MARLIN PARA TORPES

MARLIN PARA TORPES

OSCAR ASIAIN INSA

www.turtleprinter.com
https://plus.google.com/+Turtleprinter3D
https://twitter.com/Turtle3DPrinter
https://www.facebook.com/turtle3dprinter
OskarAs8 en el grupo de Clone Wars

Edición: 0
Fecha: 20/03/2015

Errores y comentarios -> marlinparatorpes@turtleprinter.com

Imagen de portada: Image courtesy of vectorolie at FreeDigitalPhotos.net

INDICE

0 Introducción

En este documento se analiza todas las funcionalidades del firmware Marlin que se usa en impresoras 3D de filamento fundido FFF (FDM). No es un documento en el que se enseñe a montar y ajustar una impresora 3D, para eso hay tutoriales en la wiki de clonewars. Solo analizaremos como funciona Marlin, como se comunica con el ordenador o como ejecuta las ordenes, además veremos todas las configuraciones posibles. En definitiva, la idea es entender perfectamente cómo funciona Marlin.

Esto es una interpretación libre del firmware Marlin, por supuesto no está exenta de errores y cualquier comentario o error se puede enviar al correo marlinparatorpes@turtleprinter.com. La mayor parte del contenido está traducido de páginas web en Ingles y de los comentarios existentes dentro del propio código de Marlin, además del estudio del código y su funcionalidad.

Marlin es un firmware de código abierto y colaboran muchísimos desarrolladores de todo el mundo mejorando y añadiendo funcionalidades constantemente. A la hora de hacer este documento, Marlin a sufrido una reactivación y actualmente existe una pagina de Wiki de Marlin desde donde podremos acceder a las versiones estables y a las ultimas modificaciones. Este documento está realizado con la ultima versión estable, que es la versión 1.0.2.

No es necesario saber de programación C o C++, aunque ayudaría a la comprensión del código.

Básicamente solo hay que saber 4 cosas:

1. Las diferentes funciones o parámetros estarán habilitados (activados) o deshabilitados (desactivados), para ello se usa la opción que nos da el código C++ de comentar las líneas. Una línea comentada empieza por una barra doble "// " lo que hará que el código escrito a continuación no se tenga en cuenta y por tanto estará deshabilitada (desactivada) esa función. En cambio el código que no empiece con la barra doble estará habilitado (activado).

   Por ejemplo, si queremos habilitar una función que esta deshabilitada (esta la línea comentada con //) solo tendremos que quitar la doble barra // y de forma contraria si queremos deshabilitar alguna función bastara con añadir // al principio de la línea y ya no se tendrá en cuenta.

   ```
   //#define ESTA_FUNCION_ESTA_DESHABILITADA

   #define ESTA_FUNCION_ESTA_HABILITADA
   ```

2. Hay que tener en cuenta 4 instrucciones básicas para entender los archivos de configuración. Son directivas del compilador y son instrucciones que le dicen al compilador que código tiene que incluir en el programa y cual no. El compilador es el programa que traduce el código del firmware al lenguaje que entiende el micro controlador. Las instrucciones son:

   - #define. Esta es la directiva principal y le dice a Marlin que lo que viene a continuación está definido, habilitado o declarado. Además tenemos la opción de darle un valor que almacenara como si fuera una variable. Veamos un ejemplo:

     #define MEDIRDIAMETRO

     En este caso la función *MEDIRDIAMETRO* esta habilitada

     #define ALTURA 25

     Aquí tenemos activado el parámetro *ALTURA* y además guarda el valor 25. Realmente lo que el compilador hace a la hora de compilar el programa es que cada vez que se encuentre el parámetro ALTURA dentro del código lo cambiara por el valor 25.

   - #undef, Esta directiva es lo contrario de #define. Si tenemos definida una opción o asignado un valor lo deshabilitara. Sería como si no se hubiera definido inicialmente la opción.

     #undef MEDIRDIAMETRO

     Si teníamos habilitada la función *MEDIRDIAMETRO,* a partir de este momento será como si no la hubiéramos habilitado, es decir, estar deshabilitada.

   - #ifdef #endif. Si el parámetro incluido en #ifdef está definido o declarado se tendrá en cuenta el código que hay entre esta directiva y el final #endif.

     ```
     #ifdef MEDIRDIAMETRO
       #define ACTIVARSENSOR
       #define DIAMETRO 1.75
     #endif
     ```

     Si la función *MEDIRDIAMETRO* esta habilitada previamente, se ejecutara el código que hay a continuación, es decir, se habilita la función *ACTIVARSENSOR* y se configura el parámetro DIAMETRO a 1.75.

- #ifndef #endif. Es la directiva contraria a la anterior. Si el parámetro incluido en #ifndef no está definido o declarado se tendrá en cuenta el código que hay entre esta directiva y el final #endif.

```
#ifndef MEDIRDIAMETRO
    #undef ACTIVARSENSOR
    #define DIAMETRO 0
#endif
```

Si la función *MEDIRDIAMETRO* **no** esta habilitada previamente, se ejecutara el código que hay a continuación, es decir, se deshabilitara la función *ACTIVARSENSOR* y se configura el parámetro DIAMETRO a 0.

3. La directiva #ifdef se emplea principalmente para configurar los parámetros necesarios para poder usar una función ya definida.

```
#define FUNCION_DECLARADA

#ifdef FUNCION_DECLARADA
    #define PARAMETRO_UNO 34
    #define PARAMETRO_DOS 45
    #define PARAMETRO_TRES_DECLARADO
#endif
```

También las directivas #ifdef y #ifndef se suelen emplear para ampliar definiciones, es decir, si tenemos habilitada una función o declarado cierto valor, añadimos configuraciones extras. En los archivos de configuración de Marlin, se añade a menudo este tipo de ampliaciones de definiciones. Cuando tengamos una en el código lo indicare y estará en rojo ya que es mejor no modificarlas.

```
#ifdef FUNCION_DECLARADA
    #define DECLARAMOS_UNA_FUNCION
    #define DECLARAMOS_OTRA_FUNCION
    #undef NO_QUIERO_TENER_DECLARADA_ESTA_FUNCION
#endif
```

4. Algunas funciones que habilites, necesitaran de librerías del sistema extras, consulta la documentación de Arduino para saber cómo incluir librerías nuevas en su IDE. Esto pasa sobre todo con el uso de los LCD, algunos precisan de librerías concretas para funcionar.

Para dar más claridad, el código que se incluye esta en 2 colores, si esta en azul es código que se puede modificar y si esta en rojo no se debería de tocar.

Si has de modificar varios parámetros u opciones, cambia uno solo cada vez y haz pruebas de funcionamiento, ya que si algo no funciona bien, no sabrás exactamente que código del que has modificado te está dando problemas.

1 ¿Que es Marlin?

Para que el software se comunique con nuestra impresora, esta deberá llevar un firmware (programa) en el micro controlador que la gestiona. Marlin es un Firmware para impresoras reprap que también funciona en equipos Ultimaker. Soporta impresión desde tarjeta SD y planificación anticipada de las trayectorias. Este firmware es una combinación entre Sprinter, grbl y mucho código original. Marlin se publica bajo licencia GPL.

El desarrollo de Marlin se ha reactivado recientemente, y se está trabajando para corregir todos los problemas encontrados. Actualmente existe una pagina de Wiki de Marlin desde donde podremos acceder a las versiones estables y a las ultimas modificaciones. Es importante que te descargues la ultima versión estable, y no la versión en desarrollo que puede incluir bugs en el funcionamiento. La ultima versión estable la puedes encontrar en la Wiki de Marlin en el apartado “Marlin Releases”.

1.1 Configurar y compilar Marlin

1. Instala la última versión no beta del IDE de Arduino desde: http://www.arduino.cc/en/Main/Software
2. Abre el repositorio de Marlin https://github.com/MarlinFirmware/Marlin/releases
3. Descárgate la ultima versión liberada pulsando en " **Source code** (zip)".
4. Algunas placas o LCD, necesitan librerías especiales en el directorio de Librerías de Arduino, consulta la documentación de tu Hardware.
5. Inicia el IDE de Arduino.
6. Configura todos los parámetros necesarios para el funcionamiento de tu impresora. En apartados posteriores te enseño como hacerlo.
7. Selecciona Herramientas/Placa/Arduino Mega 2560 (o el modelo de tu placa).
8. Elige el puerto de comunicación correcto en Herramientas/Puerto serie.
9. Abre mediante el IDE de Arduino el archivo Marlin.pde o Marlin.ino.
10. Pulsa en el botón Verificar/compilar.
11. Pulsa el botón subir, si todo está bien el firmware se cargara en el micro controlador de tu placa.

1.2 Principales características

- Interrupción basada en el movimiento con aceleración lineal real.
- Alta frecuencia de pasos.
- Planificación anticipada de movimientos. Mantiene una velocidad alta cuando es posible. Alta velocidad en curvas.

  Marlin cuenta con una planificación avanzada de movimientos, sólo acelerará y desacelerara la velocidad, de modo que el cambio en la velocidad vectorial es menor que la velocidad xy_jerk (parámetro configurable). Esto sólo es posible, si algunos movimientos futuros ya están procesados, de ahí el nombre. De este modo se produce menos sobre-deposición de plástico en las esquinas, especialmente en ángulos planos.

- Control de temperatura por interrupciones.

  Una de las interrupciones del micro controlador se utiliza para gestionar las conversiones ADC, y comprobar así las temperaturas críticas. Esto produce menos bloqueos en la rutina de gestión de la temperatura.

- Soporte preliminar para el algoritmo avanzado de Matthew Roberts
- Control completo de EndStops
- Admite impresión desde tarjetas SD
- Soporta carpetas en la tarjeta SD

  Si tienes un lector de tarjetas SD conectado a tu placa controladora, ahora también funcionaran las carpetas. El listado de los archivos en Pronterface mostrará "/path/subpath/file.g". Además, las copias de seguridad de algunos sistemas operativos estarán ocultas, así como los archivos no terminados en ".g".

- Inicio automático de la impresión desde la tarjeta SD

  Si se coloca un archivo auto[0-9].g en la raíz de la tarjeta SD, se ejecutará automáticamente cuando se enciende la impresora. El mismo archivo se ejecutará también mediante la opción "Autostart" en el menú del LCD. En primer lugar se imprimirá auto0.g después auto1.g y así sucesivamente. De esa manera, puedes pre-calentar o incluso imprimir automáticamente sin intervención del usuario.

- Soporte de display LCD (idealmente 20x4).
- Sistema de menú LCD para la impresión autónoma desde tarjetas SD, controlado por un encoder rotativo.

  Mediante el uso de un LCD, Se puede ajustar en tiempo real la temperatura, aceleraciones, velocidades, caudales, seleccionar e imprimir archivos de la tarjeta SD, precalentar, desactivar los drivers de los motores, y muchas más funciones.

- Almacenamiento de los parámetros principales en la EEPROM

  Si conoces los valores PID, la aceleración, velocidades máximas, etc. de tu impresora, puedes configurarlos, y almacenarlos en la memoria EEPROM mediante el uso de comandos Gcode. Después de cada reinicio, se cargaran desde la EEPROM, independiente de lo que este configurado en el archivo configuration.h.

- Muchas pequeñas, pero útiles funcionalidades procedentes de la versión de Bkubicek.
- Soporte para hacer Arcos con los Gcodes G2 y G3

  Slic3r puede encontrar curvas que, aunque estén divididas en segmentos, se juntan para describir un arco. Marlin es capaz de imprimir estos arcos. La ventaja es que el firmware puede elegir la resolución, y puede realizar el arco con una velocidad casi constante, dando como resultado un mejor acabado. Además, se necesitara menos comunicación serie.

- Sobre muestreo de Temperatura

  Para reducir el ruido y hacer que el control PID sea más preciso, se realiza un promedio de las lecturas del conversor ADC.

- Temperatura dinámica "AutoTemp"

  Si un gcode contiene muchos cambios en las velocidades del extrusor, o en tiempo real cambiar la velocidad de la impresión, la temperatura debe ser adecuadamente ajustada. Por lo general, una velocidad mayor requiere una temperatura más alta. Esto se puede configurar con la función AutoTemp, mediante el Gcode M109 S <mintemp> B <maxtemp> F <factor>. Más información en el apartado correspondiente al Gcode M109

- Soporte para QTMarlin, una interfaz gráfica de usuario en fase beta para ajuste de los parámetros PID, pruebas de velocidad y de aceleración.
- Envío de informes de disparo de los EndStop al host.

  Si un EndStop es pulsado mientras se realiza algún movimiento, las coordenadas en las cuales piensa el firmware que se pulso son enviadas por el puerto serie. Esto es útil, ya que el usuario recibe un mensaje de advertencia. También herramientas como QTMarlin pueden utilizarlo para encontrar combinaciones aceptables de velocidad + aceleración.

- Librería sdcardlib actualizada.
- Informes de potencia de los elementos calefactores. Útil para el control PID.
- Control de temperatura PID
- Soporta CoreXY]
- Soporta impresoras tipo Delta
- Soporta impresoras tipo SCARA
- Admite carro X Dual para impresoras con varios extrusores
- Puerto serie configurable para soportar la conexión de adaptadores inalámbricos.

- Funcionamiento automático de los ventiladores de refrigeración de los extrusores según la temperatura del HotEnd.
- Soporta Servos RC, pudiendo especificar el ángulo o la duración de la rotación continua.
- Auto nivelado de la superficie de impresión.
- Soporta un sensor de diámetro del filamento, para ajustar el volumen de extrusión.

La velocidad de comunicación del puerto serie por defecto es 250.000. Esta velocidad tiene menos fluctuaciones y errores que los habituales 115.200, pero es menos compatible con algunas placas controladoras.

2 Configuraciones mínimas necesarias para empezar

Vamos a ver cuales son los parámetros mínimos que tenemos que configurar en Marlin para que funcione nuestra impresora. Los valores recomendados son indicados para una impresora sencilla, con un extrusor, con o sin HeatedBed. Más o menos lo que sería una Prusa I3, ni una impresora Corexy ni Delta. Es **obligatorio** ajustar o revisar cada uno de los parámetros aquí descritos.

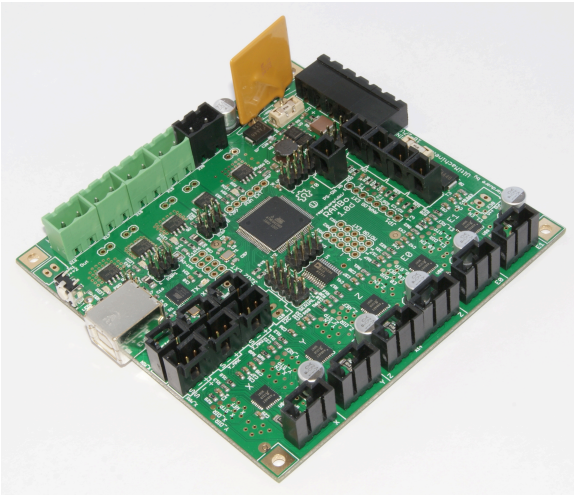
Estas son algunas guías de configuración para calibrar tu impresora:

http://reprap.org/wiki/Calibration
http://youtu.be/wAL9d7FgInk
http://calculator.josefprusa.cz
http://reprap.org/wiki/Triffid_Hunter%27s_Calibration_Guide
http://www.thingiverse.com/thing:5573
https://sites.google.com/site/repraplogphase/calibration-of-your-reprap
http://www.thingiverse.com/thing:298812

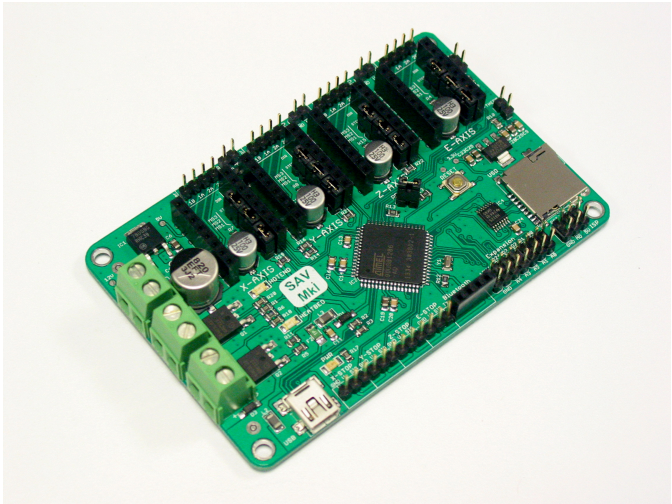
2.1 Tipo de placa base.

Configuraremos el modelo de placa base que tenemos. Este dato es usado por Marlin para asignar los pines correspondientes al Layout (conexiones exteriores) de la placa y para configuraciones adicionales.

Ejemplos de placas controladoras:



RAMBO



SAV MKI

```
#ifndef MOTHERBOARD
 #define MOTHERBOARD BOARD_ULTIMAKER
#endif
```

Seleccionaremos el nombre de nuestra placa según el listado siguiente, este listado esta también en el archivo boards.h, consúltalo antes de cambiar el nombre de la placa por si ha habido algún cambio:

- BOARD_GEN7_CUSTOM 10 // Gen7 custom
- BOARD_GEN7_12 11 // Gen7 v1.1, v1.2
- BOARD_GEN7_13 12 // Gen7 v1.3
- BOARD_GEN7_14 13 // Gen7 v1.4
- BOARD_CHEAPTRONIC 2 // Cheaptronic v1.0
- BOARD_SETHI 20 // Sethi 3D_1
- BOARD_RAMPS_OLD 3 // MEGA/RAMPS up to 1.2
- BOARD_RAMPS_13_EFB 33 // RAMPS 1.3 / 1.4 (Power outputs: Extruder, Fan, Bed)
- BOARD_RAMPS_13_EEB 34 // RAMPS 1.3 / 1.4 (Power outputs: Extruder0, Extruder1, Bed)
- BOARD_RAMPS_13_EFF 35 // RAMPS 1.3 / 1.4 (Power outputs: Extruder, Fan, Fan)
- BOARD_RAMPS_13_EEF 36 // RAMPS 1.3 / 1.4 (Power outputs: Extruder0, Extruder1, Fan)
- BOARD_DUEMILANOVE_328P 4 // Duemilanove w/ ATMega328P pin assignments
- BOARD_GEN6 5 // Gen6
- BOARD_GEN6_DELUXE 51 // Gen6 deluxe
- BOARD_SANGUINOLOLU_11 6 // Sanguinololu < 1.2
- BOARD_SANGUINOLOLU_12 62 // Sanguinololu 1.2 and above
- BOARD_MELZI 63 // Melzi
- BOARD_STB_11 64 // STB V1.1
- BOARD_AZTEEG_X1 65 // Azteeg X1
- BOARD_MELZI_1284 66 // Melzi with ATmega1284 (MaKr3d version)
- BOARD_AZTEEG_X3 67 // Azteeg X3
- BOARD_AZTEEG_X3_PRO 68 // Azteeg X3 Pro
- BOARD_ULTIMAKER 7 // Ultimaker
- BOARD_ULTIMAKER_OLD 71 // Ultimaker (Older electronics. Pre 1.5.4. This is rare)
- BOARD_ULTIMAIN_2 72 // Ultimainboard 2.x (Uses TEMP_SENSOR 20)
- BOARD_3DRAG 77 // 3Drag Controller
- BOARD_K8200 78 // Vellemann K8200 Controller (derived from 3Drag Controller)
- BOARD_TEENSYLU 8 // Teensylu
- BOARD_RUMBA 80 // Rumba
- BOARD_PRINTRBOARD 81 // Printrboard (AT90USB1286)
- BOARD_BRAINWAVE 82 // Brainwave (AT90USB646)
- BOARD_SAV_MKI 83 // SAV Mk-I (AT90USB1286)
- BOARD_TEENSY2 84 // Teensy++2.0 (AT90USB1286)

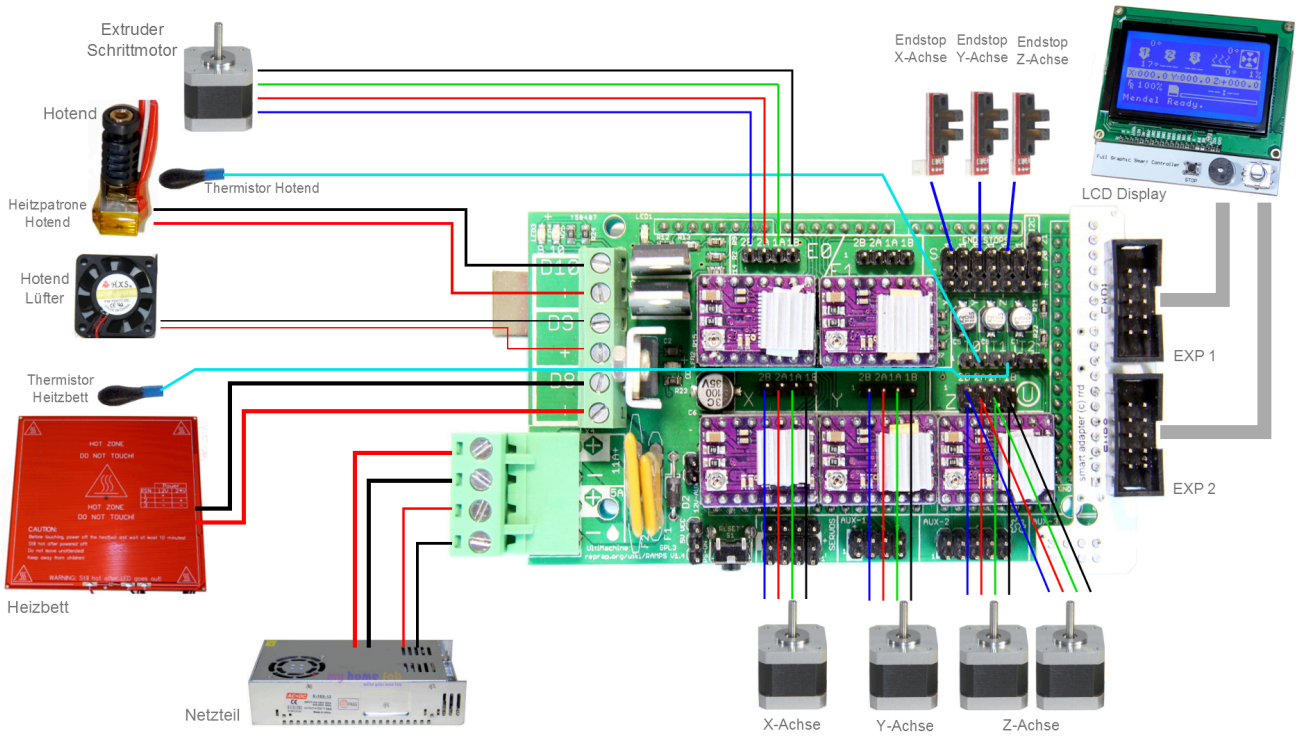
- BOARD_GEN3_PLUS 9 // Gen3+
- BOARD_GEN3_MONOLITHIC 22 // Gen3 Monolithic Electronics
- BOARD_MEGATRONICS 70 // Megatronics
- BOARD_MEGATRONICS_2 701 // Megatronics v2.0
- BOARD_MEGATRONICS_1 702 // Minitronics v1.0
- BOARD_MEGATRONICS_3 703 // Megatronics v3.0
- BOARD_OMCA_A 90 // Alpha OMCA board
- BOARD_OMCA 91 // Final OMCA board
- BOARD_RAMBO 301 // Rambo
- BOARD_ELEFU_3 21 // Elefu Ra Board (v3)
- BOARD_5DPRINT 88 // 5DPrint D8 Driver Board
- BOARD_LEAPFROG 999 // Leapfrog

Por defecto está configurado al modelo *BOARD_ULTIMAKER* que corresponde a una placa Ultimaker. **Es muy importante cambiarlo e introducir el nombre correspondiente a nuestra placa antes de conectar la placa a la tensión de alimentación, ya que podríamos dañar el micro controlador.**

No vamos a incluir todas las opciones disponibles (consulta el listado en el propio código del archivo boards.h en Marlin), ya que además constantemente se están añadiendo modelos nuevos. Pero si que hay que comentar las siguientes opciones:

#define BOARD_RAMPS_13_EFB 33 // RAMPS 1.3 / 1.4 (Power outputs: Extruder, Fan, Bed)
#define BOARD_RAMPS_13_EEB 34 // RAMPS 1.3 / 1.4 (Power outputs: Extruder0, Extruder1, Bed)
#define BOARD_RAMPS_13_EFF 35 // RAMPS 1.3 / 1.4 (Power outputs: Extruder, Fan, Fan)
#define BOARD_RAMPS_13_EEF 36 // RAMPS 1.3 / 1.4 (Power outputs: Extruder0, Extruder1, Fan)

En este caso siempre estamos usando una placa Ramps 1.3 o 1.4, pero lo que cambia es la configuración de las 3 salidas de potencia que tiene la placa. Por ejemplo la opción 33 nos permite conectar un extrusor, un ventilador de capa, y la HeatedBed. El orden que sigue es empezando desde la salida más alejada de la conexión de alimentación, en este caso sería la D10 para el extrusor, la D9 para el ventilador y la D8 para la HeatedBed.



2.2 Numero de extrusores

Número de extrusores que tiene la impresora.

#define EXTRUDERS 1

Por defecto es 1 que es lo normal. Pero si nuestra impresora tiene 2 o 3 tendremos que cambiarlo si queremos que funcionen todos, admite hasta 4 extrusores. Internamente Marlin numera los extrusores con índice 0, es decir, si tenemos por ejemplo 2 extrusores, el extrusor 0 será el primero y el extrusor 1 el segundo, y así consecutivamente. En todo el código y Gcodes, se emplea el índice 0 para referirse a un extrusor en concreto.

2.3 Definir el tipo de sensor de temperatura

En la parte de ajustes térmicos, hay que especificar el tipo de sensor de temperatura que tenemos, tanto en nuestro Hotend como en la HeatedBed (si disponemos de ella). Podemos especificarlo hasta en 3 extrusores y 1 HeatedBed.

En el código de Marlin, en el archivo Configuration.h, justo antes de estas directivas, tenemos una tabla con las distintas opciones.

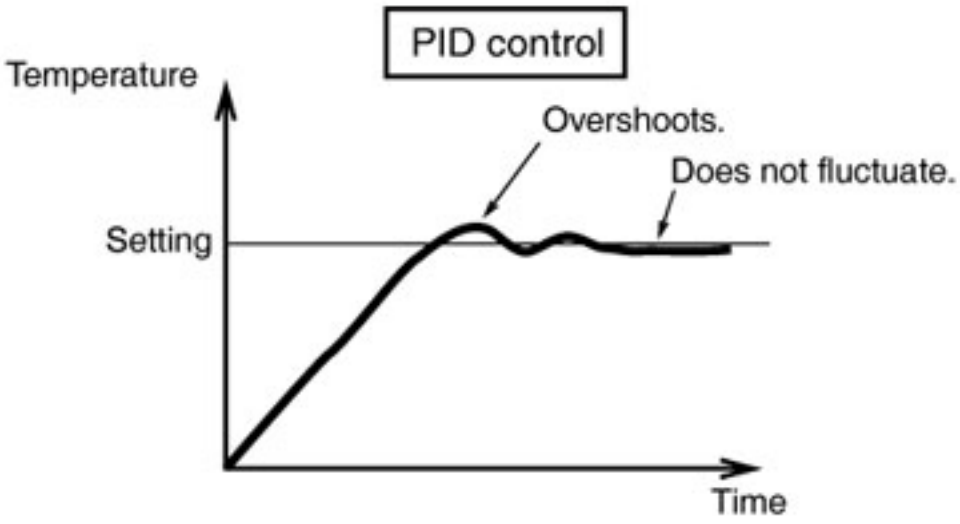
```
#define TEMP_SENSOR_0 -1
#define TEMP_SENSOR_1 -1
#define TEMP_SENSOR_2 0
#define TEMP_SENSOR_BED 0
```

Es muy importante seleccionar el adecuado, por ello, revisa la documentación de tu Impresora y si no está, consulta con el fabricante o distribuidor para saber que tipo de sensor tienes instalado. Una mala configuración puede ocasionar que la temperatura de control medida no sea la real y se queme alguna parte del extrusor o se destruya el propio sensor de temperatura, ya que por ejemplo muchos termistores no soportan más de 300 grados. Normalmente el termistor mas difundido es de 100K de EPCOS con una configuración en nuestra placa de Pullup 4,7Kohm, así que tendremos que poner el 1 que corresponde con este tipo de termistor. Si no vamos a utilizar algún extrusor, lo pondremos a 0 (no lo comentes con //). Recuerda configurar también si tienes HeatedBed el tipo de sensor que lleve, y si no tienes ponlo a 0.

2.4 Parámetros PID

El control de temperatura PID es un sistema que mediante el uso de tres parámetros ajusta la temperatura para evitar variaciones bruscas. Es un sistema de control complejo que se usa en lugar del clásico On-OFF típico de los termostatos. Mas información sobre el control PID Aquí

Ajustaremos los parámetros necesarios para que el control de temperatura PID funcione y regule perfectamente el calentamiento del extrusor. Para que el PID sea correcto, si miramos la grafica de temperatura que sale en Pronterface, veremos una rampa que sube rápidamente y que conforme llega a la temperatura objetivo va ralentizándose, llegando suavemente y manteniéndola con la mínima variación de grados. Un pequeño pico nada más llegar a la temperatura objetivo no es deseable, pero a veces es difícil no tenerlo. Lo más importante es que después de ese pico la temperatura se mantenga constante, se pueden conseguir variaciones mínimas de ± 0.2 grados, pero cada Hotend es distinto.

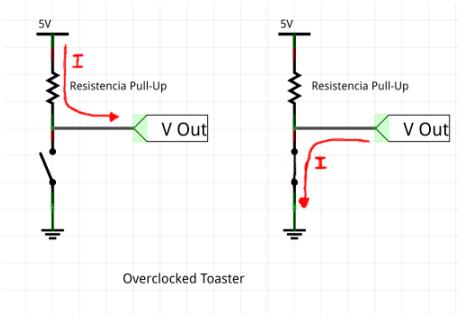



```
// Ultimaker
  #define  DEFAULT_Kp 22.2
  #define  DEFAULT_Ki 1.08
  #define  DEFAULT_Kd 114

// MakerGear
//   #define  DEFAULT_Kp 7.0
//   #define  DEFAULT_Ki 0.1
//   #define  DEFAULT_Kd 12

// Mendel Parts V9 on 12V
//   #define  DEFAULT_Kp 63.0
//   #define  DEFAULT_Ki 2.25
//   #define  DEFAULT_Kd 440
```

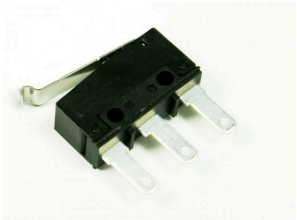
Podemos ver que en Marlin se han dejado descomentados (activos) los valores correspondientes a un Hotend Utimaker y además se incluyen los parámetros para un Hotend MakeGear y para un V9 de Mendel Parts. Lo más normal es que estos valores no nos sirvan, así que vamos a calcular los nuestros propios con el Gcode M303. Introduciremos en la consola serie o pronterface el Gcode M303, mira en el apartado correspondiente a los Gcodes para ver cómo funciona. Tras ejecutarlo nos dará los parámetros Kp, Ki y Kd correspondientes a nuestro Hotend y sustituiremos los valores en la sección de Ultimaker (que ya no emplearemos). Más información en http://reprap.org/wiki/PID_Tuning

2.5 EndStop PullUp.

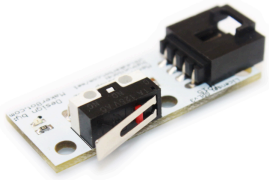
Las resistencias PullUp se emplean en electrónica digital para confirmar el estado de una entrada, mas información en http://ovtoaster.com/resistencias-pulldown-y-pullup/.



Arduino (el micro Atmel) tiene interiormente resistencias pullup en sus entradas digitales, y estas resistencias podemos activarlas o no. Desde este bloque le diremos al micro controlador si tiene que activarlas en todas las entradas correspondientes a los EndStop (finales de carrera) o solo en algunas. Solo es necesario activar las resistencias PullUp en caso de que tu EndStop sea un simple micro mecánico (imagen de abajo a la izquierda), si usas alguno de los que hay en el mercado que están montados sobre una PCB con algún componente SMD, (imagen de abajo a la derecha) no necesitas activarlas.



SI PULLUP



NO PULLUP

#define ENDSTOPPULLUPS // Comment this out (using // at the start of the line) to disable the EndStop pullup resistors

Activa las resistencias Pullup en todas las entradas de los EndStop. Por defecto están activadas, por lo que si utilizas EndStop en un PCB deberás de comentarlo con //.

```
#ifndef ENDSTOPPULLUPS
 // fine EndStop settings: Individual pullups. will be ignored if ENDSTOPPULLUPS is defined
 // #define ENDSTOPPULLUP_XMAX
 // #define ENDSTOPPULLUP_YMAX
 // #define ENDSTOPPULLUP_ZMAX
 // #define ENDSTOPPULLUP_XMIN
 // #define ENDSTOPPULLUP_YMIN
 // #define ENDSTOPPULLUP_ZMIN
#endif
```

Si no queremos activar las resistencias PullUp en todas las entradas, pero sí que queremos en alguna en concreto, lo configuraremos desde aquí. Primero comentaremos el parámetro anterior *#define ENDSTOPPULLUPS* para desactivar el general, y después iremos descomentando el parámetro correspondiente al EndStop que queramos que tenga PullUp, por ejemplo si tenemos el EndStop del origen en X con un switch mecánico, descomentaremos la línea *// #define ENDSTOPPULLUP_XMIN*. Los indicados como XMIN, YMIN, y ZMIN corresponden al Homing ó 0, y los XMAX, YMAX, y ZMAX corresponden a los EndStop del tope máximo del recorrido. Este bloque no se tendrá en cuenta si esta activo (descomentado) el parámetro general *#define ENDSTOPPULLUPS*.

```
#ifdef ENDSTOPPULLUPS
 #define ENDSTOPPULLUP_XMAX
 #define ENDSTOPPULLUP_YMAX
 #define ENDSTOPPULLUP_ZMAX
 #define ENDSTOPPULLUP_XMIN
 #define ENDSTOPPULLUP_YMIN
 #define ENDSTOPPULLUP_ZMIN
#endif
```

Este bloque es una extensión de definiciones del parámetro *#define ENDSTOPPULLUPS* y activa las resistencia PullUp de forma independiente. Como ya tenemos el parámetro general y el bloque anterior, es mejor dejar esta parte como esta.

2.6 Invertir la lógica de los EndStop

Podemos invertir la lógica de los EndStop, es decir, si tenemos algún EndStop que cuando esta accionado (pulsado) Marlin lo interpreta como que no está pulsado, es necesario invertirle la lógica. Para verificarlo, una vez montada la impresora y asegurándonos que no esté accionado (pulsado) ningún EndStop, introduciremos en la línea de comandos, o pronterface el Gcode M119 y nos devolver el estado de todos los EndStop, el resultado debería de ser que están todos "open", si hay alguno "TRIGGERED" es ese al que hay que cambiarle la lógica.

```
const bool X_MIN_ENDSTOP_INVERTING = true; // set to true to invert the logic of the EndStop.
const bool Y_MIN_ENDSTOP_INVERTING = true; // set to true to invert the logic of the EndStop.
const bool Z_MIN_ENDSTOP_INVERTING = true; // set to true to invert the logic of the EndStop.
const bool X_MAX_ENDSTOP_INVERTING = true; // set to true to invert the logic of the EndStop.
const bool Y_MAX_ENDSTOP_INVERTING = true; // set to true to invert the logic of the EndStop.
const bool Z_MAX_ENDSTOP_INVERTING = true; // set to true to invert the logic of the EndStop.
```

Si necesitamos invertir la lógica de algún EndStop, bastara con poner el parámetro correspondiente a *false*.

2.7 Invertir sentido de giro de los motores.

Si cuando se mueven los motores el eje hace el movimiento en el sentido contrario, podemos invertir la dirección de giro del motor por cada uno de los ejes y de cada extrusor presente. Por ejemplo si cuando hacemos un Homing, uno de los ejes va en sentido contrario (hacia el máximo) hay que invertir ese eje. O cuando le decimos al extrusor que extruya y lo que hace es sacar filamento, hay que invertir ese extrusor.

```
#define INVERT_X_DIR true    // for Mendel set to false, for Orca set to true
#define INVERT_Y_DIR false    // for Mendel set to true, for Orca set to false
#define INVERT_Z_DIR true    // for Mendel set to false, for Orca set to true
#define INVERT_E0_DIR false   // for direct drive extruder v9 set to true, for geared extruder set to
      false
#define INVERT_E1_DIR false    // for direct drive extruder v9 set to true, for geared extruder set to
      false
#define INVERT_E2_DIR false   // for direct drive extruder v9 set to true, for geared extruder set to
      false
```

Tenemos un parámetro por eje y por extrusor, siendo el E0 el primer extrusor. Si queremos invertir el sentido del motor y por tanto del eje, cambiaremos el parámetro a *true* si estaba en

false (o *false* si estaba en *true)*. Hay que tener precaución antes de configurar estos parámetros, ya que al hacer un Homing, podríamos dañar alguna parte de la impresora al desplazarse un eje en sentido contrario. Una opción posible para este bloque es configurar todos los motores a true o false, y después si necesitamos invertir el giro de algún motor bastaría con **darle la vuelta al conector de 4 cables que va al motor**.

