

**BRAINCODER 1.0**



## Contenido

1	Sobre los autores .....	7
2	Introducción.....	1
3	Elección de elementos .....	2
3.1	Codificador rotatorio.....	2
3.2	Adafruit OLED 128x64 i2c.....	3
3.3	Plástico.....	4
3.4	Muelle de torsión .....	4
3.5	Hilo .....	6
3.6	Rodamientos ABEC7 .....	6
3.7	Otros materiales .....	7
4	Código Arduino .....	8
5	Parte electrónica.....	12
5.1	Material necesario.....	13
5.2	Placa electrónica .....	14
5.3	Bootloader Arduino .....	16
6	Parte mecánica .....	19
6.1	Carcasa circuito.....	19
6.2	Carcasa inferior .....	19
6.3	Carcasa superior .....	21
6.4	Eje .....	22
6.5	Tapa eje.....	22
6.6	Soporte muelle de torsión .....	23
6.7	Enganche .....	23
6.8	Tapa pantalla.....	24

6.9	Detalle carrete y soporte muelle de torsión.....	24
6.9.1	Encoder completo abierto 3D .....	25
6.9.2	Encoder completo cerrado 3D .....	26
6.9.3	Lote de piezas encoder .....	27
6.9.4	Encoder completo .....	28
7	Montaje encoder .....	29
8	Presupuesto .....	32
9	Futuras mejoras .....	33
	Lista de referencias .....	34

## Índice de figuras

Figura 3.1 - Codificador rotatorio KY-040. Fuente: mactronica.com .....	2
Figura 3.2 - Detalle de las conexiones del módulo KY-040. Fuente: mactronica.com.....	2
Figura 3.3 - Display Adafruit OLED 128x64 i2c. Fuente: amazon.es.....	3
Figura 3.4 - Detalle de conexiones Display OLED 128x64. Fuente: luisllamas.es..	3
Figura 3.5 - Rollos de plástico de ABS de 1 kg cada uno. Fuente: kareca3d.com.	4
Figura 3.6 - Muelle de torsión en espiral tipo cinta métrica. Fuente: dgpowerspring.com.....	5
Figura 3.7 - Cinta métrica. Fuente: amazon.es .....	5
Figura 3.8 - Hilo de Kevlar. Fuente: amazon.es.....	6
Figura 3.9 - Rodamientos ABEC7. Fuente: decathlon.es .....	6
Figura 5.1 - Braincoder con Arduino. Fuente: propia .....	12
Figura 5.2 - Esquema del circuito de Arduino. Fuente: electrrio.com.....	12
Figura 5.3 - Atmega 328P-PU relacionado con Arduino UNO. Fuente: electrrio.com.....	13
Figura 5.4 - Piezas placa electrónica. Fuente: propia .....	14
Figura 5.5 - Placa electrónica acabada. Fuente: propia .....	15
Figura 5.6 - Placa electrónica con las conexiones externas realizadas. Fuente: propia .....	15
Figura 5.7 - Conexionado del circuito para carga de Bootloader. Fuente: untitled.es .....	16
Figura 5.8 - Carga de Bootloader primer paso. Fuente: minitronica.com.....	17
Figura 5.9 - Carga de Bootloader segundo paso. Fuente: minitronica.com .....	17
Figura 5.10 - Carga de Bootloader a varios Atmega 328P-PU. Fuente: propia....	18
Figura 6.1 - Carcasa circuito. Fuente: Propia .....	19
Figura 6.2 - Carcasa inferior. Fuente: Propia .....	20

Figura 6.3 - Carcasa superior. Fuente: Propia.....	21
Figura 6.4 - Eje. Fuente: Propia.....	22
Figura 6.5 - Tapa eje. Fuente: Propia .....	22
Figura 6.6 - Soporte muelle de torsión. Fuente: Propia .....	23
Figura 6.7 - Enganche. Fuente: Openbarbell .....	23
Figura 6.8 - Tapa pantalla. Fuente: Propia .....	24
Figura 6.9 - Detalle carrete y soporte muelle de torsión. Fuente: Propia .....	24
Figura 6.10 - Imagen 3D del despiece del encoder. Fuente: Propia .....	25
Figura 6.11 - Imagen 3D del encoder cerrado. Fuente: Propia .....	26
Figura 6.12 - Lote de piezas impresas Braincoder. Fuente: Propia .....	27
Figura 6.13 - Braincoder completo en funcionamiento. Fuente: Propia.....	28
Figura 7.1 - Despiece cinta métrica. Fuente: Propia.....	29
Figura 7.2 - Extracción muelle de torsión. Fuente: Propia.....	29
Figura 7.3 - Instalación muelle de torsión. Fuente: Propia.....	29
Figura 7.4 - Acoplamiento eje con muelle de torsión. Fuente: Propia .....	30
Figura 7.5 - Instalación elementos mecánicos. Fuente: Propia.....	30
Figura 7.6 - Parte inferior. Fuente: Propia .....	30
Figura 7.7 - Braincoder. Resultado final. Fuente: Propia.....	31

## Índice de cuadros y tablas

Tabla 5.1 - Presupuesto placa circuito Encoder. Fuente: propia .....	14
Tabla 8.1 - Presupuesto aproximado Braincoder. Fuente: propia .....	32




## 1 Sobre los autores




**Iván Alonso Molero**

**Correo:** ivalmo@usal.es

**Web:** [www.brainbuilder.es](http://www.brainbuilder.es)

 @brainbuilder

 brainbuilder


**Autor del libro: BRAINBUILDER**

(Capítulos: Psicología en deportes de fuerza)



**Juan Jesús Cembranos del Castillo**

 @juanxuso

 juanxuso (próximamente)

Mi nombre es Iván Alonso Molero, soy graduado en Psicología por la USAL y Máster en Psicología aplicada a la Actividad física y el deporte por la UAM.

Mi pasión por el culturismo y los deportes de fuerza (ambas modalidades en las cuales compito) me llevo a estudiar todo lo posible acerca de cómo mejorar el rendimiento tanto a nivel físico como mental.

Desde entonces me dedico a compartir con otras personas mi pasión a través de las redes sociales como "Brainbuilder". Y lo cierto es que lo que empezó siendo compartir reflexiones en Instagram se está convirtiendo en todo un movimiento que avanza cada vez más rápido y que mueve cada vez a más personas.

Como suelo decir, nunca dejéis que las mentes pequeñas os digan que vuestros sueños son demasiado grandes.

Yo soy Juan Cembranos, llevo ya mucho tiempo trabajando desde las sombras. Estoy graduado en ingeniería electrónica industrial y automática, también en ingeniería eléctrica en la ULE. Además estoy estudiando un MBA.

Desde hace años dedico todo mi tiempo a realizar pequeños proyectos propios, además me dedico a dar charlas sobre emprendimiento en diferentes universidades de España. Entre algunas cosas destacables, soy cofundador y CEO de LifeSlide!

A pesar de haber comenzado como ingeniero esto está cambiando abruptamente. Mi sueño es poder crear algo de valor para el mundo, y creo que esto es una buena manera de comenzar.

## 2 Introducción

Primeramente enumeramos los elementos que vamos a utilizar, esto no tiene mucha complicación. Después mostramos cómo es el código del programa de Arduino, además, estará explicado lo más posible.

Posteriormente indicamos los esquemas eléctricos y los planos de las piezas de plástico para hacer real esta majestuosidad.

Explicaremos cómo se monta el encoder paso a paso y por último haremos un presupuesto aproximado y anunciaremos algunas de las mejoras que le esperan al buen Braincoder.

**NOTA IMPORANTE:** tanto en el vídeo como en esta página del presente documento aparece un encoder negro y amarillo. Previamente lo construimos por primera vez en verde. Aquí mostraremos el paso a paso de ese encoder. Es el mismo que el otro así que no os preocupéis. Vamos a ello, ¡a tope!





## 3 Elección de elementos

### 3.1 Codificador rotatorio

En nuestro caso hemos adquirido el sensor KY-040, este es un encoder incremental bastante preciso (20 pulsos por vuelta).



Figura 3.1 - Codificador rotatorio KY-040. Fuente: [mactronica.com](http://mactronica.com)

Este aparato se puede encontrar en Ebay y en Amazon, su precio ronda los 2 €. Podemos apreciar lo barato que puede ser elaborar un encoder lineal si así lo deseamos.

A continuación mostramos una imagen de las conexiones del módulo con el Arduino:

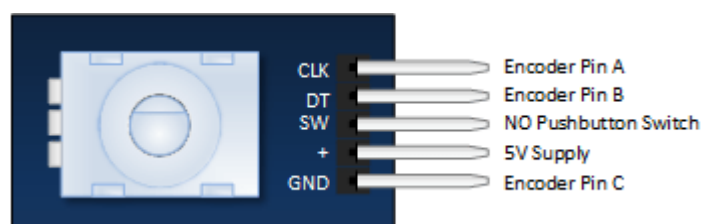


Figura 3.2 - Detalle de las conexiones del módulo KY-040. Fuente: [mactronica.com](http://mactronica.com)

Como podemos observar, claramente se marcan los pines del aparato. CLK va hacia el pin digital 2 del Arduino (D2), y DT se conecta al pin digital 3 (D3). Estos dos pines (D2 y D3) están seleccionados de manera que funcionen

correctamente, ya que están diseñados especialmente para poder programarles interrupciones externas, aspecto necesario para este proyecto.

El pin + lo alimentamos, en este caso nos sirven con 3,5 V; aunque la imagen indique 5 V se puede alimentar con menos. Por último GND va a tierra.

