



Figura 3.83: Foto del soporte con el motor atornillado.



Figura 3.84: Foto de los elementos atornillados a la plataforma giratoria.

Ya solo falta acoplar los microscopios en los soporte y la plataforma ya esta acabada mecánicamente.



Figura 3.85: Foto de los elementos atornillados a la plataforma giratoria.

9º Paso: Se une la plataforma a la maquina cartesiana mediante 4 tornillos M6X40mm. Es recomendable para realizar esta operación utilizar algún material auxiliar como apoyo.

86 CAPÍTULO 3. DISEÑO E IMPLEMENTACIÓN DEL SISTEMA MECÁNICO

10º Paso: Finalmente, se atornilla la electrónica dentro de la caja protectora.

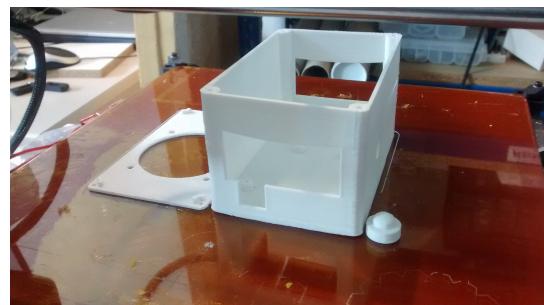


Figura 3.86: Foto de la impresión 3d la caja protectora con la RAMPS.



Figura 3.87: Foto de la caja protectora con la RAMPS.

Con esto se finaliza toda la parte mecánica y la maquina estaría (a falta del cableado) lista para usar. El cableado se explica dentro del sistema electrónica en el capítulo 4. Puede encontrarse un listado con todas las piezas y su coste aproximado en el anexo de Lista de Materiales.



Figura 3.88: Foto del montaje final.



Figura 3.89: Foto 2 del montaje final.

Capítulo 4

Diseño e implementación del sistema de control

El sistema de control es el encargado de ejecutar las órdenes dictadas por el usuario, enviando consignas a los actuadores sistema mecánico. Su esquema se muestra en la figura:

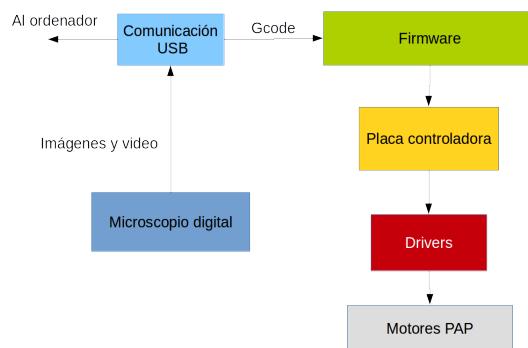


Figura 4.1: Esquema de control del sistema de control.

En este capítulo se analiza las diferentes opciones de control existentes, tanto de la unidad de control (o placa controladora), el firmware de esta placa controladora y sus comandos de control y drivers para los motores paso a paso. Finalmente se analizan las características del microscopio digital elegido, que forma parte del sistema de realimentación.

4.1. La unidad de control

En esta sección se justifica el motivo por el cual se desechó la posibilidad de utilizar la electrónica adquirida con la máquina CNC, el análisis de otras unidades controladoras, las especificaciones técnicas de la placa elegida, la fuente de alimentación a utilizar con esta placa y el cableado realizado.

4.1.1. Electrónica de control por defecto

Con la adquisición de la máquina CNC se incluye una electrónica con las siguientes características:

Especificaciones técnicas de la electrónica

- Control de los tres ejes a través de drivers más variador de frecuencia para control del giro de una fresa.
- Comandos admitidos: G code
- Sistema operativo: Windows 2000 / xp
- Voltaje de alimentación: 220v en C.A
- Drivers de motor paso a paso de 2.5A
- Conexión por puerto paralelo
- Software recomendado: Mach3
- Elementos de protección: Botón de emergencia



Figura 4.2: Imagen de la electrónica por defecto.