2.8 Medidas máximas de los ejes. Medidas zona de impresión

Hay que introducir las medidas de la zona de impresión, es decir, el máximo desplazamiento que tenemos en cada eje.

```
#define X_MAX_POS 205
#define X_MIN_POS 0
#define Y_MAX_POS 205
#define Y_MIN_POS 0
#define Z_MAX_POS 200
#define Z_MIN_POS 0
```

Define las medias máximas, el mínimo lo dejaremos a 0 y solo será necesario meter las medidas máximas de cada eje en mm. Por ejemplo, si tenemos una impresora con la zona de impresión en el eje x de 175mm, cambiaremos el parámetro *X_MAX_POS* a 175.

```
#define X_MAX_LENGTH (X_MAX_POS - X_MIN_POS)
#define Y_MAX_LENGTH (Y_MAX_POS - Y_MIN_POS)
#define Z_MAX_LENGTH (Z_MAX_POS - Z_MIN_POS)
```

Esta parte no hay que tocarla, solo esta a efectos del cálculo del recorrido máximo de cada eje y se incluye en esta sección por mantener la integridad de los parámetros.

2.9 Velocidad de desplazamiento de los ejes al hacer Homing.

Fijamos la velocidad a la que se moverán los ejes cuando hagamos un Homing. Conviene ser conservador, ya que podríamos pasarnos de largo el EndStop si lo hiciéramos muy deprisa y tener bloqueos mecánicos o romper correas al hacer tope algún eje.

```
#define HOMING_FEEDRATE {50*60, 50*60, 4*60, 0}  // set the homing speeds (mm/min)
```

Los valores están en el siguiente orden *HOMING_FEEDRATE {Eje X, Eje Y, Eje Z, Extrusor}* .El parámetro esta en mm/minuto, y como estamos acostumbrados a trabajar con mm/segundo el

parámetro se ha metido como una operación aritmética 50*60 (equivalente a 50mm/s) pero realmente le estamos diciendo a Marlin que la velocidad es 3000 mm/minuto que es el valor que guardara, ya que antes de asignarlo internamente realizara la operación aritmética que le hemos indicado.

Los valores por defecto suelen estar un poco altos, es mejor fijar la velocidad X e Y a 30*60 y el Z bajarlo a 2*60. Conforme trabajemos con nuestra impresora y veamos cómo están los ajustes podremos ir subiéndolos si queremos que cueste menos tiempo hacer el Homing. El valor configurado para el extrusor, es el mismo para todos los extrusores presentes, no se puede especificar individualmente.

2.10 Pasos por milímetro

Fijaremos los **pasos que corresponden con 1mm** de movimiento por cada uno de los ejes, incluido el extrusor. En las guías de configuración que se encuentran en la wiki de reprap te explican como calcular este dato.

*#define DEFAULT_AXIS_STEPS_PER_UNIT {78.7402,78.7402,200.0*8/3,760*1.1} // default steps per unit for Ultimaker*

Hay que calcular los correspondientes a cada eje de nuestra impresora. Podemos meter tanto un número decimal, como una operación aritmética que calculara Marlin internamente al asignar el valor a ese eje. El orden en el que tienes que introducir los datos por eje es *{Eje X,Eje Y, Eje Z, Extrusores}* .

2.11 Velocidades máximas

Aunque desde el software de control, le podemos decir la velocidad a la que queremos que imprima, desde este parámetro fijamos la velocidad máxima a la que podrá funcionar, y nunca ira más deprisa aunque así se lo indiquemos en el Software de Slicer.

#define DEFAULT_MAX_FEEDRATE {500, 500, 5, 25} // (mm/sec)

El valor esta en mm/s y el que bien por defecto es muy rápido, para impresoras tipo reprap podríamos estar sobre 150-170, pero es mejor empezar de forma conservadora con valores pequeños e ir subiendo poco a poco estos valores. Si la velocidad Z es muy alta, dependiendo del paso (avance en mm de la rosca por 1 vuelta) del eje podría incluso saltarse pasos el motor. Mis valores recomendados son *#define DEFAULT_MAX_FEEDRATE {100, 100, 2, 25}.* Y como siempre el orden de los ejes es *{Eje X, Eje Y, Eje Z, Extrusores}.*

2.12 Aceleración máxima y por defecto.

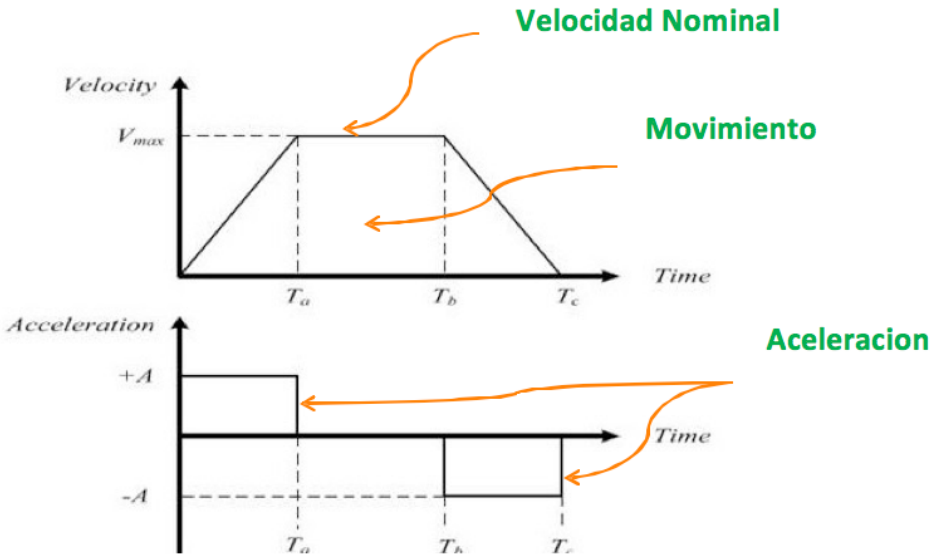
Podremos configurar la aceleración que tendrán los ejes. Si tenemos aceleraciones muy altas, tendremos vibraciones y oscilaciones al cambiar de dirección los ejes. Es mejor bajar estos valores y al igual que con la velocidad, ser conservadores e ir aumentando poco a poco.

#define DEFAULT_MAX_ACCELERATION {9000,9000,100,10000} // X, Y, Z, E maximum start speed for accelerated moves. E default values are good for Skeinforge 40+, for older versions raise them a lot.

Estos valores por defecto son altísimos para impresoras reprap, mi recomendación es la siguiente *#define DEFAULT_MAX_ACCELERATION {1200,1200,75,1200}.*

#define DEFAULT_ACCELERATION 3000 // X, Y, Z and E max acceleration in mm/s^2 for printing moves
#define DEFAULT_RETRACT_ACCELERATION 3000 // X, Y, Z and E max acceleration in mm/s^2 for retracts

Definiremos las aceleraciones por defecto, es decir, si no se indica nada en el Gcode, la impresora cogerá estas aceleraciones. El primer parámetro es para todos los ejes, y el segundo solo es para las retracciones de filamento por parte del extrusor. Lo recomendable en este caso es fijarlas al mismo valor que hemos puesto anteriormente, es decir sobre 1200 los 2 parámetros.



3 Configuraciones, detalle de la librería Configuration.h

Normalmente los valores por defecto son los adecuados, si hay que cambiar algún parámetro se indicara, antes de modificar algún parámetro asegúrate de apunta el valor que tenia previamente para poder volver hacia atrás en caso de problemas. Se ha intentado mantener el orden según aparecen en Configuration.h, no obstante, lo importante es buscar el parámetro o función adecuado si queremos modificar algo.

3.1 Delta Printer Configuration.h

Para impresoras tipo Delta, hay que sustituir los archivos Configuration.h y Configuration_adv.h que están en la carpeta *Marlin/example_configurations/delta*. No vamos a entrar en detalle de cómo configurar una impresora tipo Delta.

```
//============================= DELTA Printer ===============================
//===========================================================================
// For a Delta printer replace the configuration files with the files in the
// example_configurations/delta directory.
```

3.2 Scara Printer Configuration.h

Para impresoras tipo Scara, hay que sustituir los archivos Configuration.h y Configuration_adv.h que están en la carpeta *Marlin/example_configurations/Scara*. No vamos a entrar en detalle de cómo configurar una impresora tipo Scara.

```
//============================= SCARA Printer ===============================
//===========================================================================
// For a Delta printer replace the configuration files with the files in the
// example_configurations/SCARA directory.
```

3.3 Datos, fecha y nombre de compilación

Saca cuando se conecta la impresora, por la consola serie (Consola del Pronterface, terminal de Windows, terminal de Arduino, etc.) la información introducida. Su uso es básicamente informativo, para saber quien a compilado el firmware, podemos introducir nuestros datos o los de nuestra empresa. También es útil para asegurarnos que cada vez que cargamos una nueva versión de Marlin en nuestra placa controladora, esta se ha realizado perfectamente y está cargada en tu microprocesador la última versión que has compilado.

#define STRING_VERSION_CONFIG_H __DATE__ " " __TIME__ // build date and time

Guarda automáticamente la fecha de compilación en STRING_VERSION_CONFIG_H. Usa los macros __DATE__ y __TIME__ para meter la fecha y hora actual justo en el momento de compilación. Nos servirá para saber si la última compilación es la que está cargada en la placa controladora.

#define STRING_CONFIG_H_AUTHOR "(none, default config)" // Who made the changes.

Introduciremos entre las comillas, sustituyendo *(none, default config),* el nombre de la persona o la empresa que ha compilado el proyecto.

Estos datos podemos cambiarlos sin problema, son solo a nivel informativo y no afectan al funcionamiento, aunque recomendamos cambiar el nombre, ya que no queda muy bien que salga por nuestra consola *(none, default config)*.

3.4 Puerto Serie

Configuraremos lo referente al puerto serie. Según el modelo de placa, puede disponer de varios puertos serie, por ejemplo, podríamos emplear el puerto 0 para comunicar la placa con un display y el puerto 1 mediante un adaptador WIFI para conectarnos con el ordenador. Aunque modifiquemos el puerto, el bootloader de Arduino siempre usara el puerto 0, por lo que tendremos que tenerlo en cuenta a la hora de cargar de nuevo Marlin y conectar el ordenador a dicho puerto (0).

#define SERIAL_PORT 0

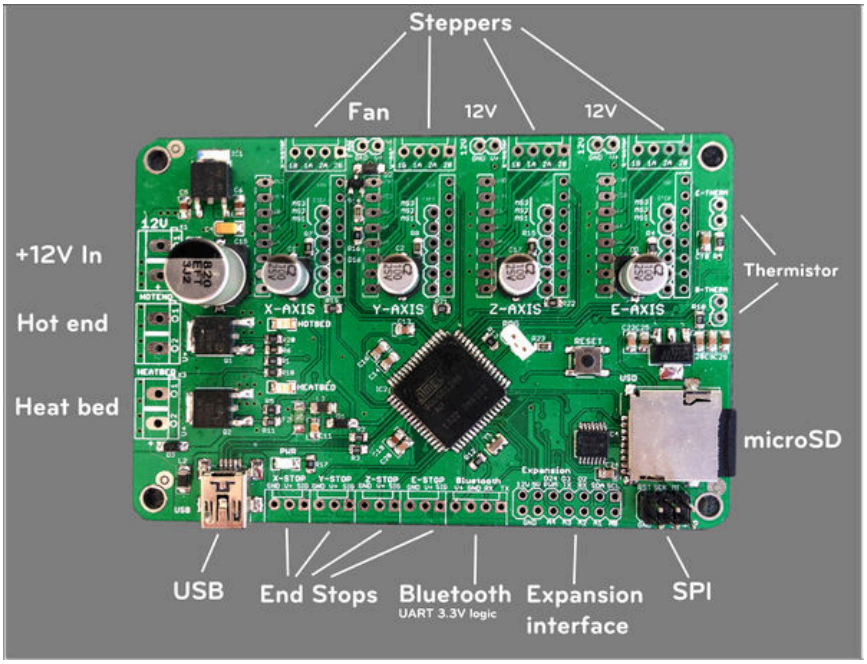
Es el número de puerto que empleamos para comunicarnos con el ordenador host. El valor por defecto es el puerto 0. Asegúrate antes de cambiarlo porque podría dejar de funcionar tu impresora si no conectas el ordenador host al puerto indicado.

#define BAUDRATE 250000

Velocidad a la que va a funcionar el puerto serie, según la placa puede ser que use velocidades inferiores. El valor por defecto es 250000. Normalmente esta velocidad por defecto es la adecuada y no hace falta cambiarla, pero si tenemos problemas con la comunicación podemos cambiarlo a 115200 sin problema.

//#define BTENABLED // Enable BT interface on AT90USB devices

En algunas placas (como la SAV MKI) que llevan un procesadores AT90USB de Atmel, se puede conectar un donge Bluetooth al 2º puerto UART (serie), desde aquí lo habilitamos para comunicarnos con el ordenador host. Por defecto esta deshabilitado (comentado con //), si lo habilitamos y no tenemos el donge Bluetooth instalado podemos dejar de comunicarnos con la impresora.



3.5 Nombre de la impresora e identificador único UUID

Podemos asignar un nombre o modelo a nuestra maquina, se empleara cuando la impresora envíe mensajes al LCD, o cuando se imprimen datos a través del puerto serie. También podemos asignarle un numero único UUID, como si fuera un numero de serie por maquina.

// #define CUSTOM_MENDEL_NAME "This Mendel"

Descomentaremos la línea y entre las comillas indicaremos el nombre o modelo de nuestra impresora. Nombre por defecto Mendel (no “This Mendel”, ya que esta directiva esta comentada). Según el tipo de placa, Marlin puede asignar otros nombres por defecto. Se puede modificar, ya que solo es a nivel estético

// #define MACHINE_UUID "00000000-0000-0000-0000-000000000000"

Descomentamos la línea e introduciremos entre las comillas el número de serie. Podemos generar nuestros propios números de serie únicos desde la web uuidgenerator. Algunos programas pueden emplear este número para diferenciar entre impresoras. No está habilitado por defecto y se puede cambiar sin problema porque no afecta al funcionamiento general.

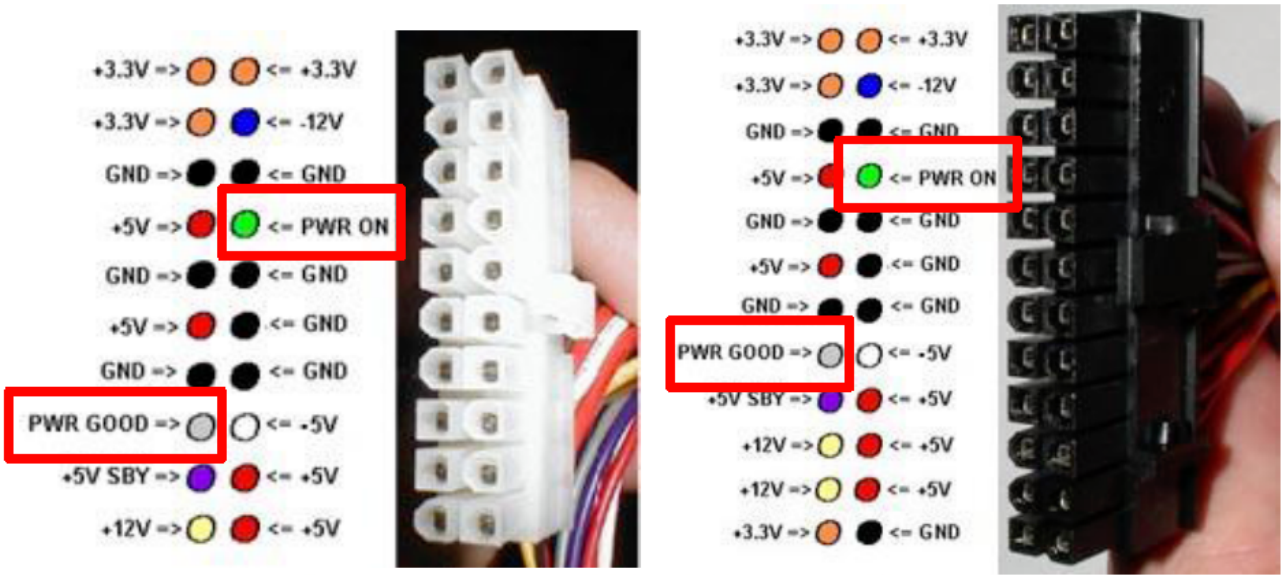
3.6 Fuente de alimentación.

Especificaremos el tipo de fuente de alimentación que tiene nuestra impresora. La intención de este parámetro es poder controlar que se encienda o apague la fuente de alimentación desde la placa de control.

// 1 = ATX
// 2 = X-Box 360 203Watts (the blue wire connected to PS_ON and the red wire to VCC)

#define POWER_SUPPLY 1

Si no vamos a controlar la fuente de alimentación desde la placa, podemos dejar esta parte por defecto. Solo es necesario especificarlo en caso de usar el cable PS-ON (verde) en la fuente ATX, o el cable SLEEP (azul) en la fuente de X-Box o si tenemos una fuente de alimentación con señal de control o entrada SUICIDE. No vamos a explicar aquí como hacer estas conexiones del Hardware, por lo que recomendamos dejarlo por defecto que nos servirá para todo tipo de fuentes de alimentación.



// #define PS_DEFAULT_OFF

Este parámetro entraría también en la parte de control de la fuente desde la placa, básicamente nos da la oportunidad de tener la fuente en Standby y controlada por los Gcodes M80 (on) y M81 (off). Para usar esta directiva, además de tener las entradas anteriormente descritas (PS-ON en ATX, o SLEEP en X-Box) conectadas, es necesario disponer de una pequeña tensión de 5V para que funcione la electrónica de control. Las fuentes ATX, por ejemplo, tienen una salida de 5V permanente mientras la fuente esta en Standby (PS-ON a 5v). Más información en http://reprap.org/wiki/Power_Supply - Controlling_the_PSU

Como ya he dicho, antes de usar estos comandos y conectar los cables de control a tu placa, estudia detenidamente la electrónica de las fuentes, una mala conexión puede destruir tu micro controlador. Además será necesario especificar en el archivo pins.h correspondiente a nuestra placa, que pines corresponden a las señales de control de la fuente. Por defecto esta inhabilitado.

3.7 Ajustes térmicos

Desde este apartado controlaremos todo lo referente a los ajustes térmicos, es decir, sensores de temperatura, temperaturas mínimas, máximas, PID, etc. Normalmente no hay que tocar nada, y dejar todas las opciones por defecto.

3.7.1 Sensor 1 como redundante del 0

//#define TEMP_SENSOR_1_AS_REDUNDANT

Fija el sensor 1 como redundante del sensor 0. Por seguridad, nos podría interesar tener 2 sensores de temperatura en el mismo extrusor. Por defecto esta desactivado, si lo activas asegúrate de que estén los 2 sensores en el mismo extrusor y de que estén configurados correctamente con el tipo que son en el parámetro *TEMP_SENSOR_0 y TEMP_SENSOR_1*.

#define MAX_REDUNDANT_TEMP_SENSOR_DIFF 10

Si hemos fijado el sensor 1 como redundante del 0, le podemos indicar la diferencia máxima que admitimos, si la diferencia es mayor que la indicada, se aborta la impresión. Por defecto esta activa, pero no tiene se tendrá en cuenta sin activar previamente la función *TEMP_SENSOR_1_AS_REDUNDANT*.

3.7.2 Parámetros de espera del Gcode M109

El comando Gcode que le dice a la placa de control que empiece a calentar el Hotend es el M109. Este Gcode es bloqueante, es decir, hasta que no se ha alcanzado a la temperatura indicada Marlin no pasara al siguiente Gcode. Con estos parámetros definimos cuando se considerara que se ha alcanzado la temperatura objetivo.

#define TEMP_RESIDENCY_TIME 10 // (seconds)

Fijamos el tiempo que esperara una vez alcanzada la temperatura destino, esto le da tiempo al extrusor y al control PID para estabilizar la temperatura. Por defecto son 10 segundos, podemos ir ajustándolo según veamos el comportamiento de la temperatura justo cuando inicia la impresión, aunque 10 segundos es un valor correcto.

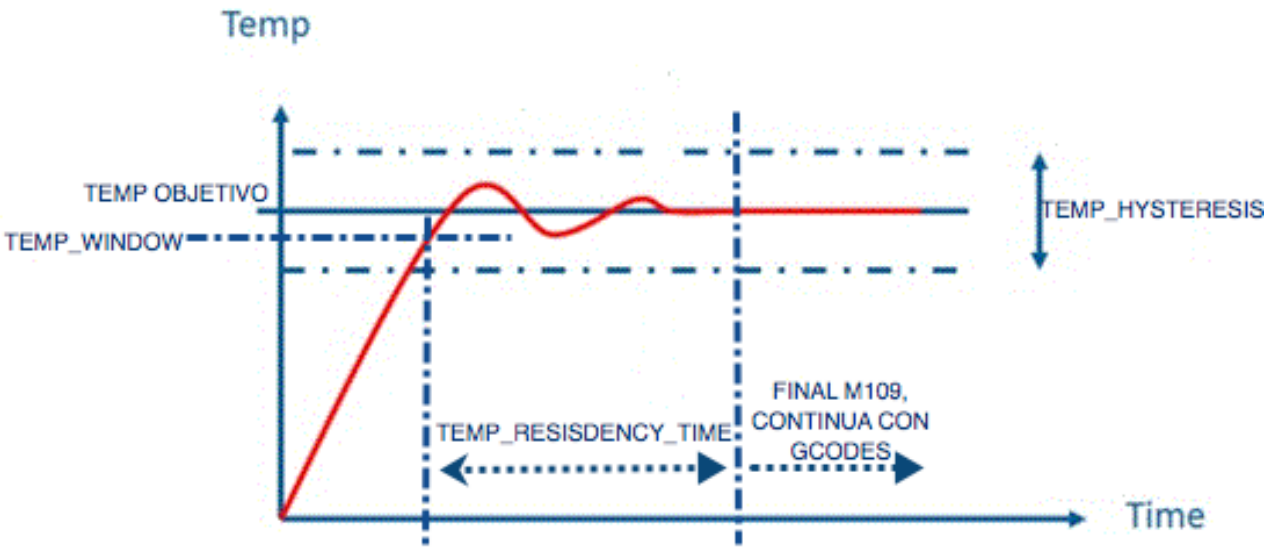
#define TEMP_HYSTERESIS 3 // (degC) range of +/- temperatures considered "close" to the target one

Es el margen de temperatura mas menos a partir del cual Marlin considera que se ha alcanzado la temperatura destino. Por ejemplo, en este caso, si estamos trabajando con PLA a 205C de temperatura objetivo, cuando la temperatura real en el extrusor llegue a 202C, Marlin considera que ya se ha alcanzado la temperatura destino.

#define TEMP_WINDOW 1 // (degC) Window around target to start the residency timer x degC early.

Ventana o margen de temperatura a partir del cual empezara a contar el tiempo de espera definido en la directiva anterior *TEMP_RESIDENCY_TIME*. Por defecto esta en 1 grado.

Tocar estos parámetros nos ayudara a estabilizar o afinar la temperatura del extrusor antes de comenzar a imprimir, aunque también dependerá en mucha medida de cómo tengamos configurados los parámetros de PID. Hay que tener en cuenta que después de un M109, hasta que no se cumplan estos parámetros, Marlin no sigue con los restantes Gcodes y por tanto la impresora estará parada calentando el extrusor.



3.7.3 Temperaturas mínimas

Con estos parámetros nos aseguramos que no se ha cortado o roto algún cable de los sensores de temperatura.

#define HEATER_0_MINTEMP 5
#define HEATER_1_MINTEMP 5
#define HEATER_2_MINTEMP 5
#define BED_MINTEMP 5

Si la lectura de los sensores de temperatura está por debajo de la que hemos fijado, se considera que algún cable está roto y por tanto no se está recibiendo la lectura del sensor, en este caso la impresora dejara de funcionar. Los valores por defecto son los adecuados, pero

puede suceder que estemos trabajando en un garaje o local en invierno y que la temperatura ambiente este por debajo de 5C y la impresora dejaría de funcionar.

3.7.4 Temperaturas máximas

La seguridad cuando se emplean fuentes de calor es muy importante, con este parámetro configuramos la temperatura máxima a la que pueden funcionar nuestros extrusores y nuestra HeatedBed.

#define HEATER_0_MAXTEMP 275
#define HEATER_1_MAXTEMP 275
#define HEATER_2_MAXTEMP 275
#define BED_MAXTEMP 150

Configuramos individualmente la temperatura máxima de cada extrusor y de la HeatedBed. Depende del tipo de extrusor si es Allmetal o tiene partes plásticas, y del tipo de sensor de temperatura que lleve ya que no todos los termistores aguantan más de 300C. Realmente con los materiales que hay hoy en día, es difícil en algún momento tener la necesidad de sobrepasar los 275C. Si se sobrepasa la temperatura se desconecta la salida que calienta ese extrusor o cama y deja de imprimir. Este parámetro esta para proteger los extrusores o HeatedBed accidentalmente, pero no para proteger de un termistor roto o que falle. Los valores por defecto suelen ser los adecuados, variarlos podría ocasionar que quemaras tu extrusor o destruyeras tu termistor. Estos parámetros son peligrosos, ya que **corres riesgo de incendio**, por favor, se cuidadoso y no dejes la impresora sin supervisión si has modificado alguno de estos parámetros.

3.7.5 Ciclos de trabajo de la HeatedBed

//#define HEATER_BED_DUTY_CYCLE_DIVIDER 4

Esta directiva **está obsoleta, ya no se aplica**, en su lugar hay que usar MAX_BED_POWER en el apartado correspondiente a las configuraciones de la HeatedBed.

3.7.6 Temperatura en vatios con M105

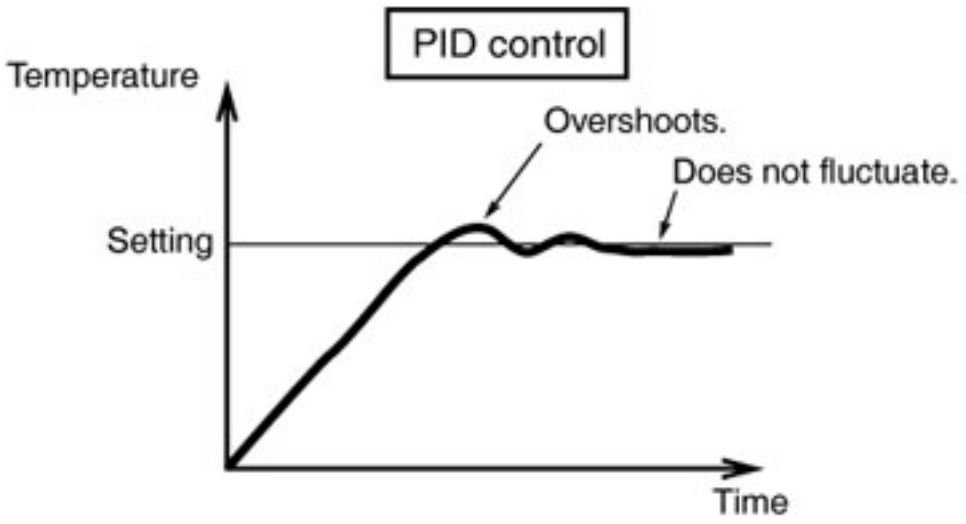
Cuando enviamos el Gcode M105 Marlin nos devuelve la temperatura actual en grados centígrados, si queremos que nos devuelva el dato en Watios podemos activar esta función.

*//#define EXTRUDER_WATTS (12.0*12.0/6.7) // P=V^2/R*
*//#define BED_WATTS (12.0*12.0/1.1)   // P=V^2/R*

Fijaremos la potencia del extrusor, que es el mismo dato para todos, y en el siguiente parámetro la potencia de la HeatedBed. Podemos introducir el dato como una formula (igual que en el ejemplo) o directamente si conocemos el dato en W del elemento. Por defecto esta desactivado, modificar este parámetro no tiene relevancia a nivel de funcionamiento, solo es el dato que vemos visualizado cuando hacemos M105 el que cambia. En la formula se ha introducido la tensión de alimentación al cuadrado (12*12) y dividido por la resistencia del elemento (/6.7) y esa formula nos da la potencia en W.

3.7.7 Parámetros PID del extrusor

En este apartado controlamos todo lo referente a los parámetros PID de los extrusores, no diferencia entre ellos, si tenemos varios extrusores les aplica los mismos valores. El sistema PID es un sistema complejo de control, si quieres saber más mira en wiki PID



#define PIDTEMP

Activa el control de temperatura por PID, si desactivamos esta función se activa en su lugar el sistema bang-bang o todo-nada, es decir, sería como un termostato, o esta encendido o está apagado. Es recomendable dejarlo como esta, el sistema bang-bang no es muy fino controlando la temperatura y tiene muchos picos de temperatura.

#define BANG_MAX 255 // limits current to nozzle while in bang-bang mode; 255=full current
#define PID_MAX BANG_MAX// limits current to nozzle while PID is active; 255=full current

Estos parámetros fijan la corriente máxima o potencia del extrusor, si nuestro extrusor es muy potente podemos decirle que aplique menos corriente y por tanto aplicara menos potencia. *BANG_MAX* es para cuando estamos funcionando en modo bang-bang y *PID_MAX* si tenemos activo el control PID. El valor por defecto es 255 que es máxima potencia. De no tener un cartucho muy potente en nuestro extrusor y tengamos que reducir la potencia porque no podemos controlarlo bien, no será necesario modificar este parámetro.

//#define PID_DEBUG // Sends debug data to the serial port.

Nos permite ver por la consola serie los datos del control PID, ira sacando constantemente los datos de control y nos da la oportunidad de ver si funciona todo correctamente. Por defecto esta desactivado, y de no necesitar depurar el funcionamiento no es necesario activarlo, aunque no tiene relevancia para el funcionamiento de la impresora. Solo funciona si esta el PID activo.

//#define PID_OPENLOOP 1

Pone el control PID en lazo abierto, es decir, no tiene en cuenta la temperatura medida por el sensor, y cuando usemos los Gcodes M104, M109 o M140 pone la potencia del extrusor o HeatedBed al valor indicado en *PID_MAX* (potencia no temperatura). Por defecto esta comentado, es mejor no activarlo porque el PID no funcionaria.

//#define SLOW_PWM_HEATERS // PWM with very low frequency (roughly 0.125Hz=8s) and minimum state time of approximately 1s useful for heaters driven by a relay

Configura la frecuencia PWM de control de los HotEnd y HeatedBed a una frecuencia muy baja, de 0.125Hz. Muy útil para cuando estén controlados por un relé¡ y no queramos altas frecuencias de encendido apagado.

#define PID_FUNCTIONAL_RANGE 10

Si la diferencia entre la temperatura de destino y la temperatura actual del Hotend, es mayor que la indicada en este parámetro, entonces el sistema PID no estará activo y el control pasa a modo bang-bang, aplicando al Hotend la potencia indicada en el parámetro *PID_MAX.*

El sistema de control PID del Hotend que emplea Marlin funciona de la siguiente forma, al comenzar a calentar no tiene en cuenta los parámetros Kp, Ki, y Kd, y aplica al cartucho calefactor la potencia indicada en le parámetro *PID_MAX*. Cuando la diferencia entre la temperatura objetivo y la temperatura real es **menor** que la indicada en el parámetro

PID_FUNCTIONAL_RANGE entonces es cuando comienza realmente el control PID de la temperatura. Esto es importante tenerlo en cuenta, ya que si no conseguimos reducir el pico inicial de temperatura cuando comienza a calentarse el Hotend, podría ser debido a que el PID empieza a controlar la temperatura demasiado tarde. Si aumentamos el parámetro *PID_FUNCTIONAL_RANGE* le estaremos indicando a Marlin que empiece antes a usar el sistema PID para controlar la temperatura, y por tanto conseguiremos una curva mas suave.

Por defecto esta en 10 grados. Solo funciona si esta el PID activo.

#define PID_INTEGRAL_DRIVE_MAX PID_MAX //limit for the integral term

Limite para el valor Ki (Integral) del control PID. Por defecto esta en 255 y se ha fijado en los parámetros anteriores *#define PID_MAX BANG_MAX* y *#define BANG_MAX 255*.

#define K1 0.95 //smoothing factor within the PID

Factor de suavizado de la curva PID. Este parámetro afecta solo al valor Kd (Diferencial)

*#define PID_dT ((OVERSAMPLENR * 10.0)/(F_CPU / 64.0 / 256.0)) //sampling period of the temperature routine*

Fija el periodo de muestreo del control PID.

Estos parámetros ayudan a suavizar la curva de temperatura. PID es un sistema de control muy complicado y entran en juego muchos parámetros, estos ayudan a suavizar la curva. Si no sabes exactamente lo que hacen mejor no los modifiques, o modifica cada uno poco a poco y experimenta con el resultado. Solo funcionan si esta el PID activo.

3.7.8 Parámetros HeatedBed

Podemos activar o no el PID de la HeatedBed, y si la activamos podremos modificar los parámetros de control. Debido a que el control PID usa una frecuencia PWM en la salida, si **no** tenemos conectada la HeatedBed directamente a la placa y al FET y hemos usado un relé intermedio, podemos tener problemas. Si el relé es un relé de estado sólido, debería de funcionar correctamente, pero con relés mecánicos no funcionara. Si experimentas cualquier problema deja desactivado el PID de la HeatedBed, realmente no es tan critico tener una temperatura constante como para tener que activarlo. Es recomendable dejar este apartado como esta y no activar el PID en la HeatedBed.

//#define PIDTEMPBED

Activa el PID en la HeatedBed, si lo dejamos desactivado funcionara en modo bang-bang, es decir, todo o nada. Si lo activamos tendremos que tener la precaución de configurar los parámetros PID.

//#define BED_LIMIT_SWITCHING

Si hemos dejado el control de la HeatedBed en modo bang-bang, esta directiva activa la histéresis (diferencia) en la temperatura. La histéresis se puede configurar en el archivo Configuration_adv.h en la directiva BED_HYSTERESIS, mirar en el apartado correspondiente a los ajustes avanzados. Por defecto esta desactivado, si no vamos a usar el PID podemos activarlo.

#define MAX_BED_POWER 255 // limits duty cycle to bed; 255=full current

Este parámetro fija la corriente máxima o potencia de la HeatedBed, si es muy potente podemos indicarle que aplique menos corriente y por tanto aplicara menos potencia. Si tenemos HeatedBed con bajas resistencias, por ejemplo 0.6 Ω, y nos funde o dispara el fusible de la placa, con este parámetro le podemos indicar que funcione con menos potencia. Si tenemos HeatedBed muy potentes es mejor usar un relé para controlarla y sacar la corriente de funcionamiento fuera de la placa. Muchos de los problemas de las placas vienen por este lado, ya que las HeatedBed consumen mucho y pueden calentar las pistas y fundir o quemar la placa con **riesgo de incendio**. Para controlar la potencia aplicada usa un sistema PWM, si tenemos un relé (tal y como hemos comentado con el control PID) podemos tener problemas. El valor por defecto es 255 que es máxima potencia. De no tener una HeatedBed muy potente y tengamos que reducir la potencia porque no podemos controlarlo bien, no es necesario modificar este parámetro.

#define DEFAULT_bedKp 10.00
#define DEFAULT_bedKi .023
#define DEFAULT_bedKd 305.4

Desde aquí configuramos los parámetros del PID de la HeatedBed, al igual que con el extrusor, usaremos el Gcode M303 para averiguar cuáles son los adecuados para el modelo que tengamos instalado. Consulta el apartado correspondiente al comando M303.

3.7.9 Seguridad extrusor

En este apartado tenemos parámetros correspondientes a seguridades extras del extrusor.

#define PREVENT_DANGEROUS_EXTRUDE

Evita que funcione el motor del extrusor en frio y por tanto intentemos extruir material con el Hotend por debajo de la temperatura indicada en el parámetro *EXTRUDE_MINTEMP* que se configura a continuación.

#define PREVENT_LENGTHY_EXTRUDE

Evita extrusiones muy largas, si la extrusión es más larga que la indicada en el parámetro *EXTRUDE_MAXLENGTH* que se configura un poco más abajo, el extrusor no extruira por seguridad.

#define EXTRUDE_MINTEMP 170

Temperatura mínima a la que tiene que estar el Hotend para que se mueva el motor del extrusor. Si vamos a realiza pruebas de funcionamiento del motor del extrusor **sin estar conectado al Hotend**, por ejemplo, cuando vayamos a medir los pasos necesarios para extruir 1mm, modificaremos este parámetro a 20-15 grados, una vez tengamos medidos los pasos por mm del extrusor, volveremos a dejar este parámetro como estaba. También se puede modificar temporalmente con el Gcode M302.

#define EXTRUDE_MAXLENGTH (X_MAX_LENGTH+Y_MAX_LENGTH) //prevent extrusion of very large distances.

Longitud máxima de seguridad que extruimos de una vez, este parámetro se usa si esta activo *PREVENT_LENGTHY_EXTRUDE* .

Todos los parámetros de este bloque es mejor dejarlos como están por defecto, menos *EXTRUDE_MINTEMP* en el caso de configurar los pasos por mm del motor del extrusor.

3.7.10 Seguridad térmica del Hotend

Podemos configurar la seguridad térmica del Hotend que protege en caso de que se mueva o salga el termistor de su alojamiento. Si el termistor se sale de su alojamiento, leerá una temperatura más baja que la real, el sistema intentara compensar y calentar en exceso el filamento y todo el Hotend con el consiguiente peligro de incendio.

Después de que el Hotend alcance la temperatura objetivo por primera vez, el sistema comenzara a monitorizar la temperatura. Cuando esta descienda por debajo de la temperatura objetivo menos la histéresis fijada más abajo empezara a controlar cuanto tiempo tarda en volver a subir, si tarda más del tiempo indicado, significara que la temperatura medida por el termistor no se puede actualizar y por tanto algo está pasando. Si falla se apagara el Hotend, y se parara la impresora.

Esta seguridad entra en acción una vez alcanzada la temperatura objetivo por primera vez, así que si tu extrusor tarda mucho en calentarse, no hay problema. Es importante que antes de imprimir chequees que este todo ok, porque si se ha movido el termistor estando la impresora parada, esta seguridad no lo detectara.