### 3.2 Adafruit OLED 128x64 i2c

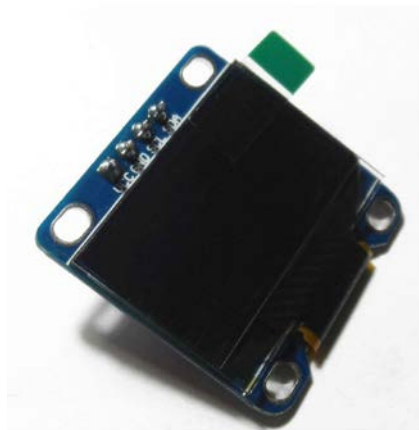


Figura 3.3 - Display Adafruit OLED 128x64 i2c. Fuente: [amazon.es](https://www.amazon.es)

A continuación se incluye un detalle de los conexiones de la pantalla escogida:

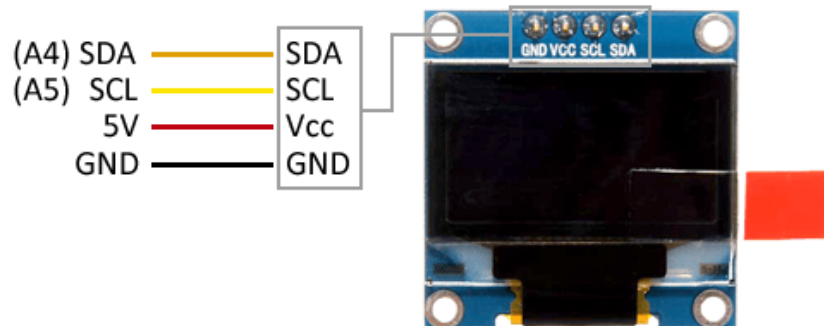


Figura 3.4 - Detalle de conexiones Display OLED 128x64. Fuente: [luisllamas.es](https://luisllamas.es)

VCC es la alimentación, esta vez sí que lo conectaremos a 5 V. GND es tierra. SDA lo llevaremos al pin analógico 4 (A4) del Arduino; SCL lo conectaremos al pin analógico 5 (A5).

Podemos encontrar esta pantalla por tiendas en internet por un precio de aproximadamente 7 €.

### 3.3 Plástico

Para construir nuestro encoder, utilizaremos una impresora 3D. De esta manera crearemos un soporte para todas nuestras piezas electrónicas y así podremos hacerlo funcionar.



Figura 3.5 - Rollos de plástico de ABS de 1 kg cada uno. Fuente: [kareca3d.com](http://kareca3d.com)

Tenemos varias opciones, entre las más conocidas podemos escoger entre ABS y PLA. El ABS es mucho más resistente y flexible, es perfecto para trabajos mecánicos como este. El plástico PLA es más barato y se imprime mejor, además de que es biodegradable, ya que no proviene del petróleo como el ABS, sino que es de origen vegetal.

El primer encoder está realizado completamente en ABS, el segundo, debido a la falta de una impresora 3D preparada para ABS al momento de imprimirlo, se realizó en PLA. Funciona perfectamente aunque lo recomendable sea ABS.

### 3.4 Muelle de torsión

Al utilizar nuestro encoder tiraremos de un hilo enrollado en un carrete, se desenrollará y posteriormente al retroceder, el hilo debe enrollarse otra vez en el carrete, pero debe hacerlo él solo.

Esto lo haremos utilizando un muelle de torsión, tal como aparece en la siguiente imagen:



Figura 3.6 - Muelle de torsión en espiral tipo cinta métrica. Fuente: [dgpowerspring.com](http://dgpowerspring.com)

El problema de esto es que en la mayoría de las tiendas online venden solamente al por mayor o el precio es elevado. La solución es sencilla: compraremos una cinta métrica y la desmontaremos, así podremos extraer el muelle de su interior.



Figura 3.7 - Cinta métrica. Fuente: [amazon.es](http://amazon.es)

### 3.5 Hilo

En una primera instancia se utilizó un hilo de pesca, pero era tan duro y fino que serraba el ABS al utilizarlo. Después se pensó en utilizar una cuerda más ancha, pero lo ideal al final fue decidirse por un hilo de Kevlar.

Este material tiene grandes propiedades mecánicas, entre ellas está su resistencia al calor de hasta 425°C concretamente para este hilo.



Figura 3.8 - Hilo de Kevlar. Fuente: [amazon.es](https://www.amazon.es)

Su precio oscila por unos 10 € cada rollo, y son 50 metros, así que da de sobra para varios encoder. Para cada uno utilizaremos aproximadamente entre 2 y 4 metros.

### 3.6 Rodamientos ABEC7

Se suelen utilizar en las tablas de Skate. Su precio oscila entre 1€ y 2€ la unidad. En el primer encoder utilizamos tres rodamientos y en el segundo solamente uno. Gracias a los rodamientos permitimos que el mecanismo funcione suavemente.



Figura 3.9 - Rodamientos ABEC7. Fuente: [decathlon.es](https://www.decathlon.es)



### 3.7 Otros materiales

Además de todo lo mencionado anteriormente, hay que tener en cuenta que construiremos un circuito que imite a un Arduino. Todo esto se mencionará en el capítulo 6 de este proyecto. Además contaremos con una placa protoboard de tamaño pequeño que sostenga el sensor, esto solamente es para el primer encoder.

También hay otros elementos de carácter obvio, como el pegamento. Utilizaremos uno muy potente para plástico, en este caso Loctite.

## 4 Código Arduino

A continuación se muestra y explica el programa para Arduino que se utiliza en este proyecto. El código es en lenguaje C++ y está lo suficientemente comentado como para que se entienda rápidamente. Primero comenzamos con las librerías:

```
//Incluimos librerías de protocolos de comunicación

#include <SPI.h>
#include <Wire.h>

//Añadimos librerías de la pantalla

#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
```

Ahora vamos a declarar algunas variables fijas, así como otras de carácter volátil, útiles a la hora de contar pulsos:

```
#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);

#define NUMFLAKES 10
#define XPOS 0
#define YPOS 1
#define DELTAY 2

//Variables para que funcione la pantalla correctamente

#define LOGO16_GLCD_HEIGHT 16
#define LOGO16_GLCD_WIDTH 16

static const unsigned char PROGMEM logo16_glcd_bmp[] =
{ B00000000, B11000000,
  B00000001, B11000000,
  B00000001, B11000000,
  B00000011, B11100000,
  B11110011, B11100000,
  B11111110, B11111000,
  B01111110, B11111111,
  B00110011, B10011111,
  B00011111, B11111100,
  B00001101, B01110000,
  B00011011, B10100000,
  B00111111, B11100000,
  B00111111, B11110000,
  B01111100, B11110000,
  B01110000, B01110000,
  B00000000, B00110000 };

const int PAUSE = 200; // Tiempo de pausa en ms
```

```

const int MIN_PULSES = 30; // Número mínimo de pulsos para que se
considera un movimiento válido

volatile long pulses = 0; // Número de pulsos
volatile unsigned long pulseTime = 0; // Tiempo del último pulso
volatile unsigned long firstTime = 0; // Tiempo del primer pulso

int value = 0; // Valor del pin 2 para determinar la rotación
int rotation = 0; // Último sentido de la rotación
boolean up = true; // Sentido del push

float velocidad;
float subida;

```

Ahora comienza el programa:

```

void setup() {
    Serial.begin(9600); //Se inicia la comunicación con el monitor serie
del PC
    display.begin(SSD1306_SWITCHCAPVCC, 0x3C); //Se inicia la comunicación
con el puerto 0x3C hacia la pantalla

    display.clearDisplay(); //Vaciamos el contenido de la pantalla

    display.setTextSize(2); //Elegimos el tamaño del texto a 2
    display.setTextColor(WHITE); //Color del texto blanco
    display.setCursor(0,0); //Colocamos el cursor al principio
    display.println("BRAINCODER"); //Mostramos un texto
    display.setTextSize(1); //Cambiamos el tamaño del texto a 1
    display.println("  si puedes medirlo");
    display.println("  PUEDES LEVANTARLO");
    display.display(); //Encendemos la pantalla con todo lo anterior
}

void printValues() {

    if(pulseTime != 0 && millis() - pulseTime > PAUSE) { // Comprobamos que
el tiempo de pulso esté iniciado y que transcurrió más de PAUSE ms

        if(pulses > MIN_PULSES) { // Comprobamos si se han alcanzado el
número mínimo de pulsos, si no descartamos

```

Mostramos los valores de la subida en el PC a través del monitor serie.

```

    if(up) { // Subida
        display.clearDisplay();
        Serial.print("Up: ");

```

Ahora hacemos que aparezca lo primero en la pantalla, en este caso haremos que salga escrito "Velocidad:", posteriormente rellenaremos con más datos.

```

        display.setTextSize(1);
        display.setTextColor(WHITE);
        display.setCursor(0,0);
        display.println("Velocidad: ");

```

Mostramos los valores de la bajada en el PC a través del monitor serie.

```

    } else { // Bajada
        Serial.print("Down: ");

```

Volvemos a hacer que aparezca lo mismo que en la subida.

```

        display.setTextSize(1);
        display.setTextColor(WHITE);
        display.setCursor(0,0);
        display.println("Velocidad: ");
    }

```

Para el cálculo de velocidad utilizaremos la distancia que recorre el hilo entre dos pulsos. En base a muchas pruebas se ha obtenido que entre cada dos pulsos hay 0,00124579518308312 metros. Este valor se utiliza para calcular la velocidad media gracias a que conocemos el tiempo.

Hay que tener en cuenta que cada fabricante crea el codificador rotatorio a su manera, así que es probable que haya que calibrarlo la primera vez. Esto se realiza haciendo pruebas y modificando la medida entre cada dos pulsos.

```

    velocidad = (float) (pulses / 2) * 0.00124579518308312 / ((float)
    (pulseTime - firstTime) / 1000);

    Serial.print(velocidad);
    Serial.println(" m/s");

```

Cuando hay una subida guardamos el valor de la velocidad en "subida". Este valor se volverá a mostrar en la bajada, así no se pierde, ya que no importa conocer la velocidad de la bajada.

```

    if(up) {
        subida = velocidad;
        display.setTextSize(2);
        display.print(" ");
        display.print(subida);
        display.println(" m/s");
        display.setTextSize(1);
        display.print("                                OK");
    }

```

Hay que advertir que al acabar la subida nos dará una señal de "OK" y al acabar la bajada (el movimiento completo) nos dirá "GO". Así sabremos que ya podemos iniciar el movimiento de nuevo.

```

    else {
        display.setTextSize(2);

```

```

        display.print(" ");
        display.print(subida);
        display.println(" m/s");
        display.setTextSize(1);
        display.print("                                GO");
    }

    up = !up;

    // text display tests

    display.display();
    display.clearDisplay();

}
pulseTime = 0; // Reiniciamos el tiempo del pulso
pulses = 0; // Reiniciamos el número de pulsos
}
}

```

Ahora podemos ver el bucle gracias al cual podremos saber si el encoder gira, hacia qué sentido y el tiempo que pasa durante esos pulsos.

```

void loop() {
    value = digitalRead(2); // Leemos el valor del pin 2 para determinar el
    estado (ver especificaciones encoder)
    if (value != rotation){ // Comprobamos si estamos en medio de un giro
        if (digitalRead(3) != value) { // Sentido horario (up)
            pulses++; // Incrementamos el número de pulsos
            if(pulseTime == 0) {
                firstTime = millis(); // Almacenamos el primer tiempo de pulso
            }
            pulseTime = millis(); // Almacenamos el último tiempo de pulso
        } else { // Sentido antihorario (down)

        }
        rotation = value; // Actualizamos el valor de la última rotación (ver
        especificaciones encoder)
    }
    printValues(); // Mostramos los valores por TTY
}
}

```



## 5 Parte electrónica

En este capítulo se desarrollará de manera concisa puesto que habrá múltiples maneras de realizar el circuito. Tenemos dos opciones: una fácil (utilizando un Arduino) y otra un poco más compleja (creando nosotros el Arduino).