El principal problema de la electrónica es su falta de firmware debido a que esta directamente controlada a través del puerto paralelo (estandarizado como IEEE 1284). Este puerto no se incluye hoy en día habitualmente.



Figura 4.3: Imagen 2 de la electrónica por defecto.

La principal consecuencia de esta falta de software es el hecho de que electrónica solo puede operar con un limitado número de programas. En este caso en concreto se recomienda Mach3, un software comercial de pago tipo Gcode-sender. Los programas Gcode-sender se encarga de enviar las instrucciones a la electrónica de control. Mach3 ([84]) permite controlar hasta 6 ejes y genera Gcode a partir de DXF, BMP, etc gracias a LazyCAM, un "slicer" de la misma compañía. Los programas tipo slicer son los encargados de generar Gcode a partir de ficheros con planos, imágenes o figuras 3d.



Figura 4.4: Aspecto de la interfaz Mach3.

92 CAPÍTULO 4. DISEÑO E IMPLEMENTACIÓN DEL SISTEMA DE CONTROL

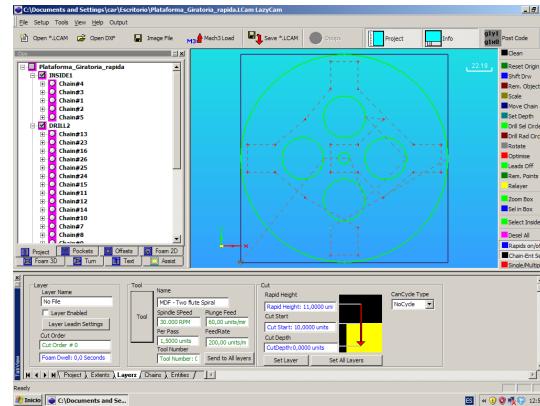


Figura 4.5: Aspecto de la interfaz LazyCam.

Se ha llevado a cabo un análisis exhaustivo del interior y se ha comprobado que a esta placa se le puede añadir un cuarto driver, pensando en un quinto eje mediante motor paso a paso.



Figura 4.6: Imagen de interior de la electrónica por defecto.

Todos estos aspectos hacen que se haya desecharido dicha electrónica y se haya buscado otra más acorde con las especificaciones como se analiza a continuación.

4.1.2. Elección de la placa controladora

Como ya se comentó en los antecedentes, el aumento del número de modelos disponibles de microcontroladores debido a la popularización de éstos, llevó al desarrollo de infinidad de fresadoras e impresoras 3D para el ámbito doméstico. Debido a esto, se tienen a disposición bastantes placas controladoras adecuadas para este proyecto. Se han analizada para tal fin placas controladoras caracterizadas por ser de fuentes abiertas (Open Source) y económicas. Además todas ellas tienen comunicación USB. Entre las placas analizadas se encuentran:

- RAMPS: (Reprap Arduino Mega Pololu Shield) Es como su propio nombre indica, una shield de Arduino Mega. Tiene capacidad para alojar a 5 drivers para motores paso a paso, 3 bucles cerrados para control térmico y capacidad de ampliación: servomotores, pantalla LCD, lector de tarjetas SD, ventiladores, etc.[23]
- RUMBA: Placa todo integrado con capacidades ampliadas. Tiene capacidad para alojar a 6 drivers para motores paso a paso, 5 bucles cerrados para control térmico, puerto específico para LCD y más capacidad de ampliación.[25]
- Grbl: Está diseñado para funcionar en un Arduino UNO, no en un Mega. Capacidad para alojar hasta 4 motores paso a paso. No tiene controles de temperatura.[24]
- SAV-mkI: Placa todo integrado diseñada en Málaga. Tiene capacidad para alojar a 5 drivers para motores paso a paso. Interesante el lector de tarjeta microSD ya integrado.[26]