//#define THERMAL_RUNAWAY_PROTECTION_PERIOD 40 //in seconds

Fija el tiempo máximo que admite que este la temperatura por debajo de la temperatura objetivo menos la histéresis. Si no llega a la temperatura objetivo menos la histéresis antes de este tiempo se parara la impresora.

//#define THERMAL_RUNAWAY_PROTECTION_HYSTERESIS 4 // in degree Celsius

Fija los grados por debajo de la temperatura objetivo, a partir de la cual comenzara a controlar el tiempo que tarda en subir.

//#define THERMAL_RUNAWAY_PROTECTION_BED_PERIOD 20 //in seconds
//#define THERMAL_RUNAWAY_PROTECTION_BED_HYSTERESIS 2 // in degree Celsius

Mismos parámetros para la HeatedBed.

Estos parámetros están comentados y por tanto desactivada la función de control, pero parece lógico activarlo, ya que la seguridad es muy importante. Para activarla hay que descomentar las 2 líneas.

3.8 Ajustes mecánicos.

Desde este apartado configuramos todos los parámetros mecánicos de la impresora, probablemente es donde más parámetros hemos tenido que configurar en la sección anterior de configuraciones básicas.

3.8.1 Corexy

Corexy es un sistema de transmisión de movimiento en los ejes X e Y que usa 2 motores fijos con una configuración especial de poleas y correas. Más datos en http://www.corexy.com/.

```
// #define COREXY
```

Por defecto esta desactivado, por lo que tendremos la impresora configurada con 1 motor para el eje X y otro para el eje Y. Si tienes una impresora corexy, además de activar este parámetro, probablemente necesites alguna configuración más.

3.8.2 Inhabilitar los EndStop MAX y MIN

Podemos deshabilitar los EndStop correspondientes al Mínimo de cada eje (0) o los del máximo recorrido por eje.

```
//#define DISABLE_MAX_ENDSTOPS
//#define DISABLE_MIN_ENDSTOPS
```

Si no tenemos EndStop de recorrido máximo del eje (MAX) desde aquí podemos inhabilitarlos. Solo tendremos que descomentar el parámetro *DISABLE_MAX_ENDSTOPS* . Por defecto están deshabilitados, el *DISABLE_MIN_ENDSTOPS* no deberíamos de tocarlo, pero si tenemos una impresora estándar con EndStop solo en el sentido del 0 (Homing) deberíamos de descomentar el parámetro *DISABLE_MAX_ENDSTOPS* .

```
#if defined(COREXY) && !defined(DISABLE_MAX_ENDSTOPS)
  #define DISABLE_MAX_ENDSTOPS
#endif
```

Si hemos definido una impresora del tipo Corexy, deshabilita por defecto los EndStop en el sentido del valor máximo del eje.

3.8.3 Invertir lógica pin Enable en los drivers de los motores

Para el control de los motores usamos unos drivers, en las impresoras normalmente se usan los pololu A4988 o los más potentes DRV8825. Estos drivers tienen un pin Enable, que sirve para activarlos, y en estos modelos se activa con un 0 lógico. Si tenemos otros drivers alternativos y se activa la entrada Enable con un 1 lógico (5V) tendrás que invertir el parámetro correspondiente al eje donde esté instalado ese driver.

```
// For Inverting Stepper Enable Pins (Active Low) use 0, Non Inverting (Active High) use 1
#define X_ENABLE_ON 0
#define Y_ENABLE_ON 0
#define Z_ENABLE_ON 0
#define E_ENABLE_ON 0 // For all extruders
```

Por defecto, con un 0 esta activa la entrada Enable (A4988 y DRV8825). Si tienes un driver que necesita activarse con un 1 lógico, hay que poner el parámetro del eje o extrusores correspondiente a 1. El valor por defecto es correcto para la electrónica estándar.

3.8.4 Deshabilitar ejes cuando no se empleen.

Podemos apagar los motores de los ejes cuando no se empleen, esto hará que la electrónica consuma menos, además de que se calentaran menos los drivers y los motores.

```
#define DISABLE_X false
#define DISABLE_Y false
#define DISABLE_Z false
#define DISABLE_E false // For all extruders
#define DISABLE_INACTIVE_EXTRUDER true //disable only inactive extruders and keep active extruder enabled
```

Cada parámetro configura un eje, por defecto están todos deshabilitados *false*, menos el parámetro que deshabilita los extrusores que no estén activos que esta a *true*.

3.8.5 Dirección del Homing

Indicamos que EndStop se pulsaran al hacer el Homing, y por tanto en qué dirección se moverán los ejes.

// Sets direction of endstops when homing; 1=MAX, -1=MIN
#define X_HOME_DIR -1
#define Y_HOME_DIR -1
#define Z_HOME_DIR -1

Podemos indicarle si queremos que el Homing sea en el EndStop del MIN (valor -1)o del MAX (valor 1) normalmente es el EndStop MIN o cero. Por defecto son los EndStop MIN (-1), si quieres cambiar algún eje tienes que poner el parámetro correspondiente a 1. Es recomendable no tocarlo.

3.8.6 Permitir movimientos fuera de las medidas de los ejes.

Podemos definir si queremos que se puedan mover los ejes fuera de las medias máximas y mínimas de los ejes. Por ejemplo, la impresora 3D Systems cube pro, cuando inicia la impresión se desplaza fuera de la zona de impresión para limpiar la punta del extrusor, estos parámetros permiten o evitan que se pueda hacer esto por seguridad.

#define min_software_endstops true // If true, axis won't move to coordinates less than HOME_POS.

Permite el movimiento por debajo del cero, es decir, coordenadas negativas. Si esta a true evita que se desplace hacia coordenadas negativas.

#define max_software_endstops true // If true, axis won't move to coordinates greater than the defined lengths below.

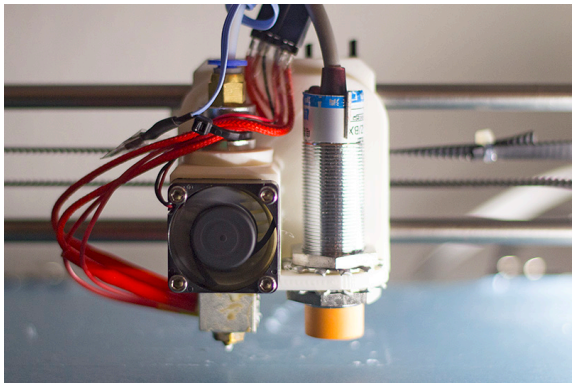
Permite movimientos por encima de las medidas máximas de los ejes (mirar configuraciones básicas). Por ejemplo, si la medida del eje x de nuestra maquina está configurada en 175, no permitirá valores superiores a esta medida si esta a true.

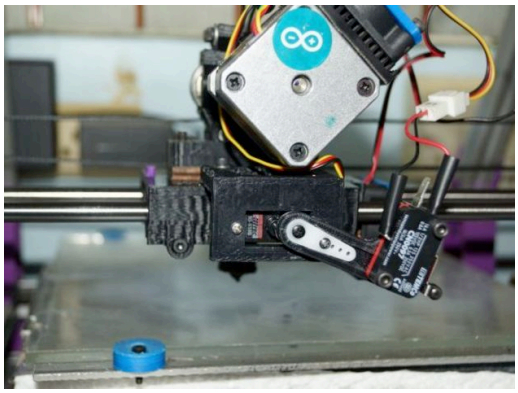
Por defecto están activos los 2 parámetros (*true*), es recomendable no cambiarlo (*false*), ya que podría cometerse un error y desplazar un eje fuera de su medida y provocar una avería.

3.8.7 Auto nivelado de la zona de impresión

Desde este bloque se configura todos los parámetros correspondientes al auto-nivelado de la zona de impresión. El que la zona de impresión este correctamente nivelada con respecto a la punta del Hotend, es de vital importancia para la perfecta adhesión de la primera capa. El auto-nivelado nos evita tener que nivelar manualmente la zona (HeatedBed) de impresión mediante la regulación de los tornillos cada vez que imprimimos alguna pieza.

Antes de imprimir ejecutamos un Gcode G28 que es un Homing, seguido de un Gcode G29 que activara la "sonda" de medida y tomara varios puntos de medida sobre la cama, una vez ha terminado calcula el error de nivelación. Una vez calculada la desviación de la nivelación, mientras imprima veremos que va corrigiendo el nivelado con pequeños movimientos del eje Z. Necesitaremos hardware extra, un servo y un EndStop, un sensor inductivo (precisa de zona de impresión metálica) o sensor capacitivo.





//#define ENABLE_AUTO_BED_LEVELING // Delete the comment to enable (remove // at the start of the line)

Activa el auto nivelado de la zona de impresión. El activar el auto nivelado precisa de hardware extra, por favor consulta antes de activarlo. Por defecto esta desactivado.

Todos los parámetros que hay a continuación en este bloque, solo se tendrán en cuenta si tenemos definido *ENABLE_AUTO_BED_LEVELING*.

#define Z_PROBE_REPEATABILITY_TEST // If not commented out, Z-Probe Repeatability test will be included if Auto Bed Leveling is Enabled.

Permite que se pueda usar el Gcode M48 para comprobar la repetitividad de disparo de nuestro EndStop o sistema de disparo que tengamos para nivelar. Básicamente nos aseguramos que el EndStop que estamos usando para tomar las medidas es de buena calidad y dispara siempre en el mismo instante. Más información en el apartado correspondiente del Gcode M48. Este parámetro esta activado, no afecta al funcionamiento, solo nos permite usar el Gcode M48.

Hay 2 maneras de fijar las coordenadas X e Y de cada uno de los puntos donde se va a tomar la medida para determinar el nivelado. Uno es en "grid" o rejilla y el otro es "3-point" o en 3 puntos.

En modo “grid” (rejilla) toma las muestras formando una rejilla rectangular, se puede especificar las medidas del rectángulo y el numero de las pruebas (puntos de muestreo). Este sistema es más exacto y por tanto será la elección más adecuada. Como desventaja tenemos que al tomar más muestras, tardaremos más tiempo en efectuar todo el proceso de nivelado.

En modo “3-point” (3 puntos) se toman 3 puntos que no sean colineales. Se puede especificar las coordenadas X e Y de los 3 puntos. Es menos preciso al tomar menos muestras, pero se realiza mas rápidamente.

#define AUTO_BED_LEVELING_GRID

Pone el modo de toma de medidas en formato rejilla. Por defecto esta activado, y como ya hemos dicho que es el sistema más exacto, lo podemos dejar como esta. Si lo desactivamos, pasa al formato “3-point” (3 puntos)

#define LEFT_PROBE_BED_POSITION 15
#define RIGHT_PROBE_BED_POSITION 170
#define BACK_PROBE_BED_POSITION 180
#define FRONT_PROBE_BED_POSITION 20

Fija las medidas del rectángulo que se usa para formar la rejilla, son las coordenadas donde tomara la medida la sonda, no son coordenadas del extrusor. Este parámetro tiene en cuenta el offset de la sonda con respecto al extrusor, hay que tomar precauciones porque podríamos tomar estar fijando medias fuera de la zona de impresión.

#define AUTO_BED_LEVELING_GRID_POINTS 2

Configura el número de puntos (medidas) que tomara en la rejilla por cada uno de los ejes. Si esta en 2, tomara 2 puntos en el eje X y 2 puntos en el eje Y en total 2*2=4 puntos, si lo cambiamos a 3 tomara 3 puntos en el eje X y 3 en el eje Y en total 3*3=9 puntos. Por defecto esta en 2 (4 puntos), es mejor cambiarlo a 3 (9 puntos en total), es una buena relación entre número de puntos y la velocidad para tomarlos, además mas puntos de medida no dará como resultado un calculo mejor.

#define ABL_PROBE_PT_1_X 15
#define ABL_PROBE_PT_1_Y 180
#define ABL_PROBE_PT_2_X 15
#define ABL_PROBE_PT_2_Y 20
#define ABL_PROBE_PT_3_X 170
#define ABL_PROBE_PT_3_Y 20

Si no hemos activado el modo rejilla *AUTO_BED_LEVELING_GRID*, el modo de prueba estará en "3-point" (3 puntos). Con estos parámetros fijamos las coordenadas X e Y de cada uno de los 3 puntos que tomara. Al igual que con el modo rejilla hay que tener cuidado porque tiene en cuenta los offset de la sonda de prueba (EndStop) con respecto al extrusor.

#define X_PROBE_OFFSET_FROM_EXTRUDER -25
#define Y_PROBE_OFFSET_FROM_EXTRUDER -29
#define Z_PROBE_OFFSET_FROM_EXTRUDER -12.35

Fija los offset de la sonda (EndStop que tenemos para tomar las medidas de nivelado) con relación a la punta del extrusor. Estos parámetros hay que cambiarlos y poner los que tengamos nosotros.

Tiene especial importancia *#define Z_PROBE_OFFSET_FROM_EXTRUDER* ya que fija la distancia desde el instante de disparo de la sonda a la punta del extrusor. Si después de nivelar, al imprimir observamos que la primera capa la imprime lejos de la cama, tendremos que aumentar este parámetro. De igual manera si observamos que la punta del extrusor queda muy pegada a la cama, tendremos que reducir este parámetro. Fíjate bien que para acercar aumentamos el offset y para alejar lo reducimos. No obstante, esto no es un tutorial para configurar el nivelado de la cama, tan solo es un análisis de Marlin y sus configuraciones.

#define Z_RAISE_BEFORE_HOMING 4 // (in mm) Raise Z before homing (G28) for Probe Clearance.

Fija la distancia que tendrá que moverse el eje Z antes de desplegar la sonda al hacer un Homing (Gcode G28). Para EndStop montados sobre servos, al tomar las medidas el servo se "despliega" poniendo la sonda en posición, de esta manera evitaremos que al desplegarse pegue con la cama y se pueda romper la sonda. Esta medida tiene que ser superior a *Z_PROBE_OFFSET_FROM_EXTRUDER* pero inferior a la medida máxima del eje Z (*Z_MAX_POS*).

#define XY_TRAVEL_SPEED 8000 // X and Y axis travel speed between probes, in mm/min

Velocidad de desplazamiento entre los puntos de prueba. Esta en mm/min, y normalmente estamos acostumbrados a trabajar con velocidades en mm/s. En este caso está configurado por defecto a una velocidad de 8000 mm/min equivalente a 133mm/s (hay que dividir por 60). Es recomendable poner este parámetro entre 4500-5000 (aproximadamente 75-85 mm/s).

#define Z_RAISE_BEFORE_PROBING 15 //How much the extruder will be raised before traveling to the first probing point.

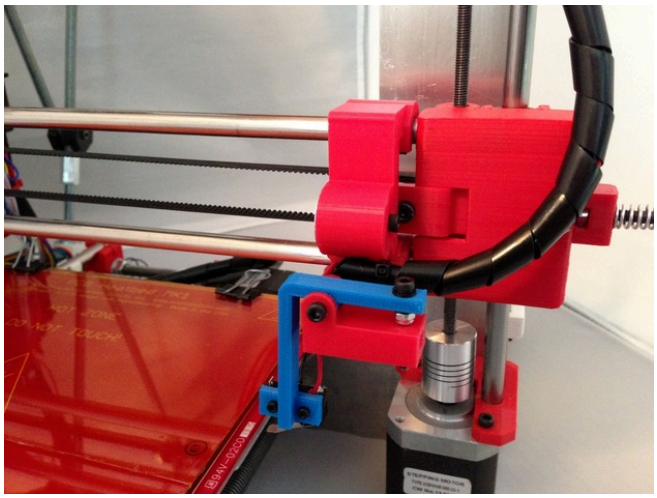
Al igual que en *Z_RAISE_BEFORE_HOMING* definimos la distancia que se desplazara el eje Z antes de desplegar la sonda, pero en este caso antes de comenzar a tomar los puntos necesarios para nivelar la cama. Se utilizaría en un Gcode G29 o en un M48.

#define Z_RAISE_BETWEEN_PROBINGS 5 //How much the extruder will be raised when traveling from between next probing points

Desplazamiento en Z que tendrá entre cada uno de los puntos que toma para nivelar la cama, como ya tenemos desplegada la sonda, un valor entre 5-10 mm es un buen desplazamiento. Por defecto esta en 5, se puede dejar así. Solo se utilizaría en un Gcode G29.

//#define Z_PROBE_SLED // turn on if you have a z-probe mounted on a sled like those designed by Charles Bell

Hay que activar esta función si tenemos una sonda montada en un carro en el lateral del eje X como la descrita aquí http://www.thingiverse.com/thing:396692.



//#define SLED_DOCKING_OFFSET 5 // the extra distance the X axis must travel to pickup the sled. 0 should be fine but you can push it further if you'd like.

Distancia extra que tendrá que desplazarse el extrusor en el eje X para poder "coger" o sujetar el carro de la sonda

// #define PROBE_SERVO_DEACTIVATION_DELAY 300

El servo de la sonda estará encendido solo cuando se mueva, mientras tanto estará apagado para evitar vibraciones o sacudidas. El valor fijado aquí es el tiempo que tarda en apagar el servo una vez se ha encendido, el valor dependerá de la velocidad del servo, pero 300 ms suele ser un valor correcto. Hay que tener al menos un *SERVO_ENDSTOPS* definido, sino te dará un error al compilar el código. Por defecto no está activado, solo tienes que configurarlo si observas que el servo hace cosas raras durante la impresión.

#define Z_SAFE_HOMING // This feature is meant to avoid Z homing with probe outside the bed area.

Si tenemos activado el auto-nivelado de la cama, y usamos la misma sonda para hacer Homing (que es lo normal) es muy recomendable dejar el Z_SAFE_HOMING activado. Esto permite que solo haga un Homing del eje Z después de hacer un Homing de X e Y. Además posicionara la sonda en unas coordenadas definidas más adelante antes de hacer el Homing Z. Por defecto está activado, es recomendable dejarlo así.

#define Z_SAFE_HOMING_X_POINT (X_MAX_LENGTH/2) // X point for Z Homing when Homing all axis (G28)
#define Z_SAFE_HOMING_Y_POINT (Y_MAX_LENGTH/2) // Y point for Z homing when homing all axis (G28)

Coordenadas X e Y donde se hará el Homing Z si está definido *Z_SAFE_HOMING*. Se usara cuando hagamos un Homing general con el Gcode G28. Para el cálculo de las coordenadas reales, tendrá en cuenta el offset de la sonda con la punta del extrusor definido anteriormente. Por defecto es el centro de la zona de impresión, es recomendable dejarlos así. Solo se tendrá en cuenta si esta activo *Z_SAFE_HOMING*

```
#ifdef AUTO_BED_LEVELING_GRID // Check if Probe_Offset * Grid Points is greater than
        Probing Range
  #if X_PROBE_OFFSET_FROM_EXTRUDER < 0
   #if (-(X_PROBE_OFFSET_FROM_EXTRUDER * (AUTO_BED_LEVELING_GRID_POINTS-1)) >=
        (RIGHT_PROBE_BED_POSITION - LEFT_PROBE_BED_POSITION))
          #error "The X axis probing range is not enough to fit all the points defined in
        AUTO_BED_LEVELING_GRID_POINTS"
        #endif
        #else
   #if ((X_PROBE_OFFSET_FROM_EXTRUDER * (AUTO_BED_LEVELING_GRID_POINTS-1)) >=
        (RIGHT_PROBE_BED_POSITION - LEFT_PROBE_BED_POSITION))
          #error "The X axis probing range is not enough to fit all the points defined in
        AUTO_BED_LEVELING_GRID_POINTS"
         #endif
        #endif
  #if Y_PROBE_OFFSET_FROM_EXTRUDER < 0
   #if (-(Y_PROBE_OFFSET_FROM_EXTRUDER * (AUTO_BED_LEVELING_GRID_POINTS-1)) >=
        (BACK_PROBE_BED_POSITION - FRONT_PROBE_BED_POSITION))
          #error "The Y axis probing range is not enough to fit all the points defined in
        AUTO_BED_LEVELING_GRID_POINTS"
         #endif
        #else
   #if ((Y_PROBE_OFFSET_FROM_EXTRUDER * (AUTO_BED_LEVELING_GRID_POINTS-1)) >=
        (BACK_PROBE_BED_POSITION - FRONT_PROBE_BED_POSITION))
          #error "The Y axis probing range is not enough to fit all the points defined in
        AUTO_BED_LEVELING_GRID_POINTS"
         #endif
        #endif
#endif
```

Código de chequeo que evita que definamos puntos de prueba mas grande que el rango que puede tomar la sonda de medida. No tocarlo.

3.8.8 Posición de los EndStop

Normalmente los EndStop están instalados al inicio o al final del recorrido de los ejes, pero si lo tuviéramos a cierta distancia del 0, podemos configurar en que coordenadas del eje correspondiente esta el EndStop. Este tipo de configuraciones está más indicada para impresoras tipo delta, en la que la posición de los EndStop no siempre coincide con los mínimos o máximos del eje.

//#define MANUAL_HOME_POSITIONS // If defined, MANUAL_*_HOME_POS below will be used

Quedara fijada la posición de los EndStop según el parámetro *MANUAL_(eje)_HOME_POS* definido un poco más abajo. Por defecto esta desactivada, si tenemos los EndStop en el inicio o final de los ejes no hay que activarlo.

//#define BED_CENTER_AT_0_0 // If defined, the center of the bed is at (X=0, Y=0)

Fija el centro de la zona de impresión en X=0 e Y=0. Empleado principalmente en impresoras tipo Delta con cama de impresión redonda. Por defecto esta desactivado.

#define MANUAL_X_HOME_POS 0
#define MANUAL_Y_HOME_POS 0
#define MANUAL_Z_HOME_POS 0
//#define MANUAL_Z_HOME_POS 402 // For delta: Distance between nozzle and print surface after homing.

Distancia del 0 al que se encuentra el EndStop de cada eje. Si hemos activado el modo *MANUAL_HOME_POSITIONS* con estos parámetros configuramos en que coordenadas se encuentran nuestros EndStop. Solo se tiene en cuenta estos parámetros si esta activo *MANUAL_HOME_POSITIONS*. Marlin con este dato, calcula la posición del origen real.

3.8.9 Número de ejes

#define NUM_AXIS 4 // The axis order in all axis related arrays is X, Y, Z, E

Numero de ejes que tiene nuestra impresora. Normalmente son 4 X, Y, Z y el Extrusor. Por defecto esta en 4.

NOTA: En todos los parámetros en los que configuremos algún valor correspondiente a los 4 ejes, el orden para introducir los datos será (Eje X, Eje Y, Eje Z, Extrusor).

3.8.10 offset entre extrusores

Si tenemos más de un extrusor, fijaremos el offset (separación) que hay entre ellos. Un dato erróneo podría ocasionar que la impresión de un extrusor estuviera desalineada con la del otro.

// #define EXTRUDER_OFFSET_X {0.0, 20.00} // (in mm) for each extruder, offset of the hotend on the X axis
// #define EXTRUDER_OFFSET_Y {0.0, 5.00} // (in mm) for each extruder, offset of the hotend on the Y axis

En el primer parámetro fijamos la distancia en el eje X que separa cada extrusor. Los valores están por orden de los extrusores, el primero siempre estará en 0. En los valores de ejemplo, están metidos los valores para 2 extrusores, a 0mm el extrusor 0 y a 20mm esta el extrusor 1, si tenemos más de 2 extrusores, solo tendremos que añadir el dato a continuación, como por ejemplo *EXTRUDER_OFFSET_X {0.0, 20.00,40.00}.* En el segundo parámetro fijamos la distancia en el eje Y, si están alineados debería de ser 0 en todos los extrusores instalados. Por defecto están comentados los 2 parámetros y solo hay que configurarlos en caso de tener más de 1 extrusor

3.8.11 Velocidad que no requerirá aceleración

Configuramos la velocidad mínima que no requerirá del cálculo de aceleración, es decir fijamos la velocidad a la que el eje se moverá instantáneamente. Si observamos movimientos bruscos en las esquinas de nuestras piezas, deberíamos de bajar esta velocidad. Pero hay que tener en cuenta que bajar esta velocidad puede influir en cómo se imprimen los radios o arcos, ya que se calculan como pequeños segmentos y se aplicara esta velocidad. Estos parámetros se pueden modificar posteriormente con el Gcode M205.

#define DEFAULT_XYJERK 20.0 // (mm/sec)
#define DEFAULT_ZJERK 0.4 // (mm/sec)
#define DEFAULT_EJERK 5.0 // (mm/sec)

Tenemos un parámetro para el eje X e Y, otro para el eje Z y un tercero para el extrusor. Los valores por defecto suelen estar bien, solo habría que reducirlos en caso de que observemos movimientos bruscos al imprimir esquinas o al inicio de un movimiento.

3.9 Custom Gcodes

En este apartado están incluidos algunos Gcodes personalizados.

```
#define CUSTOM_M_CODES
#ifdef CUSTOM_M_CODES
  #define CUSTOM_M_CODE_SET_Z_PROBE_OFFSET 851
  #define Z_PROBE_OFFSET_RANGE_MIN -15
  #define Z_PROBE_OFFSET_RANGE_MAX -5
#endif
```

Solo tenemos definido un Gcode personalizado y se usa únicamente si esta activada la auto-nivelación de la cama. Concretamente sirve para modificar el dato de offset entre la punta del extrusor y la sonda de medida. La utilidad que tiene es poder ajustar la medida real sin tener que modificar el código de Marlin y compilarlo cada vez para cargarlo en nuestra placa. De esta manera desde la consola serie podemos ir ajustando ese offset hasta encontrar la medida adecuada. De igual forma si modificamos algo en la impresora o se mueve la sonda, podemos volver a ajustar le offset sin modificar Marlin.

La forma de usarlos es poner en la consola serie o en pronterface el Gcode *CUSTOM_M_CODE_SET_Z_PROBE_OFFSET Z12.4*, aunque también es equivalente usar el Gcode M851, *por lo que* M851 *Z12.4* seria el mismo comando, donde 12.4 es el offset introducido. Requiere tener activado el uso de la Eeprom sino perderemos el dato cuando apaguemos la impresora. Después de modificar el dato y que este correctamente ajustado, enviaremos el Gcode M500 para guardarlo.

Podemos ajustar el valor máximo y mínimo que admitiremos de offset, si el valor introducido no está entre este margen, no se tendrá en cuenta, de esta manera nos aseguramos cierto rango de valores. En *Z_PROBE_OFFSET_RANGE_MIN* introduciremos el valor mínimo que admitiremos y en *Z_PROBE_OFFSET_RANGE_MAX* el máximo.

También seria posible ajustar el offset de la sonda con el Gcode M206, pero aunque el resultado es el mismo, con M206 estamos ajustando el offset del Homing y no el parámetro *Z_PROBE_OFFSET_FROM_EXTRUDER* , así que la forma correcta de ajustarlo es con el Gcode *CUSTOM_M_CODE_SET_Z_PROBE_OFFSET* o su equivalente *M851.*

3.10 Eeprom

Activaremos el uso de la Eeprom. Podremos guardar, leer, revertir o imprimir (por puerto serie) los datos guardados con los Gcodes M500, M501, M502 y M503. Mas información en el apartado correspondiente de la sección de Gcodes.

//#define EEPROM_SETTINGS

Activaremos el uso de la Eeprom, es recomendable activarlo para poder ir ajustando algún parámetro sin necesidad de tener que modificar el código de Marlin y compilarlo para cargarlo en nuestra placa.

//#define EEPROM_CHITCHAT

Esta función, hace que se amplie la cantidad de información enviada por el puerto serie a la consola cuando usemos el Gcode M501, si esta activo el resultado seria equivalente a enviar un Gcode M503 a continuación del M501. Si queremos reducir la respuesta serie del micro controlador y además reducir el programa en 1700 byte, lo dejaremos desactivada, pero si podemos por capacidad de la placa, cuando activemos *#define EEPROM_SETTINGS* deberíamos de activar también este parámetro.

3.11 Valores predefinidos de temperatura según el material.

Podemos definir los valores por defecto de temperatura que queremos según el material seleccionado, estos valores se usan cuando tenemos un display con encoder rotativo y tenemos la capacidad de imprimir sin ordenador. Cuando controlamos al impresora con un LCD podemos precalentar el Hotend y la HeatedBed sin necesidad de tener una sd con un archivo.

```
// Preheat Constants
#define PLA_PREHEAT_HOTEND_TEMP 180
#define PLA_PREHEAT_HPB_TEMP 70
#define PLA_PREHEAT_FAN_SPEED 255   // Insert Value between 0 and 255

#define ABS_PREHEAT_HOTEND_TEMP 240
#define ABS_PREHEAT_HPB_TEMP 100
#define ABS_PREHEAT_FAN_SPEED 255   // Insert Value between 0 and 255
```

Como podemos ver, se pueden configura la temperatura para el Hotend, la temperatura para la HeatedBed, y la velocidad del ventilador del extrusor, tanto para PLA como para ABS. Los valores por defecto suelen ser correctos, si queremos podemos ajustar la temperatura según necesidades de nuestro material, ya que según el proveedor necesitaremos temperaturas diferentes. En el caso de que con ABS no queramos que funcione el ventilador de capa, cambiaremos este parámetro a 0.

3.12 LCD y SD.

En este apartado podremos configurar el tipo de LCD que tenemos y algunos de sus parámetros de control. Especificaremos si es un LCD solo, o es una placa que incluye LCD+controles. Además podremos activar el uso de tarjetas SD en nuestra impresora, ya que algunos modelos de LCD+controles ya incluyen lector SD.

3.12.1 Idioma

Este parámetro no se encuentra en el archivo Configuration.h, pero se puede configurar el idioma del menú mostrado en le LCD. Este parámetro esta en el archivo **languaje.h**

```
//#define LANGUAGE_INCLUDE GENERATE_LANGUAGE_INCLUDE(en)
```

Cambiaremos el idioma de los menús del LCD y de algunos de los mensajes que salen por la consola serie. Por defecto esta en Ingles, y los idiomas disponibles son:

- en English
- pl Polish
- fr French
- de German
- es Spanish
- ru Russian
- it Italian
- pt Portuguese

- pt-br Portuguese (Brazil)
- fi Finnish
- an Aragonés
- nl Dutch
- ca Catalan
- eu Basque-Euskera

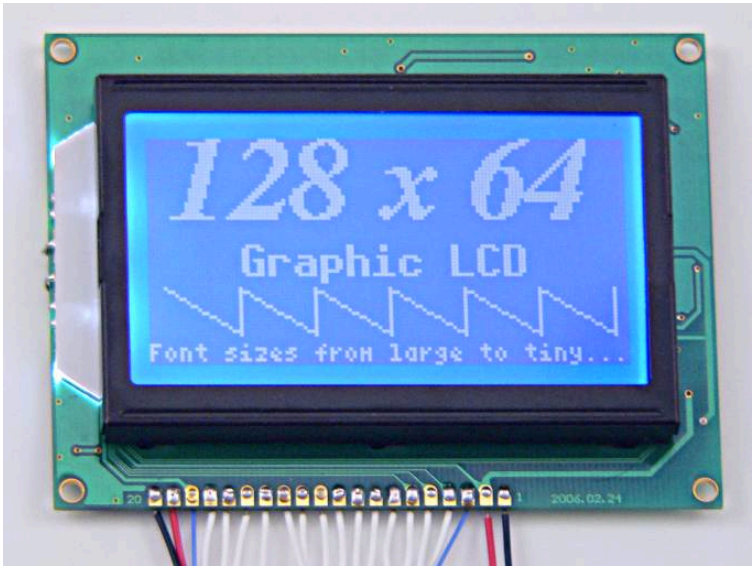
3.12.2 LCDs generales

//#define ULTRA_LCD //general LCD support, also 16x2

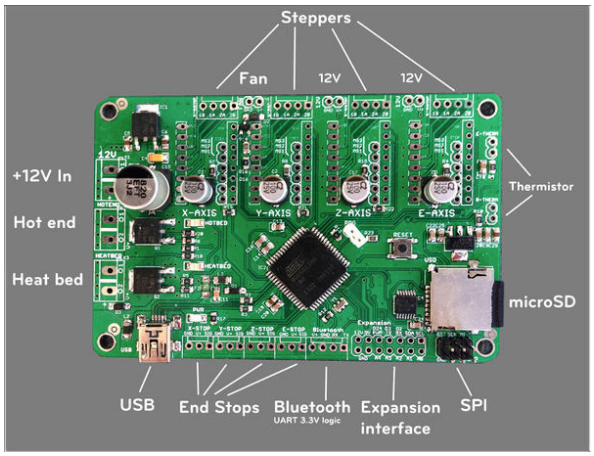
Activaremos esta función si tenemos un display sin controles de 16 caracteres x 2 líneas (o de 20x4, se puede configurar más adelante). Solo si es un LCD genérico, si está incluido en algún tipo de placa especifico para impresoras 3D con encoder o botones, deberíamos de buscar el modelo concreto más adelante y activar el correspondiente, no este.

//#define DOGLCD // Support for SPI LCD 128x64 (Controller ST7565R graphic Display Family)

Activaremos esta función si tenemos un display sin controles del tipo grafico de 128x64 puntos. Indicado para la familia de LCD controlados con un chip ST7565R. Solo se activara esta función si es un display suelto, si está integrado en alguna placa con encoder o botones buscaremos el modelo concreto que tenemos en el código que hay a continuación.



3.12.3 Tarjeta SD



//#define SDSUPPORT // Enable SD Card Support in Hardware Console

Nos permitirá usar tarjetas SD, siempre que dispongamos de un lector de tarjetas. Esta deshabilitada por defecto. Si nuestra placa controladora tiene ranura SD, o tenemos un LCD de los descritos a continuación (un LCD+controles que incluya ranura SD) activaremos esta función.

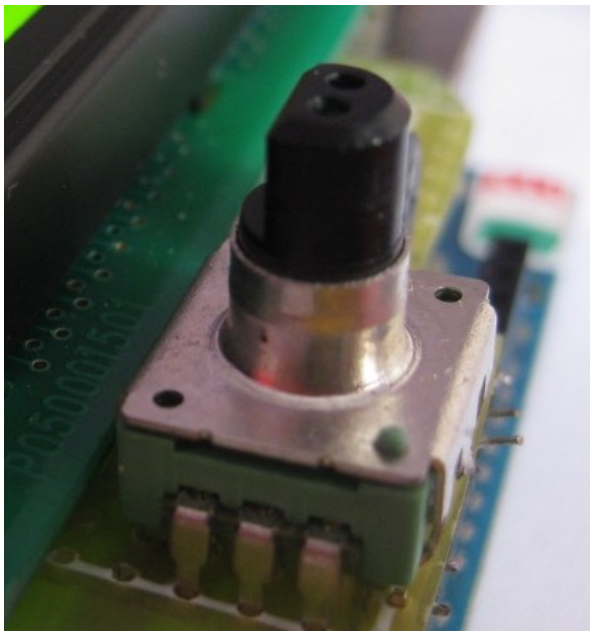
//#define SDSLOW // Use slower SD transfer mode (not normally needed - uncomment if you're getting volume init error)

Si tenemos problemas al iniciar la tarjeta SD, deberemos de habilitar este parámetro que pone la SD en modo de transferencia lento. En condiciones normales no será necesario. Sirve tanto para lectores de SD en placa controladora como los que llevan los LCD.

//#define SD_CHECK_AND_RETRY // Use CRC checks and retries on the SD communication

Activa el uso de CRC check en la comunicación con la SD, es un sistema que comprueba si hay errores a la hora de enviar o recibir datos. Si tenemos problemas al ejecutar archivos desde la SD, es recomendable activarlo. Por defecto esta desactivado. Sirve tanto para lectores de SD en placa controladora como los que llevan los LCD.

3.12.4 Ajuste del Encoder rotativo con click



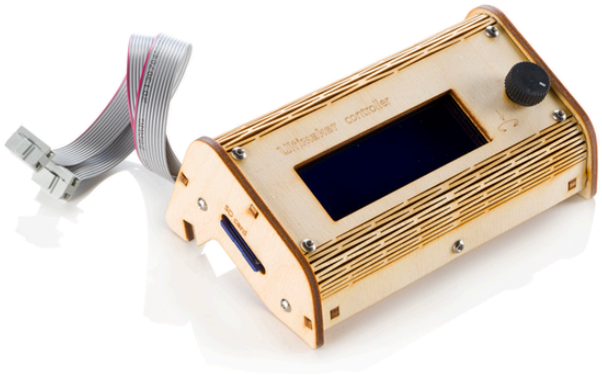
//#define ENCODER_PULSES_PER_STEP 1 // Increase if you have a high resolution encoder
//#define ENCODER_STEPS_PER_MENU_ITEM 5 // Set according to ENCODER_PULSES_PER_STEP or your liking

Si nuestro LCD dispone de un encoder rotativo, podemos fijar el comportamiento del mismo. Cuando giramos el encoder, notamos unos pequeños saltos, con estos parámetros podremos configurar la cantidad de estos pequeños saltos que tenemos que notar para que el menú cambie de línea o aumente o disminuya algún parámetro vía LCD.

Primero fijaremos el numero de pulsos para cada paso con *ENCODER_PULSES_PER_STEP*, si tenemos encoder de mucha resolución lo cambiaremos. Por defecto esta en 1 que para encoder de 20-24 pasos por vuelta es lo adecuado.

Después configuraremos el número de pasos para cada ítem del menú con *ENCODER_STEPS_PER_MENU_ITEM* , nos fijara el número de step que tenemos que girar el encoder para que se cambie el ítem del menú seleccionado en ese momento. Si vemos que el comportamiento del encoder es un poco raro y no conseguimos que con cada paso del encoder cambie de menú, hay que modificar estos parámetros y probar.