Tanto en el vídeo como aquí haremos la manera compleja. Aún así, os mostramos un esquema de cómo sería el circuito utilizando directamente un Arduino:

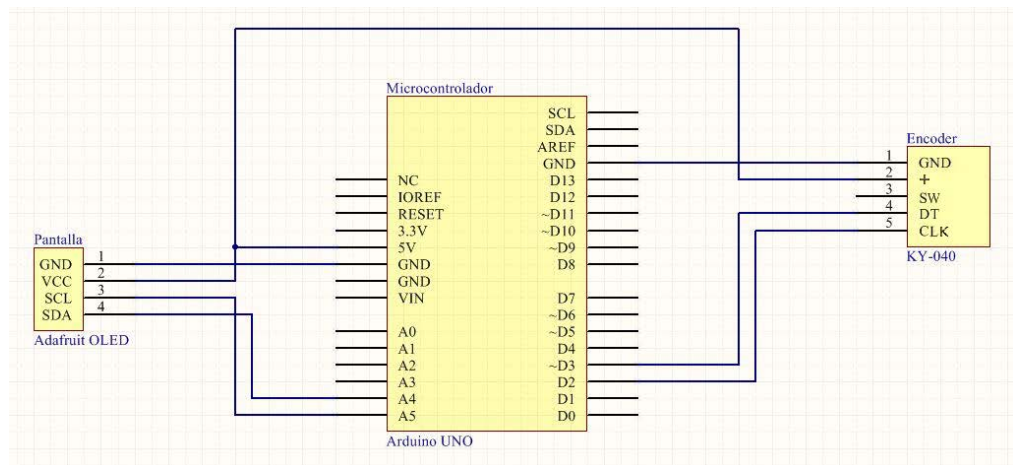


Figura 5.1 - Braincoder con Arduino. Fuente: propia

Si decidimos construir nosotros el Arduino, lo que haríamos sería simplemente replicar un Arduino en una placa PCB perforada con un ATMEGA 328P-PU, de esta manera ocupará menos espacio. El circuito a realizar es más o menos el siguiente:

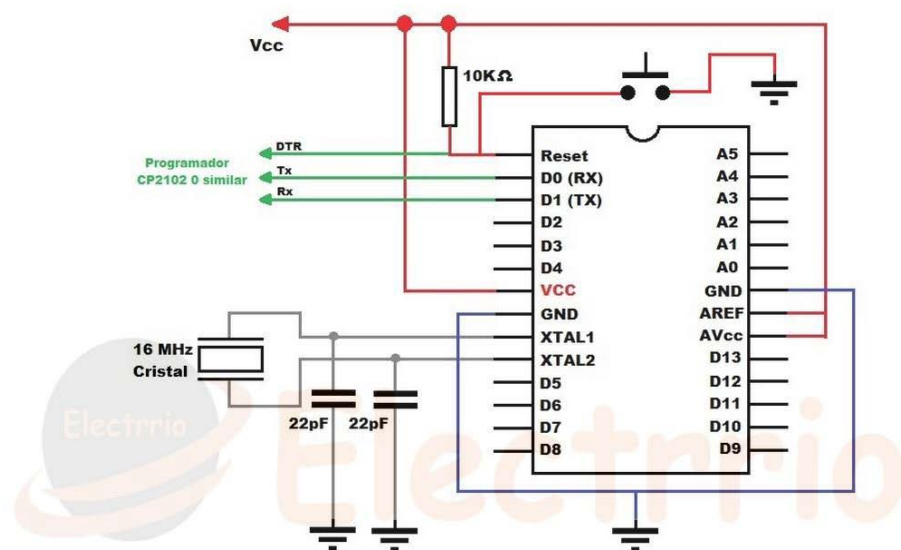


Figura 5.2 - Esquema del circuito de Arduino. Fuente: electrrio.com

Y sobre eso montaríamos el codificador rotatorio y la pantalla OLED según lo indicado en la figura 5.1.

## 5.1 Material necesario

Enumeraremos cada uno de los elementos que componen nuestro circuito y su precio, para así posteriormente hacer un presupuesto sencillo.

- **Atmega 328P-PU:** Este microcontrolador fue creado por la empresa ATMEL, se utiliza en el Arduino UNO y es muy versátil y rápido trabajando con programas sencillos. La relación entre los pines del Arduino y de este elemento es esta:



Figura 5.3 - Atmega 328P-PU relacionado con Arduino UNO. Fuente: [electrrio.com](http://electrrio.com)

- **Zócalo Atmega 328:** Con esta pieza podemos conectar y desconectar rápidamente un microcontrolador Atmega 328 en cualquier momento sin necesidad de retirar la soldadura y volver a soldar de nuevo.
- **Oscilador de cuarzo de 16 MHz:** Gracias a este elemento podemos hacer que todo el programa funcione. El oscilador es el reloj que marcará los tiempos del microcontrolador.
- **2 Condensadores cerámicos 18-22 pF:** Ayudan a normalizar las irregularidades de los tiempos del oscilador.
- **Condensador electrolítico 100 µF 16 Vdc:** Estabiliza las variaciones de tensión del regulador.
- **Regulador de tensión LM7805:** Introduciendo entre 5 y 35 voltios hará que siempre salgan 5 voltios hacia el microcontrolador.
- **Resistencia 10 KΩ:** Es necesario por si en un futuro instalamos un botón de Reset.

- **Clip para pila 9V y pila de 9V:** De esta manera alimentamos a nuestro encoder.
- **Placa PCB perforada:** En esta pieza irá todo soldado y conectado.

También habría que contar con un soldador y estaño, elementos que daremos por supuesto

Tabla 5.1 - Presupuesto placa circuito Encoder. Fuente: propia

Elemento	Precio
Atmega 328P-PU	3,00 €
Zócalo Atmega 328	0,30 €
Oscilador 16 MHz	0,25 €
2 Condensadores 22 pF	0,15 €
Condensador 100 µF	0,25 €
Regulador LM7805	0,50 €
Resistencia 10 KΩ	0,05 €
Clip para pila de 9V	0,20 €
Pila de 9V	2,00 €
Placa PCB perforada	0,50 €
<b>Total</b>	<b>7,20 €</b>

## 5.2 Placa electrónica

Primeramente cogemos todas las piezas que necesitamos. Cabe destacar que esta placa solamente tiene un lado para soldar.

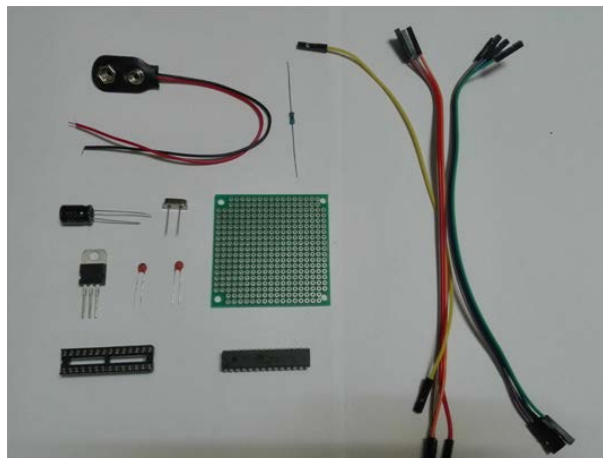


Figura 5.4 - Piezas placa electrónica. Fuente: propia

Además de todas las piezas que se ven en la foto contaremos con 5 pares de pines típicos de Arduino. No vamos a poner el esquema exacto ya que hay

muchas maneras de hacerlo y no es necesario. Una vez soldadas las piezas principales tendremos lo siguiente:

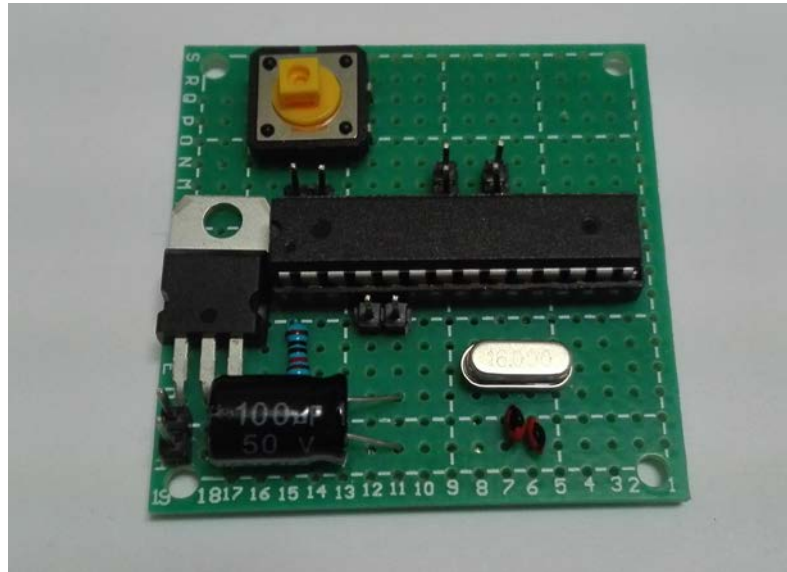


Figura 5.5 - Placa electrónica acabada. Fuente: propia

Como se aprecia en la foto, hemos añadido el botón de Reset. Gracias a esta configuración en las conexiones, cuando decidamos conectar el codificador rotatorio, la pantalla y la alimentación; simplemente utilizaremos cables con conectores “jumper” típicos de Arduino.