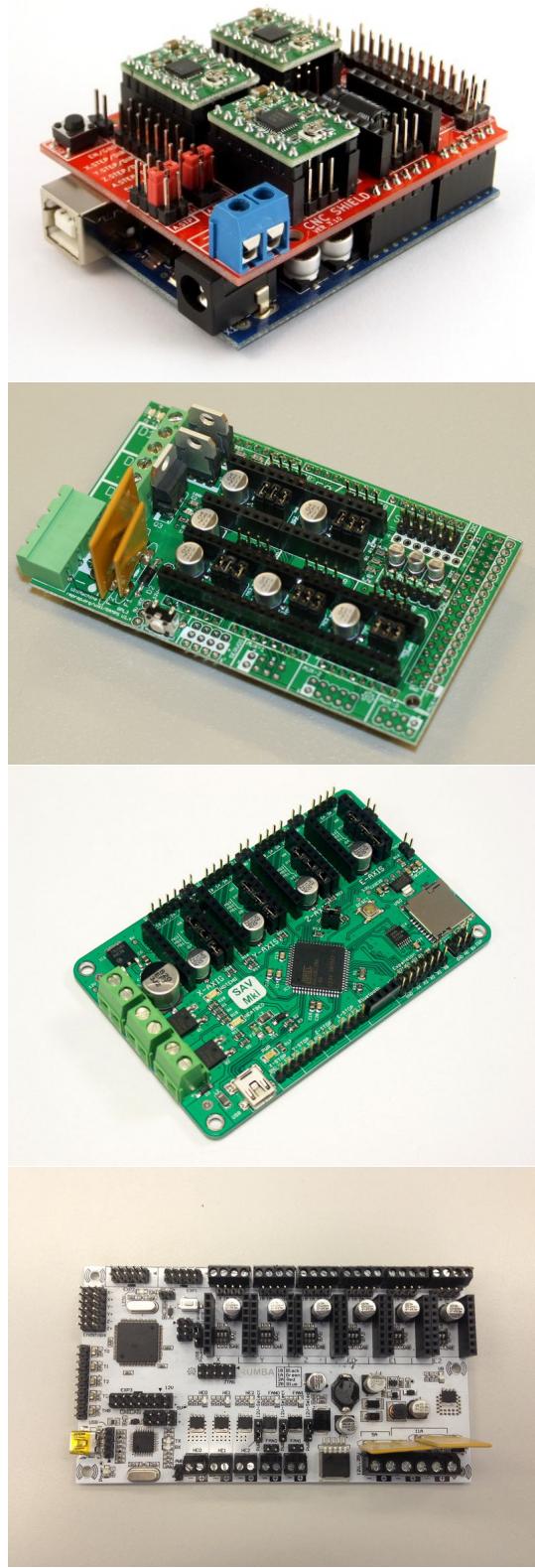


Figura 4.7: Imágenes de las placa analizadas. Fuentes:[23], [24], [25],[26]

96 CAPÍTULO 4. DISEÑO E IMPLEMENTACIÓN DEL SISTEMA DE CONTROL

Se han descartado todas las de 4 por no tener suficientes grados de libertad, ya que la máquina tiene al menos 5 grados de libertad mínimo: 3 para la traslación , una para la plataforma y otro para la herramienta.

Se ha elegido la RAMPS debido a su quinto driver y al hecho de ser una placa muy popular, existe mucha mayor documentación ante posibles eventualidades. Las primeras pruebas para ver si los drivers eran capaces de mover los motores NEMA23 que monta el sistema de posicionamiento cartesiano por defecto se realizaron en una Grbl Shield. Esta placa funciona sobre un Arduino Uno, que además era más económico. Se escogió además por la facilidad de modificación del firmware y del host. Se comentan estas pruebas en el capítulo 5.