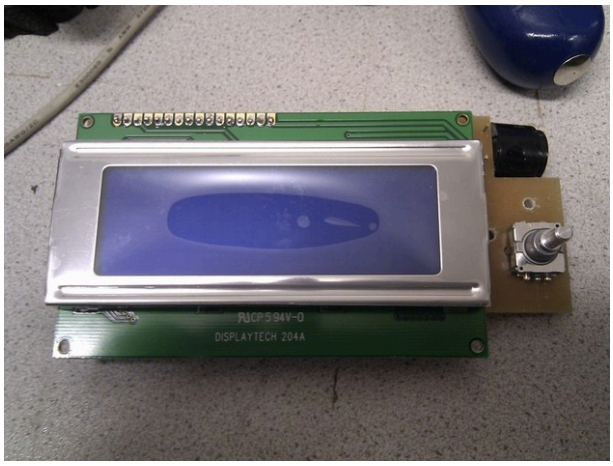
3.12.5 Ultimaker Controller

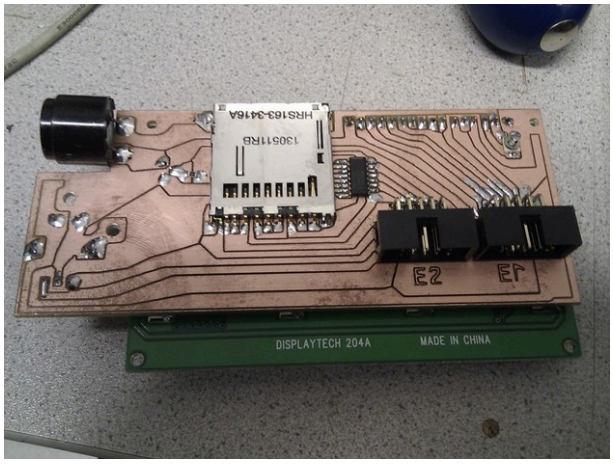


//#define ULTIMAKERCONTROLLER //as available from the Ultimaker online store.

Si tenemos un controlador con LCD de Ultimaker, lo activaremos. Más datos en https://shop.ultimaker.com/product/7/UltiController

3.12.6 Ultipanel

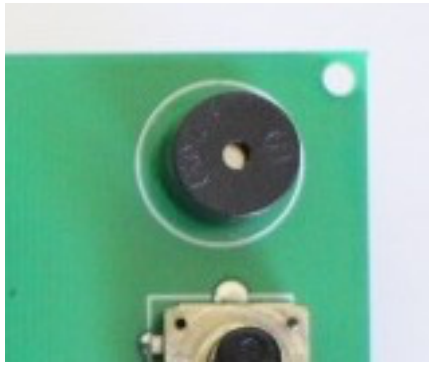




//#define ULTIPANEL //the UltiPanel as on Thingiverse

Si tenemos un Ultipanel como http://www.thingiverse.com/thing:15081.

3.12.7 Configuración Buzzer en LCD



//#define LCD_FEEDBACK_FREQUENCY_HZ 1000 // this is the tone frequency the buzzer plays when on UI feedback. ie Screen Click
//#define LCD_FEEDBACK_FREQUENCY_DURATION_MS 100 // the duration the buzzer plays the UI feedback sound. ie Screen Click

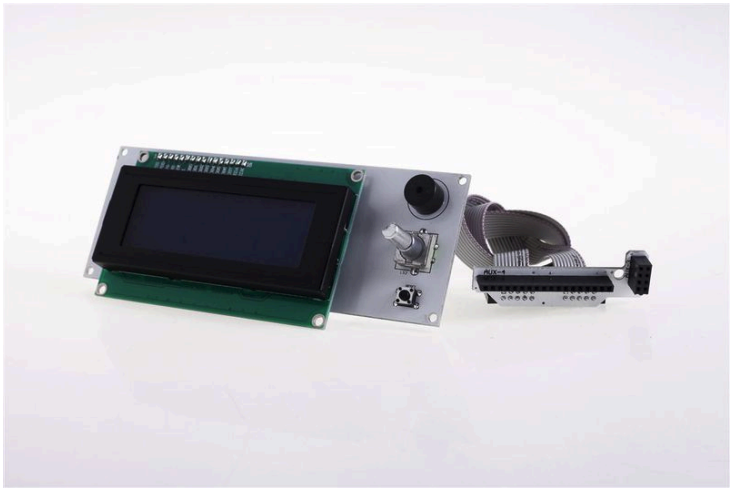
Algunas placas LCD incluyen un buzzer o micro altavoz, desde estos parámetros podemos configurar el tono del sonido y la duración del mismo.

3.12.8 Makr Panel

//#define MAKRPANEL

LCD Makr-panel de MaKr3d, más información http://reprap.org/wiki/MaKrPanel

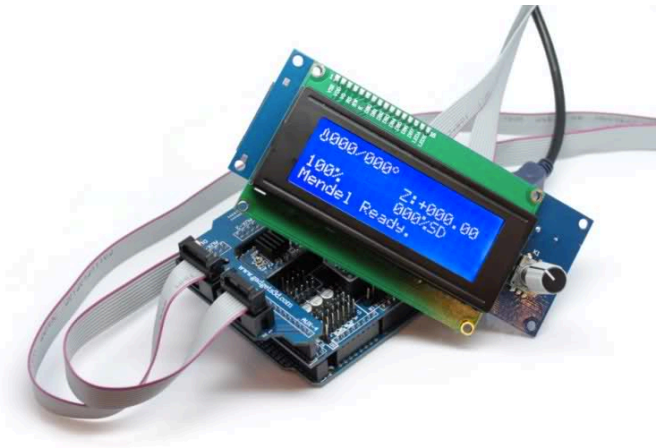
3.12.9 LCD Reprap Discount Smart Controller



//#define REPRAP_DISCOUNT_SMART_CONTROLLER

RepRap discount LCD Smart controller, el que tiene un LCD de 20x4. Más datos http://reprap.org/wiki/RepRapDiscount_Smart_Controller

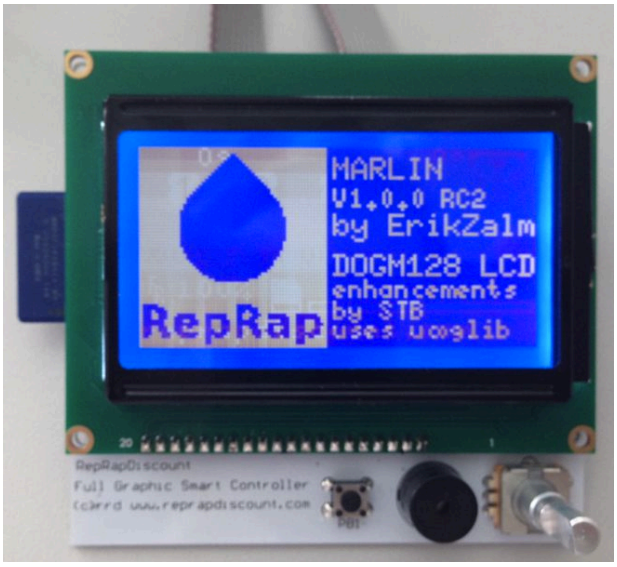
3.12.10 G3d LCD/SD



//#define G3D_PANEL

G3D LCD/SD de GADGET3D. Más datos en
http://reprap.org/wiki/RAMPS_1.3/1.4_GADGETS3D_Shield_with_Panel y en
https://gadgets3d.eu/index.php?route=product/product&path=61&product_id=65

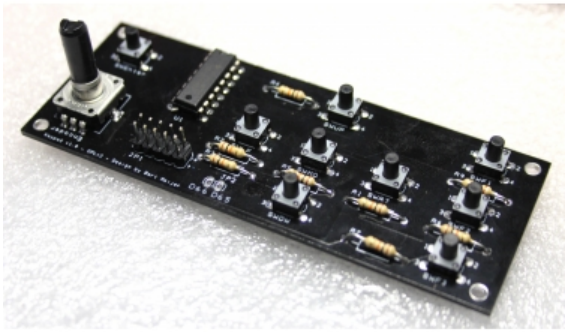
3.12.11 LCD Reprap Discount Full Graphics Smart Controller



//#define REPRAP_DISCOUNT_FULL_GRAPHIC_SMART_CONTROLLER

ReprapDiscount Full graphics, el que tiene la pantalla grafica de 128x64 puntos. Hay que instalar la librería U8glib en nuestra carpeta de librerías de Arduino. Más información http://reprap.org/wiki/RepRapDiscount_Full_Graphic_Smart_Controller

3.12.12 Teclado Reprap World



//#define REPRAPWORLD_KEYPAD

Si tenemos el teclado RepRapWorld Keypad. No lleva LCD. Más datos http://reprapworld.com/?products_details&products_id=202&cPath=1591_1626

//#define REPRAPWORLD_KEYPAD_MOVE_STEP 10.0 // how much should be moved when a key is pressed, eg 10.0 means 10mm per click

Si tenemos el teclado RepRapWorld Keypad, podemos definir cuantos milímetros se mueve el eje con cada pulsación. Medida en mm.

3.12.13 LCD Elefu

//#define RA_CONTROL_PANEL

Si tenemos un LCD de Elefu. Hay que instalar la librería LiquidCrystal_I2C.h en nuestra carpeta de librerías de Arduino. Más información en http://www.elefu.com/index.php?route=product/product&product_id=53

3.12.14 Ampliación de definiciones

A partir de este punto, según el LCD que hemos activado, se cargan ciertos parámetros o habilitan funciones. Generalmente no hay que tocar nada, lo incluyo a nivel informativo.

```
//automatic expansión

#if defined (MAKRPANEL)
 #define DOGLCD
 #define SDSUPPORT
 #define ULTIPANEL
 #define NEWPANEL
 #define DEFAULT_LCD_CONTRAST 17
#endif

#if defined (REPRAP_DISCOUNT_FULL_GRAPHIC_SMART_CONTROLLER)
 #define DOGLCD
 #define U8GLIB_ST7920
 #define REPRAP_DISCOUNT_SMART_CONTROLLER
#endif

#if defined(ULTIMAKERCONTROLLER) || defined(REPRAP_DISCOUNT_SMART_CONTROLLER) ||
defined(G3D_PANEL)
 #define ULTIPANEL
 #define NEWPANEL
#endif

#if defined(REPRAPWORLD_KEYPAD)
  #define NEWPANEL
  #define ULTIPANEL
#endif
#if defined(RA_CONTROL_PANEL)
 #define ULTIPANEL
 #define NEWPANEL
 #define LCD_I2C_TYPE_PCA8574
 #define LCD_I2C_ADDRESS 0x27   // I2C Address of the port expander
#endif
```

3.12.15 LCD I2C

I2c es un protocolo de comunicación con el micro controlador que solo usa 2 cables, mas información http://es.wikipedia.org/wiki/I²C. En este apartado se configuran LCDs I2C.

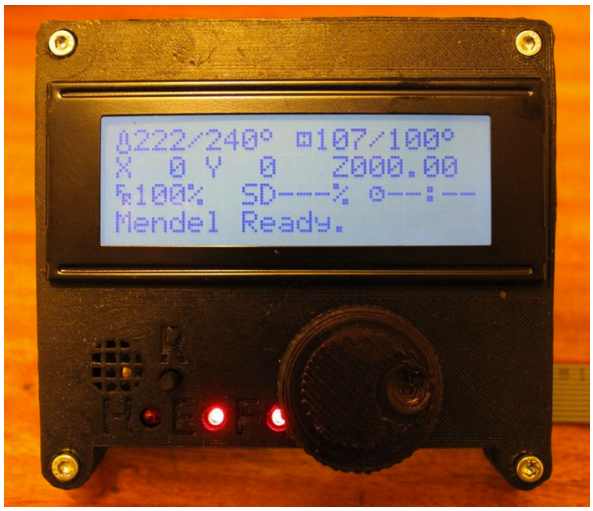
//#define LCD_I2C_SAINSMART_YWROBOT

LCD que usa una pequeña placa soldada a los 16 pines de la pantalla y que proporciona una conexión I2C con la placa controladora. Está función esta indicada para placas interface de SainSmart o de YWRobot. Hay que cargar la librería LiquidCrystal_I2C de FMalpartida en la carpeta de librerías de Arduino. Un ejemplo de LCD http://www.sainsmart.com/sainsmart-iic-i2c-twi-serial-2004-20x4-lcd-module-shield-for-arduino-uno-mega-r3.html, y la plaquita en cuestión aquí

```
#ifdef LCD_I2C_SAINSMART_YWROBOT
  #define LCD_I2C_TYPE_PCF8575
  #define LCD_I2C_ADDRESS 0x27   // I2C Address of the port expander
  #define NEWPANEL
  #define ULTIPANEL
#endif
```

Si tenemos un LCD I2C que use algún modulo de SainSmart o de YWRobot, hay que configurar la dirección I2C correcta en *LCD_I2C_ADDRESS,* consulta la documentación del modulo.

3.12.16 Panelolu 2



//#define LCD_I2C_PANELOLU2

LCD Panelolu2 que incluye LCD, encoder, botones y leds. Más información http://reprap.org/wiki/Panelolu2. Usa la librería LiquidTWI2, asegúrate de que este instalada en la carpeta de librerías del entorno de programación Arduino.

```
#ifdef LCD_I2C_PANELOLU2
 #define LCD_I2C_TYPE_MCP23017
 #define LCD_I2C_ADDRESS 0x20 // I2C Address of the port expander
 #define LCD_USE_I2C_BUZZER //comment out to disable buzzer on LCD
 #define NEWPANEL
 #define ULTIPANEL

 #ifndef ENCODER_PULSES_PER_STEP
      #define ENCODER_PULSES_PER_STEP 4
 #endif

 #ifndef ENCODER_STEPS_PER_MENU_ITEM
      #define ENCODER_STEPS_PER_MENU_ITEM 1
 #endif


 #ifdef LCD_USE_I2C_BUZZER
      #define LCD_FEEDBACK_FREQUENCY_HZ 1000
      #define LCD_FEEDBACK_FREQUENCY_DURATION_MS 100
 #endif
#endif
```

Solo para el LCD Panelolu2, podemos configurar ciertos parámetros desde este bloque. En *#define LCD_I2C_ADDRESS* introduciremos la dirección I2C del modulo, necesaria para que el micro controlador se pueda comunicar con el LCD

#define ENCODER_PULSES_PER_STEP y *#define ENCODER_STEPS_PER_MENU_ITEM* definen el comportamiento del encoder rotativo al igual que en los parámetros explicados anteriormente.

#define LCD_FEEDBACK_FREQUENCY_HZ y *#define LCD_FEEDBACK_FREQUENCY _DURATION_MS* configuran el tono y la duración del sonido que se reproduce en el buzzer incluido en la placa, cada vez que se pulsa el encoder.

3.12.17 LCD Panucatt

//#define LCD_I2C_VIKI

LCD de Panucatt. Más información http://www.panucatt.com/product_p/vikilcd.htm. Hay que instalar la librería LiquidTWI2 versión v1.2.3 o posterior en la carpeta librerías de Arduino.

```
#ifdef LCD_I2C_VIKI
  #define LCD_I2C_TYPE_MCP23017
  #define LCD_I2C_ADDRESS 0x20 // I2C Address of the port expander
  #define LCD_USE_I2C_BUZZER //comment out to disable buzzer on LCD (requires LiquidTWI2 v1.2.3 or later)
  #define NEWPANEL
  #define ULTIPANEL
#endif
```

Si usamos el LCD de Panucatt, aquí podemos cambiar la dirección I2C en *LCD_I2C_ADDRESS que será necesaria para que el micro controlador se comunique con el LCD.*

Si queremos podemos quitar el sonido del buzzer comentando la línea #define *LCD_USE_I2C_BUZZER.*

3.12.18 LCD SAV con registro de desplazamiento

//#define SAV_3DLCD

Si usamos un LCD SAV con registros de desplazamiento. Más información en http://3g1l.com/blog-cheap-arduino-2-wire-lcd-display-0 y en https://bitbucket.org/fmalpartida/new-liquidcrystal/wiki/schematics - !shiftregister-connection

```
#ifdef SAV_3DLCD
  #define SR_LCD_2W_NL    // Non latching 2 wire shiftregister
  #define NEWPANEL
  #define ULTIPANEL
#endif
```

Ampliación de definiciones para este tipo de LCD.

3.12.19 Ampliación de configuraciones

Estos 2 bloques de código terminan de configurar los LCD según el modelo.

```
#ifdef ULTIPANEL
//  #define NEWPANEL  //enable this if you have a click-encoder panel
 #define SDSUPPORT
 #define ULTRA_LCD
 #ifdef DOGLCD // Change number of lines to match the DOG graphic display
  #define LCD_WIDTH 22
  #define LCD_HEIGHT 5
 #else
  #define LCD_WIDTH 20
  #define LCD_HEIGHT 4
 #endif
```

El primer bloque se usa si tenemos instalado cualquier LCD+controles (encoder, botones o leds) de los arriba descritos (ULTIMAKERCONTROLLER, ULTIPANEL, MAKRPANEL, REPRAP_DISCOUNT_SMART_CONTROLLER, G3D_PANEL, REPRAP_DISCOUNT_FULL_ GRAPHIC_SMART_CONTROLLER, RA_CONTROL_PANEL, LCD_I2C_SAINSMART_YWROBOT, LCD_I2C_PANELOLU2, LCD_I2C_VIKI)

Bloque 1 (LCD+controles):

- *// #define NEWPANEL*, lo activaremos en caso de que tengamos un encoder rotativo para navegar por los menús. Realmente este parámetro solo es necesario activarlo si tenemos un modelo ULTIPANEL de LCD+controles, ya que para el resto de LCD se activa en su sección correspondiente. Por defecto esta desactivado.
- *#define SDSUPPORT* , activa el uso de la SD. Si nuestro LCD+controles tiene un lector de tarjetas SD, hay que dejarlo activado, en caso de que no lo lleve o simplemente no queramos que funcione, comentaremos esta línea.
- Las 2 líneas *#define LCD_WIDTH 20* y *#define LCD_HEIGHT 5* configuran las líneas de texto que tendrá el LCD grafico (el de 128x64 puntos). Solo es a efectos del menú de control, por defecto esta en 5 líneas de 20 caracteres. Es mejor dejarlo como esta, ya que otras medidas requeriría de otro tipo de fuente grafica que no está disponible en el código.
- Las 2 siguientes (después de *#else*), se refiere a un LCD no grafico, es decir de los de 16x2 o 20x4 líneas. *#define LCD_WIDTH 20* y *#define LCD_HEIGHT 4* configuran el número de líneas y caracteres del LCD. Normalmente las placas de LCD+controles incluyen un display de 20x4 líneas, pero si tiene otra medida será necesario cambiar estos parámetros.

```
#else //no panel but just LCD
 #ifdef ULTRA_LCD
 #ifdef DOGLCD // Change number of lines to match the 128x64 graphics display
  #define LCD_WIDTH 22
  #define LCD_HEIGHT 5
 #else
  #define LCD_WIDTH 16
  #define LCD_HEIGHT 2
 #endif
 #endif
#endif
```

El segundo bloque se aplica en caso de tener unos simples LCD y no hemos activado ningún LCD+controles de los anteriores. Veamos como:

Bloque 2 (LCD solo):

- Las 2 líneas *#define LCD_WIDTH 20* y *#define LCD_HEIGHT 5* configuran las líneas de texto que tendrá el LCD grafico (el de 128x64 puntos). Solo es a efectos del menú de control, por defecto esta en 5 líneas de 20 caracteres. Es mejor dejarlo como esta, ya que otras medidas requeriría de otro tipo de fuente grafica que no está disponible en el código.
- Las 2 siguientes (después de *#else*), se refiere a un LCD no grafico, es decir de los de 16x2 o 20x4 líneas. *#define LCD_WIDTH 20* y *#define LCD_HEIGHT 4* configuran el número de líneas y caracteres del LCD. Normalmente las placas de LCD+controles incluyen un display de 20x4 líneas, pero si tiene otra medida será necesario cambiar estos parámetros.

3.12.20 Ajuste de contraste en LCD Gráficos

```
#ifdef DOGLCD
# ifndef DEFAULT_LCD_CONTRAST
#  define DEFAULT_LCD_CONTRAST 32
# endif
#endif
```

Configura el contraste por defecto solo para los LCDs gráficos (128x64 puntos). Este parámetro se puede ajustar luego con el Gcode M250. Por defecto esta en 32, valores aceptados entre 0-63.

3.13 Frecuencia PWM, y control de velocidad del ventilador de capa

//#define FAST_PWM_FAN

Si usamos el control de velocidad del ventilador de capa y observamos que hace ruido , podemos aumentar la frecuencia de control PWM, aunque esto aumentara la temperatura en el transistor FET que controla el ventilador. Si no es porque haga mucho ruido, mejor dejarlo como está.

//#define FAN_SOFT_PW

Podemos activar y controlar los parámetros del ventilador de capa. Si activamos el control PWM, podremos variar la velocidad de funcionamiento del ventilador desde el software, es decir, no solo estar encendido o apagado, podremos tenerlo a distintas velocidades, eso sí, necesitamos conectarlo a una salida de la placa controladora y no directamente a 12V. Consulta la documentación de tu placa para saber cuál es la salida para esto. Por defecto esta desactivado.

#define SOFT_PWM_SCALE 0

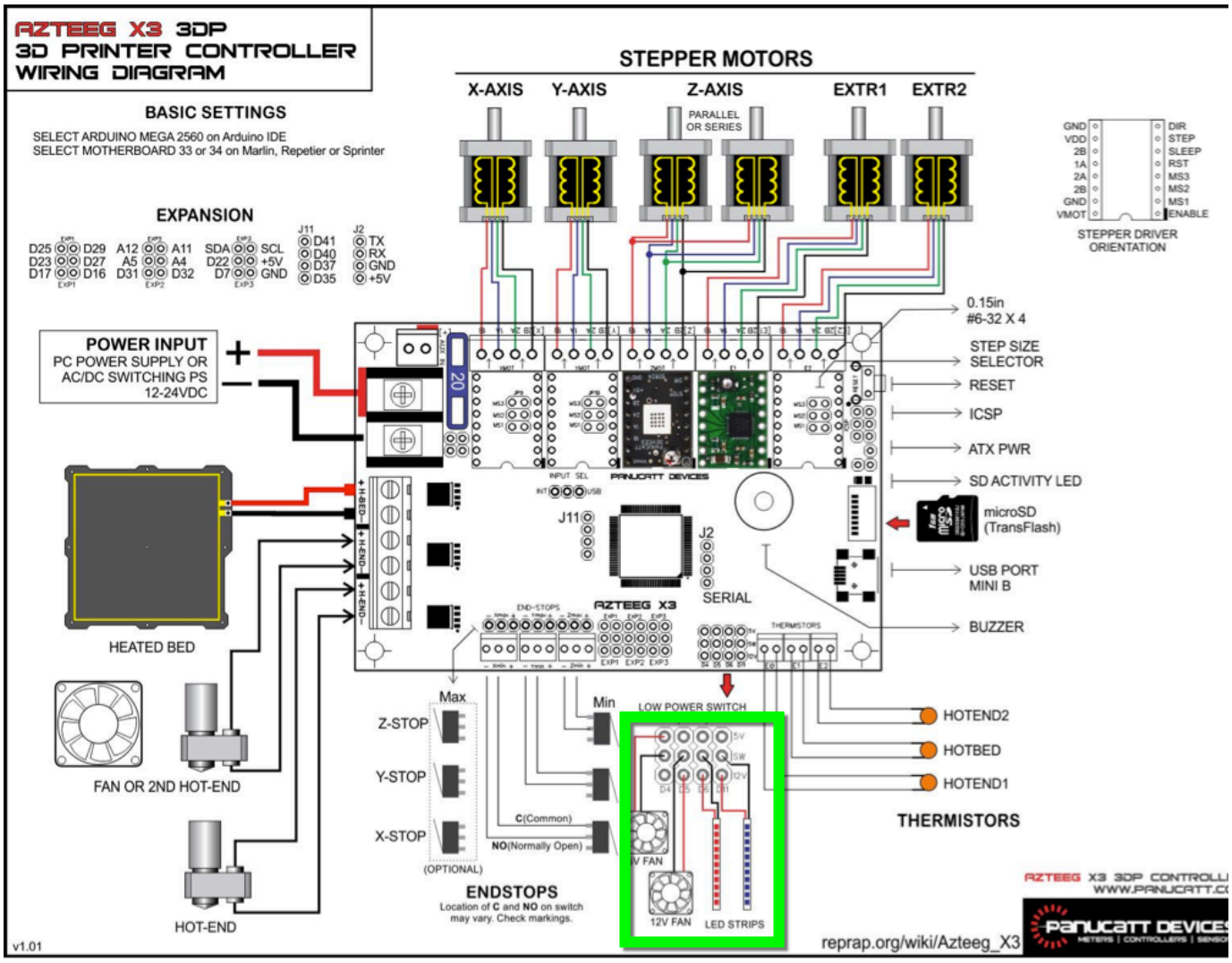
Para controlar la temperatura en los Hotend y la HeatedBed se usa una frecuencia PWM. Normalmente se utiliza una frecuencia baja para no interferir en el Hardware, pero si esta frecuencia es demasiado baja se puede incrementar modificando el parámetro SOFT_PWM_SCALE. Incrementarlo en 1 dobla la frecuencia PWM que afecta a los Hotend, HeatedBed y al ventilador de capa (si FAN_SOFT_PWM está habilitado).

No obstante aumentar este parámetro disminuye la resolución de la señal, con un valor 0 hay 128 puntos efectivos de control, si aumentamos este valor disminuirán los puntos. Si no estás seguro de lo que haces mejor no tocar este parámetro.

3.14 Leds de color según temperatura.

//#define TEMP_STAT_LEDS

En la placa Azteeg X3 pro, se pueden conectar 2 tiras de leds, una azul y otra roja que se encenderán en función de la temperatura del Hotend y de la HeatedBed. Sirve de aviso para saber si hay peligro en tocar el Hotend. Si la temperatura de cualquier elemento calefactor es menor de 54C entonces estará encendido el led azul, si es superior se encenderá el led rojo. En el esquema siguiente podemos ver como se conectan las tiras de leds, la azul al pin D11 y la roja al D6. Este parámetro solo funciona en la placa Azteeg X3 Pro 8. Por defecto esta deshabilitada.



3.15 Tomar fotos con Gcode M240

// #define PHOTOGRAPH_PIN 23

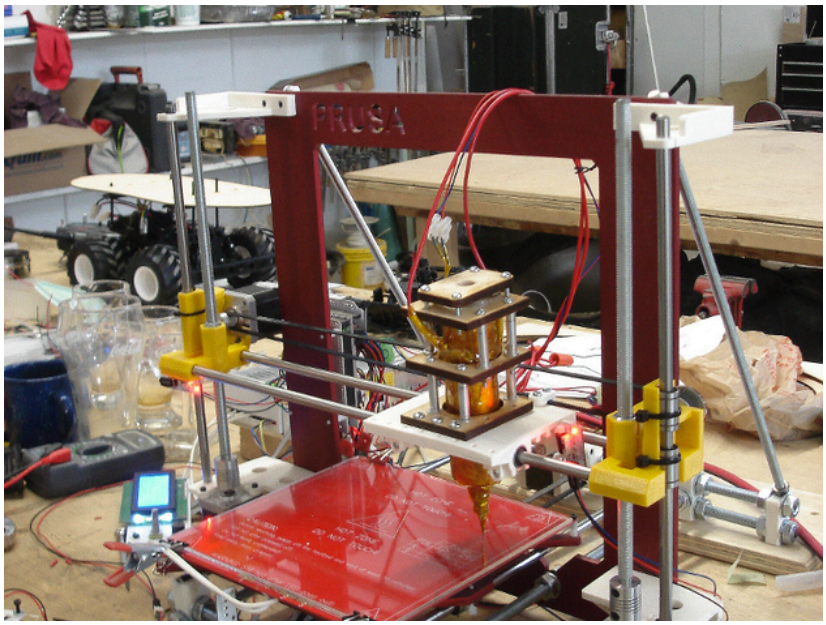
Si lo activamos, el Gcode M240 disparara una cámara de fotos emulando un control remoto Canon RC-1. El disparador estará conectado al pin indicado en este parámetro. Más datos en http://www.doc-diy.net/photo/rc-1_hacked/ y en http://captain-slow.dk/2014/03/09/3d-printing-timelapses/. Por defecto esta deshabilitado.

3.16 Repara errores en Gcode de SkeinForge.

//#define SF_ARC_FIX

SkeinForge envía códigos Gcode erróneos cuando hace arcos, si habilitamos este parámetro, Marlin corrige estos errores. Por defecto esta deshabilitado.

3.17 Uso del extrusor de pasta BariCUDA



//#define BARICUDA

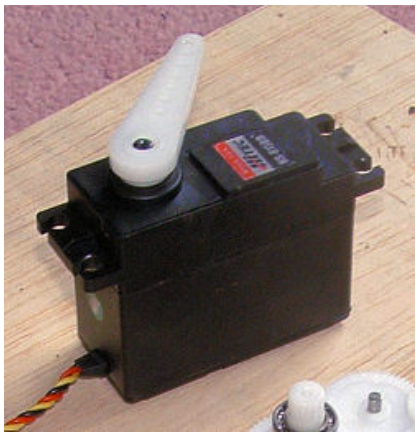
Permite que se pueda instalar un el extrusor de pasta tipo BariCUDA. Al habilitarlo permitimos el control de las válvulas de aire necesarias usando los Gcodes M126, M127, M128 y M129. Más información http://www.thingiverse.com/thing:26343 y en https://github.com/jmil/BariCUDA

3.18 Uso de BlinkM led

//#define BLINKM

Permite el uso del led BlinkM controlado por I2C, con el Gcode M150. Más información en http://thingm.com/products/blinkm/

3.19 Configuración de Servos RC



Si para el nivelado de la cama usamos un servo con un EndStop en la punta, que tiene que girar cuando haga las mediciones y luego se retrae, se configura desde esta sección.

//#define NUM_SERVOS 3 // Servo index starts with 0 for M280 command

Numero de servos que tenemos en nuestra impresora, podemos tener un servo por eje. Por defecto esta comentado, si tenemos un servo lo habilitaremos y cambiaremos el valor a 1. El número máximo de servos que puede controlar Marlin es de 3.

Si usamos el Gcode M280, el índice de servos empieza en 0, es decir el servo 1 seria el índice 0, ejemplo M280 P0 S180.

//#define SERVO_ENDSTOPS {-1, -1, 0} // Servo index for X, Y, Z. Disable with -1

Configuramos que servo está instalado en que eje. Usamos el índice correspondiente a cada servo para asignarlo al eje en el que este instalado. Si esta a -1 esta desactivado para ese eje, 0 u otro valor activado. Por defecto esta Deshabilitado.

Los datos introducidos por defecto *SERVO_ENDSTOPS {-1, -1, 0}* nos configurarían el servo con el índice 0 como servo para el eje Z.

//#define SERVO_ENDSTOP_ANGLES {0,0, 0,0, 70,0} // X,Y,Z Axis Extend and Retract angles

Configuramos los ángulos del servo, a que grados estará desplegado y a que otros estará recogido. Son pares de valores y están especificados uno para cada eje, siendo el primer valor los grados extendido y el segundo los grados recogido. Por defecto esta deshabilitado.

Los datos introducidos por defecto *SERVO_ENDSTOP_ANGLES {0,0, 0,0, 70,0}* nos configuran solo para el eje Z un angulo extendido de 70º y otro angulo para cuando este recogido de 0º

3.20 Sensor de diámetro de filamento

Podemos instalar un sensor de diámetro para poder ajustar el flujo de material según varíe. De momento solo tiene soporte para 1 extrusor (extruder 0), y la medida que tiene que proporcionar el sensor será un voltaje de 0-5V equivalente al diámetro, por ejemplo con un filamento de 1,75mm de diámetro el sensor dará una lectura de 1,75V. Más información en https://www.youmagine.com/designs/filament-diameter-sensor.

Las placas compatibles son:

• RAMSP1.4 que usa la entrada analógica 5 en el conector AUX2.
• Printrboard que usa la entrada analógica 2 en el conector Aux2.
• Rambo que usa la entrada analógica 3.
• Cualquier otra placa requerirá tener una entrada analógica libre en la placa y configurar la librería Pins.h correspondiente a nuestra placa.

//#define FILAMENT_SENSOR

Habilita el uso del sensor de diámetro. Si lo habilitamos podremos usar los Gcodes M404, M405, M406, M407 para configurarlo. Por defecto esta deshabilitado.

#define FILAMENT_SENSOR_EXTRUDER_NUM 0 //The number of the extruder that has the filament sensor (0,1,2)

Como ya hemos comentado, solo soporta el uso en 1 extrusor, con este parámetro podemos indicarle que extrusor lleva el sensor de diámetro. Lo más normal es que sea el 0. Por defecto está configurado el 0, pero no tendrá efecto si no está habilitado *FILAMENT_SENSOR.*

#define MEASUREMENT_DELAY_CM 14 //measurement delay in cm. This is the distance from filament sensor to middle of barrel

Fijamos la distancia en cm que hay entre el sensor que mide el diámetro y la mitad del Hotend. Esto lo emplea Marlin para saber en qué momento justo llega al extrusor el filamento medido y poder ajustar el flujo de material. El valor por defecto no sirve, este parámetro hay que cambiarlo y ajustarlo a nuestra instalación.

#define DEFAULT_NOMINAL_FILAMENT_DIA 3.0 //Enter the diameter (in mm) of the filament generally used (3.0 mm or 1.75 mm) - this is then used in the slicer software. Used for sensor reading validation

Definimos el diámetro nominal de nuestro extrusor, lo normal es 1,75 o 3mm.

#define MEASURED_UPPER_LIMIT 3.30 //upper limit factor used for sensor reading validation in mm
#define MEASURED_LOWER_LIMIT 1.90 //lower limit factor for sensor reading validation in mm

Medidas máximas y mínimas que admitiremos como lecturas del sensor. Si el valor leído esta fuera de ese margen se dará por hecho que el sensor no funciona correctamente y se despreciara la lectura. Por defecto están los valores para filamento de 3mm, para filamento de 1,75 podríamos emplear, por ejemplo, 2 y 1,15 mm.

#define MAX_MEASUREMENT_DELAY 20 //delay buffer size in bytes (1 byte = 1cm)- limits maximum measurement delay allowable (must be larger than MEASUREMENT_DELAY_CM and lower number saves RAM)

Configura el tamaño del buffer donde se guardan los datos del diámetro medido, hay que tener en cuenta que por cada cm que tengamos configurado en *MEASUREMENT_DELAY_CM* hay que añadir 1 byte, y añadirle además un pequeño margen extra. Por otro lado, si configuramos el buffer muy grande corremos el riesgo de llenar la memoria RAM del sistema

//defines used in the code
#define DEFAULT_MEASURED_FILAMENT_DIA DEFAULT_NOMINAL_FILAMENT_DIA //set measured to nominal initially

Uso a nivel interno, no tocar.

//#define FILAMENT_LCD_DISPLAY

Si tenemos un LCD y queremos que muestre en la línea inferior el diámetro medido.

4 Configuraciones Avanzadas. Detalle de la librería Configuration_Adv.h

Vamos a ver los parámetros avanzados que podemos configurar en el archivo Configuration_Adv.h. Normalmente no hay que tocar ningún parámetro de este archivo, son ajustes más avanzados y algunos de ellos precisan de un conocimiento mayor del funcionamiento de la impresora.

4.1 Ajustes Térmicos

4.1.1 Ajustes del control de la HeatedBed en modo bang-bang

```
#ifdef BED_LIMIT_SWITCHING
  #define BED_HYSTERESIS 2 //only disable heating if T>target+BED_HYSTERESIS and enable heating if T>target-BED_HYSTERESIS
#endif
```

Si en Configuration.h hemos deshabilitado el modo PID, y por tanto activo el modo bang-bang para el control de la temperatura de la HeatedBed, podemos configurar el uso de la histéresis (diferencia de temperatura).

La HeatedBed estará ON si la temperatura medida es menor que la temperatura objetivo menos BED_HYSTERESIS, y estará OFF si la temperatura medida es mayor que la temperatura objetivo más BED_HYSTERESIS. Esto sirve para limitar la cantidad de veces que se encenderá y apagara la HeatedBed, un valor muy alto hará que no tenga una temperatura más o menos constante, y por otro lado un valor pequeño producirá muchos ciclos de ON OFF. Por defecto esta en 2 grados y solo funciona si hemos activado el parámetro BED_LIMIT_SWITCHING en el archivo Configuration.h

```
#define BED_CHECK_INTERVAL 5000 //ms between checks in bang-bang control
```

Con este parámetro configuramos el ciclo de tiempo en el que el control de la HeatedBed verifica la temperatura y la enciende ON o la apaga OFF. Esto reduce el número de veces que se encenderá y apagara la HeatedBed, pero un valor muy alto puede hacer que la temperatura varíe mucho, sobre todo si nuestra HeatedBed es muy potente. Este parámetro es útil si tenemos un relé mecánico para controlar nuestra HeatedBed y no queremos oír muchas veces el “click” del relé. Por defecto esta en 5 segundos (5000 ms) y solo funciona en modo bang-bang.

4.1.2 Verificación del estado del Hotend

Podemos configurar la verificación del estado del Hotend, es un sistema que verifica que al inicio del ciclo, cuando enviamos un Gcode M104 o M109, empiece a calentar el Hotend. Si estuviera fundido el cartucho cerámico o cortado un cable, nos daría un error y apagaría el Hotend.

No es una verificación continua solo se produce al inicio del ciclo, es decir, una vez pasado un tiempo, verifica la temperatura y si es menor que la fijada se apaga el Hotend y si es mayor deja de verificar.

Solo se iniciara este control si la diferencia entre la temperatura objetivo y la temperatura actual del Hotend es de al menos 2 veces la indicada en WATCH_TEMP_INCREASE.

//#define WATCH_TEMP_PERIOD 40000 //40 seconds
//#define WATCH_TEMP_INCREASE 10 //Heat up at least 10 degree in 40 seconds

Tendremos que descomentar las 2 líneas, en WATCH_TEMP_PERIOD fijaremos el tiempo y en WATCH_TEMP_INCREASE los grados. Una vez iniciado el ciclo de calentamiento, y pasado el tiempo que hemos fijado medirá la temperatura que tiene que haber aumentado como mínimo los grados indicados, si no los ha alcanzado se apaga el extrusor y enviara un mensaje por consola y por el LCD de “Heating failed”.