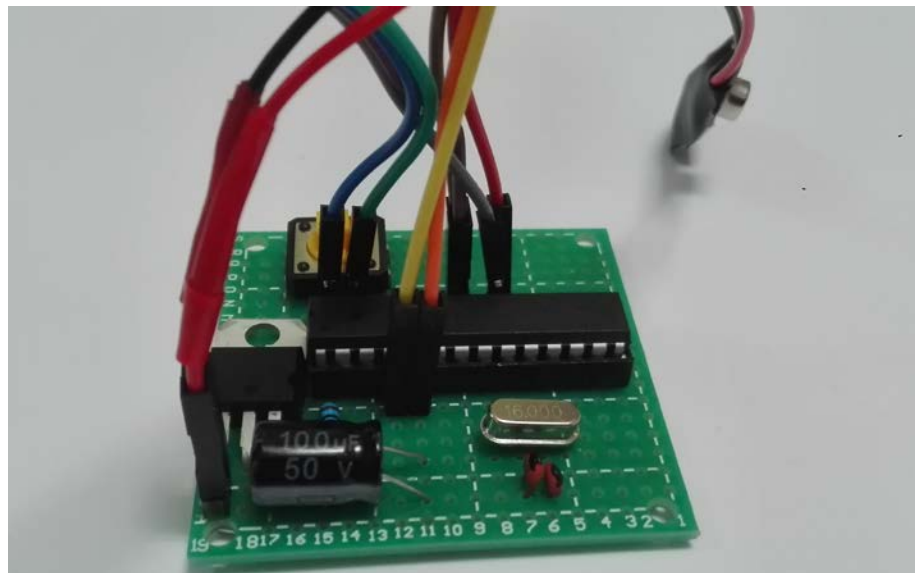


Figura 5.6 - Placa electrónica con las conexiones externas realizadas. Fuente: propia

### 5.3 Bootloader Arduino

Como no vamos a utilizar el Arduino al completo, vamos a adquirir un microcontrolador Atmega 328P-PU. Este chip cuando lo compramos viene vacío, podemos programarle algo a través de un programador básico, con lenguaje C, ensamblador o lo que decidamos. En este caso tendríamos que configurar las entradas y salidas, así como abrir los puertos que se necesiten en el programa.

Para evitar tener que realizar todo este trabajo, y además aprovechar el programa en C++ del Arduino, lo que haremos será cargar el Bootloader de Arduino. Un Bootloader es el programa por defecto que va a sustentar todo el código, es un gestor de arranque con un conjunto de instrucciones que es necesario que tenga el microcontrolador para que funcione el programa de Arduino.

Para grabar el Bootloader primeramente debemos conectar el Arduino a una protoboard, y en esta estará el microcontrolador Atmega 328P-PU con algunos elementos para que funcione.

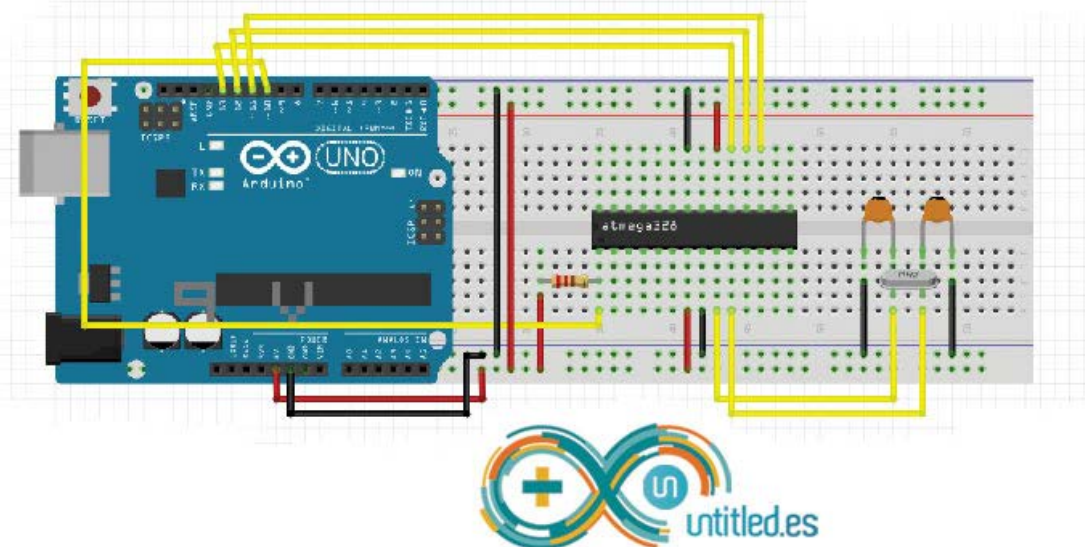


Figura 5.7 - Conexión del circuito para carga de Bootloader. Fuente: [untitled.es](https://untitled.es)

Conectamos el Arduino al PC y abrimos el IDE de Arduino. Vamos a Archivo, ejemplos y hacemos clic en ArduinoISP. Después cargamos el programa al Arduino.

- **Primer paso:** Cuando se haya cargado el programa hacemos clic en herramientas, vamos a programador y seleccionamos "Arduino as ISP".



- **Segundo paso:** Ahora vamos a herramientas de nuevo y hacemos clic en "Quemar Bootloader". Esperamos y ya tendremos nuestro microcontrolador preparado para utilizarlo como un Arduino.

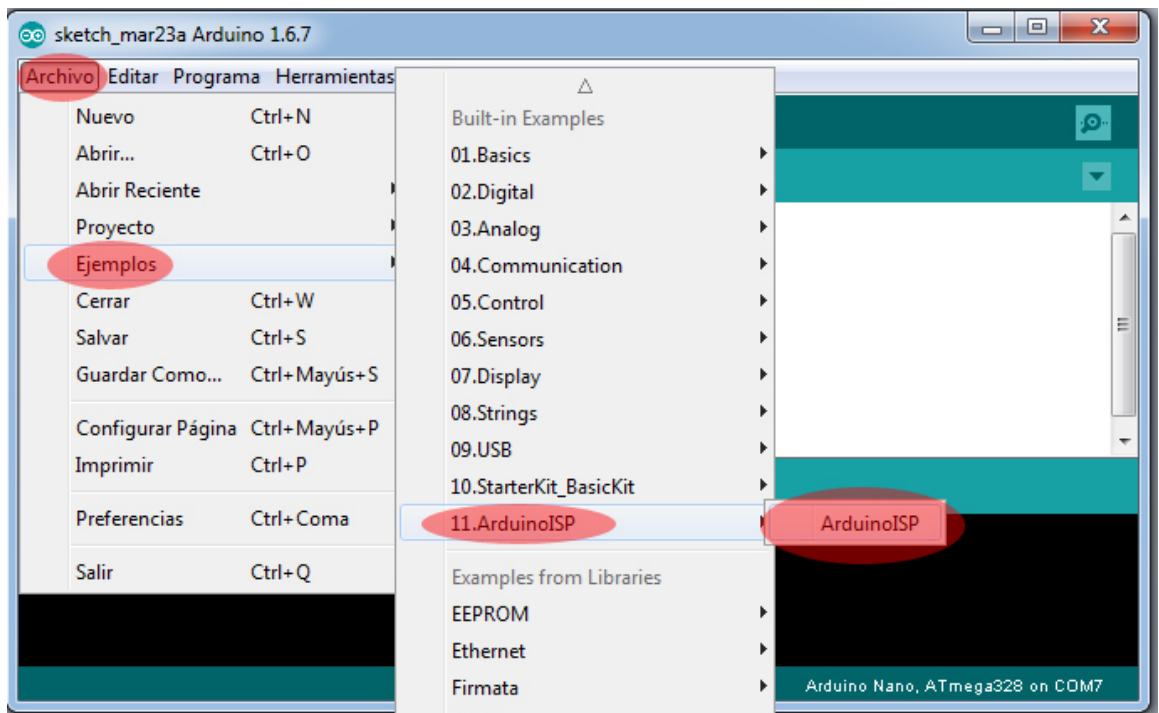


Figura 5.8 - Carga de Bootloader primer paso. Fuente: minitronica.com

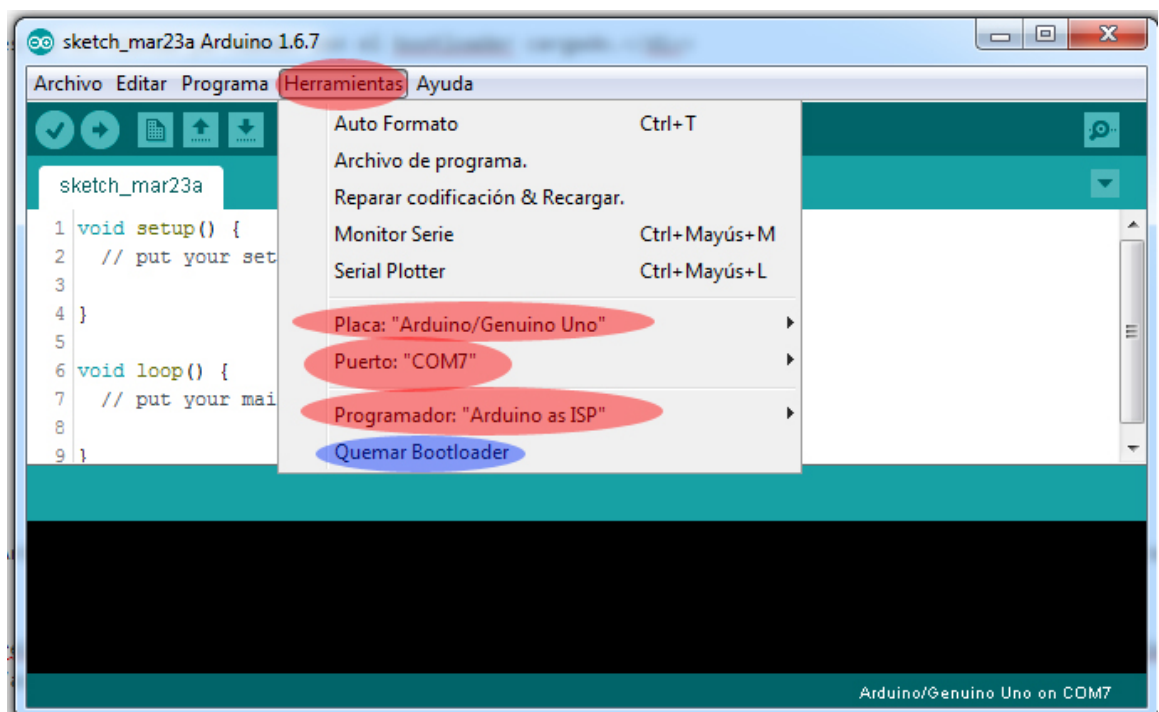


Figura 5.9 - Carga de Bootloader segundo paso. Fuente: minitronica.com

En nuestro caso estamos muy locos y hemos comprado 5 microcontroladores Atmega 328P-PU, así que cargaremos el Bootloader a todos ellos.

