2011

NUNCHUCK + SERVO

COVARRUBIAS BAZUA JUAN FRANCISCO

SAMARIO MARTINEZ ERIK

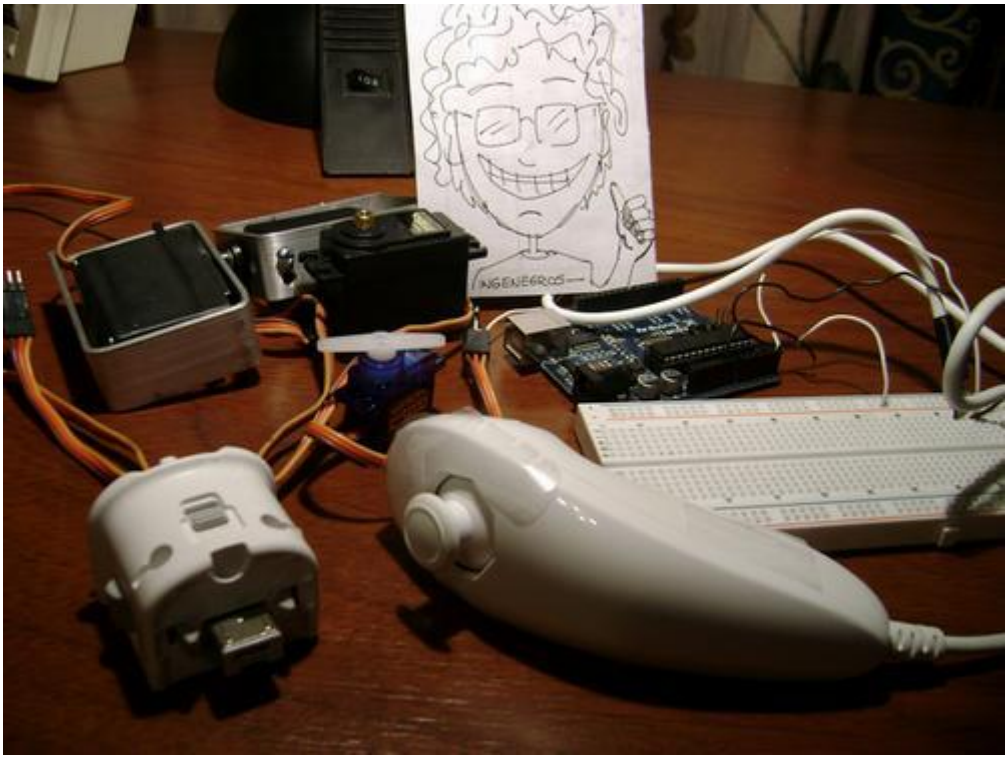
SALVADOR GONZALEZ

INSTITUTO TECNOLOGICO DE TIJUANA

01/01/2011

COMO MOVER MOTORSITOS CON CONTROL DE WII

MATERIALES:



NOSOTROS ESTUBIMOS BATAYANDO UN POCO CON EL ARDUINO PERO DESPUES DE UNOS 10 MINUTOS TODO QUEDO RESUELTO Y YA FUNCIONA

CONEXIÓN DEL WII CHUCK

GDN EN PIN ANALOGO

5V EN PIN ANALOGO

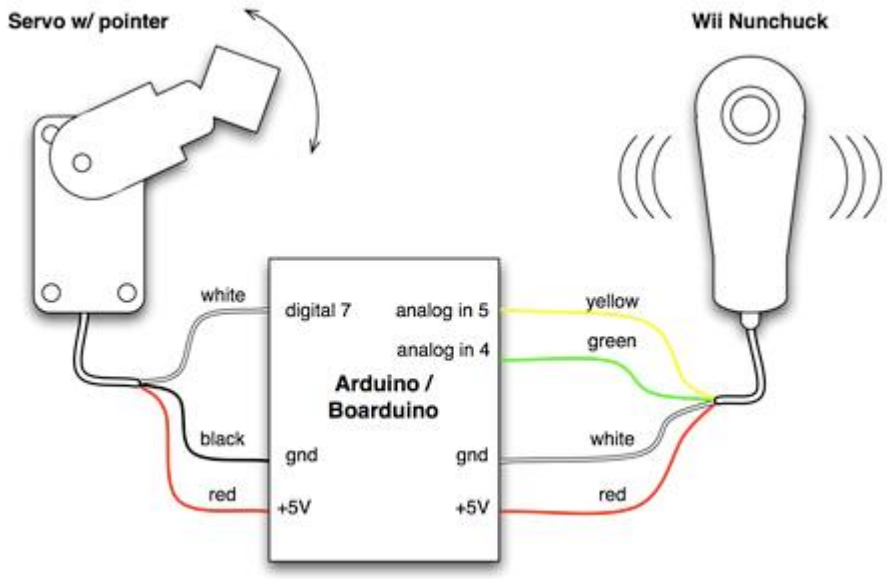
D EN PIN 4

C EN PIN 5

CONEXIÓN DEL MOTOR

5V

GND EN PIN 7 DIGITAL LA OTRA ENTRADA DEL MOTOR



CODIGO;

```
001.#include <Wire.h>
002.#include <string.h>
003.
004.#undef int
005.#include <stdio.h>
006.
007.uint8_t outbuf[6];      // array to store arduino output
008.int cnt = 0; //contador
009.int ledPin = 13;
010.
011.void setup () //recuerden que esto se ejecuta 1 sola vez al inicio
012.{
013.  beginSerial (19200); //abrimos el puerto serie a 19200 baudios
014.  Serial.print ("Finished setup\n"); //imprimimos un mensaje en la
pantalla
015.  Wire.begin ();        // juntamos el bus de datos i2c con la
direccion 0x52
016.  nunchuck_init (); // mandamos un handshake para inicializar el
dispositivo
017.}
018.
019.void nunchuck_init () // esto indica una funcion que luego sera
llamada
020.{
021.  Wire.beginTransmission (0x52);    // transmite al dispositivo 0x52
022.  Wire.send (0x40);     // envia la direccion de la memoria
```

```
023.  Wire.send (0x00);     // manda un cero
024.  Wire.endTransmission ();  // deja de transmitir
025.}
026.
027.void send_zero ()
028.{
029.  Wire.beginTransmission (0x52);    // transmite al dispositivo 0x52
030.  Wire.send (0x00);     // manda 1 byte, un cero
031.  Wire.endTransmission ();  // deja de transmitir
032.}
033.
034.void loop ()
035.{
036.  Wire.requestFrom (0x52, 6);   // pide datos del nunchuck al
dispositivo 0x52, y se recibirán 6 bytes
037.  while (Wire.available ()) //similar al Serial.available() nos indica
que se estan recibiendo datos por el puerto.
038.    {
039.      outbuf[cnt] = nunchuk_decode_byte (Wire.receive ());  // recibe
el byte como un integer
040.      digitalWrite (ledPin, HIGH);  // prende el led
041.      cnt++; //incrementa el contador
042.    }
043.
044.  // Si recibimos los 6 bytes, entonces los imprimimos
045.  if (cnt >= 5)
046.    {
047.      print (); // llamamos a la funcion print
048.    }
049.
050.  cnt = 0; //volvemos a dejar el contador en 0
051.  send_zero (); // manda el pedido para recibir mas bytes
052.  delay (100);
053.}
054.
055.// Print the input data we have recieved << Imprime la informacion
que recibimos
056.// accel data is 10 bits long << La informacion de los acelerometros
tiene una longitud de 10 bits
057.// so we read 8 bits, then we have to add << entonces leemos 8 bits y
le agregamos los ultimos 2 bits
058. // on the last 2 bits.  That is why I << por eso los multiplico por
2 * 2
059.// multiply them by 2 * 2 << Los bits que agregamos vienen en el 6to
Byte que recibimos.
060.void print ()
061.{
062.  int joy_x_axis = outbuf[0];
063.  int joy_y_axis = outbuf[1];
064.  int accel_x_axis = outbuf[2] * 2 * 2;
065.  int accel_y_axis = outbuf[3] * 2 * 2;
066.  int accel_z_axis = outbuf[4] * 2 * 2;
067.
068.  int z_button = 0;
069.  int c_button = 0;
```

```
070.
071. // byte outbuf[5] contiene los bits de los botones C y Z
072. // tambien contiene los ultimos bits significativos de los
acelerometros
073. // por lo que tenemos que ver cada bit del byte outbuf[5]
074.
075.  if ((outbuf[5] >> 0) & 1)
076.    {
077.      z_button = 1;
078.    }
079.  if ((outbuf[5] >> 1) & 1)
080.    {
081.      c_button = 1;
082.    }
083.
084.  if ((outbuf[5] >> 2) & 1)
085.    {
086.      accel_x_axis += 2;
087.    }
088.  if ((outbuf[5] >> 3) & 1)
089.    {
090.      accel_x_axis += 1;
091.    }
092.
093.  if ((outbuf[5] >> 4) & 1)
094.    {
095.      accel_y_axis += 2;
096.    }
097.  if ((outbuf[5] >> 5) & 1)
098.    {
099.      accel_y_axis += 1;
100.    }
101.
102.  if ((outbuf[5] >> 6) & 1)
103.    {
104.      accel_z_axis += 2;
105.    }
106.  if ((outbuf[5] >> 7) & 1)
107.    {
108.      accel_z_axis += 1;
109.    }
110.
111.  Serial.print (joy_x_axis, DEC);
112.  Serial.print ("\t");
113.
114.  Serial.print (joy_y_axis, DEC);
115.  Serial.print ("\t");
116.
117.  Serial.print (accel_x_axis, DEC);
118.  Serial.print ("\t");
119.
120.  Serial.print (accel_y_axis, DEC);
121.  Serial.print ("\t");
122.
```

```
  Serial.print (accel_z_axis, DEC);
  Serial.print ("\t");

  Serial.print (z_button, DEC);
  Serial.print ("\t");

  Serial.print (c_button, DEC);
  Serial.print ("\t");

  Serial.print ("\r\n");
}

// Encode data to format that most wiimote drivers except
// only needed if you use one of the regular wiimote drivers
char
nunchuk_decode_byte (char x)
{
  x = (x ^ 0x17) + 0x17;
  return x;
}
```

Mas información en

http://ingenegros.com.ar/Microcontroladores/leer-un-wii-nunchuck-con-arduino.html