Por defecto está apagado, si queremos tener un mayor control por seguridad podemos activarlos, pero hay que estar seguros que la potencia de nuestro Hotend es la suficiente para subir esos grados en ese tiempo holgadamente, sino habrá que ajustar estos parámetros

Esta función también se podría emplear para verificar al inicio del calentamiento que el termistor esta en su sitio y no se ha salido. Pero solo al inicio, una vez alcanzada la temperatura objetivo ya no funciona este control, en caso de querer seguir monitorizando por seguridad el funcionamiento tendremos que activar la función *THERMAL_RUNAWAY_PROTECTION_PERIOD* en el archivo Configuration.h.

4.1.3 Corrección de temperatura según la velocidad de extrusión

Si tenemos definido el modo PID para el Hotend, podemos usar un parámetro experimental que ajusta la potencia del Hotend en función de la velocidad de extrusión.

```
#ifdef PIDTEMP
#define PID_ADD_EXTRUSION_RATE
 #ifdef PID_ADD_EXTRUSION_RATE
   #define  DEFAULT_Kc (1) //heating power=Kc*(e_speed)
 #endif
#endif
```

Hay que introducir el parámetro Kc, si Kc se ha calculado bien el aumento de velocidad de extrusión será compensado aumentando proporcionalmente la potencia aplicada. Por defecto esta activado y a 1.

Aunque esto suena muy bien, no he sido capaz de encontrar en el código como se realiza esa corrección, así que **creo que no es funcional** y se ha quedado colgado en el código de alguna modificación posterior en la que se ha eliminado. En mi opinión se ha sustituido por el parámetro que hay a continuación AUTOTEMP.

```
#define AUTOTEMP
#ifdef AUTOTEMP
 #define AUTOTEMP_OLDWEIGHT 0.98
#endif
```

Si tenemos habilitada la función AUTOTEMP, podemos controlar la temperatura del Hotend en función de la velocidad. Se calcula teniendo en cuenta todas las líneas de gcode almacenadas en el buffer, y se obtienen los máximos steps/sec del motor del extrusor. Por lo general se requiere de una mayor temperatura del Hotend a velocidades superiores.

Para activar el modo autotemp hay que enviar un Gcode M109 S<mintemp> B<maxtemp> F<factor>. La temperatura objetivo se calcula con la formula mintemp+factor*[steps/sec] y estará limitada por la mintemp y la maxtemp. Un ejemplo seria M109 S215 B260 F1. Para salir del modo autotemp solo hay que enviar un Gcode M109 sin el valor F. El parámetro AUTOTEMP_OLDWEIGHT fijara cuanta importancia le damos a la temperatura anterior, es un parámetro que suavizara el control autotemp. Por defecto esta habilitada a falta de activarla con M109

4.1.4 Mostrar valores ADC de la temperatura

//#define SHOW_TEMP_ADC_VALUES

Al enviar un Gcode M105, devolverá la temperatura en grados centígrados y además el valor ADC leído. Se puede usar para ver las lecturas reales y calibrar termistores. Por defecto esta desactivado.

4.1.5 Extrusión preventiva

Si la impresora esta inactiva, pero la temperatura está por encima de la temperatura mínima indicada, cada vez que pase el tiempo asignado, se extruira una cantidad de filamento. Su principal función es que si tenemos la impresora esperando, pero el Hotend está caliente y no queremos que se queme el filamento en el interior.

//#define EXTRUDER_RUNOUT_PREVENT

Activa la extrusión preventiva. Por defecto esta desactivada. Los parámetros que hay a continuación se usan para configurarla.

#define EXTRUDER_RUNOUT_MINTEMP 190

Fija la temperatura mínima que tendrá que tener el Hotend para que extruya periódicamente. Por defecto esta en 190 grados.

#define EXTRUDER_RUNOUT_SECONDS 30.

Cada vez que pase este tiempo, se extruira filamento, siempre y cuando la temperatura sea superior a la MINTEMP.

#define EXTRUDER_RUNOUT_ESTEPS 15.
#define EXTRUDER_RUNOUT_EXTRUDE 100

Con estos 2 parámetros configuramos la cantidad de filamento que extruira, la formula que aplica es la siguiente :

cantidad a extruir=EXTRUDER_RUNOUT_EXTRUDE * EXTRUDER_RUNOUT_ESTEPS/pasos por unidad del extrusor.

Mi consejo es poner en EXTRUDER_RUNOUT_ESTEPS los pasos por unidad que hemos fijado para nuestro extrusor, y en EXTRUDER_RUNOUT_EXTRUDE la cantidad de filamento en mm, y ya no tendremos que hacer cálculos complicados

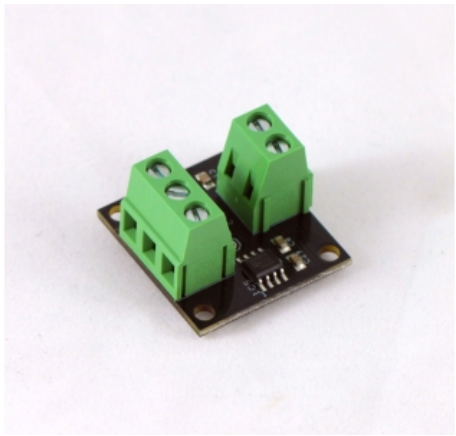
#define EXTRUDER_RUNOUT_SPEED 1500. //extrusion speed

Velocidad a la que extruira el filamento en mm por minuto, la formula que aplica es:

Velocidad = EXTRUDER_RUNOUT_SPEED/60.*EXTRUDER_RUNOUT_ESTEPS/pasos por unidad del extrusor.

De ahí la importancia de fijar en EXTRUDER_RUNOUT_ESTEPS los pasos por unidad de nuestro extrusor, ya que lo tiene en cuenta para calcular la velocidad y de esta manera la velocidad introducida en EXTRUDER_RUNOUT_SPEED será real y nos evitaremos al igual que antes tener que calcular la velocidad.

4.1.6 Ajustes de Termopares con sensor AD595



Si tenemos configurado el sensor de temperatura como -1, es decir, un termopar con un AD595 para convertir la señal, con estos parámetros ajustamos o afinamos la lectura de datos.

#define TEMP_SENSOR_AD595_OFFSET 0.0
#define TEMP_SENSOR_AD595_GAIN 1.0

Con TEMP_SENSOR_AD595_OFFSET configuramos la cantidad extra de grados que tiene que sumar o restar (numero negativo) a la lectura de temperatura para que se acerque a la real. Y en TEMP_SENSOR_AD595_GAIN configuramos la ganancia o coeficiente de corrección. Para más claridad la formula que aplica es la siguiente:

Temperatura actual = (Temperatura medida * TEMP_SENSOR_AD595_GAIN) + TEMP_SENSOR_AD595_OFFSET

Para ajustar estos parámetros sería necesario tomar la temperatura real con un termómetro fiable y compararlo con la lectura del AD595.

4.1.7 control de velocidad del ventilador de la placa controladora

Si tenemos un ventilador para enfriar los drivers de los motores, con estos parámetros podemos configurarlo. Se encenderá el ventilador en cuanto se encienda un driver y además podemos fijar el tiempo que permanecerán encendidos después de que se apague el último de los drivers. También podemos fijar la velocidad si tenemos control por PWM.

#define CONTROLLERFAN_PIN -1 //Pin used for the fan to cool controller (-1 to disable)

En que pin está conectado el ventilador, por defecto esta en -1 que es deshabilitado. Para habilitarlo bastara con cambiar el -1 por el numero correspondiente al pin de la placa controladora donde este conectado el ventilador.

#define CONTROLLERFAN_SECS 60 //How many seconds, after all motors were disabled, the fan should run

Tiempo extra que permanecerá encendido el ventilador una vez se ha apagado el último de los drivers de los motores. Por defecto esta en 60 segundos.

#define CONTROLLERFAN_SPEED 255 // == full speed

Velocidad a la que funcionara el ventilador. Por defecto esta en 255 que es la máxima velocidad.

4.1.8 Arranque del ventilador a máxima potencia

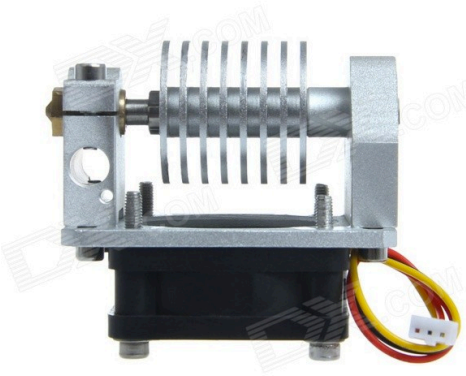
En los controles de velocidad de los motores por PWM, para que sean más fiable, es muy recomendable arrancar primero el ventilador a máxima velocidad y después controlarla por PWM.

Este parámetro configura el ventilador principal, que suele ser el que ya tiene una salida digital definida (FAN_PIN), en el caso de la RAMPS1.4 seria la D09. Este parámetro no funciona con el control PWM por software que usa la placa Sanguinololu.

//#define FAN_KICKSTART_TIME 100

Tiempo en ms que el ventilador principal funcionara a máxima velocidad, una vez transcurrido la velocidad la ajustara la placa según la tengamos configurada. Por defecto esta deshabilitada.

4.1.9 Configuración de los ventiladores de los Hotend



Podemos configurar a que pines hemos conectado los ventiladores que enfrían los Hotend, a que temperatura se encenderán y a la velocidad que giraran.

#define EXTRUDER_0_AUTO_FAN_PIN -1
#define EXTRUDER_1_AUTO_FAN_PIN -1
#define EXTRUDER_2_AUTO_FAN_PIN -1

A que pines está conectado el ventilador de cada uno de los extrusores, si esta a -1 no están habilitados. Para habilitarlos solo será necesario sustituir el -1 por el numero de pin al que esta conectado el ventilador. Si solo tenemos un extrusor hay que configurar el extrusor 0. Se pueden configurar varios extrusores al mismo pin. Por defecto están los 4 deshabilitados.

#define EXTRUDER_AUTO_FAN_TEMPERATURE 50

Cuando la temperatura de cualquiera de los Hotends sea superior al valor fijado se encenderá el ventilador correspondiente, y se apagara cuando sea inferior. Por defecto está fijada en 50 grados.

```
#define EXTRUDER_AUTO_FAN_SPEED   255  // == full speed
```

Velocidad que giraran los ventiladores una vez se enciendan. Por defecto esta en 255 que es la máxima velocidad.

4.2 Ajustes mecánicos

4.2.1 Uso de los EndStop solo para Homing

```
#define ENDSTOPS_ONLY_FOR_HOMING // If defined the endstops will only be used for homing
```

Solo se usaran los EndStop para el Homing. Una vez se ha hecho el Homing, la placa sabrá exactamente en qué posición esta el extrusor y ya no usara mas los EndStop aunque se pulsen mientras imprime la maquina. Por defecto esta activado. Si lo deshabilitamos y por ejemplo en algún momento mientras estamos imprimiendo pasamos por el X0 o Y0 se pararía la impresión.

4.2.2 Configuración de la posición de los EndStop en modo manual

Si en el archivo Configuration.h hemos habilitado la opción de MANUAL_HOME_POSITIONS, desde este bloque se configuran los límites. Es una ampliación de definiciones, es recomendable no tocar nada.

```
//// AUTOSET LOCATIONS OF LIMIT SWITCHES
//// Added by ZetaPhoenix 09-15-2012
#ifdef MANUAL_HOME_POSITIONS  // Use manual limit switch locations
 #define X_HOME_POS MANUAL_X_HOME_POS
 #define Y_HOME_POS MANUAL_Y_HOME_POS
 #define Z_HOME_POS MANUAL_Z_HOME_POS
#else //Set min/max homing switch positions based upon homing direction and min/max travel limits
 //X axis
 #if X_HOME_DIR == -1
  #ifdef BED_CENTER_AT_0_0
   #define X_HOME_POS X_MAX_LENGTH * -0.5
  #else
   #define X_HOME_POS X_MIN_POS
  #endif //BED_CENTER_AT_0_0
 #else
  #ifdef BED_CENTER_AT_0_0
   #define X_HOME_POS X_MAX_LENGTH * 0.5
```

```
  #else
    #define X_HOME_POS X_MAX_POS
  #endif //BED_CENTER_AT_0_0
#endif //X_HOME_DIR == -1

//Y axis
#if Y_HOME_DIR == -1
  #ifdef BED_CENTER_AT_0_0
    #define Y_HOME_POS Y_MAX_LENGTH * -0.5
  #else
    #define Y_HOME_POS Y_MIN_POS
  #endif //BED_CENTER_AT_0_0
#else
  #ifdef BED_CENTER_AT_0_0
    #define Y_HOME_POS Y_MAX_LENGTH * 0.5
  #else
    #define Y_HOME_POS Y_MAX_POS
  #endif //BED_CENTER_AT_0_0
#endif //Y_HOME_DIR == -1

// Z axis
#if Z_HOME_DIR == -1 //BED_CENTER_AT_0_0 not used
  #define Z_HOME_POS Z_MIN_POS
#else
  #define Z_HOME_POS Z_MAX_POS
 #endif //Z_HOME_DIR == -1
#endif //End auto min/max positions
//END AUTOSET LOCATIONS OF LIMIT SWITCHES -ZP
```

4.2.3 Habilitar eje Z en el último momento.

```
//#define Z_LATE_ENABLE // Enable Z the last moment. Needed if your Z driver overheats.
```

Si el driver de nuestro motor Z se calienta en exceso, podemos configurarlo para que en los momentos en los que no se mueva el eje Z durante la impresión se desconectara el driver y el motor. Por defecto esta deshabilitado.

4.2.4 Usar 2 drivers independientes para el mismo eje Z o Y

Normalmente usamos un solo driver para mover los 2 motores del eje Z, pero dependiendo de la corriente que tengamos configurada puede calentarse en exceso. Podemos configurar la placa para usar 2 drivers independientes uno para cada motor. En algunas placas como por ejemplo, la RAMPS que soporta hasta 2 extrusores, a menudo el segundo extrusor no se usa, y podríamos usar ese driver para controlar el otro motor Z.

```
//#define Z_DUAL_STEPPER_DRIVERS

#ifdef Z_DUAL_STEPPER_DRIVERS
  #undef EXTRUDERS
  #define EXTRUDERS 1
#endif
```

Usaremos 2 drivers independientes para controlar los motores del eje Z. Por defecto esta deshabilitado. La parte en rojo es una ampliación de la definición y fija (y limita) el numero de extrusores en 1 ya que en las placas de 5 drivers (como la RAMPS) no es compatible tener más de un extrusor con el uso de drivers dobles para el eje Z.

Es importante tener en cuenta que será **necesario definir los pines Z2** en el archivo pins.h correspondiente a nuestra placa, o nos dará un error de compilación el programa. Hay que incluir el siguiente código:

```
#define Z2_STEP_PIN 29 // Mismo numero de pin que E1_STEP_PIN
#define Z2_DIR_PIN 39 // Mismo numero de pin que E1_DIR_PIN
#define Z2_ENABLE_PIN 28 // Mismo numero de pin que E1_ENABLE_PIN
```

Donde los números de pin deberán de coincidir con los configurados en nuestro segundo extrusor, es decir, deberían de ser los mismos que se han fijado para el extrusor E1.

```
//#define Y_DUAL_STEPPER_DRIVERS

// Define if the two Y drives need to rotate in opposite directions
#define INVERT_Y2_VS_Y_DIR true

#ifdef Y_DUAL_STEPPER_DRIVERS
  #undef EXTRUDERS
  #define EXTRUDERS 1
#endif
```

Misma función que la anterior pero por si necesitamos 2 drivers independientes para el eje Y. La única diferencia es INVERT_Y2_VS_Y_DIR que configura si el 2º motor Y gira en el sentido contrario al 1º, girando los 2 en direcciones opuestas. Por defecto esta deshabilitado y la dirección de giro está configurada como giro en direcciones opuestas (true).

Es importante tener en cuenta que será **necesario definir los pines Y2** en el archivo pins.h correspondiente a nuestra placa, o nos dará un error de compilación el programa. Hay que incluir el siguiente código:

```
#define Y2_STEP_PIN 29 // Mismo numero de pin que E1_STEP_PIN
```

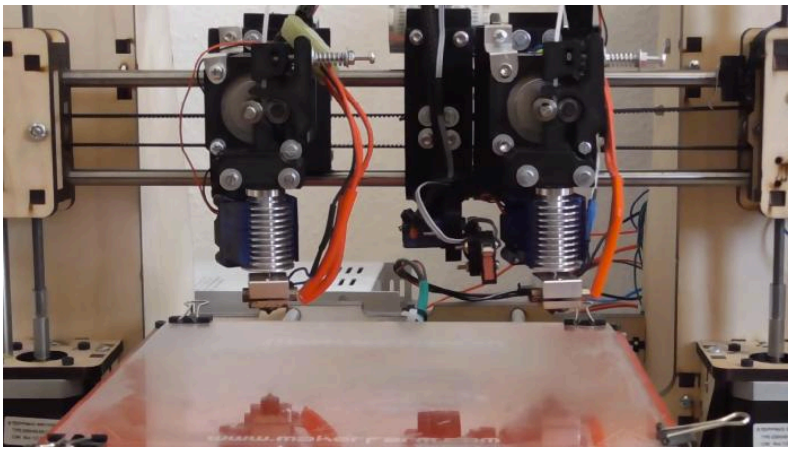
#define Y2_DIR_PIN 39 // Mismo numero de pin que E1_DIR_PIN
#define Y2_ENABLE_PIN 28 // Mismo numero de pin que E1_ENABLE_PIN

Donde los números de pin deberán de coincidir con los configurados en nuestro segundo extrusor, es decir, deberían de ser los mismos que se han fijado para el extrusor E1.

#if defined (Z_DUAL_STEPPER_DRIVERS) && defined (Y_DUAL_STEPPER_DRIVERS)
#error "You cannot have dual drivers for both Y and Z"
#endif

Ampliación de definiciones, no podemos tener activos a la vez los drivers dobles para el eje Z y el eje Y.

4.2.5 Carro dual en el eje X



Podemos configurar si tenemos un carro dual para el eje X. En algunos modelos con 2 extrusores podemos "aparcar" el extrusor inactivo mientras imprimimos con el otro y así evitar contaminaciones de la pieza con otro filamento. Además reduce el peso del carro X permitiendo impresiones más rápidas.

//#define DUAL_X_CARRIAGE

Activaremos el uso dual del carro X. Por defecto esta deshabilitado.

#ifdef DUAL_X_CARRIAGE

El resto de configuraciones solo tendrán valor si hemos activado previamente el uso de carro dual en X. El primer carro X se aparca en la posición del EndStop mínimo (home) y el segundo carro se aparcara en la posición del EndStop máximo (final recorrido máximo).

#define X2_MIN_POS 80 // set minimum to ensure second x-carriage doesn't hit the parked first X-carriage

Distancia mínima en la que podrá estar el 2º carro sin que golpee con el 1º carro que estará aparcado, seria equivalente al offset entre extrusores. Si hemos definido el uso del carro dual será imprescindible configurar este parámetro.

#define X2_MAX_POS 353 // set maximum to the distance between toolheads when both heads are homed

Fija la distancia máxima que habrá entre las puntas de los 2 Hotends cuando ambos estén "aparcados". Necesario configurarlo.

#define X2_HOME_DIR 1 // the second X-carriage always homes to the maximum EndStop position

Dirección en la que el 2º carro hará el Homing. Por defecto esta en 1 que será hacia el lado del máximo, y lo normal es dejarlo así.

#define X2_HOME_POS X2_MAX_POS // default home position is the maximum carriage position

La posición por defecto del Homing del 2º carro será la posición máxima del eje. Es recomendable dejarlo como esta por defecto.

#define X2_ENABLE_PIN 29
#define X2_STEP_PIN 25
#define X2_DIR_PIN 23

Configuramos los pines del driver del motor del 2º extrusor. Se hace en este apartado para evitar tener que buscar en la librería pins.h el apartado correspondiente a nuestra placa. Sera necesario configurarlo.

#define DEFAULT_DUAL_X_CARRIAGE_MODE 0

Podemos elegir entre varios modos de funcionamiento cuando inicia la maquina, que luego podremos cambiar con el Gcode M605 S<modo>.

Modo 0: Control total. El software Slicer tiene el control total sobre ambos carros y puede conseguir resultados óptimos siempre que soporte el uso de carro dual (M605 S0).

Modo 1: Auto-aparcado. El firmware (Marlin) puede aparcar automáticamente los carros X y cambiar así de extrusor. No requiere soporte por parte del software Slicer (M605 S1).

Modo 2: Duplicado. El 2º carro copia todos los movimientos del primer carro, realizando así la impresión simultanea de 2 piezas iguales. El x-offset del segundo carro y el ófset de la temperatura pueden fijarse con el Gcode M605 S2 [Xnnn] [Rmmm].

Por defecto esta en el modo 0, asegúrate que el software de Slicer que usas soporta el uso de carro dual, sino usa el modo 1.

#define TOOLCHANGE_PARK_ZLIFT 0.2 // the distance to raise Z axis when parking an extruder

Distancia que elevaremos el eje Z cuando aparquemos alguno de los 2 carros. Solo en Modo 1 Auto-aparcado.

#define TOOLCHANGE_UNPARK_ZLIFT 1 // the distance to raise Z axis when unparking an extruder

Distancia que elevaremos el eje Z cuando "des-aparquemos" alguno de los 2 carros. Solo en Modo 1 Auto-aparcado.

#define DEFAULT_DUPLICATION_X_OFFSET 100

Distancia X por defecto entre los 2 Hotend cuando estamos en Modo 2 duplicado.

#endif //DUAL_X_CARRIAGE

4.2.6 Distancia para 2º Homing ultra lento

#define X_HOME_RETRACT_MM 5
#define Y_HOME_RETRACT_MM 5
#define Z_HOME_RETRACT_MM 2

Cuando hacemos un Homing, los ejes se desplazan a la velocidad configurada en Configuration.h en el parámetro HOMING_FEEDRATE. Cuando pulsan el EndStop, se retraen una distancia y vuelven a hacer el Homing más lento. Desde estos parámetros fijamos la distancia que se retraerá una vez pulsado por primera vez el EndStop.

4.2.7 Homing rápido

//#define QUICK_HOME //if this is defined, if both x and y are to be homed, a diagonal move will be performed initially.

Si se activa, el Homing se hará en modo rápido. Al hacer el Homing se realizara un movimiento en diagonal, en vez de hacer primero el Homing X y después el Homing Y. Por defecto esta deshabilitado.

4.2.8 Modo relativo de los ejes

#define AXIS_RELATIVE_MODES {false, false, false, false}

Configura el movimiento de los ejes en modo relativo o en modo absoluto por defecto. Para más información ver los Gcodes G90 y G91. Por defecto están todos los ejes en modo absoluto.

4.2.9 Configuración frecuencia máxima de pasos

#ifdef CONFIG_STEPPERS_TOSHIBA
#define MAX_STEP_FREQUENCY 10000 // Max step frequency for Toshiba Stepper Controllers
#else
#define MAX_STEP_FREQUENCY 40000 // Max step frequency for Ultimaker (5000 pps / half step)
#endif

Los drivers marca Toshiba, no admiten frecuencias de pasos superiores a 10.000 Hz, pero los utilizados normalmente en las impresoras reprap (pololus) admiten hasta 40.000 Hz. Desde esta ampliación de definición configuramos la frecuencia máxima de funcionamiento. No tocar.

4.2.10 Lógica del pin STEP en los drivers de los motores

#define INVERT_X_STEP_PIN false
#define INVERT_Y_STEP_PIN false
#define INVERT_Z_STEP_PIN false
#define INVERT_E_STEP_PIN false

Por defecto, los drivers usados normalmente (pololus) requieren una señal lógica 1 en el pin de pasos (step), pero si tenemos otro tipo de drivers que funcionen con lógica negativa y precisen un 0 lógico para la entrada de pasos, podemos configurarlo desde aquí. Por defecto están los 4 ejes en lógica positiva (un 1 lógico por cada paso), si usamos los drivers habituales no hay que tocar nada.

4.2.11 Tiempo desconexión Drivers

#define DEFAULT_STEPPER_DEACTIVE_TIME 60

Tiempo en segundos que tardaran en desconectarse los drivers de los motores una vez que la impresora entre en estado de reposo. Por defecto está configurado a 60 segundos.

4.2.12 Velocidades mínimas

#define DEFAULT_MINIMUMFEEDRATE 0.0 // minimum feedrate
#define DEFAULT_MINTRAVELFEEDRATE 0.0

Configuraremos las velocidades mínimas a las que se desplazaran los ejes, en DEFAULT_MINIMUMFEEDRATE configuraremos la velocidad mínima mientras se imprime, y en DEFAULT_MINTRAVELFEEDRATE es la velocidad mínima en desplazamientos (sin imprimir). Estos parámetros también se pueden configurar con el Gcode M205. Hay que tener precaución porque esta velocidad se aplica a todos los ejes. Por defecto esta en 0, es mejor dejarlo como esta.

4.2.13 Velocidades de desplazamiento de los ejes en modo manual

#ifdef ULTIPANEL
*#define MANUAL_FEEDRATE {50*60, 50*60, 4*60, 60} // speeds for manual moves (mm/min)*
#endif

Configuraremos la velocidad a la que se desplazaran los ejes cuando se realicen movimientos manuales desde el LCD con encoder. Es imprescindible tener un LCD con encoder rotativo. Las velocidades fijadas por defecto pueden ser adecuadas, pero sería recomendable bajar la velocidad del eje Z a 2, dejando los valores en MANUAL_FEEDRATE {50*60, 50*60, 2*60, 60}.

#ifdef ULTIPANEL
#define ULTIPANEL_FEEDMULTIPLY
#endif

Cuando tenemos un LCD con encoder, si lo giramos mientras imprime podemos aumentar la velocidad de impresión. Desde este parámetro podemos desactivar esta función, bastara con comentar la línea #define ULTIPANEL_FEEDMULTIPLY y ya no podremos modificar la velocidad desde el encoder rotativo. Por defecto esta activado.

4.2.14 Movimientos según estado del buffer

El buffer es la memoria intermedia donde el micro controlador almacena los movimientos que se van a producir, de esta manera si hay algún micro corte de datos, aun tendrá movimientos por hacer.

#define DEFAULT_MINSEGMENTTIME 20000

Tiempo mínimo en microsegundos que tardara en realizarse los movimientos cuando el buffer esta vacío. Por defecto está configurado en 20000 microsegundos. Este retraso ayudara a que se vaya llenando mientras el buffer para realizar el siguiente movimiento. Es recomendable dejarlo como esta. Se puede modificar mas tarde con el Gcode M205

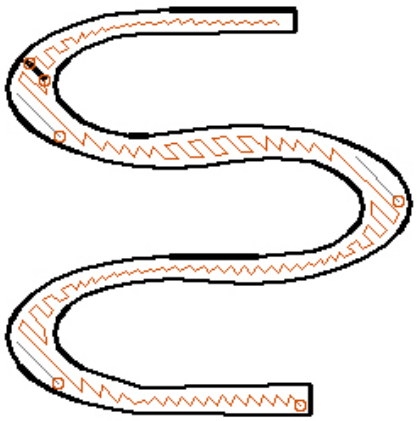
#define SLOWDOWN

Si está habilitado, se Reducirá la velocidad de los movimientos cuando de búfer comienza a vaciarse, en lugar de esperar en una esquina a que se recargue el búfer. Por defecto está habilitado y es recomendable dejarlo así.

4.2.15 Frecuencia limite

//#define XY_FREQUENCY_LIMIT 15

Este parámetro ajusta la frecuencia de cambio de dirección cuando se hace un relleno entre paredes estrechas y el extrusor tiene que cambiar rápidamente de dirección, para más información visita el blog de nophead. Por defecto esta deshabilitado.

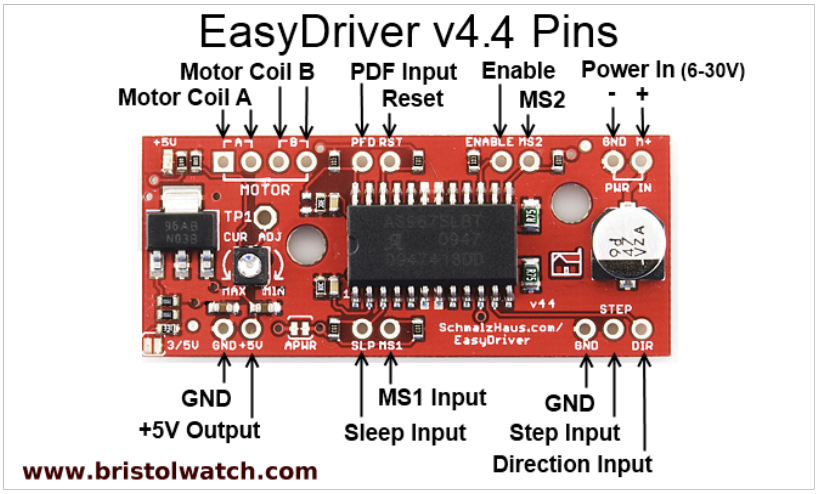


4.2.16 Velocidad mínima de unión del planificador

#define MINIMUM_PLANNER_SPEED 0.05// (mm/sec)

El planificador es la parte de Marlin que se encarga de calcular los movimientos y las aceleraciones. Con este parámetro configuramos la velocidad mínima al final del buffer y cuando se para. Este parámetro no debería de ser mucho mayor que 0, y solo tiene que cambiarse si se observa un comportamiento no deseado cuando se imprime a velocidades muy bajas. Por defecto 0,05 mm/sec.

4.2.17 Configuración de drivers con entradas MS1 y MS2 para micro pasos



Si tenemos algún driver de motor con entradas MS1 y MS2 para configurar los micro pasos y están conectadas al micro controlador, desde estos parámetros podemos configurar la lógica de como deberán de estar esas entradas para seleccionar el modo de micro pasos. Más datos en http://www.bristolwatch.com/arduino/easy_driver.htm. También se pueden configurar con los Gcodes M350 y M351. Solo están definidos los pines de conexión en pins.h para la placa 5DPrint D8 y para la Rambo.

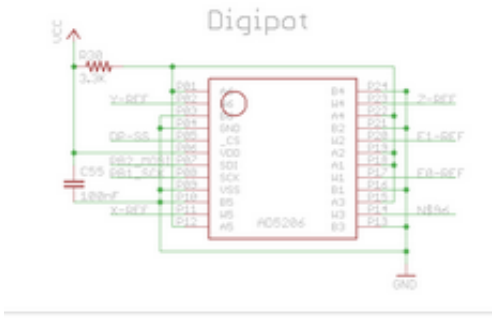
// MS1 MS2 Stepper Driver Microstepping mode table
#define MICROSTEP1 LOW,LOW
#define MICROSTEP2 HIGH,LOW
#define MICROSTEP4 LOW,HIGH
#define MICROSTEP8 HIGH,HIGH
#define MICROSTEP16 HIGH,HIGH

Configuración de la lógica de las entradas MS1 y MS2. Por defecto suele estar bien configurado, pero compruébalo con la documentación de tus drivers. Si el driver soporta 1/16 de microesteping, no suele soportar entonces 1/8.

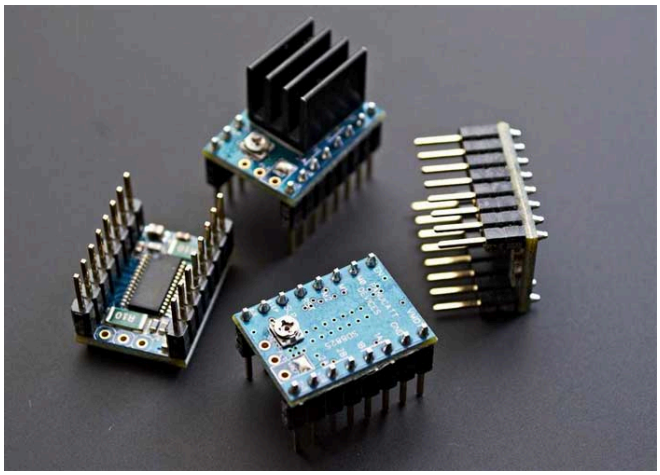
#define MICROSTEP_MODES {16,16,16,16,16} // [1,2,4,8,16]

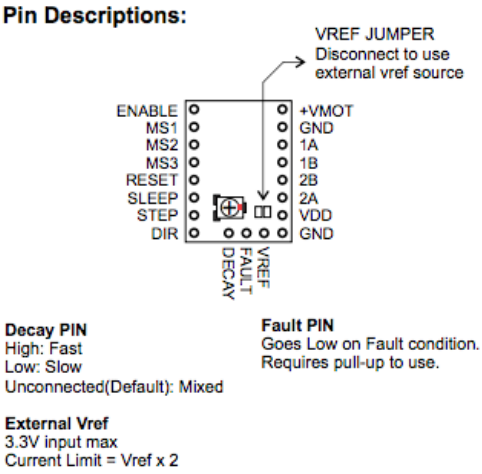
Configuración de micro pasos por defecto de los drivers, valores para los ejes X, Y, Z, extrusor 0 y extrusor 1. Solo se pueden configurar de este modo 5 drivers. Los valores admitidos son 1, 2, 4, 8,16 que en un motor de 200 pasos corresponden a 200, 400, 800, 1600, 3200 micro pasos por vuelta. Por defecto están configurados todos a 16 (3200 pasos).

4.2.18 Configuración de DIGIPOT



Algunas placas tiene la posibilidad de modificar la corriente de configuración de los drivers de los motores electrónicamente. En lugar de usar un potenciómetro en el driver que regulamos manualmente, la placa de control dispone de un potenciómetro electrónico que podemos configurar. Es necesario que la entrada correspondiente Vref de los drivers este conectada a la placa controladora. Actualmente solo esta soportada por la placa Rambo y por la placa Azteeg X3 Pro (en este caso solo por comunicación I2C). El driver que soporta el uso de la Vref externa para el control de la corriente es el SD8825.





#define DIGIPOT_MOTOR_CURRENT {135,135,135,135,135} // Values 0-255 (RAMBO 135 = ~0.75A, 185 = ~1A)

Configuración por defecto de la corriente de los drivers, teniendo en cuenta que 255 será el máximo. Por ejemplo, para la Rambo el valor 135 equivale a 0,75 Amperios y el 185 equivaldría a 1 Amperio. Se pueden cambiar posteriormente con el Gcode M907 y con el M908. Por defecto esta en 135 que equivale a 0,75 amperios en una Rambo.

//#define DIGIPOT_I2C

Si lo descomentamos habilitaremos el uso del potenciómetro digital controlado por I2C que es el que dispone la placa Azteeg X3 Pro para controlar las corrientes de los drivers. Por defecto esta deshabilitado.

#define DIGIPOT_I2C_NUM_CHANNELS 8

Número de canales que tiene el potenciómetro digital, en el caso del que tiene la Azteeg es de 8.

#define DIGIPOT_I2C_MOTOR_CURRENTS {1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0}

Configuración por defecto de la corriente en Amperios de los drivers. En este caso el dato estará en Amperios. Se puede cambiar con el Gcode M907 y con el M908. Por defecto esta en 1 Amperio.

4.3 Ajustes adicionales

4.3.1 Configuración CHDK para tomar fotos

//#define CHDK 4 //Pin for triggering CHDK to take a picture

Configuración del pin usado para tomar fotos con CHDK, mas datos en http://captain-slow.dk/2014/03/09/3d-printing-timelapses/. Se puede disparar la foto con el Gcode M240

#define CHDK_DELAY 50 //How long in ms the pin should stay HIGH before going LOW again

Tiempo en ms que permanecerá en HIGH el pin antes de pasar a LOW de nuevo. Por defecto esta en 50ms.

4.3.2 Configuración avanzada SD

Parámetros de configuración avanzados en la SD.

#define SD_FINISHED_STEPPERRELEASE true //if sd support and the file is finished: disable steppers?

Si tenemos una SD y la impresión ha terminado, se deshabilitaran los drivers y por tanto los motores. Por defecto está configurado en true, es decir, en cuanto termine de imprimir el archivo vía SD, se deshabilitaran los drivers.

#define SD_FINISHED_RELEASECOMMAND "M84 X Y Z E" // You might want to keep the z enabled so your bed stays in place.

Es el comando Gcode que se envía si tenemos habilitado *SD_FINISHED_STEPPERRELEASE true.* Se enviara el Gcode "M84 X Y Z E" que deshabilita los drivers de los motores X, Y, Z y extrusores. Por ejemplo, podríamos modificar el código y eliminar la Z si queremos mantener habilitado el eje Z. Por defecto se deshabilitan todos los ejes.

#define SDCARD_RATHERRECENTFIRST //reverse file order of sd card menu display. Its sorted practically after the file system block order.

Ordena los archivos visualizados en el LCD de forma que los más recientes estén primeros, si la comentamos se ordenaran al revés. Por defecto está configurado para que aparezcan los más recientes primero.

//#define MENU_ADDAUTOSTART

Podemos configurar la impresora para que empiece a imprimir directamente. Si cuando se enciende o introducimos una SD, tiene un archivo con el nombre auto0.g empezara a imprimirlo directamente. si quieres más información mira en https://github.com/ErikZalm/Marlin/wiki/Autostart. Por defecto esta deshabilitada.

Los 2 parámetros siguientes no están a continuación en el código del archivo Configuration_adv.h, están un poco más abajo, pero se han puesto aquí ya que tienen que ver con el uso de la SD.

//#define ABORT_ON_ENDSTOP_HIT_FEATURE_ENABLED

Habilita la posibilidad de que se detenga la impresión desde SD cuando se pulse algún EndStop. Si la hemos habilitado, se puede controlar (activar o desactivar) con el Gcode M540 y desde el LCD ya que aparecerá un menú extra. Por defecto esta deshabilitada.