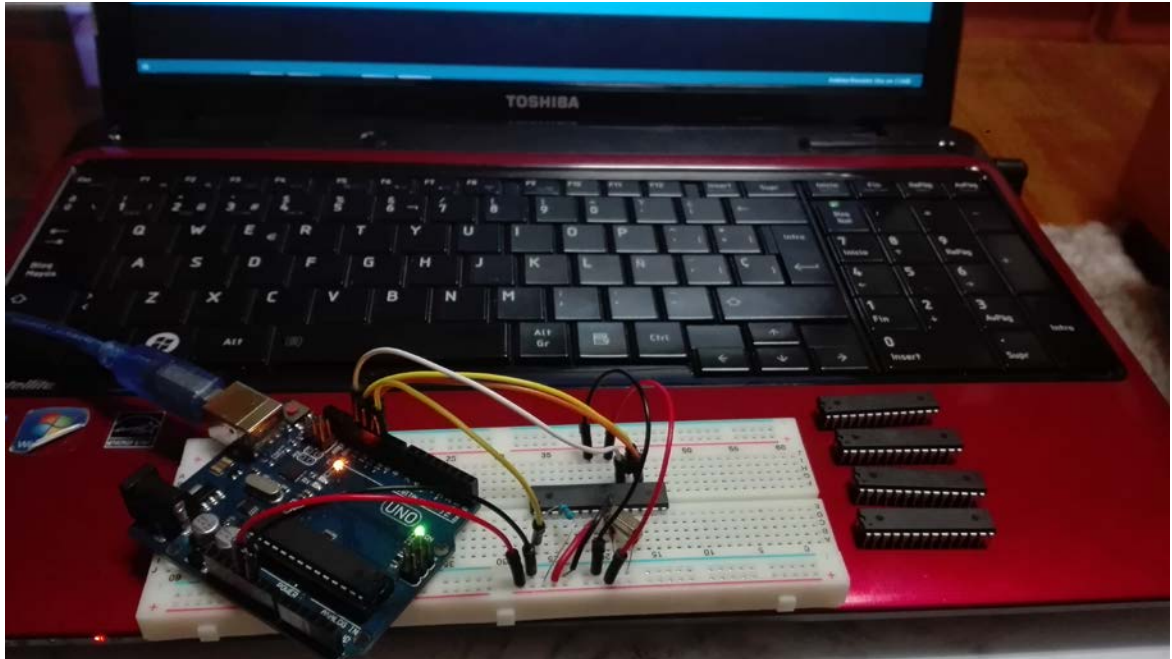


Figura 5.10 - Carga de Bootloader a varios Atmega 328P-PU. Fuente: propia

## 6 Parte mecánica

A continuación os mostramos las famosas piezas de plástico que componen a nuestro mágico encoder.

### 6.1 Carcasa circuito

Como su nombre indica esta pieza será el soporte de nuestro circuito electrónico, aquí irá conectado todo.

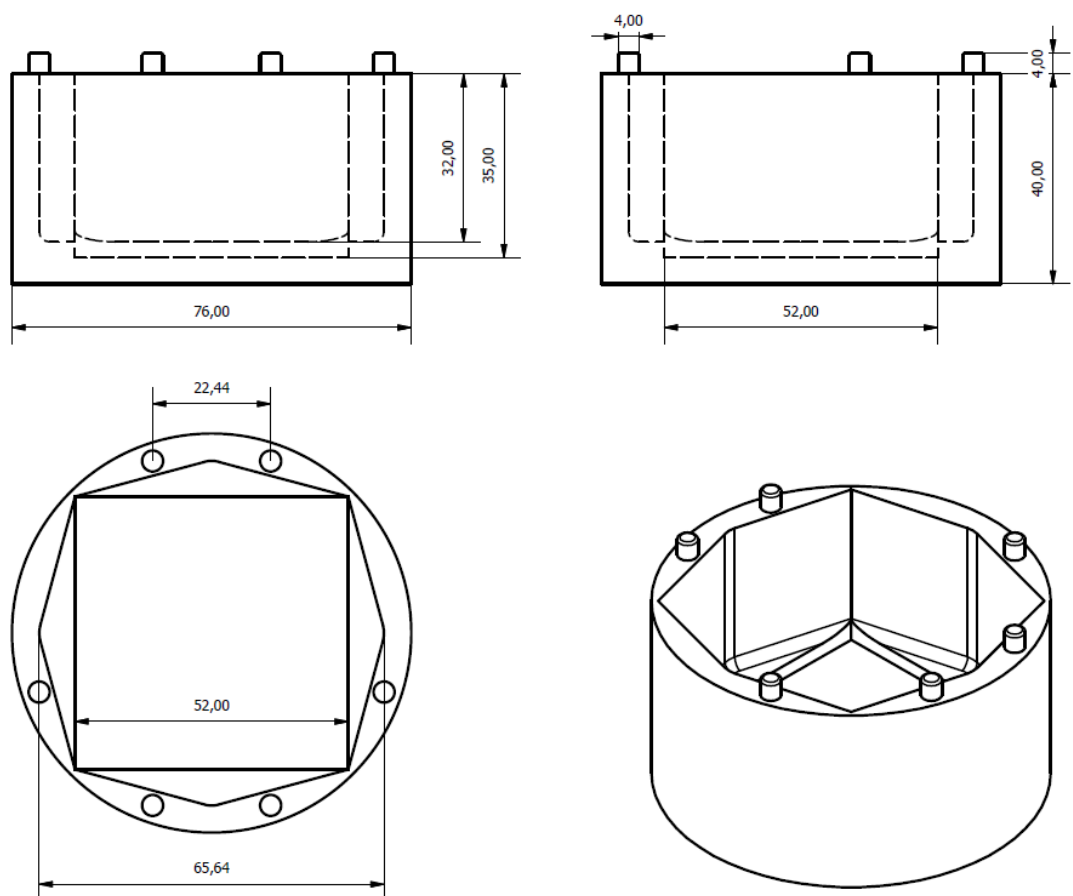


Figura 6.1 - Carcasa circuito. Fuente: Propia

### 6.2 Carcasa inferior

Dentro de esta pieza va encajado todo el sistema mecánico del encoder. Tiene un agujero en la parte inferior por el que pasan los cables del encoder y los de la pantalla.

En su interior va alojado también el codificador rotatorio. Este va apoyado en un pequeño saliente y atornillado a través de dos agujeros previamente realizados en la carcasa. Los tornillos que usaremos serán los de la cinta métrica al desmontarla, ya que encajan perfectamente.

Esta es una de las partes más importantes porque solamente con esta ya tendríamos todo el soporte para hacer funcionar el encoder lineal al completo.

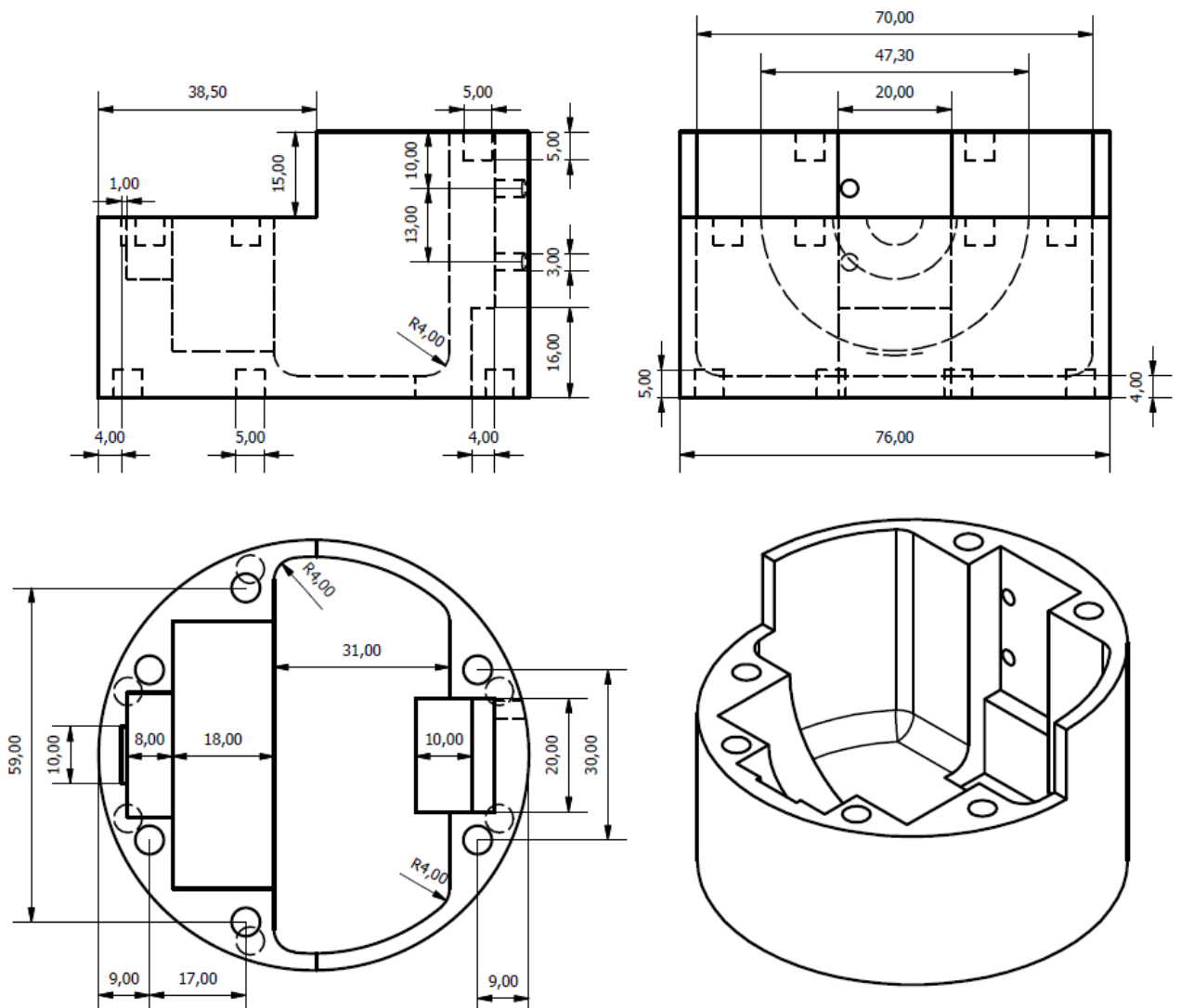


Figura 6.2 - Carcasa inferior. Fuente: Propia

### 6.3 Carcasa superior

El siguiente elemento tiene una función doble: no solamente cierra completamente el encoder protegiendo el sistema mecánico, sino que también hace de soporte para la pantalla.

