk

////

/////// para mover motores
int pulso = 4;
int act1 = 3;
int act2 = 5;
int act3 = 6;
int act4 = 11;

// Valores de PWM
int e = 150;
int v= 0;

// 4 motores a controlar
int a = 0;
int b = 0;
int c = 0;
int d = 0;

int xx  = 0;
int xx2 = 0;
int yy  = 0;
int yy2 = 0;
```

```
///////

void setup()
{
  beginSerial (19200);
  Serial.print ("Finished setup\n");
  Wire.begin ();        // se une la el bus i2c bus con la
dirección 0x52
  Nunchuk_init ();             // se manda la iniciación para el
Nunchuk
  pulseWidth = minPulse;         // el ancho de pulso se reduce
al minimo

}



void loop()
{
  // Cuando el contador llega a 25 se lee el Nunchuk
  t++;
  if( t == 25 )
  {

    inclinacion();

  }

  delay(20);

}


void inclinacion()
{
   t = 0;
        Wire.requestFrom (0x52, 6); // se hace la petición de
datos al Nunchuk
        while (Wire.available ())
          {
            // se recibe un byte como un entero
             outbuf[cnt] = nunchuk_decode_byte (Wire.receive
());
             cnt++;
          }

        if (cnt >= 5)
        {
            ObtenerX();
        }

        if (cnt >= 5)
        {
          ObtenerY();
        }

        //moverMotor();
      cnt = 0;
      send_zero(); // se manda una petición para los siguientes
bytes
```

```
}

void moverMotor()
{
   a=xx2;
   Serial.print("Motor 1: ");  Serial.print(a);
   Serial.print("  \t");
   analogWrite(act1, a);

   b=yy2;
   Serial.print("Motor 2: ");  Serial.println(b);
   analogWrite(act2, b);

}

void ObtenerX()
{
   float tilt = outbuf[2];
          tilt = (tilt - 70) * 1.5;                 // se
convierte wn grados aproximadamente
          pulseWidth = (tilt * 9) + minPulse;        // los
ángulos se convierten en microsegundos
          pwbuff[pwbuffpos] = pulseWidth;            // se
guarda para sacar promedios
          if( ++pwbuffpos == pwbuffsize ) pwbuffpos = 0;
            pulseWidth=0;
              for( int p=0; p<pwbuffsize; p++ )
                  pulseWidth += pwbuff[p];
                  pulseWidth /= pwbuffsize;


            // aquí se asignan los datos del eje X
            // tilt2 es el dato que contiene los valores de
inclinación que da el
            //acelerómetro
            xx2=((tilt*100)/165);
            Serial.print("\tx%: ");   Serial.print(xx2);
}

void ObtenerY()
{
   float tilt2 = outbuf[3];
          tilt2 = (tilt2 - 70) * 1.5;
          pulseWidth2 = (tilt2 * 9) + minPulse;
          pwbuff2[pwbuffpos2] = pulseWidth2;

           if( ++pwbuffpos2 == pwbuffsize ) pwbuffpos2 = 0;
               pulseWidth2=0;
                 for( int p2=0; p2<pwbuffsize; p2++ )
                     pulseWidth2 += pwbuff2[p2];
                     pulseWidth2 /= pwbuffsize;

      // aquí se asignan los datos del eje Y
      // tilt2 es el dato que contiene los valores de
inclinación que da el
      //acelerómetro
      yy2=((tilt2*100)/165);
      Serial.print("\ty%: ");
      Serial.print(yy2);
      Serial.print("\n");
}
```

```
void Nunchuk_init()
{
  Wire.beginTransmission (0x52);    // se trasmite al dispositivo
0x52
  Wire.send (0x40);                   // se manda la dirección de
memoria
  Wire.send (0x00);                   // se manda un cero
  Wire.endTransmission ();            // transmisión terminada
}

void send_zero()
{
  Wire.beginTransmission (0x52);    // se trasmite al dispositivo
0x52
  Wire.send (0x00);                   // se manda un byte
  Wire.endTransmission ();            // transmisión terminada
}

char nunchuk_decode_byte (char x)
{
  x = (x ^ 0x17) + 0x17;
  return x;
}
```