#define SDCARDDETECTINVERTED

```
#ifdef ULTIPANEL
 #undef SDCARDDETECTINVERTED
#endif
```

Si estamos usando una RAMPS, o alguna otra placa de control que no detecta cuando hay una SD insertada, puedes solucionarlo conectando un pulsador al pin definido como SDCARDCARDDETECT en el archivo pins.h correspondiente a tu placa. Cuando usas un botón para poner a 0 lógico el pin, puedes necesitar invertir el valor lógico de entrada para que lo interprete la placa como un 1 lógico. Comentando este parámetro conseguiremos invertir el valor. Por defecto esta invertido. El código en rojo no hay que tocarlo y deshabilita la inversión si tenemos instalado un LCD con SD.

4.3.3 Muestra una barra de progreso en el LCD

//#define LCD_PROGRESS_BAR

Muestra una barra de progreso con el tiempo que lleva le impresión y el que le falta para terminar. Solo para los LCD controlados con el chip HD447890. Por defecto está desactivada

```
#ifdef LCD_PROGRESS_BAR
 // Amount of time (ms) to show the bar
 #define PROGRESS_BAR_BAR_TIME 2000
```

Tiempo en milisegundos que se mostrará la barra de progreso. Por defecto está en 2000 (2 segundos).

```
 // Amount of time (ms) to show the status message
 #define PROGRESS_BAR_MSG_TIME 3000
```

Tiempo en milisegundos que mostrará el mensaje de estado, que es el que sale siempre en la última línea del LCD. Por defecto está en 3000 (3 segundos).

```
 // Amount of time (ms) to retain the status message (0=forever)
 #define PROGRESS_MSG_EXPIRE   0
```

Cuando termina la impresión, o se pausa, el mensaje en la barra de estado se mostrará el tiempo establecido. Si está en 0 se mostrará continuamente hasta comenzar otra impression. Por defecto está en 0, es decir, se mostará hasta la siguiente impresión.

```
 // Enable this to show messages for MSG_TIME then hide them
 //#define PROGRESS_MSG_ONCE
```

Si activamos está function, se usara la línea donde sale el tiempo que lleva la impresión para mostrar mensajes y por tanto se ocultará el tiempo de la impresión.

4.3.4 Configuración Perro Guardian (Watchdog)

//#define USE_WATCHDOG

Habilitaremos el uso del perro guardián. En electrónica, un perro guardián (en inglés watchdog) es un mecanismo de seguridad que provoca un reset del micro controlador en caso de que éste se haya bloqueado. Básicamente es un temporizador que se pone a cero periódicamente, si se atasca el código en algún punto por algún error, no se pondrá a cero el

temporizador y se reiniciara el micro controlador y se apagara todo. Para más información Aquí. Por defecto esta deshabilitado, sería recomendable activarlo.

```
#ifdef USE_WATCHDOG
//#define WATCHDOG_RESET_MANUAL
#endif
```

Si tenemos un reinicio producido por el perro guardián en un Arduino Mega2560, el dispositivo se puede bloquear en vez de reiniciarse, en este caso es recomendable usar la opción de WATCHDOG_RESET_MANUAL que evitara usar el reset por hardware. El sistema apagara todo y saldrá un mensaje en el LCD para que apaguemos manualmente la impresora. **Este tipo de control no es seguro**, ya que precisa que las interrupciones estén deshabilitadas y el mico controlador podría colgarse en el código de una interrupción cuando se deshabiliten las interrupciones.

4.3.5 BabyStepping

BabyStepping nos permite controlar los motores de los ejes con pequeños movimientos en tiempo real, es decir, mientras se está imprimiendo, es independiente del proceso de impresión y no respeta los EndStop. Por supuesto es necesario un LCD con encoder rotativo. Se añade un nuevo menú al LCD de BabyStepping desde donde podremos seleccionar el eje a mover. Se puede utilizar por ejemplo para ajustar el eje Z mientras estamos imprimiendo.

```
//#define BABYSTEPPING
```

Habilita el control BabyStepping. Por defecto esta deshabilitado. El código siguiente solo se tiene en cuenta si está habilitada esta opción

```
#ifdef BABYSTEPPING
  #define BABYSTEP_XY  //not only z, but also XY in the menu. more clutter, more functions
```

Habilita el control BabyStepping para los ejes X e Y, y no solo para el eje Z. Por defecto está habilitado.

```
  #define BABYSTEP_INVERT_Z false  //true for inverse movements in Z
```

Si esta a *true*, se invertirán los movimientos del eje Z. Por defecto esta en *false*, es decir, sin invertir.

```
#define BABYSTEP_Z_MULTIPLICATOR 2 //faster z movements
```

Configuramos un parámetro multiplicador para que las modificaciones de altura en el eje Z sean más rápidas. Por defecto está en 2.

```
#ifdef COREXY
  #error BABYSTEPPING not implemented for COREXY yet.
#endif

#ifdef DELTA
  #ifdef BABYSTEP_XY
    #error BABYSTEPPING only implemented for Z axis on deltabots.
  #endif
#endif
#endif
```

Ampliaciones de definiciones en caso de tener habilitado BabyStepping.

4.3.6 Modo Avanzado de extrusión

Al extruir filamento, realmente lo que hacemos es aumentar la presión en el interior del Hotend que provoca que salga plástico por la punta que es la única salida que tiene. Pero las aceleraciones que aplicamos al filamento, no tienen una repercusión directa en la presión interior del filamento, esto es debido a que la presurización (fase de aceleración), y la despresurización (fase de desaceleración), no son simétricas en efecto. Esto provoca que al inicio de una aceleración tengamos un retardo en la salida del plástico y por tanto una escasez de material y de manera contraria en la desaceleración, durante un pequeño instante sigue saliendo plástico por lo que tendremos un exceso de material. Para corregir este efecto tenemos el modo avanzado de extrusión, que intenta solventarlo con el factor K. Ajustar este valor será cuestión de pruebas, ya que dependerá de la geometría de nuestro Hotend, pero como referencia tenemos valores aproximados de 0,005 para una replicator hasta 0,007 para una Thing-o-Matic o Cupcake. Más información en http://www.makerbot.com.

```
//#define ADVANCE
```

Habilita el modo avanzado. Por defecto esta deshabilitado.

```
#ifdef ADVANCE
 #define EXTRUDER_ADVANCE_K .0
 #define D_FILAMENT 2.85
 #define STEPS_MM_E 836
 #define EXTRUTION_AREA (0.25 * D_FILAMENT * D_FILAMENT * 3.14159)
 #define STEPS_PER_CUBIC_MM_E (axis_steps_per_unit[E_AXIS]/ EXTRUTION_AREA)
#endif // ADVANCE
```

Tendremos que introducir ciertos parámetros para ajustar nuestra extrusión:

- *EXTRUDER_ADVANCE_K* será el factor de corrección que aplicamos, habrá que ir probando con distintos valores en una pieza conocida para determinar que factor K es el más adecuado para nuestro Hotend. Por defecto esta en 0.
- *D_FILAMENT* es el diámetro real de nuestro filamento no el teórico (3 ó 1,75) sino el real para que el funcionamiento del sistema sea el correcto. Habrá que medirlo con un pie de rey o micrómetro. Por defecto está en 2,85 que sería el diámetro real aproximado del filamento de 3 mm.
- *STEPS_MM_E* pasos por milímetro de nuestro extrusor.
- No tocar el código en rojo, se emplea para calcular los factores de corrección.

4.3.7 Interpretación de arcos

```
#define MM_PER_ARC_SEGMENT 1
#define N_ARC_CORRECTION 25
```

Marlin no hace los arcos o círculos siguiendo una línea curva, en su lugar hace pequeños segmentos, de esta manera reduce la carga del micro controlador. Con estos parámetros configuramos como serán esos segmentos, en *MM_PER_ARC_SEGMENT* fijamos la longitud de los segmentos, y en *N_ARC_CORRECTION* la corrección en el número de segmentos que tendremos. Por defecto es mejor dejar estos valores como están, pero si tienes problemas con agujeros pequeños, puedes experimentar para ver el resultado que te da.

4.3.8 Movimiento mínimo descartable

```
const unsigned int dropsegments=5; //everything with less than this number of steps will be ignored as move and joined with the next movement
```

Cualquier movimiento que sea un número de pasos menor que el indicado, será ignorado individualmente y se añadirá y realizara en el siguiente movimiento. Por defecto esta en 5 pasos, es recomendable dejarlo como esta.

4.3.9 Ampliación de definiciones de las señales de control de las FA

```
// Power Signal Control Definitions
// By default use ATX definition
#ifndef POWER_SUPPLY
  #define POWER_SUPPLY 1
#endif
// 1 = ATX
#if (POWER_SUPPLY == 1)
  #define PS_ON_AWAKE  LOW
  #define PS_ON_ASLEEP HIGH
#endif
// 2 = X-Box 360 203W
#if (POWER_SUPPLY == 2)
  #define PS_ON_AWAKE  HIGH
  #define PS_ON_ASLEEP LOW
#endif
```

Ampliación de las definiciones según el tipo de Fuente de alimentación seleccionada. No tocar esta parte.

4.3.10 Hotend 0 y 1 se controlan en paralelo

```
//#define HEATERS_PARALLEL
```

Función que controla el uso de 2 extrusores (el 0 y el 1) en paralelo. Solo tiene en Cuenta el sensor de temperatura del extrusor 0 y enciende y apaga las salidas de los 2 Hotend a la vez. Por defecto esta deshabilitado.

4.4 Buffers

Configuración de los buffers que usa el micro controlador. Desde estos parámetros configuramos el tamaño de los buffer, si no sabemos muy bien que estamos haciendo es mejor dejarlos como están.

```
// THE BLOCK_BUFFER_SIZE NEEDS TO BE A POWER OF 2, i.g. 8,16,32 because shifts and ors are used to do the ring-buffering.
#if defined SDSUPPORT
  #define BLOCK_BUFFER_SIZE 16   // SD,LCD,Buttons take more memory, block buffer needs to be smaller
#else
  #define BLOCK_BUFFER_SIZE 16 // maximize block buffer
#endif
```

//The ASCII buffer for receiving from the serial:
#define MAX_CMD_SIZE 96
#define BUFSIZE 4

4.5 Configuración de la retracción con Gcode G10 y G11

Podemos habilitar y configurar la retracción con los Gcodes G10 y G11. Permite que Marlin controle todos los parámetros de la retracción y no el software de Slicer, se usaran también en caso de control de la impresión desde el LCD. Normalmente los software de Slicer usan el comando G1 con movimientos negativos del motor del extrusor para realizar la retracción, pero algunos admiten el uso de los comandos G10 y G11.

// #define FWRETRACT //ONLY PARTIALLY TESTED

Habilita el uso de la retracción controlada por Marlin. Una vez habilitado podemos configurar además que cuando el software de Slicer envíe G1 al extrusor para hacer la retracción, Marlin tome el control descartando los valores del Slicer y usando los suyos propios, para habilitar la auto retracción usaremos el Gcode M209.

Fijamos los parámetros por defecto de configuración de la retracción, además podremos configurarlos posteriormente con los Gcodes M207 y M208. No se tendrán en cuenta si no tenemos habilitado *FWRETRACT*. La configuración de la retracción es una de las cosas más delicadas de la impresión, una mala configuración puede ocasionar múltiples atascos del Hotend, así que infórmate bien antes de cambiar los parámetros.

Veamos que hace cada parámetro:

#ifdef FWRETRACT
#define MIN_RETRACT 0.1 //minimum extruded mm to accept a automatic gcode retraction attempt

Longitud mínima en mm de retracción para hacer una retracción automática. Cuando tenemos habilitada la auto retracción (Gcode M209) y Marlin sustituye los valores generados por el Slicer G1 de retracción por los suyos propios, si el movimiento G1 es menor del valor aquí indicado no se realizara la auto retracción. Por defecto esta en 0,1 mm que es un valor adecuado.

#define RETRACT_LENGTH 3 //default retract length (positive mm)

Longitud en mm que se retraerá el filamento al hacer un Gcode G10. Por defecto esta en 3mm que suele ser adecuada para todo tipo de extrusores.

#define RETRACT_LENGTH_SWAP 13 //default swap retract length (positive mm), for extruder change

Cuando tenemos 2 extrusores y realizamos el cambio en la impresión de uno a otro, el filamento del extrusor que no se vaya a usar se retraerá esta longitud en mm, esto evitara goteos no deseados del extrusor que no esté funcionando. Por defecto esta en 13mm pero dependerá de nuestro Hotend, aunque este valor debería de ser mayor que *RETRACT_LENGTH.* Si solo tenemos un extrusor no se tendrá en cuenta.

#define RETRACT_FEEDRATE 45 //default feedrate for retracting (mm/s)

Velocidad en mm/s a la que se retraerá el filamento. Por defecto esta en 45mm/s que es un valor adecuado.

#define RETRACT_ZLIFT 0 //default retract Z-lift

Distancia que se levantara el eje Z para que al moverse el extrusor sobre la parte ya impresa de la pieza no quede un rastro. Normalmente la retracción se produce cuando el extrusor tiene que cambiar de posición sobre la pieza sin expulsar filamento, si tenemos configurado este parámetro obtendremos impresiones más limpias al evitar el roce del poco filamento que tengamos en la punta del Hotend con la pieza ya impresa. Por defecto esta a 0, es decir, no se moverá, un buen valor seria sobre 0,25 – 0,4 mm.

#define RETRACT_RECOVER_LENGTH 0 //default additional recover length (mm, added to retract length when recovering)

Longitud adicional en mm que se extruira el filamento cuando vuelva a ponerse en posición de extruir después de haber hecho una retracción. Si tenemos la longitud de retracción en 3 mm (*RETRACT_LENGTH*) y hemos configurado la longitud adicional al extruir (*RETRACT_RECOVER_LENGTH)* en 0,1 (por ejemplo), el extrusor extruira 3,1 mm de filamento antes de seguir con la impresión. Por defecto esta en 0, un buen valor seria sobre 0,05. Si ponemos valores muy altos podemos tener un goteo inicial al comenzar de nuevo a imprimir, y si es muy pequeño podemos encontrarnos que al comenzar le falte un poco de filamento.

#define RETRACT_RECOVER_LENGTH_SWAP 0 //default additional swap recover length (mm, added to retract length when recovering from extruder change)

Tiene la misma función que el parámetro anterior, pero en este caso cuando volvemos a cambiar de extrusor y comenzamos a imprimir, en este caso extruira la distancia retraída antes (*RETRACT_LENGTH_SWAP*) más la distancia extra aquí indicada. Por defecto está en 0.

#define RETRACT_RECOVER_FEEDRATE 8   //default feedrate for recovering from retraction (mm/s)
#endif

Velocidad a la que se extruira el filamento cuando se recupera de una retracción. Por defecto esta en 8 mm/s y es recomendable tener valores bajos para evitar un aumento de presión muy rápido en el interior del Hotend.

4.6 Soporte para el cambio de filamento con Gcode M600

Tenemos la opción de configurar la impresora para que se detenga y así poder cambiar el filamento. Es una función experimental y hay que tener precaución al usarla. Se usa el Gcode M600 y hay que tener display para poder controlarlo.

#ifdef ULTIPANEL
#define FILAMENTCHANGEENABLE

Habilita el uso del Gcode M600. Por defecto está habilitado, pero solo si tenemos un LCD con encoder rotativo, por eso está detrás de *#ifdef ULTIPANEL.*

Con el Gcode M600 se pueden indicar las coordenadas de pausa X, Y, Z, E y la longitud de retracción final, si no se envían se usaran los valores por defecto que se indican a continuación.

#ifdef FILAMENTCHANGEENABLE
#define FILAMENTCHANGE_XPOS 3
#define FILAMENTCHANGE_YPOS 3
#define FILAMENTCHANGE_ZADD 10

Coordenas X e Y donde se desplazara el extrusor, y mm extras que se añadirán al eje Z. Por defecto esta en X=3, Y=3 y el eje Z= Z+10.

```
    #define FILAMENTCHANGE_FIRSTRETRACT -2
    #define FILAMENTCHANGE_FINALRETRACT -100
  #endif
#endif
```

Longitud en mm que se retraerá el filamento, en *FILAMENTCHANGE_FIRSTRETRACT* configuraremos la longitud en mm que se retraerá antes de desplazarse el extrusor a las coordenadas X e Y, y en *FILAMENTCHANGE_FINALRETRACT* configuramos la retracción final para sacar el filamento totalmente del extrusor y poderlo cambiar. Los valores están en mm e irán **en negativo**. Por defecto están en -2 y en -100.

Los pasos que se ejecutaran cuando se use el Gcode M600 son los siguientes:

1. Se parara la impresión.
2. Se levantara el eje Z (*FILAMENTCHANGE_ZADD).*
3. *S*e retraerá el filamento (*FILAMENTCHANGE_FIRSTRETRACT).*
4. Irá a las coordenadas X (*FILAMENTCHANGE_XPOS)* e Y (*FILAMENTCHANGE_YPOS).*
5. Se volverá a retraer el filamento una segunda vez para sacarlo del todo (*FILAMENTCHANGE_FINALRETRACT*).
6. Una vez este hecho el cambio del filamento, lo confirmaremos desde el LCD.
7. Se extruira de nuevo la cantidad retraída de filamento(*FILAMENTCHANGE_FINALRETRACT*)
8. Se desplazara a la posición X e Y donde se paró la impresora.
9. Se desplazara a su posición el eje Z, se extruira la cantidad inicialmente retraída (*FILAMENTCHANGE_FIRSTRETRACT).*
10. *V*olverá a imprimir.

```
#ifdef FILAMENTCHANGEENABLE
  #ifdef EXTRUDER_RUNOUT_PREVENT
    #error EXTRUDER_RUNOUT_PREVENT currently incompatible with FILAMENTCHANGE
  #endif
#endif
```

Comprobación de seguridad. Si hemos habilitado previamente *EXTRUDER_RUNOUT_PREVENT* no son compatibles las 2 opciones a la vez.

4.7 Ampliación de definiciones

En este bloque solo hay ampliaciones de definiciones, es recomendable no tocar nada.

```
#if defined (ENABLE_AUTO_BED_LEVELING) && defined (DELTA)
  #error "Bed Auto Leveling is still not compatible with Delta Kinematics."
#endif

#if EXTRUDERS > 1 && defined TEMP_SENSOR_1_AS_REDUNDANT
  #error "You cannot use TEMP_SENSOR_1_AS_REDUNDANT if EXTRUDERS > 1"
#endif

#if EXTRUDERS > 1 && defined HEATERS_PARALLEL
  #error "You cannot use HEATERS_PARALLEL if EXTRUDERS > 1"
#endif

#if TEMP_SENSOR_0 > 0
  #define THERMISTORHEATER_0 TEMP_SENSOR_0
  #define HEATER_0_USES_THERMISTOR
#endif
#if TEMP_SENSOR_1 > 0
  #define THERMISTORHEATER_1 TEMP_SENSOR_1
  #define HEATER_1_USES_THERMISTOR
#endif
#if TEMP_SENSOR_2 > 0
  #define THERMISTORHEATER_2 TEMP_SENSOR_2
  #define HEATER_2_USES_THERMISTOR
#endif
#if TEMP_SENSOR_BED > 0
  #define THERMISTORBED TEMP_SENSOR_BED
  #define BED_USES_THERMISTOR
#endif
#if TEMP_SENSOR_0 == -1
  #define HEATER_0_USES_AD595
#endif
#if TEMP_SENSOR_1 == -1
  #define HEATER_1_USES_AD595
#endif
#if TEMP_SENSOR_2 == -1
  #define HEATER_2_USES_AD595
#endif
#if TEMP_SENSOR_BED == -1
  #define BED_USES_AD595
#endif
```

```
#if TEMP_SENSOR_0 == -2
  #define HEATER_0_USES_MAX6675
#endif
#if TEMP_SENSOR_0 == 0
  #undef HEATER_0_MINTEMP
  #undef HEATER_0_MAXTEMP
#endif
#if TEMP_SENSOR_1 == 0
  #undef HEATER_1_MINTEMP
  #undef HEATER_1_MAXTEMP
#endif
#if TEMP_SENSOR_2 == 0
  #undef HEATER_2_MINTEMP
  #undef HEATER_2_MAXTEMP
#endif
#if TEMP_SENSOR_BED == 0
  #undef BED_MINTEMP
  #undef BED_MAXTEMP
#endif
```

5 Gcodes

5.1 Introducción

En este apartado ampliamos la información de los comandos Gcode que Marlin utiliza. El objetivo principal es la fabricación aditiva de piezas utilizando procesos de deposición de filamento fundido, para ello el extrusor seguirá los movimientos mediante los Gcodes según la norma NIST RS274NGC G-code standard. Marlin, y por tanto la placa controladora solo entiende Gcodes, es la forma de recibir las ordenes, bien sea por el puerto serie con un ordenador o mediante un archivo .gcode en una tarjeta SD, lo que se envía a la placa son Gcodes. Un archivo .gcode contiene un conjunto de ordenes en formato Gcode que resulta con el movimiento del extrusor y la fabricación de la correspondiente pieza.

Hay varias maneras de generar un Gcode para una impresora, la primera y mas utilizada es usar un programa de Slicer como Slic3r , Skeinforge o Cura. Estos programas necesitan un modelo CAD que cortaran en rodajas y generan el Gcode necesario para imprimir cada capa. Otra opción es usar una librería de nivel inferior como Mecode. Mecode te da un control preciso sobre la trayectoria del extrusor y por tanto son útiles si tienes una impresión compleja que no es adecuada para el programa de Slicer. Y la ultima opción es escribir tu mismo tu propio Gcode, esta puede ser la mejor opción si lo que necesitas es ejecutar unas pocas líneas de prueba durante la calibración de la impresora.

Un trozo de código típico de Gcode enviado a una máquina RepRap podría tener este aspecto:

```
T0
G92 E0
G28
G1 F1500.0
G1 X2.0 Y2.0 F3000.0
X3.0 Y3.0 G1
```

El significado de todos esos símbolos y números (y más) se explica a continuación.

En la pagina de la wiki de reprap se incluyen todos los Gcodes que soportan los diferentes firmwares, pero aquí solo incluimos los comandos y parámetros que soporta Marlin, si quieres mas información visita http://reprap.org/wiki/G-code

5.2 Tipos de comandos Gcode

Vamos a ver los tipos de comandos Gcodes y los parámetros o campos que los acompañan, y que van precedidos de una letra seguidos de una serie de números. Los números están representados por **nnn,** y pueden ser números enteros (128) o decimales (12.42) (con punto no coma) dependiendo del contexto. Por ejemplo, una coordenada X puede ser con números enteros (**X175**) o decimales (**X17.62**), pero seleccionar el numero de extrusor 2.76 no tendría ningún sentido.

Comando	Funcionalidad
Gnnn	Comandos Gcode estándar, como el movimiento a un punto determinado
Mnnn	Comandos RepRap definidos, como encender un ventilador de refrigeración
Tnnn	Seleccionar nnn herramienta. En RepRap, las herramientas son los extrusores
Snnn	Parámetro de comando, como la tensión para enviar a un motor
Pnnn	Parámetro de comando, como un tiempo en milisegundos
Xnnn	Coordenada X, por lo general para moverse a ese punto. Puede ser un número entero o decimal.
Ynnn	Coordenada Y, por lo general para moverse a ese punto. Puede ser un número entero o decimal
Znnn	Coordenada Z, por lo general para moverse a ese punto. Puede ser un número entero o decimal
Fnnn	Avance en mm por minuto. (Velocidad de movimiento del cabezal de impresión)
Rnnn	Parámetro - se utiliza para temperaturas
Ennn	Longitud de extrusión en mm. Es exactamente igual que X, Y, Z, pero para la longitud del filamento a extruir. Puede ser un número entero o decimal
Nnnn	Número de línea. Se utiliza para solicitar la repetición de la transmisión en el caso de errores de comunicación.
* nnn	Suma de comprobación. Se utiliza para comprobar si hay errores de comunicación.

5.3 Comentarios

En los archivos .gcode también se pueden incluir comentarios, comienzan con un punto y coma ";" y terminan al final de la línea:

*N3 T0 * 57; Esto es un comentario*
*N4 G92 E0 * 67*
; Este también
*N5 G28 * 22*

Los comentarios y espacios en blanco serán ignorados por la impresora. Si es posible, es mejor evitarlos o no hacerlos muy largos, ya que son enviados por el puerto serie del equipo host consumiendo ancho de banda, y es la propia placa controladora de la impresora la que tiene que decidir si es un Gcode o no.

5.4 Comandos de chequeo

Ejemplo:

*N123 [... Gcode aquí ...] * 71*

Número de línea, Gcode y la suma de comprobación o checksum.

Marlin comprueba el checksum con un valor calculado localmente en la placa y si son diferentes solicita una retransmisión de la misma línea.

Para calcular la suma de comprobación "cs" o checksum para una comando Gcode "cmd" (incluyendo su número de línea) se realiza mediante una función EXOR con los bytes de la cadena "cmd" hasta y sin incluir el carácter * como sigue:

```
int cs = 0;
for (i = 0; cmd[i] != '*' && cmd[i] != NULL; i ++)
cs = cs ^ cmd[i];
cs &= 0xff; // La programación defensiva ...
```

y el valor se añade como un entero decimal al comando después del carácter *.

El firmware espera que los números de línea aumenten de 1 en 1 en cada línea, y si no sucede se marca como un error.

5.5 Comandos G Buffered

Marlin almacena estos comandos en un búfer internamente para su ejecución. Esto significa que no existe retardo (apreciable) mientras un comando es reconocido y se transmite el siguiente y de esta forma los movimientos de impresión se pueden hacer sin una temporización entre uno y el siguiente. Tan pronto como uno de estos comandos buffer es recibido se reconoce y almacena localmente, si el buffer local esta completo entonces el acuse de recibo se retrasa hasta que quede espacio disponible para el almacenamiento. Es así como se logra el control de flujo del movimiento sin interrupciones.

5.5.1 G0 y G1: Movimiento

Realiza un movimiento lineal desde la posición actual a las coordenadas indicadas. Marlin considera G0 y G1 como el mismo comando, por lo que no haremos distinciones.

Uso

G0 Xnnn Ynnn Znnn Ennn Fnnn
G1 Xnnn Ynnn Znnn Ennn Fnnn

Parámetros

*No todos las parámetros tienen que utilizarse a la vez, pero por lo menos tiene que llevar **uno.***

Xnnn La coordenada X a donde se tendrá que mover.
Ynnn La coordenada Y a donde se tendrá que mover.
Znnn La coordenada Z a donde se tendrá que mover.
Ennn La longitud de filamento en mm que se tendrá que extruir entre el punto donde se ha iniciado el comando y el punto donde termina.
Fnnn Velocidad de avance en mm/min.

Ejemplos

G0 X12 (Mover 12 mm el eje X)
G0 F1500 (Ajuste de la velocidad de avance a 1500 mm/min)
G1 X90.6 Y13.8 E22.4 (Mover 90.6mm el eje X, 13.8mm el eje Y mientras se extruyen 22.4mm de material)

Veamos como funciona con las velocidades de avance:

1. G1 F1500
2. G1 X50 Y25.3 E22.4

En el ejemplo anterior hemos fijado el avance en 1500 mm/minuto en la línea 1, a continuación movemos 50 mm el eje X, 25.3mm el eje Y mientras que se extruyen 22.4mm de filamento entre los dos puntos.

1. G1 F1500
2. G1 X50 Y25.3 E22.4 F3000

Sin embargo, en este ejemplo hemos establecido un avance de 1500 mm/min en la línea 1, y luego hace el movimiento **acelerando** hasta una velocidad final de avance de 3000 mm/minuto. La extrusión se acelerará junto con el movimiento en X y en Y, por lo que todo se mantiene sincronizado.

La especificación RepRap trata la velocidad de avance F como otra variable que interpolara linealmente al igual que hace con X, Y, Z y E. Esto le da un control completo sobre la aceleración y la desaceleración del movimiento del extrusor de una manera tal que asegura que todo se mueve suavemente y el volumen de material se extruye correctamente en todos los puntos.

Para retraer el filamento en el extrusor(para reducir su presión interna, mientras que hace un movimiento en el aire de modo que no gotee) simplemente utiliza G0 o G1 con un valor de E que sea menor que la longitud actualmente extruida, o lo que es lo mismo, un valor negativo.

Ejemplo:

G1 E-4

5.5.2 G2 y G3: Movimiento de arco

Realiza un movimiento en forma de arco.

Condición:

Marlin **NO** tiene en cuenta estos comandos si tiene definido el modo Scara en el archivo Configuration.h. Es decir, funciona para las impresoras tipo Reprap y Delta.

Uso

G2 Xnnn Ynnn Innn Jnnn Ennn (Arco en dirección horaria)
G3 Xnnn Ynnn Innn Jnnn Ennn (Arco en dirección anti horaria)

Parámetros

Xnnn La coordenada X a donde se tendrá que mover.
Ynnn La coordenada Y a donde se tendrá que mover.
Innn El punto en el espacio X desde la posición actual para mantener una distancia constante.
Jnnn El punto en el espacio Y desde la posición actual para mantener una distancia constante.
Ennn La longitud de filamento en mm que se tendrá que extruir entre el punto donde se ha iniciado el comando y el punto donde termina.

Ejemplos

G2 X90.6 Y13.8 I5 J10 E22.4 (Mover en forma de arco en dirección horaria desde el punto actual al punto (X = 90,6, Y = 13,8), con un punto central (X = current_X + 5, Y = current_Y + 10) mientras se extruyen 22.4mm de material entre los 2 puntos)

G3 X90.6 Y13.8 I5 J10 E22.4 (Mover en forma de arco en dirección anti horaria desde el punto actual al punto (X = 90,6, Y = 13,8), con un punto central (X = current_X + 5, Y = current_Y + 10) mientras se extruyen 22.4mm de material entre los 2 puntos)

5.5.3 G28: Mover al origen (Homing)

Este comando hace que la impresora mueva el extrusor a las coordenadas X, Y, Z cero, donde están los EndStop en un proceso conocido como "Homing". Lo hará con al aceleración indicada en Configuration.h en el parámetro *HOMING_FEEDRATE* con el fin de llegar rápido a pulsar el EndStop, pero cuando llega retrocede 5 mm, y vuelve lentamente a pulsar el EndStop. Esto asegura un posicionamiento más preciso.

Uso

G28 X Y Z

Parámetros

Se puede usar con parámetros o sin ellos.

X Homing del eje X.
Y Homing del eje Y.
Z Homing del eje Z.

Ejemplo

G28 (Mueve los ejes X, Y, Z a su posición de origen o Homing)
G28 X Y (Mueve solo los ejes X, Y a su posición de origen o Homing)

5.5.4 G29 a G32: Auto nivelación de cama

Comandos relativos al uso del auto nivelado de cama.

G29 Inicio de auto nivelado

Inicia el proceso de auto nivelado, el numero de "muestras" dependerá de cómo hemos configurado al auto nivelación, si en modo rejilla o en modo 3 puntos. Es preciso haber hecho previamente un Homing del eje X e Y o dará un mensaje de error y se descartara el comando.

Condición:

Marlin solo tiene en cuenta este comando si tiene habilitado la opción ENABLE_AUTO_BED_LEVELING de auto nivelado automático en el archivo Configuration.h, en caso contrario no tendrá efecto.

Uso

G29

Parámetros

No tiene parámetros.

Ejemplo

G28 (Previamente hacemos un Homing)
G29 (Comenzara el proceso de auto nivelado)

G30 Prueba en un solo punto con la sonda Z

Forma más simple de hacer un Homing Z con la sonda en la posición X Y actual. Si la sonda (el EndStop) esta montado sobre un servo, este comando bajara previamente la sonda y una vez tomada la media lo retraerá.

Condición:

Marlin solo tiene en cuenta este comando si tiene **habilitada** la opción ENABLE_AUTO_BED_LEVELING de auto nivelado automático en el archivo Configuration.h, y **no tiene habilitada** la opción Z_PROBE_SLED en el archivo Configuration.h, en caso contrario no tendrá efecto.

Uso

G30

Parámetros

No tiene parámetros.

Ejemplo

G1 X25 Y30 (Si queremos hacer la prueba en un punto concreto (X=25 Y=30), moveremos el extrusor a esas coordenadas)(No es obligatorio el uso de este comando G1)
G30 (Hará la prueba con la sonda Z)

G31 Acopla el carro de la sonda Z

Si tenemos un la sonda Z montada sobre un carro que aparcamos en un lateral y acoplamos para tomar las medidas, con este comando lo acoplaremos al carro del extrusor.

Condición:

Marlin solo tiene en cuenta este comando si tiene **habilitadas** las opciones ENABLE_AUTO_BED_LEVELING de auto nivelado automático y la opción Z_PROBE_SLED en el archivo Configuration.h, en caso contrario no tendrá efecto.

Uso

G31

Parámetros

No tiene parámetros.

Ejemplo

G31 (Acoplamos el carro de la sonda)
G30 (Hará la prueba con la sonda Z en las coordenadas X e Y actuales)

G32 Aparca el carro de la sonda Z

Es el comando contrario al anterior G31, en este caso aparcara el carro de la sonda en el lateral.

Condición:

Marlin solo tiene en cuenta este comando si tiene **habilitadas** las opciones ENABLE_AUTO_BED_LEVELING de auto nivelado automático y la opción Z_PROBE_SLED en el archivo Configuration.h, en caso contrario no tendrá efecto.

Uso

G32 (Aparcamos el carro una vez hecha la prueba)

Parámetros

No tiene parámetros.

Ejemplo

G31 (Acoplamos el carro de la sonda)
G30 (Hará la prueba con la sonda Z en las coordenadas X e Y actuales)
G32 (Aparcamos el carro en un lateral)

5.6 Comandos G Unbuffered

Los siguientes comandos no se almacenan en el búfer interno. Cuando se recibe uno se almacena, pero la placa controladora no lo reconoce hasta que el búfer se vacía, es entonces cuando el comando se ejecuta y la placa se detiene hasta que lo haya finalizado. Estas pausas cortas entre estos comandos y las que podrían derivarse de ellos no afectan en el rendimiento de la impresora.

5.6.1 G4: Retardo

Tiempo de espera en el que la maquina no hará nada. Para los que saben más de programación, esto **no es equivalente** a un delay, ya que mientras espera, Marlin sigue controlando todas las rutinas importantes (Temperatura, refresco de LCD, etc.). Simplemente es una parada en la ejecución de los Gcodes.

Uso

G4 Snnn ó Pnnn

Parámetros

Snnn Tiempo de espera en segundos.
Pnnn Tiempo de espera en milisegundos.

Ejemplo

G4 P200 (esperar sin hacer nada durante 200 milisegundos)
G4 S1 (esperar sin hacer nada durante 1 segundo)

5.6.2 G10 y G11: Retracción de filamento

Estos Gcodes no son soportados por todos los software de Slicer, normalmente se envía un Gcode *G1 E-3* (valores en negativo) para indicarle que tiene que retraer 3 mm el extrusor, por lo que es el propio software de Slicer el que tendrá el control total sobre la retracción del filamento. No obstante si hemos activado la auto retracción con el Gcode M209 (ver mas adelante), Marlin convertirá los G1 E-nnn en G10.

G10 Retracción de filamento

Se retraerá el filamento según los parámetros por defecto configurados en Configuration_adv.h o con los parámetros introducidos con los Gcodes M207 y M208.

Condición:

Marlin solo tiene en cuenta este comando si tiene **habilitada** la opción FWRETRACT en el archivo Configuration_adv.h, en caso contrario no tendrá efecto.

Uso

G10 S1

Parámetros

Parámetro no obligatorio.

S1 Indica que es una retracción para el cambio de filamento.

Ejemplo

G10 (Se retraerá el filamento)
G10 S1 (Es un cambio de filamento, y se retrae mas cantidad de mm)

Si usamos el parámetro S1, se especifica que es una retracción para el cambio de filamento, y el filamento se retraerá la cantidad configurada en RETRACT_LENGTH_SWAP en el archivo Configuration_adv.h que es una longitud mayor.

G11 Recuperación de filamento

Se recupera el filamento extruido con G10 según los parámetros por defecto configurados en Configuration_adv.h o con los parámetros introducidos con los Gcodes M207 y M208.

Condición:

Marlin solo tiene en cuenta este comando si tiene **habilitada** la opción FWRETRACT en el archivo Configuration_adv.h, en caso contrario no tendrá efecto.

Uso

G11

Parámetros

No hay parámetros

Ejemplo

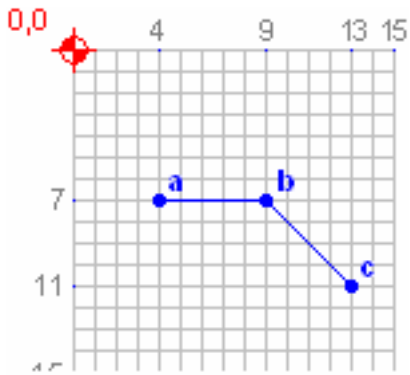
G10 (Se retraerá el filamento previamente)
G11 (Se recupera el filamento retraído)

Si previamente hemos usamos el parámetro S1 con el G10, se recuperara la cantidad indicado para el cambio de filamento en RETRACT_LENGTH_SWAP en el archivo Configuration_adv.h.

5.6.3 G90 y G91: Ajuste de Posicionamiento absoluto o relativo

Cuando antes hablamos del comando G1 los valores que le acompañan son los desplazamientos sobre cada eje (X, Y, Z), pero la máquina debe tener siempre un punto de referencia. Por lo general, el punto de referencia es la coordenada 0,0 también conocida como punto de origen, o Punto Cero. El comando G90 indica Modo Absoluto, es decir que todos los valores de las coordenadas X, Y, Z serán referidos a ese punto de origen. El comando G91 indica Modo Incremental o Relativo, o sea, utilizará el Punto Cero sólo cuando comience el trazado, de ahí en adelante, el último punto, se convertirá en punto de origen para el próximo desplazamiento.