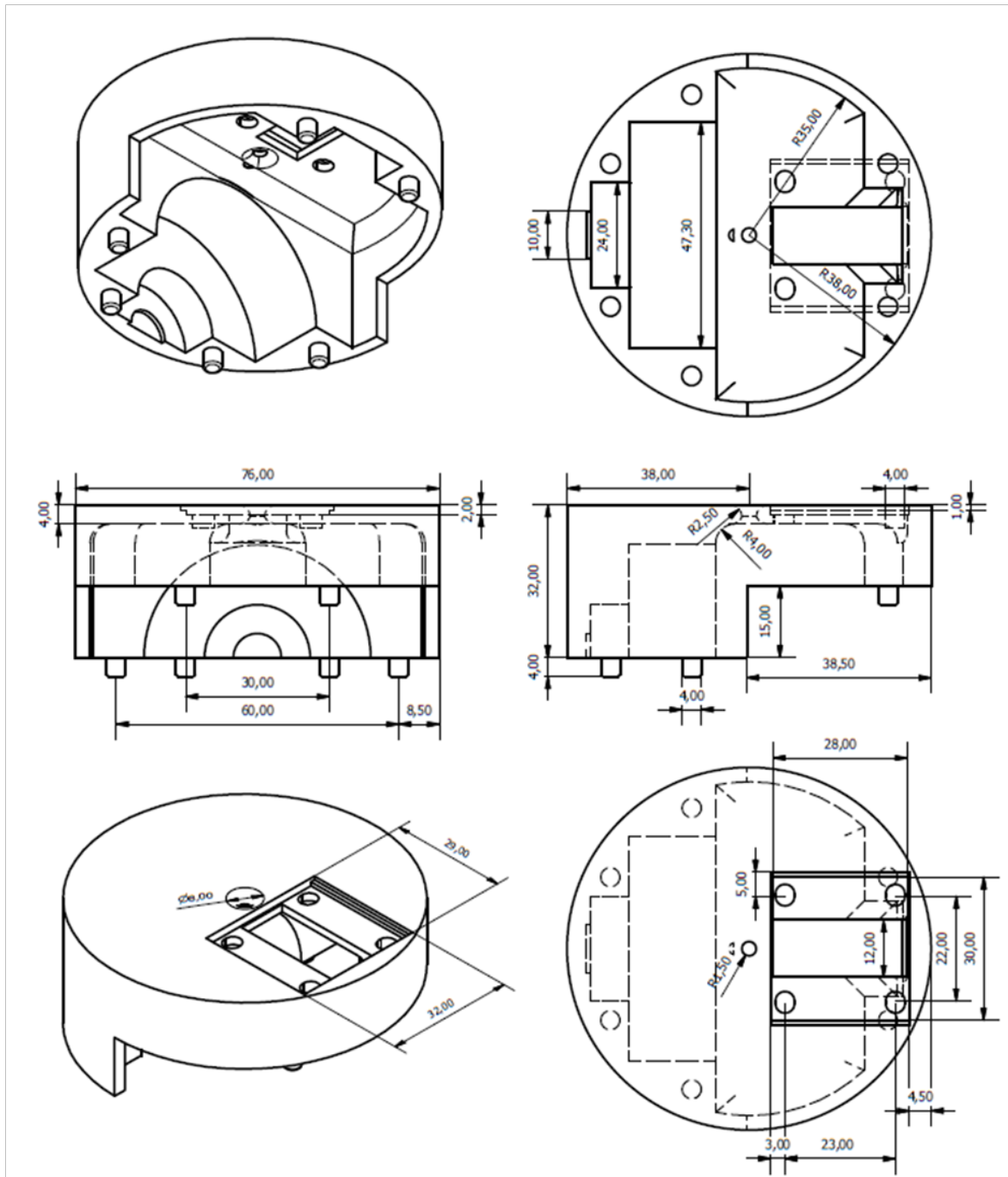


Figura 6.3 - Carcasa superior. Fuente: Propia



## 6.4 Eje

En este encoder solamente hay un eje, este comunicará directamente el movimiento giratorio del carrete con el del codificador y con el muelle de torsión.

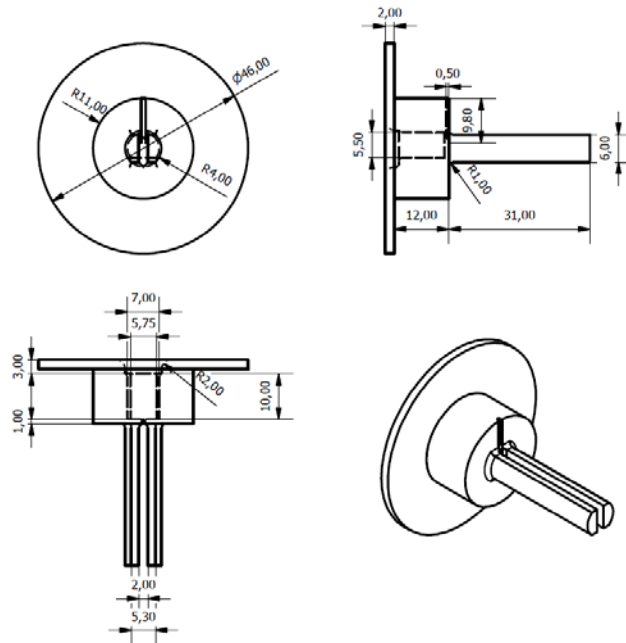


Figura 6.4 - Eje. Fuente: Propia

## 6.5 Tapa eje

Con esta pieza cerraremos el eje creando un carrete donde enrollar el hilo.

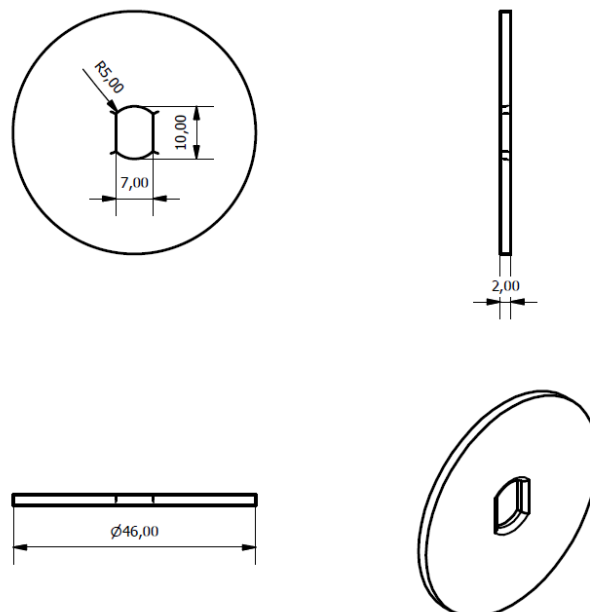


Figura 6.5 - Tapa eje. Fuente: Propia

## 6.6 Soporte muelle de torsión

Con esta pieza fijaremos la parte externa del muelle de torsión a los agujeros que tiene en uno de sus lados, y a través del otro agujero redondo pasará el eje secundario que se sujetará al centro del muelle. Mientras el eje gire, en el muelle se acumulará tensión que luego se liberará al dejar de tirar del hilo. Esta pieza ha sido creada por Openbarbell y la hemos modificado un poco para que encaje mejor en nuestro caso.

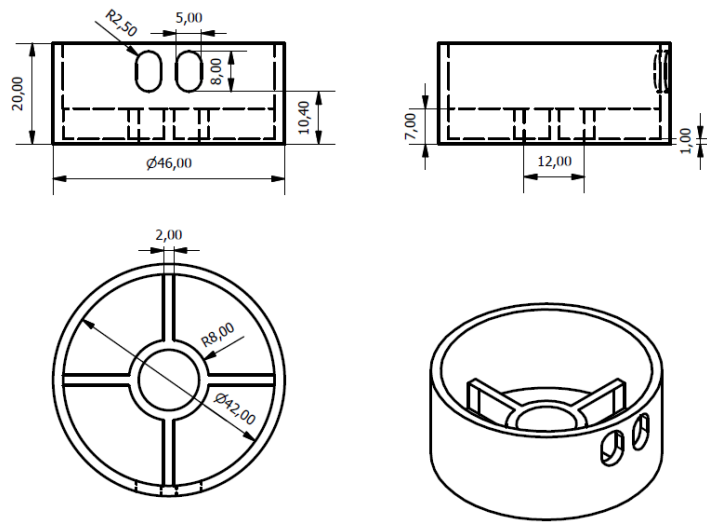


Figura 6.6 - Soporte muelle de torsión. Fuente: Propia

## 6.7 Enganche

Esta pieza es crucial, gracias a esto podremos unir nuestro hilo a la barra. De esta manera el encoder estará al completo ya que podremos detectar el movimiento. Esta pieza ha sido creada enteramente por Openbarbell.