La RAMPS es una placa diseñada para el control de impresora 3D y otras máquinas CNC. Se ha utilizado la versión 1.4 que es la más actual. La RAMPS se monta sobre un Arduino Mega 2560, donde está la microcontroladora. Algunas de las especificaciones de estas placas son:

Especificaciones para Arduino MEGA 2560 Rev3 [85] :

- Microcontroladora: ATmega2560
- Voltaje de alimentación: 5V
- Voltaje de entrada(recomendado): 7-12V
- Voltaje de entrada(límite): 6-20V
- Pines Digitales Entrada/Salida: 54 (de los cuales 15 tienen salida PWM)
- Pines Analógicos de entrada: 16
- Corriente para pines E/S Pin 20 mA
- Memoria Flash: 256 KB con 8 KB usado por el bootloader
- SRAM: 8 KB
- EEPROM: 4 KB
- Velocidad de reloj: 16 MHz
- Medidas: 101.52 mm x 53.3 mm
- Peso 37 g



Figura 4.8: Foto de la placa compatible Arduino utilizada.

Como ya se ha comentado anteriormente, la comunicación con el ordenador se realiza a través de USB. El tipo de conector USB que dispone el Arduino Mega es de tipo B (común en impresoras [86]):



Figura 4.9: Foto de USB tipo B.

Podemos citar además algunas especificaciones interesantes de la RAMPS [87]:

- Expandible
- Tres controles de temperatura con mosfets. Uno de ellos con fusible adicional de 11A.
- Fusible de 5A para protección de componentes
- Capacidad para 5 drivers tipo Pololu.
- Pines para comunicaciones I2C y SPI.
- Mosfets son controlados mediante PWM .
- Lectores de tarjeta SD disponibles como expansión
- LEDs indicadores cuando se encienden los controles de temperatura
- Dos conectores para motores en el eje Z.

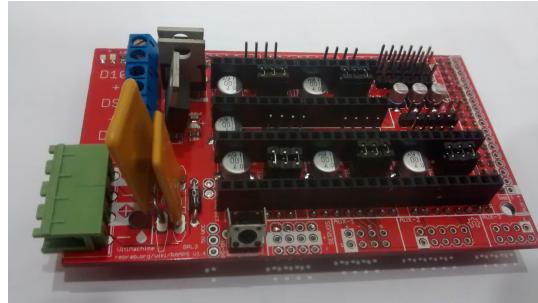


Figura 4.10: Foto de la RAMPS utilizada.

4.1.3. Fuente de alimentación

La RAMPS se alimenta independientemente del Arduino. Esta alimentación se realiza entre 12 y 24 voltios. Para este propósito se ha escogido una fuente de alimentación con formato ATX (habitual en ordenadores de sobremesa). El formato ATX tiene el tamaño 15 cm x 14 cm x 8.6 cm [27].



Figura 4.11: Foto de la fuente ATX.

Además de conectores molex para dispositivos IDE, HDD o SATA entre otros, incluye para CPU P4, combinado para el conector de la placa base de 4 pines a 12V. y ATX2 de 24 pines. Los pines 15 y 16 deben de ser puenteados para que la fuente se mantenga encendida.

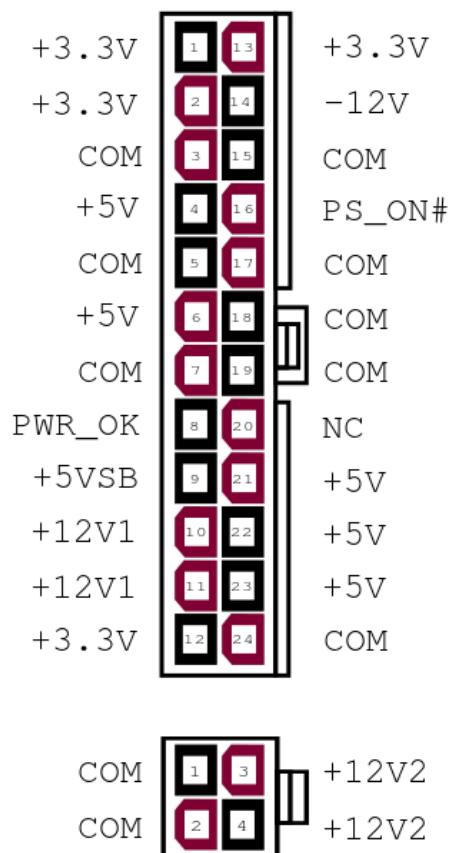


Figura 4.12: Esquema los pines de la fuente. Fuente:[27]