Bueno, lo aclaremos un poco más, mira estos dos gráficos, a pesar de que se trata del mismo, las instrucciones de trazado serán distintas, en un caso lo haremos en Modo Absoluto (G90), y en el otro lo haremos en Modo Incremental o Relativo (G91). Ambos son solo con desplazamientos en los ejes X e Y a fin de hacer más fácil la explicación.

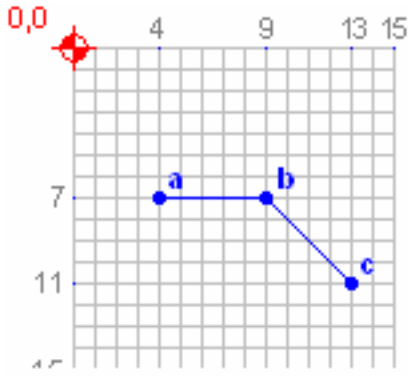


Desplazamiento Absoluto

1. G90
2. G1 X4 Y7
3. G1 X9
4. G1 X13 Y11
5. G1 X0 Y0

Línea 1: G90 Los desplazamientos que siguen serán referidos al punto 0,0.
Línea 2: G1 Se debe de dirigir al punto X=4,Y=7 (a).
Línea 3: G1 Se debe de dirigir hasta el punto X=9 (b).
Línea 4: G1 Continúa con el movimiento hasta el punto X=13,Y=11 (c).
Línea 5: G1 Desplázate hasta el punto X=0,Y=0 (Regresa al punto de origen 0,0).

Ahora veamos el otro Modo de Desplazamiento:



Desplazamiento Relativo o incremental

1. G91
2. G1 X4 Y7
3. G1 X5
4. G1 X4 Y4
5. G1 X-13 Y-11

Línea 1: G91 Los desplazamientos que siguen serán en modo incremental.
Línea 2: G1 Se debe mover 4 puntos en el eje X y 7 puntos en el eje Y (a).
Línea 3: G1 Debe moverse 5 puntos en el eje X (b).
Línea 4: G1 Continúa con el movimiento 4 puntos en el eje X y 4 puntos en el eje Y (c).
Línea 5: G1 Desplázate 13 puntos hacia atrás en el eje X, y 11 puntos hacia atrás en el eje Y (Regresa al punto de origen 0,0).

Como podemos observar, aunque el movimiento realizado ha sido el mismo en los 2 casos, las instrucciones cambian mucho.

G90 Modo absoluto

Todas las coordenadas a partir de este momento son tomadas como absolutas con respecto al origen de la impresora (0,0,0). Es la forma de trabajo por defecto.

Uso

G90

Parámetros

No hay parámetros

Ejemplo

G90 (Coordenadas en modo absoluto)
G1 X4 Y7 (Ve al punto X=4 Y=7)

G91 Modo relativo

Todas las coordenadas a partir de este momento son en relación a la última posición.

Uso

G91

Parámetros

No hay parámetros

Ejemplo

G91 (Coordenadas en modo relativo)
G1 X4 Y7 (Desde donde estas mueve el eje X 4 mm y el eje Y 7 mm)

5.6.4 G92: Establecer coordenadas de la posición actual

Establece la posición actual con las coordenadas fijadas, si no se incluyen parámetros establece la posición a 0 en todos los ejes. No produce movimiento, solo cambia los valores de coordenadas de la posición actual.

Uso

G92 Xnnn Ynnn Znnn Ennn

Parámetros

Se puede usar con parámetros o sin ellos.

Xnnn Valor que tomara la coordenada del eje X.
Ynnn Valor que tomara la coordenada del eje Y.
Znnn Valor que tomara la coordenada del eje Z.
Ennn Valor que tomara la coordenada del extrusor E.

Ejemplos

G92 X90 Y10 *(La posición actual tendrá las coordenadas X=90, Y=10 y el resto quedaran como estaban)*

G92 Z10 *(La posición actual tendrá la coordenada Z=10 y el resto como estaban)*

G92 (La posición actual tendrá las coordenadas X=0, Y=0, Z=0, E=0)

Hay que tener precaución al usar este comando, ya que al cambiar las coordenadas con G92, realmente estamos engañando a la impresora con la posición actual del carro del extrusor, y se pueden producir movimientos fuera de la zona de impresión llegando a los limites de los ejes.

5.7 Comandos M Unbuffered

Los siguientes comandos no se almacenan en el búfer interno. Este tipo de comandos se usa para enviar ordenes que no son de movimientos, o para asignar valores a algunos parámetros.

5.7.1 M0 Y M1: Stop

Marlin no hace distinción entre los 2 comandos, se pueden usar indistintamente. La impresora terminara cualquier movimiento que quedan en su memoria intermedia (buffer), y luego se parara hasta que el usuario presione un botón en el LCD (o pulse el encoder). Se puede introducir una cantidad de tiempo durante el cual ni pulsando el botón del LCD continuara la impresión.

Condición:

Marlin solo tiene en cuenta este comando si tiene algún tipo de LCD con encoder rotativo con posibilidad de pulsarlo, en caso contrario no tendrá efecto.

Uso

M0 Snnn ó Pnnn
M1 Snnn ó Pnnn

Parámetros

Snnn Tiempo de espera en segundos.
Pnnn Tiempo de espera en milisegundos.

Ejemplo

M0 S10 (Parara la impresora y durante 10 segundos no continuara aunque pulsemos el encoder, terminado ese tiempo espera a que se pulse el encoder)
M1 (Parara la impresora y espera a que se pulse el encoder)

Ver también M112.

5.7.2 M17: Activar todos los motores paso a paso

Activa los motores paso a paso de todos los ejes, lo que los dejara bloqueados en la posición actual sin poder actuar manualmente sobre ellos.

Uso

M17

Parámetros

No tiene parámetros

Ejemplo

M17 (Activa los motores de todos los ejes)

5.7.3 M18: Deshabilitar los motores paso a paso

Ver M84.

5.7.4 M31: Tiempo transcurrido desde que se ha comenzado la impresión con un comando M109 o desde un archivo en una tarjeta SD

Envía por el puerto serie (al terminal o Pronterface) el tiempo transcurrido desde que se inició la impresión con un comando M109 o desde un archivo almacenado en una tarjeta SD.

Uso

M31

Parámetros

No tiene parámetros

Ejemplo

M31 (Saca por el monitor serie o por Pronterface el tiempo transcurrido)

El mensaje enviado será parecido al siguiente:

eco: 54 min, 38 seg

5.7.5 M42: Enciende/Apaga un pin de salida digital o analógico

Enciende o apaga el pin digital que indiquemos, si es un pin analógico del microcontrolador también se puede enviar un valor entre 0-255 para sacar el valor analógico correspondiente (por PWM). Si no se indica pin se usara el led que suelen incluir las placas controladoras. Si el pin indicado es digital y no admite PWM (salida analógica) cualquier número superior a 0 será considerado HIGH +5V.

Uso

M42 Pnnn Snnn

Parámetros

Pnnn Numero de pin sobre el que queremos actuar.
Snnn Valor entre 0-255 que enviaremos al pin.

Ejemplo

M42 P009 S100 (Pone a 5V (1 lógico) la salida digital 9)

M42 P009 S000 (Apaga (0 lógico) la salida digital 9)

M42 P034 S125 (Pone a 2,5V (aprox) la salida analógica 34)

M42 S100 (Enciende el led de la placa controladora)

Hay que tener en cuenta que si el pin indicado en Pnnn corresponde con el pin de control del ventilador FAN_PIN, ajustara la velocidad del ventilador al valor enviado en S.

5.7.6 M48: Verificación de la repetitividad del EndStop Z

Cuando tengamos habilitado el auto nivelado de la cama de impresión, usaremos el Gcode M48 para comprobar la repetitividad de disparo de nuestro EndStop o sistema de disparo que tengamos para nivelar. Básicamente nos aseguramos que el EndStop que hemos usado para tomar las medidas es de buena calidad y dispara casi siempre en el mismo instante. Realizara un numero de pruebas y después sacara por el puerto serie (Terminal o Pronterface) la desviación del EndStop, cuanto mayor sea el numero peor será nuestro sistema de disparo.

Podemos configurar las coordenadas X Y donde queremos hacer al prueba, el numero de veces que se hará, si queremos que se recoja y extienda en cada prueba el servo (si tenemos), si después de cada medida se moverá una cantidad aleatoria para volver al punto inicial y repetir la prueba y finalmente la cantidad de mensajes que sacara por el puerto serie para no saturar la consola. Mas información en el foro de www.3dprintboard.com

Condición:

Marlin solo tiene en cuenta este comando si tiene **habilitadas** las opciones ENABLE_AUTO_BED_LEVELING de auto nivelado automático y la opción Z_PROBE_REPEATABILITY_TEST en el archivo Configuration.h, en caso contrario no tendrá efecto.

Uso

M48 nnnn Xnnn Ynnn Vnnn E Lnnn

Parámetros

No tiene por que llevar todos los parámetros, puede ir sin ninguno.

nnnn Numero de muestras que tomara para calcular la desviación. Es n minúscula, ya que la N mayúscula la usa Marlin para otros comandos y puede dar error. El numero de muestras tiene que estar entre 4 y 50 y por defecto es 10.
Xnnn Coordenada X donde queremos que se haga la prueba.
Ynnn Coordenada Y donde queremos que se haga la prueba.
Vnnn Verbosidad, o cantidad de información que enviara por el puerto serie (Consola). Se puede introducir un valor entre 0 y 4, siendo por defecto 1.
E Recoge y extiende el servo del EndStop en cada disparo. De esta manera sabremos si el sistema de servo que tenemos montado afecta a la repetitividad del disparo del EndStop. Por defecto no esta activado.
Lnnn Numero de movimientos que hará antes de repetir la medida. De esta manera sabremos si los ajustes de nuestros ejes afectan a la repetitividad del disparo del EndStop. Se puede usar un numero entre 0 y 15, y por defecto será 0 (sin movimientos)

Ejemplo

M48 (Realiza la prueba con los parámetros por defecto, suficiente la mayoría de las veces)

M48 n25 X50 Y50 E (Tomara 25 muestras en las coordenadas X=50, Y=50 y retraerá y desplegara el servo cada vez)

M48 n15 L5 (Tomara 15 muestras y entre cada medida realizara 5 movimientos)

5.7.7 M80: Encender fuente de alimentación

Si estamos usando el cable PS-ON (verde) en la fuente ATX, o el cable SLEEP (azul) en la fuente de X-Box para controlar el encendido de la fuente de alimentación con el pin PS_ON_PIN, podremos cambiar el estado de la fuente de modo en espera a encendida con el comando M80. No vamos a explicar aquí como hacer estas conexiones del Hardware, mas información en la wiki de reprap. Si además hemos definido el SUICIDE_PIN, este se pondrá a 5V (1 lógico).

Condición:

Marlin solo tiene en cuenta este comando si tiene definido el pin PS_ON_PIN en el archivo pins.h correspondiente a tu placa, en caso contrario no tendrá efecto.

Uso

M80

Parámetros

No tiene parámetros.

Ejemplo

M80 (La fuente de alimentación pasara de estar en StandBy a encenderse)

5.7.8 M81: Apaga la fuente de alimentación

Si estamos usando el cable PS-ON (verde) en la fuente ATX, o el cable SLEEP (azul) en la fuente de X-Box para controlar el encendido de la fuente de alimentación con el pin PS_ON_PIN, la pondrá en StandBy (espera). Si hemos definido el SUICIDE_PIN, se pondrá a 0 (0 lógico). Si no tenemos conectados ninguno de estos pines, este comando aun así, apaga los extrusores y la HeatedBed, deshabilita los motores y apaga el ventilador. Este comando seria el contrario al anterior M80.

Uso

M81

Parámetros

No tiene parámetros.

Ejemplo

M81 (La fuente de alimentación pasara a estar en standby)

5.7.9 M82: Establecer las coordenadas del extrusor en modo absoluto

Establece las coordenadas referentes al extrusor (E) en modo absoluto. Para más información mira G90. G90 afecta a todos los ejes, y M82 solo afecta al extrusor. El modo absoluto es que usa Marlin por defecto.

Uso

M82

Parámetros

No tiene parámetros.

Ejemplo

M82 (pone las coordenadas del eje E (extrusor) en modo absoluto)

5.7.10 M83: Establecer las coordenadas del extrusor en modo relativo

Establece las coordenadas referentes al extrusor (E) en modo relativo. Para más información mira G91. G91 afecta a todos los ejes, y M83 solo afecta al extrusor.

Uso

M83

Parámetros

No tiene parámetros.

Ejemplo

M83 (pone las coordenadas del eje E (extrusor) en modo relativo)

5.7.11 M84: Deshabilita los motores y fija el tiempo de inactividad de los motores

Deshabilita los motores hasta el próximo movimiento. También se puede utilizar para modificar el tiempo de inactividad, que es el tiempo en segundos que tardaran en desconectarse los motores una vez que la impresora entre en estado de reposo. Por defecto esta configurado en el archivo Configuration_adv.h a *DEFAULT_STEPPER_DEACTIVE_TIME 60* segundos., pero podemos modificar el tiempo de desconexión con este mismo Gcode y el parámetro Snnn.

Uso

Puede ir sin parámetros.

M84 X Y Z E Snnn

Parámetros

X Deshabilitara el motor del eje X.
Y Deshabilitara el motor del eje Y.
Z Deshabilitara el motor del eje Z **.**
E Deshabilitara los motores de los extrusores E**.**
Snnn Tiempo de inactividad en segundos que tardar en deshabilitar los motores.

Si no lleva parámetro X,Y,Z, o E deshabilitara los motores de todos los ejes.

Ejemplos

M84 (Deshabilita los motores de todos los ejes)

M84 X Y S25 (Deshabilita los motores de los ejes X, Y y cambia el tiempo de inactividad a 25 segundos)

M84 S25 (Deshabilita los motores de todos los ejes y cambia el tiempo de inactividad a 25 segundos)

M84 S0 (Deshabilita los motores de todos los ejes y cambia el tiempo de inactividad a 0 segundos, es decir, no se deshabilitaran nunca los motores)

5.7.12 M85: Tiempo máximo de inactividad

Fija el tiempo máximo que estará la impresora en modo de espera, sin haber recibido ningún comando (modo inactivo). Pasado este tiempo, se deshabilitarán los motores, se apagara todo incluso la fuente de alimentación si la estamos controlando a través de la placa. Por defecto este tiempo esta en 0 segundos, es decir, no se tiene en cuenta el tiempo en modo inactivo que lleve la impresora.

Uso

M85 Snnn

Parámetros

Snnn Tiempo en segundos transcurrido el cual sin actividad se apagara la impresora.

Ejemplos

M85 S300 (Fija el tiempo de inactividad en 300 segundos (5 minutos))

M85 S0 (Deshabilita el control del tiempo en modo de inactividad, modo por defecto)

5.7.13 M92: Fija el numero de pasos por milímetro de cada eje

Fija los **pasos del motor que corresponden con 1mm** de movimiento por cada uno de los ejes, incluido el extrusor. Este parámetro *DEFAULT_AXIS_STEPS_PER_UNIT* se configura en el archivo Configuration.h, pero se puede modificar sin tener que compilar el programa usando este comando y guardándolo posteriormente en la Eeprom. También se puede usar durante la calibración de la impresora, y una vez encontrado el valor adecuado modificar el parámetro en Configuration.h definitivamente.

Uso

M92 Xnnn Ynnn Znnn Ennn

Parámetros

No hace falta que lleve todos los parámetros, pero al menos necesita uno.

Xnnn Pasos por mm del eje X.
Ynnn Pasos por mm del eje Y.
Znnn Pasos por mm del eje Z.
Ennn Pasos por mm del extrusor E, mismo para todos los extrusores.

Ejemplos

M92 X85.20 Y85.20 (85,20 pasos en los ejes X, Y equivalen a 1mm de desplazamiento)

M92 Z3200 (Se necesitan 3200 pasos para desplazar 1mm el eje Z)

5.7.14 M104: Fija la temperatura objetivo del extrusor (Hotend). No bloqueante

Enciende el Hotend y fija la temperatura objetivo que tendrá que alcanzar. No es un comando bloqueante, se envía el dato y la placa controladora seguirá ejecutando el siguiente Gcode aunque no se haya alcanzado la temperatura en el Hotend. Ver M109.

Uso

M104 Snnn

Parámetros

Snnn Temperatura objetivo del extrusor en grados centígrados.

Ejemplos

M104 S195 (Enciende el Hotend y fija la temperatura objetivo en 195 grados)

5.7.15 M105: Leer las temperaturas actuales

Devuelve la temperatura en grados centígrados de los Hotend y de la HeatedBed por el puerto serie a la consola o a Pronterface.

Uso

M105

Parámetros

No tiene parámetros.

Ejemplos

M105 (solicitamos las lecturas de las temperaturas actuales)

El mensaje enviado será parecido al siguiente:

ok: T:204 B:73

Donde T será la temperatura del Hotend y B la de la HeatedBed.

5.7.16 M106: Ventilador On

Enciende el ventilador conectado a la placa, generalmente es el del extrusor 0, y fija la velocidad a la que girara, si no se incluye ningún parámetro girara a la máxima velocidad.

Condición:

Marlin solo tiene en cuenta este comando si tiene definido el pin FAN_PIN en el archivo pins.h correspondiente a tu placa, en caso contrario no tendrán efecto.

Uso

M106 Snnn

Parámetros

Snnn Velocidad a la que girara el ventilador. Valor entre 0-255. S0 lo apagara, ver M107.

Ejemplos

M106 S255 (Enciende el ventilador a la máxima velocidad)

M106 (Enciende el ventilador a la máxima velocidad)

M106 S127 (enciende el ventilador a velocidad media)

M106 S0 (Apaga el ventilador)

5.7.17 M107: Ventilador Off

Apaga el ventilador conectado a la placa, generalmente es el del extrusor 0. Este Gcode esta depreciado (en desuso) hay que utilizar M106 S0.

Condición:

Marlin solo tiene en cuenta este comando si tiene definido el pin FAN_PIN en el archivo pins.h correspondiente a tu placa, en caso contrario no tendrán efecto.

Uso

M107

Parámetros

No tiene parámetros.

Ejemplos

M107 (Apaga el ventilador)

5.7.18 M109: Fija la temperatura objetivo del extrusor (Hotend) y espera

Enciende el Hotend, fija la temperatura objetivo y espera hasta que se haya alcanzado la temperatura en el Hotend. Mientras espera no ejecutara ningún Gcode.

Si usamos el parámetro **R** en lugar de **S** la placa controladora esperara también tanto para calentar como para que se enfrie el Hotend. Podría ser útil a la hora de cambiar de filamento o al utilizar filamentos que cambian sus propiedades con al temperatura, si queremos tener una temperatura inferior a la actual para empezar a extruir.

Si tenemos habilitada la función AUTOTEMP en el archivo Configuration_adv.h se puede activar el modo AUTOTEMP enviando un Gcode M109 junto con el parámetro **F** y apagarlo enviando un M109 sin parámetro **F**. Como ya hemos comentado AUTOTEMP ajusta la temperatura del Hotend en función de la velocidad de extrusión. Más información en el apartado correspondiente a las "Configuraciones avanzadas. Detalle de la librería Configuration_adv.h" en este mismo documento.

Condición:

Marlin solo tendrá en cuenta los parámetros **B** y **F** si tiene activa la función AUTOTEMP en el archivo Configuration_adv.h, en caso contrario no tendrán efecto.

Uso

M109 Snnn ó Rnnn Bnnn Fnnn

Parámetros

Snnn Temperatura objetivo del extrusor en grados centígrados. Si tenemos activo AUTOTEMP fija la temperatura mínima.
Rnnn Temperatura objetivo del extrusor en grados centígrados pero teniendo en cuenta también si hay esperar a que se enfríe el Hotend.
Bnnn Si tenemos activo AUTOTEMP fija la temperatura máxima.
Fnnn Si tenemos activo AUTOTEMP fija el factor de corrección.

Ejemplos

M109 S195 (Enciende el Hotend y fija la temperatura objetivo en 195 grados)

M109 R195 (Enciende el Hotend o espera que se enfríe hasta alcanzar 195 grados)

M109 S190 B210 F0.95 (Si tenemos activo AUTOTEMP lo conecta y fija la temperatura mínima en 190 grados, la temperatura máxima en 210 y el factor en 0,95)

5.7.19 M112: Parada de emergencia

Cualquier movimiento en proceso terminará inmediatamente y se apagaran todos los motores, calentadores, fuentes de alimentación y la impresora. Se puede iniciar de nuevo pulsando el botón de reinicio de la placa controladora o apagando la impresora. Ver también M0 y M1. Este tipo de Gcode seria el usado para enviar desde un botón programado en el LCD, y se pulsaría para una parada de emergencia ante enganchones, bloqueos, calentones, etc.

Uso

M112

Parámetros

No tiene parámetros.

Ejemplos

M112 (Parada de emergencia)

5.7.20 M114: Obtener Posición actual

Devuelve las coordenadas X, Y, Z, E actuales del extrusor por el puerto serie a la consola o a Pronterface.

Uso

M114

Parámetros

No tiene parámetros.

Ejemplos

M114 (Solicita las coordenadas actuales del extrusor)

La máquina devuelve una cadena, tal como:

ok X: 0.00 Y: 0.00 Z: 0.00 E: 0.00 Count X:0.00 Y:0.00 Z:0.00

Los 4 primeros números son las coordenadas. Los otros 3 números son las posiciones de la función paso a paso, esto ayuda a depurar una función anterior si hemos tenido un error.

5.7.21 M115: Obtener la versión de firmware y principales características

Devuelve la versión del firmware y los datos de identificación de la maquina por el puerto serie a la consola o a Pronterface.

Uso

M115

Parámetros

No tiene parámetros.

Ejemplos

M115 (Solicita los datos de la impresora)

La máquina devuelve una cadena, tal como:

ok FIRMWARE_NAME:Marlin V1; Sprinter/grbl mashup for gen6 FIRMWARE_URL https://github.com/ErikZalm/Marlin/ PROTOCOL_VERSION: 1.0 MACHINE_TYPE: *Mendel EXTRUDER_COUNT: 1 UUID: 00000000-0000-0000-0000-000000000000*

Algunos datos dependerán del tipo de placa y maquina que tengamos configurada.

5.7.22 M117: Muestra un mensaje por el LCD

Muestra el mensaje adjunto como parámetro, por la línea de estado del LCD.

Uso

M117 Mensaje

Parámetros

Mensaje Es el texto que queremos mostrar por el LCD. Hay que tener precaución con la longitud del texto, no debería de ser mas largo de 16 caracteres para que se visualice en todos los LCD

Ejemplos

M117 Comenzando… (Muestra por el LCD el mensaje "Comenzando…")

5.7.23 M119: Obtener el estado de los EndStop

Devuelve el estado (pulsado o no) de los EndStop de X, Y, Z por el puerto serie a la consola o a Pronterface.

Uso

M119

Parámetros

No tiene parámetros.

Ejemplos

M119 (Solicita los datos del estado de los EndStop)

La máquina devuelve una cadena, tal como:

Ok Reporting EndStop status x_min: TRIGGERED y_min: open z_min: open

El numero de EndStop dependerá de la configuración de nuestra maquina. TRIGGERED es pulsado y open no

5.7.24 M120: Deshabilita los EndStop

Deshabilita los EndStop, si se pulsa alguno durante un movimiento no se detendrá.

Uso

M120

Parámetros

No tiene parámetros.

Ejemplos

M120 (Deshabilita los EndStop)

5.7.25 M121: Habilita los EndStop

Habilita los EndStop, si se pulsa alguno durante un movimiento se detendrá.

Uso

M121

Parámetros

No tiene parámetros.

Ejemplos

M121 (Habilita los EndStop)

5.7.26 M140: Fija la temperatura objetivo de la cama (HeatedBed). No bloqueante

Enciende la HeatedBed y fija la temperatura objetivo que tendrá que alcanzar. No es un comando bloqueante, se envía el dato y la placa controladora seguirá ejecutando el siguiente Gcode aunque no se haya alcanzado la temperatura en la HeatedBed. Ver M190.

Uso

M140 Snnn

Parámetros

Snnn Temperatura objetivo de la HeatedBed en grados centígrados.

Ejemplos

M140 S70 (Enciende la cama y fija la temperatura objetivo en 70 grados)

5.7.27 M150: Enciende el led del BlinkM

Si tenemos un modulo BlinkM (http://thingm.com/products/blinkm/) controlado mediante un puerto I2C conectado a nuestra placa, podemos enviarle comandos para encender el led y cambiarlo de color, siempre y cuando este activo el parámetro MBLINK en el archivo Configuration.h,

Condición:

Marlin solo tiene en cuenta estos comandos si tiene habilitado el parámetro BLINKM en el archivo Configuration.h, en caso contrario no tendrán efecto.

Uso

M150 Rnnn Unnn Bnnn

Parámetros

Rnnn Proporción de color Rojo (Red) con la que lucirá el led, entre 0-255.
Unnn Proporción de color verde (Green, pero la G puede dar problemas) con la que lucirá el led, entre 0-255.
Bnnn Proporción de color Azul (Blue) con la que lucirá el led, entre 0-255.

Ejemplos

M150 R255 U255 B255 (Encenderemos el led de color blanco)

M150 R255 (Encenderemos el led de color Rojo)

M150 R0 U0 B0 (Apagara el led)

5.7.28 M190: Fija la temperatura objetivo de la cama (HeatedBed) y espera

Enciende la HeatedBed, fija la temperatura objetivo y espera hasta que se haya alcanzado la temperatura en la HeatedBed. Mientras espera no ejecutara ningún Gcode.

Si usamos el parámetro **R** en lugar de **S** la placa controladora esperara también tanto para calentar como para que se enfríe la HeatedBed. Podría ser útil si queremos tener una temperatura inferior a la actual para empezar a extruir.

Uso

M190 Snnn ó Rnnn

Parámetros

Snnn Temperatura objetivo de la cama (HeatedBed) en grados centígrados.
Rnnn Temperatura objetivo de la cama (HeatedBed) en grados centígrados pero teniendo en cuenta también si hay que esperar a que se enfríe.

Ejemplos

M190 S110 (Enciende la cama y fija la temperatura objetivo en 110 grados)

M190 R70 (Enciende la cama o espera que se enfríe hasta alcanzar 70 grados)

5.7.29 M200: Fija el diámetro del filamento y cambia las unidades del extrusor a volumétricas

Con este Gcode, fijamos el diámetro del filamento y cambiamos las unidades que manejara Marlin del extrusor a unidades volumétricas (mm^3). La utilidad de este comando es generar un código .gcode universal que sea independiente del diámetro del filamento, con lo que será mas fácil portar ese archivo a otras impresoras. El único ajuste que tendremos que hacer antes de generar el .gcode universal, es configurar el diámetro del filamento en nuestro programa de Slicer en 1,28379 para que todos los movimientos generados por el Slicer se correspondan con 1mm de longitud = $1mm^3$. Para volver a usar las unidades en mm lineales, cambiaremos el diámetro en el programa de Slicer y fijaremos con M200 el diámetro en 0. Más información:

- http://wooden-mendel.blogspot.com/2011/09/volumetric-stage-two.html
- http://reprap.org/wiki/Triffid_Hunter%27s_Calibration_Guide - Optional:_Switch_to_volumetric_E_units

Uso

M200 Dnnn Tnnn

Parámetros

Dnnn Diámetro del filamento en mm.
Tnnn Extrusor al que estamos fijando el diámetro del filamento, si tenemos uno solo no es necesario este parámetro, siendo 0 el extrusor uno, 1 el extrusor dos, etc.

Ejemplos

M200 D1.75 (Fija el diámetro del filamento en 1,75 mm)

M200 D2.85 T1 (Fija el diámetro del filamento que se usa en el segundo extrusor en 2,85 mm)

M200 D0 (Vuelve a poner las unidades del extrusor en mm lineales)

5.7.30 M201: Aceleración máxima.

Fija la aceleración máxima de los ejes en mm/s^2. Modifica el parámetro *DEFAULT_MAX_ACCELERATION* del archivo Configuration.h.

Uso

M201 Xnnn Ynnn Znnn Ennn

Parámetros

No hace falta que lleve todos los parámetros, pero al menos necesita uno.

Xnnn Aceleración máxima para el eje X.
Ynnn Aceleración máxima para el eje Y.
Znnn Aceleración máxima para el eje Z.
Ennn Aceleración máxima para el extrusor E, la misma para todos los extrusores.

Ejemplos

M201 X1500 Y1500 Z800 E1000 (Fija la aceleración máxima en 1200mm/s² para el eje X e Y, 800mm/s² para el Z y en 1000mm/s² para el extrusor)

M201 Z1200 (Fija la aceleración máxima en 1200mm/s² para el eje Z)

5.7.31 M203: Velocidad máxima

Fija la velocidad máxima de los ejes en mm/s. Modifica el parámetro *DEFAULT_MAX_FEEDRATE* del archivo Configuration.h.

Uso

M203 Xnnn Ynnn Znnn Ennn

Parámetros

No hace falta que lleve todos los parámetros, pero al menos necesita uno.

Xnnn Velocidad máxima para el eje X.
Ynnn Velocidad máxima para el eje Y.
Znnn Velocidad máxima para el eje Z.
Ennn Velocidad máxima para el extrusor E, la misma para todos los extrusores.

Ejemplos

M203 X100 Y100 Z4 E80 (Fija la velocidad máxima en 100mm/s para el eje X e Y, 4mm/s para el Z y en 80mm/s para el extrusor)

M203 Z6 (Fija la velocidad máxima en 6mm/s para el eje Z)

5.7.32 M204: Aceleración por defecto

Fija la aceleración por defecto de los ejes en mm/s^2, si antes de un movimiento G1 no se especifica ninguna aceleración, será esta la que se emplee. También se puede fijar la aceleración empleada para los movimientos que afectan solo al extrusor (cuando se envía un G1 solo con el parámetro E), incluidas las retracciones de filamento con el parámetro T. Modifica el parámetro *DEFAULT_ACCELERATION* del archivo Configuration.h.

Uso

M204 Snnn Tnnn

Parámetros

No hace falta que lleve todos los parámetros, pero al menos necesita uno.

Snnn Aceleración por defecto para los ejes X, Y, Z, y el extrusor.
Tnnn Aceleración por defecto para los movimientos (cuando se envía un G1 solo con el parámetro E) y retracciones del extrusor.

Ejemplos

M204 S1000 (Aceleración por defecto para X, Y, Z , E, $1000mm/s^2$)

M204 S1200 T800 (Aceleración por defecto para X, Y, Z , E, $1200mm/s^2$, y para los movimientos del extrusor en $800mm/s^2$)

5.7.33 M205: Parámetros avanzados de movimiento

Con este comando podemos modificar varios parámetros avanzados utilizados en el movimiento. velocidades mínimas de desplazamiento, tiempo mínimo en realizar los movimientos, velocidad que no precisara aceleración para los ejes X, Y, Z y E. Estos parámetros están configurados en los archivos Configuration.h y Configuration_adv.h y con este comando los modificamos.

Uso

M205 Snnn Tnnn Bnnn Xnnn Znnn Ennn

Parámetros

No hace falta que lleve todos los parámetros, pero al menos necesita uno.

Snnn Velocidad mínima en mm/s a las que se moverán los ejes mientras se imprime. Parámetro *DEFAULT_MINIMUMFEEDRATE* en Configuration_adv.h.
Tnnn Velocidad mínima en mm/s a las que se moverán los ejes en desplazamientos sin imprimir. Parámetro *DEFAULT_MINTRAVELFEEDRATE* en Configuration_adv.h.
Bnnn Tiempo mínimo en microsegundos que tardara en realizarse los movimientos cuando el buffer esta vacío. Parámetro *DEFAULT_MINSEGMENTTIME* en Configuration_adv.h.
Xnnn Velocidad mínima en mm/s que no requerirá del cálculo de aceleración, es decir a la velocidad aquí indicada el eje se moverá instantáneamente, para los ejes X, Y. Parámetro *DEFAULT_XYJERK* en Configuration.h.
Znnn Velocidad mínima en mm/s que no requerirá del cálculo de aceleración, es decir a la velocidad aquí indicada el eje se moverá instantáneamente, para el eje Z. Parámetro *DEFAULT_ZJERK* en Configuration.h.
Ennn Velocidad mínima en mm/s que no requerirá del cálculo de aceleración, es decir a la velocidad aquí indicada el eje se moverá instantáneamente, para el extrusor E. Parámetro *DEFAULT_EJERK* en Configuration.h.

Ejemplos

M205 S15 T40 X25 (Fijaremos la velocidad mínima mientras imprime en 15mm/s, la velocidad en desplazamientos en 40 mm/s y la velocidad instantánea de los ejes X, Y en 25 mm/s)

5.7.34 M206: Adicional offset al Homing

Los valores especificados se añaden a la posición de disparo de los EndStop cuando se hace un Homing. Es decir, una vez pulsado el Homing, el eje se desplazara la distancia aquí indicada y se tomara como el 0 del eje.

Uso

M206 Xnnn Ynnn Znnn

Parámetros

No hace falta que lleve todos los parámetros, pero al menos necesita uno.

Xnnn Offset que se añadirá al Homing del eje X.
Ynnn Offset que se añadirá al Homing del eje Y.
Znnn Offset que se añadirá al Homing del eje Z.

Ejemplos

Ejemplo:

M206 X10.0 Y10.0 Z-0,4 (Se añade al Homing del eje X, Y 10mm y al eje Z -0,4 mm)

M206 Z-0,3 (Se añade al Homing del eje Z -0,3 mm)

5.7.35 M207: Parámetros de configuración de la auto retracción en G10

Fijamos los parámetros de la auto retracción que se ejecutara mediante el Gcode G10. Podemos fijar la longitud de retracción, la velocidad y la distancia que se desplazara el eje Z. Estos parámetros están configurados también en el archivo Configuration_adv.h.

Condición:

Marlin solo tiene en cuenta este comando si tiene definido el modo FWRETRACT en el archivo Configuration_adv.h, en caso contrario no tendrán efecto.

Uso

M207 Snnn Fnnn Znnn

Parámetros

No hace falta que lleve todos los parámetros, pero al menos necesita uno.

Snnn Longitud en mm que se retraerá el filamento al hacer un Gcode G10. Parámetro *RETRACT_LENGTH* en el archivo Configuration_adv.h.
Fnnn Velocidad en mm/minuto a la que se retraerá el filamento. Parámetro *RETRACT_FEEDRATE* en el archivo Configuration_adv.h.
Znnn Distancia que se levantara el eje Z para que al moverse el extrusor sobre la parte ya impresa de la pieza no quede un rastro. Parámetro *RETRACT_ZLIFT* en el archivo Configuration_adv.h.

Ejemplos

M207 S4.5 F2700 (Fijamos la longitud de retracción en 4,5 mm y la velocidad en 2700 mm/min (45mm/sec))

5.7.36 M208: Parámetros de configuración de la auto retracción en G11

Fijamos los parámetros de la recuperación después de una auto retracción que se ejecutara mediante el Gcode G11. Podemos fijar la longitud de recuperación después de la retracción y la velocidad. Estos parámetros están configurados también en el archivo Configuration_adv.h.

Condición:

Marlin solo tiene en cuenta este comando si tiene definido el modo FWRETRACT en el archivo Configuration_adv.h, en caso contrario no tendrán efecto.

Uso

M208 Snnn Fnnn

Parámetros

No hace falta que lleve todos los parámetros, pero al menos necesita uno.

Snnn Longitud adicional en mm que se recuperara el filamento después de una retracción al hacer un Gcode G11. Parámetro *RETRACT_RECOVER_LENGTH* en el archivo Configuration_adv.h.
Fnnn Velocidad en mm/minuto a la que se recuperara el filamento. Parámetro *RETRACT_RECOVER_FEEDRATE* en el archivo Configuration_adv.h.

Ejemplos

M207 S0.1 F600 (Fijamos la longitud de recuperación en 0,1 mm y la velocidad en 600 mm/min (10mm/sec))

5.7.37 M209: Habilita/deshabilita la retracción automática

Este parámetro permite habilitar la retracción automática si el software de Slicer no soporta los Gcodes G10 y G11. Cada movimiento individual de extrusión sólo se clasificará como retracción dependiendo de si son valores negativos.

Condición:

Marlin solo tiene en cuenta este comando si tiene definido el modo FWRETRACT en el archivo Configuration_adv.h, en caso contrario no tendrán efecto.

Uso

M209 S1 ó S0

Parámetros

S1 Habilita la auto retracción.
S0 Deshabilita la auto retracción.

Ejemplos

M209 S1 (Habilita la auto retracción)

M209 S0 (deshabilita la auto retracción)

5.7.38 M218: Offset entre extrusores

Fija el offset en mm de los extrusores cuando tenemos mas de uno. Introduciremos los offset entre ellos en los ejes X, Y y en caso de disponer de carro dual X también el offset Z. Estos parámetros se pueden configurar en los archivos Configuration.h y Configuration_adv.h.

Condición:

Marlin solo tiene en cuenta este comando si tiene definido mas de 1 extrusor *EXTRUDERS* en el archivo Configuration.h, en caso contrario no tendrán efecto. Y el offset Z solo se tendrá en cuenta si esta definido el modo de carro dual *DUAL_X_CARRIAGE* del eje X en el archivo Configuration_adv.h.

Uso

M218 Xnnn Ynnn Znnn

Parámetros

No hace falta que lleve todos los parámetros, pero al menos necesita uno.

Xnnn Offset entre extrusores en el eje X. Parámetro *EXTRUDER_OFFSET_X* en el archivo Configuration.h.
Ynnn Offset entre extrusores en el eje Y. Parámetro *EXTRUDER_OFFSET_Y* en el archivo Configuration.h.
Znnn Offset entre extrusores en el eje Z cuando se usa el carro dual en X. Parámetro *EXTRUDER1_Z_OFFSET* en el archivo Configuration_adv.h..