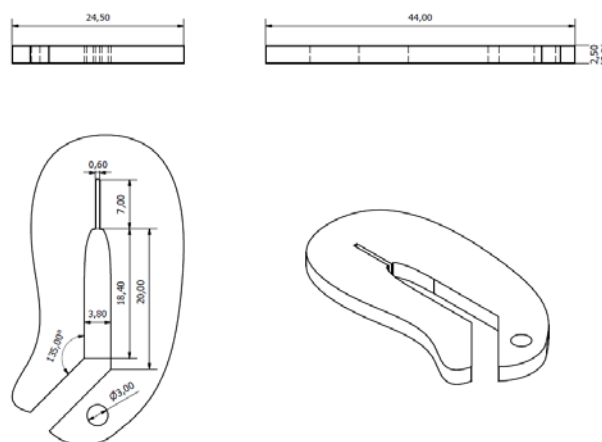


Figura 6.7 - Enganche. Fuente: Openbarbell

## 6.8 Tapa pantalla

Gracias a esto fijaremos la pantalla a la parte superior del encoder. Además, la protegeremos de los golpes, así no se romperá fácilmente.

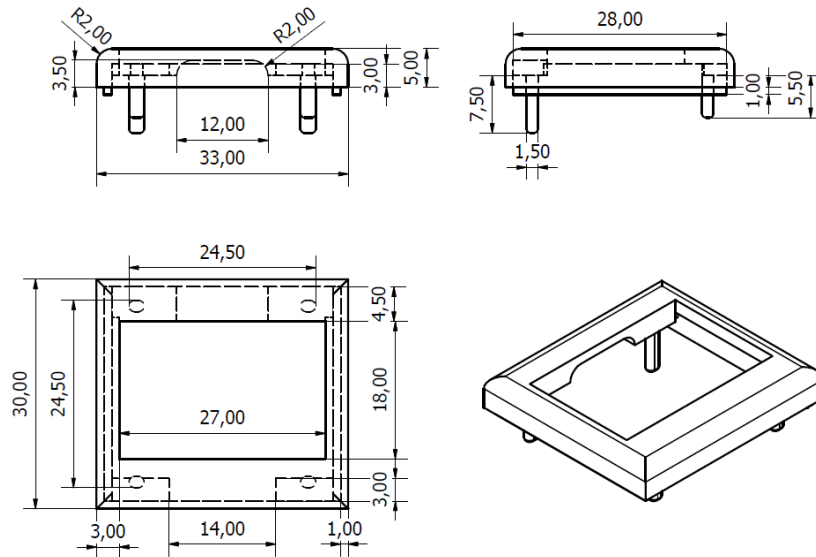


Figura 6.8 - Tapa pantalla. Fuente: Propia

Con todo esto ya estarían todas las piezas. Ahora pasaremos a mostrar algunas imágenes del encoder, tanto en 3D con Autodesk Inventor, como fotografías del encoder y de las piezas ya impresas.

## 6.9 Detalle carrete y soporte muelle de torsión

Se puede ver perfectamente que esos elementos van encajados y que el codificador rotatorio va insertado en el carrete, al lado contrario del eje.

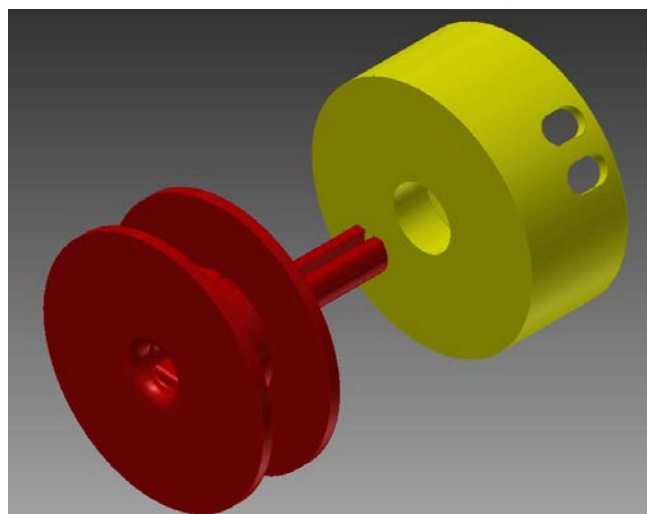


Figura 6.9 - Detalle carrete y soporte muelle de torsión. Fuente: Propia

### 6.9.1 Encoder completo abierto 3D

En la imagen siguiente veremos el encoder despiezado. Está dispuesto de tal manera que se sobreentiende en qué lugar va cada pieza.

En verde tendríamos la carcasa inferior, la superior y la del circuito electrónico. En rojo está el carrete con su eje, en amarillo el soporte del muelle de torsión y en morado la tapa de la pantalla.

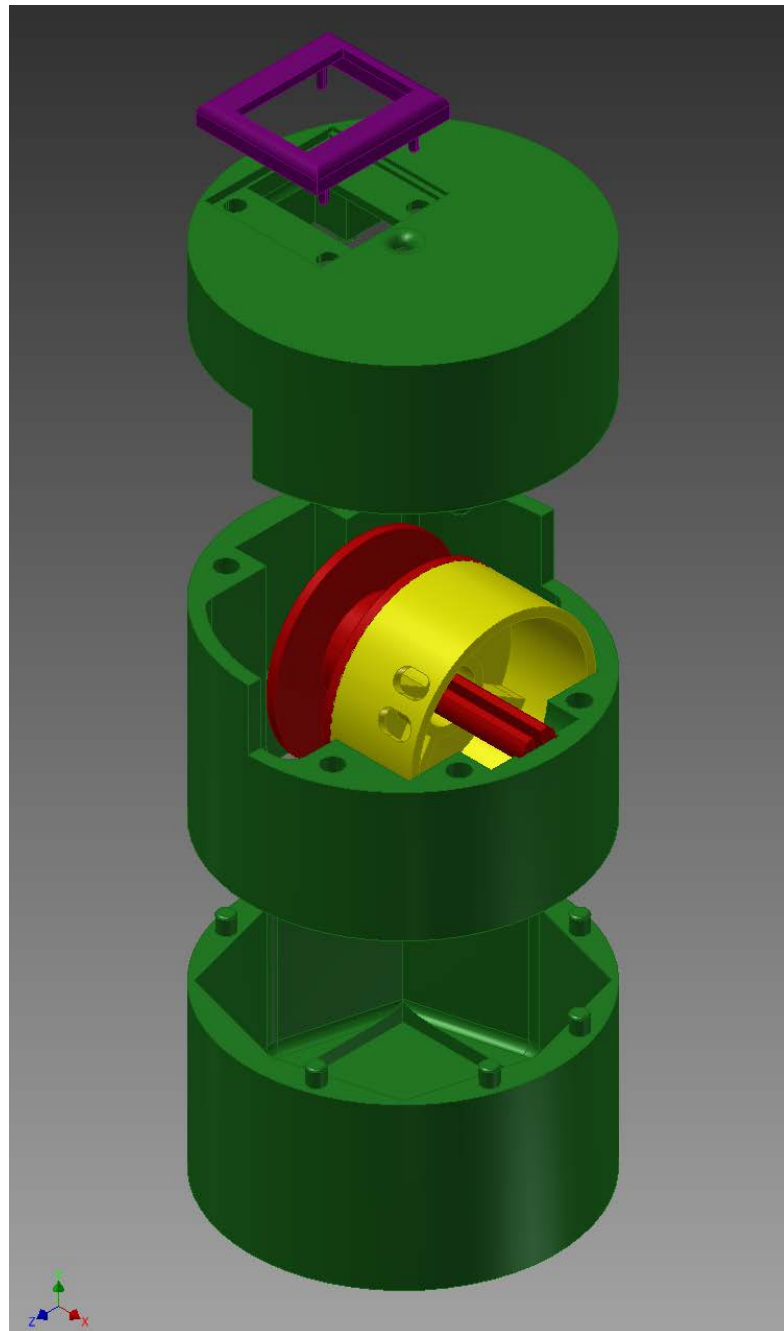


Figura 6.10 - Imagen 3D del despiece del encoder. Fuente: Propia

### 6.9.2 Encoder completo cerrado 3D

Ahora pasamos a cómo sería el encoder montado. Podemos observar que queda hermético y compacto. Además, tanto el circuito electrónico como el mecanismo mecánico quedan completamente protegidos, no como con el primer encoder. Nótese la ausencia del enganche en todos los diseños en 3D, ya que no parecía importante en estos casos