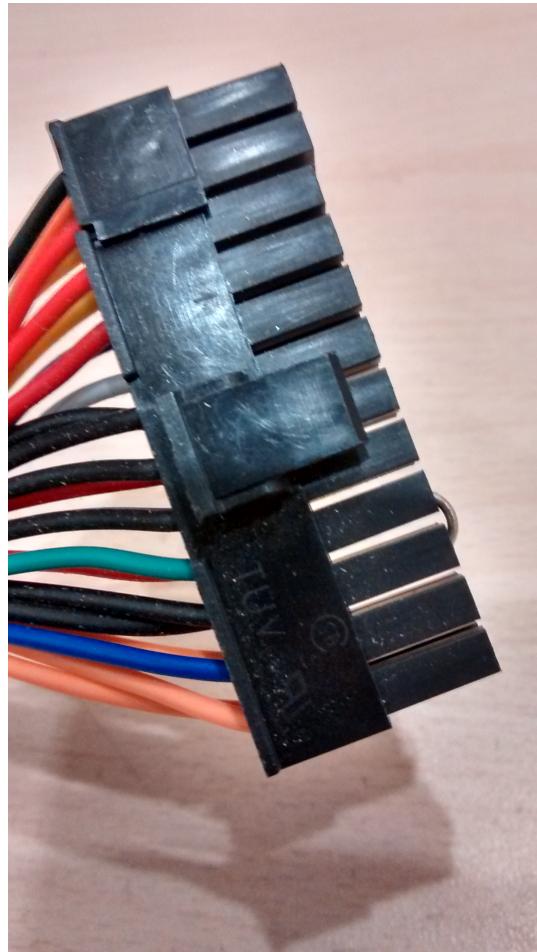


Figura 4.13: Imagen de los pines de la fuente puenteada.

El conector CPU P4 se corta y los cables se conectan a la clema de 4 contactos para alimentación de la RAMPS. Este terminal es capaz de soportar hasta 15A.



Figura 4.15: Soldado de uno de los conectores de seguridad.



Figura 4.14: Imagen de la clema ya cableada.

4.1.4. Cableado

Los conectores de los motores son del modelo GX16-4 ([88]). Es un conector de seguridad, utilizado en aviación. Tiene 4 pines y una muesca para asegurar la posición correcta. Es circular con un diámetro de 16mm. Sin embargo a la RAMPS los motores se conectan mediante pines de 2.54mm. Por ello se ha realizado una conexión con pines ya crimpados:

El adaptador tiene cables de color verde, negro, rojo y blanco. Donde la muesca se dispone en la parte superior, visto desde atrás. La disposición es: en la fila superior derecha amarillo, fila superior izquierda azul, fila inferior derecha rojo y en la fila inferior izquierda verde. No hacer una correcta ejecución de las fases puede conllevar a un cortocircuito y un sobrecalentamiento excesivo en los drivers, aparte de un mal funcionamiento.

En el esquema de conexión se puede comprobar como la mayoría de los pines quedan sin utilizar (se ha modificado el utilizado para impresoras 3D, creado por Neil Underwood [89]), con vista a futuras ampliaciones:

Esquema de conexión

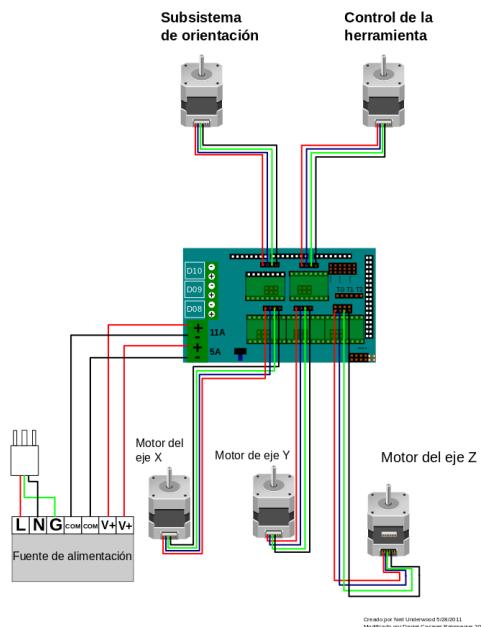


Figura 4.16: Esquema de conexión de la RAMPS modificado.

4.2. Firmware

El firmware es [90] *un bloque de instrucciones de máquina para propósitos específicos, grabado en un chip, normalmente de lectura/escritura (ROM, EEPROM, flash, etc.), que establece la lógica de más bajo nivel que controla los circuitos electrónicos de un dispositivo de cualquier tipo*. Por tanto, el firmware de este proyecto debe la electrónica ya descrita, sirviendo como unión entre la misma y el software que controlará el usuario.

En este apartado se justifica la elección del firmware y la descripción de como llevar a cabo su configuración para adaptarlo a las necesidades del proyecto.

4.2.1. Elección del firmware

Existe una gran multitud de firmwares para impresoras 3D y CNC específico para las placas analizadas en el apartado anterior. Entre los firmwares más populares se pueden encontrar Sprinter (uno de los más populares), Grbl (orientado a CNC, no a impresoras 3D) o Marlin. Marlin es un firmware para impresoras 3D

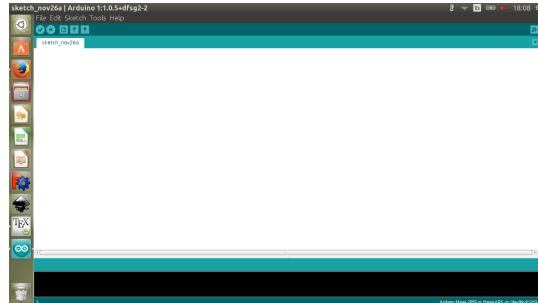


Figura 4.17: Arduino IDE.

y otras máquinas. La microcontroladora controla la entrada/salidas, el voltaje y los movimientos. Fue creada a partir de versiones anteriores de los firmwares Sprinter y Grbl [91].

Algunos de las características interesantes de Marlin ([28]) son:

- Control de Servomotores para posicionamiento de la plataforma: Nivelación automática o Manual de la plataforma
- Rutinas de protección termal
- Administración de memoria EEPROM para restauración de datos
- Configuración de distintos termistores y termopares
- Controladores para LCD y lector SD
- Acepta cinemática Delta, SCARA, y cartesiano
- Comunicación USB
- G-Code ampliado
- Configurable hasta 4 ejes nuevos.

Se va a utilizar el firmware sin modificar, solo debemos configurarlo para nuestra maquina en concreto. Para ello se va a utilizar la IDE de Arduino. El IDE de Arduino compila y envía el programa al microntrolador. Se debe seleccionar el puerto y el tipo de placa en la pestaña correspondiente antes de usarlo.

Para ello debemos irnos a Tools y seleccionar Arduino Mega 2560. Para seleccionar el puerto se debe tener conectada la placa al ordenador. Tras ello se carga el firmware que debemos haber descargado de su pagina en Github [28].