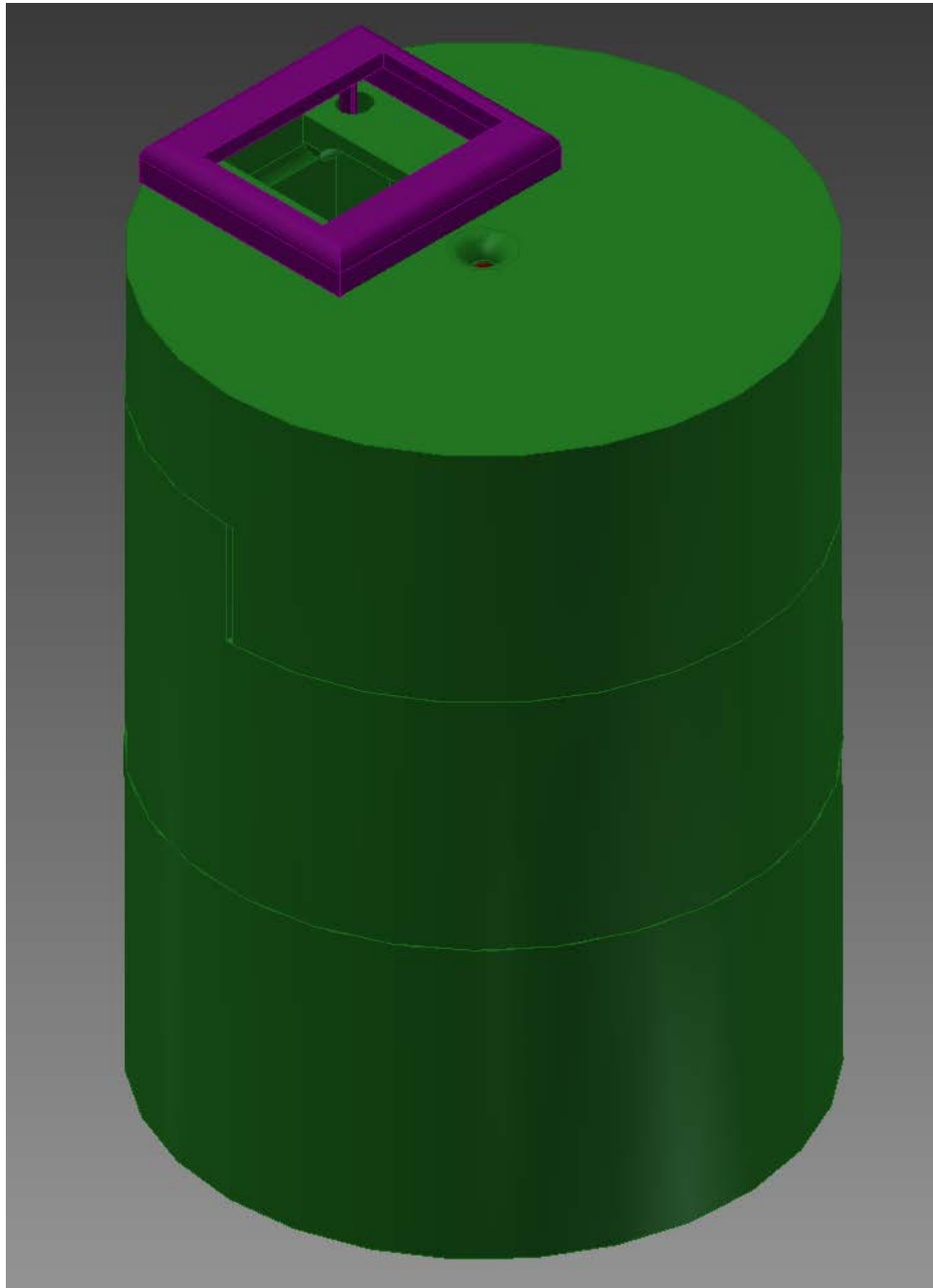


Figura 6.11 - Imagen 3D del encoder cerrado. Fuente: Propia

### 6.9.3 Lote de piezas encoder

Una vez exportamos nuestros diseños a STL podemos enviarlos a una impresora 3D para que los fabrique. Cuando hayamos impreso todas las piezas nos encontraremos con algo así:



Figura 6.12 - Lote de piezas impresas Braincoder. Fuente: Propia

Podemos ver que hay un par de piezas diferentes a los diseños, concretamente la carcasa del circuito electrónico y la tapa de la pantalla.

En cuanto al primero simplemente fue un error en el primer diseño, al imprimirse se pudo ver que no cabía todo dentro y hubo que aumentar la altura. La tapa de la pantalla era poco sólida, así que se aumentó la cantidad de plástico y así la pieza queda más robusta.

#### 6.9.4 Encoder completo

En la siguiente imagen se muestra el encoder ya completamente montado y en funcionamiento:



Figura 6.13 - Braincoder completo en funcionamiento. Fuente: Propia



## 7 Montaje encoder

En este capítulo mostraremos paso a paso cómo se monta el encoder 2. Es un proceso sencillo así que no nos detendremos mucho en los detalles. Primeramente necesitamos un muelle de torsión, utilizaremos una cinta métrica como la de la foto. La desmontaremos.



Figura 7.1 - Despiece cinta métrica. Fuente: Propia

Ahora retiraremos todas las piezas para así poder extraer el muelle de torsión con su carcasa. Después, con unos alicates, romperemos su carcasa y sacaremos el muelle de torsión.



Figura 7.2 - Extracción muelle de torsión. Fuente: Propia

Una vez retiramos el muelle lo introducimos en el soporte del muelle de torsión.



Figura 7.3 - Instalación muelle de torsión. Fuente: Propia

A continuación cogemos el eje y le atamos el hilo. Hay que tener especial cuidado con el sentido en el enrollamos el hilo, así como en qué sentido se enrollaría nuestro muelle de torsión. Después acoplaremos con cuidado el eje al muelle.

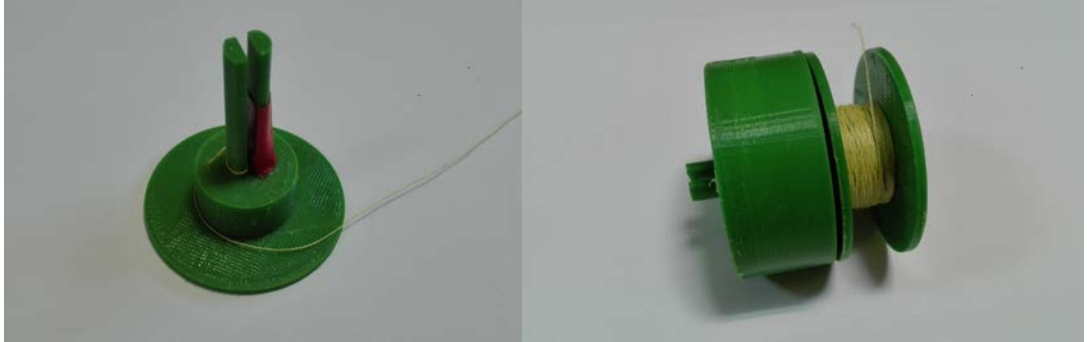


Figura 7.4 - Acoplamiento eje con muelle de torsión. Fuente: Propia

Insertamos el codificador rotatorio en el dorso del eje y lo atornillamos a este; todo ello lo encajamos en la carcasa inferior del encoder. Previamente hemos introducido todos los cables necesarios.



Figura 7.5 - Instalación elementos mecánicos. Fuente: Propia

Encajamos el circuito en su carcasa y esta a la carcasa inferior del encoder.



Figura 7.6 - Parte inferior. Fuente: Propia

Le ponemos la tapa a la pantalla, la conectamos y cerramos el encoder con la carcasa superior. Por último hacemos un nudo en el enganche con el extremo del hilo que sobra. El resultado final es el siguiente:



Figura 7.7 - Braincoder. Resultado final. Fuente: Propia

## 8 Presupuesto

Primeramente tenemos que hacer cálculos, evidentemente el resultado saldrá muy superior al que tendremos si pagamos a empresas de fabricación en serie para que nos hagan todas las piezas necesarias y el circuito electrónico.

Tabla 8.1 - Presupuesto aproximado Braincoder. Fuente: propia

Elemento	Precio
Atmega 328P-PU	3,00 €
Zócalo Atmega 328	0,30 €
Oscilador 16 MHz	0,25 €
2 Condensadores 22 pF	0,15 €
Condensador 100 µF	0,25 €
Regulador LM7805	0,50 €
Resistencia 10 KΩ	0,05 €
Clip para pila de 9V	0,20 €
Pila de 9V	2,00 €
Placa PCB perforada	0,50 €
Codificador rotatorio	2,00 €
Pantalla OLED	7,00 €
260g de plástico PLA o ABS	5,20 €
Cinta métrica	2,50 €
3 metros de hilo de kevlar	0,60 €
Rodamiento ABEC7	1,50 €
Loctite	3,00 €
Cables y conectores	3,00 €
<b>Total</b>	<b>32,00 €</b>

Evidentemente los precios del PLA y del ABS son diferentes, así que hemos redondeado al alza, hemos calculado su valor en 20 € el kg. El pegamento es un valor aproximado, ya que no gastamos un bote entero.

## 9 Futuras mejoras

Enumeraremos algunos de los cambios que haremos a nuestro encoder, así como todo lo que podemos añadirle para mejorar la experiencia del usuario. Cuánto más apoyo recibamos, más cambios haremos 😊

- Haremos que el encoder se pueda montar y desmontar fácilmente, con piezas que encajen entre sí y se fijen si queremos.
- Cubriremos el encoder un cilindro de metacrilato, gracias al cual aumentaremos la resistencia mecánica de este aparato en gran medida.
- Le acoplaremos un módulo de Bluetooth para así conectarlo con un teléfono móvil, también crearemos una app para registrar nuestros levantamientos.
- Cambiaremos completamente el sistema de medida ya que el codificador rotatorio que tenemos actualmente no es muy preciso.
- Colocaremos unos imanes en la base del encoder para que quede fijo en un disco en el suelo mientras se utiliza.
- Cambiaremos el enganche por unos imanes que se fijen a la barra.
- Haremos algunos cambios en la disposición de todos los elementos del encoder: colocaremos un interruptor en la parte superior de este y una batería en su interior. Podremos también disponer de un enchufe micro USB para cargar la batería siempre que lo deseemos.
- Instalaremos un muelle de goma en la salida del hilo; así, si soltásemos de golpe al tirara del hilo, este no se rompería.
- Y, durante todo es te proceso, actualizaremos el programa en varias ocasiones para optimizar más las medidas.

## Lista de referencias

- [1] G-SE. Encoder lineal. Qué es y para qué sirve. Tipos de encoder. Disponible en: <https://g-se.com/encoder-lineal-bp-L57cfb26e815cd>
- [2] Abm-industrial. Qué es un encoder. Disponible en: <http://www.abm-industrial.com/2013/02/07/que-es-un-encoder/>
- [3] Demaquinasyherramientas. Tipos de encoder. Disponible en: <http://www.demaquinasyherramientas.com/mecanizado/encoder-tipos>
- [4] Youtube. Cómo funciona un encoder rotatorio y cómo usarlo con Arduino. Disponible en: <https://www.youtube.com/watch?v=v4BbSzJ-hz4>
- [5] Makershopbcn. Diferencia entre ABS y PLA. Disponible en: <https://makershopbcn.com/abs-vs-pla-que-diferencia-existe-entre-estos-dos-filamentos-para-impresora-3d>
- [6] Squatsandscience. Encoder lineal Openbarbell. Disponible en: <http://squatsandscience.com/openbarbell/>
- [7] Luisllamas. Cómo utilizar Arduino para reprogramar el Bootloader de otro Arduino. Disponible en: <https://www.luisllamas.es/usar-arduino-para-reprogramar-el-bootloader/>
- [8] Thisisbeast. Dispositivo para gestionar y controlar cargas de entrenamiento. Disponible en: <https://www.thisisbeast.com/>