104 CAPÍTULO 4. DISEÑO E IMPLEMENTACIÓN DEL SISTEMA DE CONTROL

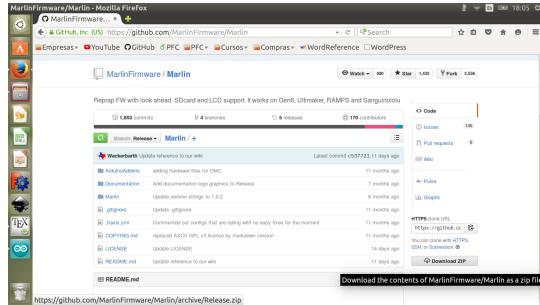


Figura 4.18: Página en Github.Fuente: [28]

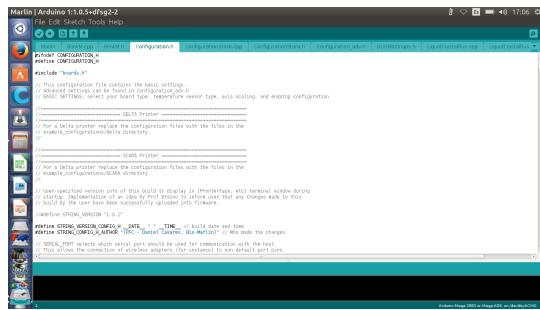


Figura 4.19: Arduino IDE con Marlin cargado.

Se puede apreciar que el lenguaje de Arduino esta basado en Procesing, el cual a su vez se basa en Java. La parte que se va a modificar es en el archivo Config.h. Se selecciona la pestaña y se pasa a describir la forma de configurarlo propiamente dicha.

4.2.2. Configuración principal

El baudrate se pone a 115200 Se utiliza por defecto esta velocidad de comunicación que no es la más alta pero funciona bastante bien. Se elige la placa RAMPS con 2 extrusores (EE) ya que uno va a estar dedicado al posicionamiento angular de la plataforma y otro al posicionamiento angular de la herramienta. Debajo se vuelven a definir 2 extrusores y la fuente de alimentación tipo ATX.

El código debe quedar de la siguiente forma:

```
#define STRING_CONFIG_H_AUTHOR "( Daniel_Casares , _default_config )"
// Who made the changes.

#define BAUDRATE 115200
```

```
#ifndef MOTHERBOARD
#define MOTHERBOARD BOARD_RAMPS_13_EEB
#endif

#define EXTRUDERS 2
//Segun la configuracion elegida este debe ser el numero para no
//dar error.

#define POWER_SUPPLY 1
```

4.2.3. Configuración de los sensores de temperaturas

En las propiedades termales se va a desactivar los sensores para evitar medidas de prevención de encender los motores, ya que estos están conectados. Esto se debe a que en una impresora 3D, la temperatura de extrusión debe de ser la adecuada para no dañar su extrusor.

Se disponen todos los sensores a 0, la temperatura mínima también a 0 y la temperatura máxima a 1000. Se comentan las variables de *PREVENT_DANGEROUS_EXTRUDE* y *PREVENT_LENGTHY_EXTRUDE*. El código queda de la forma:

```
#define TEMP_SENSOR_0 0
#define TEMP_SENSOR_1 0
#define TEMP_SENSOR_2 0
#define TEMP_SENSOR_BED 0

// This makes temp sensor 1 a redundant sensor for sensor 0. If the
// temperatures difference between these sensors is to high the
// print will be aborted.
//#define TEMP_SENSOR_1_AS_REDUNDANT
#define MAX_REDUNDANT_TEMP_SENSOR_DIFF 10

// Actual temperature must be close to target for this long before
// M109 returns success
#define TEMP_RESIDENCY_TIME 10 // (seconds)
#define TEMP_HYSTERESIS 3 // (degC) range of +/- temperatures
// considered "close" to the target one
#define TEMP_WINDOW 1 // (degC) Window around target to
// start the residency timer x degC early.

// The minimal temperature defines the temperature below which the
// heater will not be enabled It is used
// to check that the wiring to the thermistor is not broken.
// Otherwise this would lead to the heater being powered on all the
// time.

#define HEATER_0_MINTEMP 0
#define HEATER_1_MINTEMP 0
#define HEATER_2_MINTEMP 0
#define BED_MINTEMP 0
```