Ejemplos

M218 X25.4 Y0 (Fija el offset entre extrusores X=25,4 mm y el Y=0)

5.7.39 M220: Porcentaje de ajuste de la velocidad

Factor en porcentaje que se aplicara a la velocidad de los movimientos. Un factor de 100% es un multiplicador por 1, un factor de 200% es un multiplicador por 2.

Uso

M220 Snnn

Parámetros

Snnn Factor en % que se aplicara a la velocidad de movimiento.

Ejemplos

M220 S150 (La velocidad se aumentara un 50%, o lo que es lo mismo se multiplicara por 1,5)

M220 S75 (La velocidad se reducirá un 25% (100-75), o lo que es lo mismo se multiplicara por 0,75)

5.7.40 M221: Porcentaje de ajuste de la extrusión

Factor en porcentaje que se aplicara a la cantidad de filamento a extruir. Un factor de 100% es un multiplicador por 1, un factor de 200% es un multiplicador por 2. Es equivalente al parámetro "flow" del software Slicer Cura.

Uso

M221 Snnn

Parámetros

Snnn Factor en % que se aplicara a la cantidad de filamento a extruir.

Ejemplos

M221 S105 (La cantidad extruida se aumentara un 5%, o lo que es lo mismo se multiplicara por 1,05)

M221 S95 (La cantidad extruida se reducirá un 5% (100-95), o lo que es lo mismo se multiplicara por 0,95)

5.7.41 M226: Comprobación de estado de un pin determinado, bloqueante.

Este comando comprobara el estado del pin indicado y esperara (no continuara ejecutando mas Gcodes) hasta que sea igual al valor especificado. Solo está disponible para valores digitales LOW ó HIGH.

Uso

M226 Pnnn Snnn

Parámetros

Pnnn Numero de pin digital que se configurara como entrada y del que leeremos el valor.
Snnn Estado del pin que hará que se continúe con la ejecución de Gcodes. Los valores posibles son 0 (LOW), 1 (HIGH) y -1 (TOGGLE o cambio de estado).

Ejemplos

M226 P55 S0 (Esperaremos hasta que el valor leído en el pin 55 sea LOW)

M226 P55 S1 (Esperaremos hasta que el valor leído en el pin 55 sea HIGH)

M226 P55 S-1 (Esperaremos hasta que el valor leído en el pin 55 cambie de estado, si esta a LOW hasta que pase a HIGH y si esta en HIGH hasta que pase a LOW)

5.7.42 M240: Disparo de la cámara para tomar fotos

Si tenemos una cámara de fotos conectada a nuestra placa controladora, disparara una foto emulando un control remoto Canon RC-1. Más datos en http://www.doc-diy.net/photo/rc-1_hacked/ y en http://captain-slow.dk/2014/03/09/3d-printing-timelapses/. Hay que tener habilitada la función CHDK o haber configurado el pin PHOTOGRAPH_PIN.

Condición:

Marlin solo tiene en cuenta este comando si tiene definido el pin PHOTOGRAPH_PIN en el archivo Configuration.h o definido el modo CHDK en el archivo Configuration_adv.h, en caso contrario no tendrá efecto.

Uso

M240

Parámetros

No tiene parámetros.

Ejemplos

M240 (tomara una foto)

5.7.43 M250: Ajuste del contraste del LCD, solo para LCD gráficos.

Ajusta el contraste del LCD. Solo funciona con los LCD gráficos de 128x64 puntos. Indicado para la familia de LCD controlados con un chip ST7565R.

Condición:

Marlin solo tiene en cuenta este comando si tiene configurado algún LCD grafico en el archivo Configuration.h, en caso contrario no tendrá efecto.

Uso

M250 Cnnn

Parámetros

Cnnn Valor del contraste. Valores aceptados entre 0-63.

Ejemplos

M250 C40 (Fija el contraste en 40)

5.7.44 M280: Posiciona el servo en un ángulo determinado

Si tenemos algún servo para retraer o posicionar los EndStop, con este comando podemos cambiarlo de posición. Le indicaremos que servo queremos mover y la posición destino.

Condición:

Marlin solo tiene en cuenta este comando si tiene definido al menos 1 servo en el parámetro NUM_SERVOS en el archivo Configuration.h, en caso contrario no tendrá efecto.

Uso

M280 Pnnn Snnn

Parámetros

Pnnn Índice del servo que queremos mover. El índice comienza en 0 para el servo 1, 1 para el servo 2, etc.
Snnn Posición a la que queremos mover el servo en grados o microsegundos.

Ejemplos

M280 P0 S90 (Pone el primer servo en la posición 90 grados)

M280 P1 S0 (Pone el segundo servo en la posición de 0 grados)

5.7.45 M300: Emite un pitido por el buzzer de la placa o LCD.

Emite un pitido por el buzzer de la placa controladora o por el del LCD si están disponibles. Le indicaremos la frecuencia en HZ y la duración.

Condición:

Marlin solo tiene en cuenta este comando si tiene configurado algún LCD o un LCD I2C con buzzer en el archivo Configuration.h o también si la placa controlador tiene un buzzer y por tanto tiene asignado el pin BEEPER en el archivo pins.h correspondiente a tu placa, en caso contrario no tendrá efecto.

Uso

M300 Snnn Pnnn

Parámetros

No hace falta que lleve todos los parámetros, incluso puede ir sin ninguno.

Snnn Frecuencia en HZ, si no se indica por defecto es 110Hz.
Pnnn Tiempo que durara el sonido en microsegundos, si no se indica 1000 microsegundos (1 segundo).

Ejemplos

M300 S200 P500 (Emite un pitido de 200Hz durante 0,5 segundos)

M300 (Emite un pitido de 110Hz durante 1 segundo)

5.7.46 M301: Establece los parámetros PID, Kp, Ki y Kd del Hotend

Establece los parámetros Kp (valor proporcional), Ki (valor integral) y Kd (valor diferencial) que usa el control PID para ajustar la temperatura del Hotend. Estos parámetros están configurados también en el archivo Configuration.h y los indicado aquí los sobrescriben. Mas información sobre PID en http://reprap.org/wiki/PID_Tuning y en http://es.wikipedia.org/wiki/Proporcional_integral_derivativo.

Condición:

Marlin solo tiene en cuenta este comando si tiene definido el modo PIDTEMP en el archivo Configuration.h, en caso contrario no tendrá efecto.

Uso

M301 Ennn Pnnn Innn Dnnn

Parámetros

No hace falta que lleve todos los parámetros.

Ennn Numero de extrusor que vamos a configurar.
Pnnn Parámetro Kp (valor proporcional).
Innn Parámetro Ki (valor integral).
Dnnn Parámetro Kd (valor diferencial).

Ejemplos

M301 P26.13 I0.13 (Fija los parámetros Kp=26,13 y Ki=0,13)

5.7.47 M302: Permite extrusiones en frío, o fija temperatura mínima de extrusión

Si tenemos activa la opción *PREVENT_DANGEROUS_EXTRU*DE en el archivo Configuration.h, no se permite que funcione el motor del extrusor si el Hotend esta por debajo de la temperatura indicada en el parámetro *EXTRUDE_MINTEMP* (en el mismo archivo Configuration.h). Con este comando podemos modificar este comportamiento o también podemos cambiar la temperatura que hemos fijado en le parámetro *EXTRUDE_MINTEMP.*

Condición:

Marlin solo tiene en cuenta este comando si tiene definido el modo *PREVENT_DANGEROUS_EXTRU*DE en el archivo Configuration.h, en caso contrario no tendrá efecto.

Uso

M302 Snnn

Parámetros

No hace falta que lleve todos los parámetros.

Snnn Temperatura mínima en grados necesaria en el Hotend para que empiece a funcionar le motor del extrusor.

Ejemplos

M302 S175 (Fija el parámetro EXTRUDE_MINTEMP en 175 grados)

M302 (Permite la extrusión a cualquier temperatura)

5.7.48 M303 Auto ajuste de los parámetros PID

Los parámetros Kp, Ki y Kd que necesita el sistema PID para controlar la temperatura dependen directamente de la forma física de nuestro Hotend o de nuestra HeatedBed. Con este comando podemos iniciar un proceso de pruebas que dará con los parámetros adecuados para nuestra impresora. Este Auto ajuste es necesario realizarlo al configurar nuestra impresora por primera vez, y cada vez que cambiemos algo (Cartucho, termistor, bloque soporte, HeatedBed, etc). Se realizaran una serie de ciclos de encendido y nos dará por la consola serie o Pronterface los datos Kp, Ki, y Kd óptimos. Mas información en http://reprap.org/wiki/PID_Tuning

Uso

M302 Ennn Snnn Cnnn

Parámetros

No hace falta que lleve todos los parámetros.

Ennn Extrusor sobre el que queremos hacer el ajuste. Si no tenemos mas que uno no hay que indicarlo, ya que si no se indica ningún valor por defecto E vale 0 (primer extrusor). Para hacer el cálculo sobre la HeatedBed hay que poner -1.
Snnn Temperatura a la que queremos hacer los cálculos. Si no se indica ningún valor, por defecto es 150 grados para el Hotend y 70 grados para la HeatedBed.
Cnnn Numero de ciclos que realizara de calentamiento para afinar los parámetros. Cuantos mas ciclos hagamos mas exactos serán los parámetros obtenidos. Si no se indica ningún valor, por defecto son 5 ciclos.

Ejemplos

M303 (Empieza el auto ajuste del extrusor 1, con una temperatura objetivo de 150 grados y hará 5 ciclos de calentamiento)

M303 E1 S200 C10 (Empieza el auto ajuste del extrusor 2, con una temperatura objetivo de 200 grados y hará 10 ciclos de calentamiento)

M303 E-1 (Empieza el auto ajuste de la HeatedBed, con una temperatura objetivo de 70 grados y hará 5 ciclos de calentamiento)

5.7.49 M304: Establece los parámetros PID, Kp, Ki y Kd de la HeatedBed

Establece los parámetros Kp (valor proporcional), Ki (valor integral) y Kd (valor diferencial) que usa el control PID para ajustar la temperatura de la cama caliente o HeatedBed. Estos parámetros están configurados también en el archivo Configuration.h y los indicado aquí los sobrescriben.
Mas información sobre PID en http://reprap.org/wiki/PID_Tuning y en http://es.wikipedia.org/wiki/Proporcional_integral_derivativo.

Condición:

Marlin solo tiene en cuenta este comando si tiene definido el modo PIDTEMPBED en el archivo Configuration.h, en caso contrario no tendrá efecto.

Uso

M304 Pnnn Innn Dnnn

Parámetros

No hace falta que lleve todos los parámetros.

Pnnn Parámetro Kp (valor proporcional).
Innn Parámetro Ki (valor integral).
Dnnn Parámetro Kd (valor diferencial).

Ejemplos

M303 P6.18 I1.24 (Fija los parámetros Kp=6,18 y Ki=1,24)

5.7.50 M350: Cambia el modo de micro pasos de los drivers

Si tenemos algún driver de motor con entradas MS1 y MS2 para configurar los micro pasos y están conectadas al micro controlador, con este comando podemos configurar el modo de micro pasos (1, 1/2, 1/4, 1/8, 1/16). Más datos en http://www.bristolwatch.com/.

Solo están definidos los pines de conexión para las placas 5DPrint D8 y para la Rambo. Los valores admitidos son 1, 2, 4, 8 y 16 que en un motor de 200 pasos corresponden a 200, 400, 800, 1600 y 3200 micro pasos por vuelta. Generalmente si el driver soporta 1/16 micro pasos no soportara 1/8, consulta la documentación de tu driver. Hay que tener precaución porque el numero de pasos por unidad (mm) se mantiene, así que si modificamos el modo de los micro pasos con este comando, es preciso ajustar también el numero de pasos por unidad con el Gcode M92.

Condición:

Marlin solo tiene en cuenta este comando si tiene definido el pin X_MS1_PIN en el archivo pins.h correspondiente a tu placa, en caso contrario no tendrá efecto.

Uso

M350 Snnn Xnnn Ynnn Znnn Ennn Bnnn

Parámetros

No hace falta que lleve todos los parámetros, pero al menos necesita uno.

Snnn Modifica el modo de micro pasos de los drivers de todos los ejes (X, Y, Z, Extrusor).
Xnnn Modifica el modo de micro pasos del driver del eje X.
Ynnn Modifica el modo de micro pasos del driver del eje Y.
Znnn Modifica el modo de micro pasos del driver del eje Z.
Ennn Modifica el modo de micro pasos del driver del Extrusor 0.
Bnnn Modifica el modo de micro pasos del driver del Extrusor 1.

Ejemplos

M350 S16 (Pone el modo micro pasos de todos los drivers en 16 (3200 micro pasos))

M350 X4 Y4 (Pone el modo micro pasos de los drivers X e Y en 4 (800 micro pasos))

M350 B16 (Pone el modo micro pasos del driver del extrusor 1 en 16 (3200 micro pasos))

5.7.51 M351: Control directo de los pines MS1 y MS2 para cambiar el modo de micro pasos

Según el estado de los pines MS1 Y MS2, el driver trabaja en un modo determinado de micro pasos. Al asignar un modo con el Gcode M350 es Marlin el que se encarga de configurar MS1 y MS2 del correspondiente driver, pero con este comando podemos configurar nosotros directamente los pines MS1 y MS2 para seleccionar el modo de micro pasos. Generalmente la tabla de estado de los pines es la siguiente:

MS1	MS2	MODO
LOW	LOW	Sin micro pasos
HIGH	LOW	½ micro pasos
LOW	HIGH	¼ micro pasos
HIGH	HIGH	Según el modelo de driver 1/8 ó 1/16 micro pasos

Condición:

Marlin solo tiene en cuenta este comando si tiene definido el pin X_MS1_PIN en el archivo pins.h correspondiente a tu placa, en caso contrario no tendrá efecto.

Uso

M351 Sn Xn Yn Zn En Bn

Parámetros

No hace falta que lleve todos los parámetros, pero al menos necesita uno.

Sn Determina el pin que modificaremos, 1 para MS1 y 2 para MS2.
Xn Estado en el que se pondrá el pin del driver del eje X. 0 para LOW y 1 para HIGH.
Yn Estado en el que se pondrá el pin del driver del eje Y. 0 para LOW y 1 para HIGH.
Zn Estado en el que se pondrá el pin del driver del eje Z. 0 para LOW y 1 para HIGH.
En Estado en el que se pondrá el pin indicado del driver del extrusor 0. 0 para LOW y 1 para HIGH.
Bn Estado en el que se pondrá el pin indicado del driver del extrusor 1. 0 para LOW y 1 para HIGH.

Ejemplos

M351 S1 X0 Y0 Z0 E1 B1 (Pone el pin MS1 de los drivers de los ejes X, Y, Z a LOW y de los drivers de los extrusores 0 y 1 a HIGH*)*

M351 S2 X1 Y1 Z1 E1 B1 (Pone el pin MS2 de los drivers de todos los ejes X, Y, Z y de los extrusores 0 y 1 a HIGH*)*

Con estos 2 comandos hemos puesto los drivers de los ejes X, Y, y Z en el modo de ¼ de micro pasos (MS1=LOW y MS2= HIGH) y los drivers de los 2 extrusores en 1/16 o 1/8 según el driver (MS1= HIGH y MS2 = HIGH).

5.7.52 M400: Espera a que se vacíe el buffer antes de continuar

Esperara a que se ejecuten todos los comandos almacenados en el buffer, no continuara hasta que no este vacío.

Uso

M400

Parámetros

No lleva parámetros.

Ejemplos

M400 (Espera a que este vacío el buffer)

5.7.53 M401: Posiciona el servo del EndStop Z si esta presente

Si tenemos montado el EndStop del eje Z en un servo para usarlo cuando nivelemos la HeatedBed, este comando lo pondrá en posición de medir, listo para que se pueda pulsar.

Condición:

Marlin solo tiene en cuenta este comando si tiene definido el modo *ENABLE_AUTO_BED_LEVELING,* algún servo en el parámetro *SERVO_ENDSTOPS* y **no tiene habilitada** la opción Z_PROBE_SLED en el archivo Configuration.h, en caso contrario no tendrá efecto.

Uso

M401

Parámetros

No lleva parámetros.

Ejemplos

M401 (Gira el servo y posiciona el EndStop listo para usarse)

5.7.54 M402: Retrae el servo del EndStop Z si esta presente

Si tenemos montado el EndStop del eje Z en un servo para usarlo cuando nivelemos la HeatedBed, este comando lo retraerá, es decir, lo guardara. Este comando es el contrario a M401.

Condición:

Marlin solo tiene en cuenta este comando si tiene definido el modo *ENABLE_AUTO_BED_LEVELING,* algún servo en el parámetro *SERVO_ENDSTOPS* y **no tiene habilitada** la opción Z_PROBE_SLED en el archivo Configuration.h, en caso contrario no tendrá efecto.

Uso

M402

Parámetros

No lleva parámetros.

Ejemplos

M402 (Retrae el servo y "guarda" el EndStop)

5.7.55 M540: Habilita o deshabilita la parada cuando pulsa un EndStop al imprimir desde SD

Podemos configurar que cuando imprima desde un archivo en una SD, si pulsa un EndStop se para la impresión o no.

Condición:

Marlin solo tiene en cuenta este comando si tiene definido el modo ABORT_ON_ENDSTOP_HIT_FEATURE_ENABLED en el archivo Configuration_adv.h, en caso contrario no tendrá efecto.

Uso

M540 Sn

Parámetros

Sn Habilita o deshabilita la parada cuando se pulse un EndStop. Los valores aceptados son 0= deshabilitado (no parara) y 1= habilitado (se parara la impresión).

Ejemplos

M540 S0 (Deshabilita la parada en caso de que se pulse un EndStop)

5.7.56 M600: Pausa para cambiar de filamento

Tenemos la opción de indicarle a la impresora que se detenga y así poder cambiar el filamento. Es una función experimental y hay que tener precaución al usarla. Hay que tener LCD con encoder rotativo para poder controlarla, ya que una vez reemplazado el filamento, hay que decirle desde el LCD que continúe con la impresión. En el archivo Configuration_adv.h se pueden configurar todos los parámetros de este comando, no obstante junto con el comando M600 se pueden volver a enviar por si queremos modificar alguno.

Condición:

Marlin solo tiene en cuenta este comando si tiene habilitado el modo *FILAMENTCHANGEENABLE* en el archivo Configuration_adv.h, en caso contrario no tendrá efecto.

Uso

No hace falta que lleve todos los parámetros,

M600 Xnnn Ynnn Znnn E-nnn L-nnn

Parámetros

Xnnn Coordenada del eje X donde se desplazara el extrusor para cambiar el filamento. Parámetro *FILAMENTCHANGE_XPOS* en el archivo Configuration_adv.h.
Ynnn Coordenada del eje Y donde se desplazara el extrusor para cambiar el filamento. Parámetro *FILAMENTCHANGE_YPOS* en el archivo Configuration_adv.h.
Znnn Distancia extra en mm que se añadirán al eje Z antes de desplazar el extrusor a las coordenadas X e Y. Parámetro *FILAMENTCHANGE_ZADD* en el archivo Configuration_adv.h.
E-nnn Longitud en mm que se retraerá el filamento antes de desplazarse el extrusor a las coordenadas X e Y indicadas. **El numero debe ser negativo**. Parámetro *FILAMENTCHANGE_FIRSTRETRACT* en el archivo Configuration_adv.h.
L-nnn Longitud en mm que se retraerá el filamento cuando el extrusor haya llegado a las coordenadas X e Y indicadas, se usa para sacar del todo el filamento. **El numero debe ser negativo**. Parámetro *FILAMENTCHANGE_FINALRETRACT* en el archivo Configuration_adv.h.

Ejemplos

M600 (Pausa la impresión para cambiar el filamento y usa los parámetros por defecto indicados en el archivo Configuration_adv.h)

*M600 X2 Y2 Z5 E-2 L-100 (*Pausa la impresión para cambiar el filamento, retrae el filamento 2 mm, desplaza el eje Z 5mm, se mueve a las coordenadas X=2, Y=2 y finalmente retrae el filamento 100mm)

5.7.57 M605: Fija el modo de desplazamiento en carros X duales

En algunos modelos con 2 extrusores podemos tener 2 carros independientes en el eje X y así poder “aparcar” el extrusor inactivo mientras imprimimos con el otro y evitar contaminaciones de la pieza con el filamento. Además reduce el peso del carro X permitiendo impresiones más rápidas.

Podemos elegir entre varios modos de funcionamiento, en el archivo Configuration_adv.h se puede configurar el modo por defecto, pero podremos cambiarlo con este comando:

Modo 0: Control total. El software Slicer tiene el control total sobre ambos carros y puede conseguir resultados óptimos siempre que soporte el uso de carro dual (M605 S0).

Modo 1: Auto-aparcado. El firmware (Marlin) puede aparcar automáticamente los carros X y cambiar así de extrusor. No requiere soporte por parte del software Slicer (M605 S1).

Modo 2: Duplicado. El 2º carro copia todos los movimientos del primer carro, realizando así la impresión simultanea de 2 piezas iguales. El x-offset del segundo carro y el ófset de la temperatura pueden fijarse con el Gcode M605 S2 [Xnnn] [Rnnn].

Condición:

Marlin solo tiene en cuenta este comando si tiene habilitado el modo *DUAL_X_CARRIAGE* en el archivo Configuration_adv.h, en caso contrario no tendrá efecto.

Uso

No hace falta que lleve todos los parámetros, pero al menos necesita el Sn

M600 Sn Xnnn Rnnn

Parámetros

Sn Selección de modo, valores aceptados 0, 1, 2.
Xnnn Distancia X entre los 2 Hotend cuando estamos en Modo 2 duplicado. Solo se tiene en cuenta este parámetro en modo 2.
Rnnn Diferencia de temperatura en grados del segundo extrusor con respecto a la temperatura fijada en el primero, el valor puede ser positivo o negativo. Solo se tiene en cuenta este parámetro en modo 2.

Ejemplos

M605 S0 (Cambia el modo de control a “total por software Slicer”)

*M600 S2 X50 R-5 (*Cambia el modo de control a duplicado, fija el offset X entre extrusores en 50mm y el segundo extrusor estará a 5 grados menos que el primero)

5.7.58 M907: Fija la corriente de los drivers con los potenciómetros digitales

Algunas placas tiene la posibilidad de modificar la corriente de configuración de los drivers de los motores, en vez de usar un potenciómetro en los drivers que regulamos manualmente, la placa de control dispone de un potenciómetro electrónico que podemos configurar con este comando. Es necesario que la entrada correspondiente Vref de los drivers este conectada a la placa controladora. Actualmente solo esta soportada por la placa Rambo (código de placa 301) y por la placa Azteeg X3 Pro (por un canal I2C). El driver que soporta el uso de la Vref externa para el control de la corriente es el SD8825.

Según el tipo de placa que tengamos los valores de los parámetros variaran. Para una Rambo los valores están entre 0 - 255. Por ejemplo, para la Rambo el valor 135 equivale a 0,75 Amperios y el 185 equivaldría a 1 Amperio. En cambio si tenemos una placa en el que los potenciómetros digitales se controlan mediante un canal I2C, como por ejemplo la Azteeg pro, el parámetro lo introduciremos en Amperios directamente.

Este comando usa los códigos de los ejes para fijar la corriente.

Condición:

Marlin solo tiene en cuenta este comando si nuestra placa controladora tiene definido el pin *DIGIPOTSS_PIN* en el archivo pins.h o esta habilitado el modo *DIGIPOT_I2C* en el archivo Configuration_adv.h, en caso contrario no tendrá efecto.

Uso

No hace falta que lleve todos los parámetros.

M907 Xnnn Ynnn Znnn Ennn Bnnn Snnn

Parámetros

Xnnn Fija la corriente del driver correspondiente al eje X.
Ynnn Fija la corriente del driver correspondiente al eje Y.
Znnn Fija la corriente del driver correspondiente al eje Z.
Ennn Fija la corriente del driver correspondiente al extrusor 0.
Bnnn Fija la corriente del driver correspondiente al extrusor 1.
Snnn Fija la corriente de los drivers de todos los ejes y el extrusor 0.

Ejemplos

M907 X185 Y185 E135 (Fija la corriente para los drivers X Y en 185 (1 Amperio) y para el driver del extrusor en 135 (0,75 Amperios). Valores adecuados para una placa Rambo)

M907 X0.75 Y0.75 Z0.75 E1.0 (Fija la corriente para los drivers X Y Z en 0,75 Amperios y para el driver del extrusor en 1 Amperio. Valores adecuados para una placa Azteeg Pro)

5.7.59 M908: Fija directamente la corriente de los drivers con los potenciómetros digitales.

Este comando es similar al anterior M907, pero en este caso fijamos la corriente haciendo referencia directa al canal del potenciómetro digital. Es necesario revisar la documentación de nuestra placa para ver que canal correspondería con que eje. Solo valido para placas con potenciómetro digital que no este controlado mediante I2C, por ejemplo la placa Rambo. Para la placa Rambo la asignación de canales es la siguiente:

- Canal 1, extrusor 0.
- Canal 2, extrusor 1.
- Canal 4; eje Z.
- Canal 5, eje X.
- Canal 6, eje Y.

Condición:

Marlin solo tiene en cuenta este comando si nuestra placa controladora tiene definido el pin *DIGIPOTSS_PIN* en el archivo pins.h, en caso contrario no tendrá efecto.

Uso

M908 Pnnn Snnn

Parámetros

Pnnn Canal del potenciómetro digital.
Snnn Fija la corriente, valores entre 0-255.

Ejemplos

M908 P1 S185 (Fija la corriente del canal 1 o extrusor 0 en 185, es decir 1 Amperio)

5.7.60 M999: Reiniciar después de haberse detenido por error

Si hemos tenido una parada de emergencia por algún error, como por ejemplo, exceso de temperatura, pulsado un EndStop, error de comunicación, etc. Con este comando se reiniciaría la impresión donde se quedo.

Uso

M999

Parámetros

No lleva parámetros.

Ejemplos

M999 (reinicia la impresión donde se quedo al pararse)

5.7.61 CUSTOM_M_CODE_SET_Z_PROBE_OFFSET: Adicional offset al eje Z

Los valores especificados se añaden a la posición de disparo del EndStop del eje Z cuando se hace un Homing. Es decir, una vez pulsado el Homing, el eje se desplazara la distancia aquí indicada y se tomara como el 0 del eje. Este comando es similar al Gcode M206, pero en este caso esta únicamente enfocado al eje Z para ajustar la altura cuando tenemos habilitado *ENABLE_AUTO_BED_LEVELING*. Normalmente son valores negativos, ya que en realidad le estamos indicando la distancia real que hay entre el momento en el que se pulsa el EndStop y la punta del Hotend. En el archivo Configuration.h tienen que estar habilitados los CUSTOM_M_CODES, además por seguridad, se puede configurar un valor máximo y mínimo para este valor en el archivo Configuration.h.

Condición:

Marlin solo tiene en cuenta este comando si tiene habilitado los CUSTOM_M_CODES y el modo *ENABLE_AUTO_BED_LEVELING* ambos en el archivo Configuration.h, en caso contrario no tendrá efecto.

Uso

CUSTOM_M_CODE_SET_Z_PROBE_OFFSET Znnn

Parámetros

Znnn Offset en el eje Z entre el disparo del EndStop y la punta del Hotend en milímetro.

Ejemplos

CUSTOM_M_CODE_SET_Z_PROBE_OFFSET Z-3.6 (Fija el offset en el eje Z en -3,6 mm)

5.8 T: Selección de Extrusor

Cuando disponemos de más de un extrusor y enviamos un Gcode que modifique de alguna manera su estado (lo mueva, cambie temperatura, etc.), es necesario indicar a que numero de extrusor nos referimos, para ello se emplea previamente a cualquier comando que afecte al extrusor el Gcode T. Hay que tener en cuenta que el primer extrusor será el 0, el segundo el 1, etc.

Uso

Tn

Parámetros

Tn Numero de extrusor seleccionado.

Ejemplos

T0 (Selecciona el extrusor 0)
G1 E23.5 (extruye por el extrusor 0 filamento)
T1 (Selecciona el extrusor 1)
G1 E12 (extruye por el extrusor 1 filamento)

T1 (Selecciona el extrusor 1)
M600 (Pausa el extrusor 1 y saca el filamento para poder reemplazarlo)

T2 (Selecciona el extrusor 2, en caso de tener 3 extrusores)
G10 (Hace una retracción controlada del extrusor 2)

5.9 Gcodes específicos de impresoras tipo Delta

Para facilitar la calibración de las impresoras tipo Delta, se usan los siguientes Gcodes. Más información de cómo usarlos en el siguiente video.

5.9.1 M665: Configuración parámetros en impresoras tipo Delta

Con este comando podemos modificar ciertos parámetros exclusivos de las impresoras tipo Delta. Estos parámetros ya están configurados por defecto en el archivo Configuration.h especifico del modelo Delta.

Condición:

Marlin solo tiene en cuenta este comando si tiene activo el modo *DELTA* en el archivo Configuration.h, en caso contrario no tendrá efecto.

Uso

No hace falta que lleve todos los parámetros.

M665 Lnnn Rnnn Snnn

Parámetros

Lnnn Fija la longitud de las barras diagonales que dirigen el extrusor. Es la distancia entre los centros de los puntos por donde se fijan cada una de las 3 barras que posicionan el extrusor. Modifica el parámetro *DELTA_DIAGONAL_ROD* en el archivo Configuration.h.

Rnnn Fija la distancia efectiva horizontal entre los extremos de las varillas diagonales. Modifica el parámetro *DELTA_RADIUS* en el archivo Configuration.h.

Snnn Cuando hacemos un arco, se convierte en muchos segmentos rectos tangentes al centro, este parámetro fija el numero de segmentos por segundo que se imprimirán cuando hacemos un arco o circulo. Modifica el parámetro *DELTA_SEGMENTS_PER_SECOND* en el archivo Configuration.h

Ejemplos

M665 L250 R125 S200 (Fija los parámetros en: DELTA_DIAGONAL_ROD=250 DELTA_RADIUS=125 y *DELTA_SEGMENTS_PER_SECOND=200)*

5.9.2 M666: Ajuste del offset de los EndStop en impresoras tipo Delta

Con este comando podremos ir ajustando el offset de la posición de los EndStop de cada eje en impresoras tipo delta. Sirve para ajustar los 0 de cada eje, teniendo en cuenta que en las impresoras tipo Delta los EndStop no están ni al principio ni al final del recorrido del eje correspondiente.

Condición:

Marlin solo tiene en cuenta este comando si tiene activo el modo *DELTA* en el archivo Configuration.h, en caso contrario no tendrá efecto.

Uso

No hace falta que lleve todos los parámetros.

M666 Xnnn Ynnn Znnn

Parámetros

Xnnn Fija el offset del EndStop del eje X.
Ynnn Fija el offset del EndStop del eje Y
Znnn Fija el offset del EndStop del eje Z

Ejemplos

M666 X-0.2 Y-0.3 Z5.9 (Fija el offset del EndStop del eje X en -0,2 del eje Y en -0,3 y del eje Z en 2,9)

5.10 Gcodes específicos de impresoras tipo Scara

Con el fin de facilitar la calibración de las impresoras tipo Scara como la Morgan, se utilizan los siguientes Gcodes. Más información sobre como calibrar una impresora tipo Scara en http://reprap.harleystudio.co.za/. Todos estos Gcodes son específicos de impresoras tipo Scara, y por tanto Marlin solo tiene en cuenta estos comandos si tiene activo el modo *SCARA* en el archivo Configuration.h, en caso contrario no tendrán efecto.

5.10.1 M360: Mueve el brazo Theta a la posición de 0 grados

Los brazos se mueven en una posición donde el brazo de dirección Theta es paralelo al borde superior de la plataforma.

Uso

M360

Parámetros

No lleva parámetros.

Ejemplos

M360 (Pone el brazo Theta paralelo al borde superior de la plataforma)

5.10.2 M361: Mueve el brazo Theta a la posición de 90 grados

Los brazos se mueven en una posición donde el brazo de dirección Theta esta a 90 grados con respecto al borde superior de la plataforma.

Uso

M361

Parámetros

No lleva parámetros.

Ejemplos

M361 (Pone el brazo Theta a 90 grados del borde superior de la plataforma)

5.10.3 M362: Mueve el brazo Psi a la posición de 0 grados

Los brazos se mueven en una posición donde el brazo de dirección Psi es paralelo al borde superior de la plataforma.

Uso

M362

Parámetros

No lleva parámetros.

Ejemplos

M362 (Pone el brazo Psi paralelo al borde superior de la plataforma)

5.10.4 M363: Mueve el brazo Psi a la posición de 90 grados

Los brazos se mueven en una posición donde el brazo de dirección Psi esta a 90 grados con respecto al borde superior de la plataforma.

Uso

M363

Parámetros

No lleva parámetros.

Ejemplos

M363 (Pone el brazo Psi a 90 grados del borde superior de la plataforma)

5.10.5 M364: Mueve los brazos a una posición de 90 grados entre Psi y Theta

Los brazos se mueven en una posición donde los brazos de dirección Psi y Theta forman un ángulo de 90 grados entre ellos.

Uso

M364

Parámetros

No lleva parámetros.

Ejemplos

M364 (Pone los brazos Psi y Theta a 90 grados entre ellos)

5.10.6 M365: factor de escala SCARA

Ajuste del factor de escala para los ejes X, Y y Z. 100% de escala (valor por defecto) está representado por 1. Sirve para ajustar que los movimientos de cada eje correspondan con la distancia real buscada.

Uso

No hace falta que lleve todos los parámetros.

M365 Xnnn Ynnn Znnn

Parámetros

Xnnn Fija el factor de ajuste del eje X.
Ynnn Fija el factor de ajuste del eje Y
Znnn Fija el factor de ajuste del eje Z

Ejemplos

M365 X0.99 Y1.02 Z1 (fija el factor de ajuste de los ejes X=0,99 (99%) Y=1,02 (102%) y Z=1 (100%))

5.11 Gcodes de uso de la EEPROM

En la práctica cada impresora tiene una serie de parámetros físicos que deben ser persistentes pero fácilmente configurables, tales como steps/mm del extrusor, diversos valores máximos, ajustes de velocidades, etc. Estos parámetros están codificados actualmente en el firmware (Marlin), de modo que un usuario tiene que modificar, recompilar y cargar el firmware cada vez que quiera hacer un ajuste. Pero podemos almacenar en la EEPROM del micro controlador estas configuraciones y posteriormente se podrán modificar a través de algunos comandos Gcode M. La memoria EEPROM es una memoria no volátil que tienen los micro controladores y en la que podemos guardar datos que queramos conservar, aunque desenchufemos la alimentación del micro controlador. Es imprescindible para poder usar estos comandos, tener habilitado el uso de Eeprom con el parámetro *EEPROM_SETTINGS* del archivo Configuration.h, de otra manera no tendrán efecto.

5.11.1 M500: Almacena los parámetros en la EEPROM

Guarda los parámetros actuales en la memoria Eeprom del micro controlador. Los parámetro que se guardaran son los siguientes:

- Pasos por milímetro de todos los ejes. Parámetro *DEFAULT_AXIS_STEPS_PER_UNIT* en el archivo Configuration.h. Modificable con Gcode M92.
- Velocidad máxima de todos los ejes. Parámetro *DEFAULT_MAX_FEEDRATE* en el archivo Configuration.h. Modificable con Gcode M203.
- Aceleración máxima de todos los ejes. Parámetro *DEFAULT_MAX_ACCELERATION* en el archivo Configuration.h. Modificable con el Gcode M201.
- Aceleración por defecto de todos los ejes. Parámetro *DEFAULT_ACCELERATION* en el

archivo Configuration.h. Modificable con el Gcode M204.

- Aceleración del extrusor para la retracción. Parámetro *DEFAULT_RETRACT_ACCELERATION* en el archivo Configuration.h. Modificable con el Gcode M204.
- Velocidad mínima de todos los ejes mientras se imprime. Parámetro *DEFAULT_MINIMUMFEEDRATE* en el archivo Configuration_adv.h. Modificable con el Gcode M205.
- Velocidad mínima de todos los ejes cuando no se imprime. Parámetro *DEFAULT_MINIMUMFEEDRATE* en el archivo Configuration_adv.h. Modificable con el Gcode M205.
- Tiempo mínimo en microsegundos que tardara en realizarse los movimientos cuando el buffer esta vacío. Parámetro *DEFAULT_MINSEGMENTTIME* en el archivo Configuration_adv.h. Modificable con el Gcode M205.
- Velocidad mínima que no requerirá del cálculo de aceleración para los ejes X e Y. Parámetro *DEFAULT_XYJERK* en el archivo Configuration.h. Modificable con el Gcode M205.
- Velocidad mínima que no requerirá del cálculo de aceleración para el eje Z. Parámetro *DEFAULT_ZJERK* en el archivo Configuration.h. Modificable con el Gcode M205.
- Velocidad mínima que no requerirá del cálculo de aceleración para el extrusor E. Parámetro *DEFAULT_EJERK* en el archivo Configuration.h. Modificable con el Gcode M205.
- Distancia añadida al Homing en cada eje. Parámetro solo modificable con el Gcode M206.
- Temperatura por defecto para el Hotend con PLA. Parámetro *PLA_PREHEAT_HOTEND_TEMP* en el archivo Configuration.h. Modificable desde el LCD.
- Temperatura por defecto para la Heatedbed con PLA. Parámetro *PLA_PREHEAT_HPB_TEMP* en el archivo Configuration.h. Modificable desde el LCD.
- Velocidad por defecto del ventilador de capa con PLA. Parámetro *PLA_PREHEAT_FAN_SPEED* en el archivo Configuration.h. Modificable desde el LCD.
- Temperatura por defecto para el Hotend con ABS. Parámetro *ABS_PREHEAT_HOTEND_TEMP* en el archivo Configuration.h. Modificable desde el LCD.
- Temperatura por defecto para la Heatedbed con ABS. Parámetro *ABS_PREHEAT_HPB_TEMP* en el archivo Configuration.h. Modificable desde el LCD.
- Velocidad por defecto del ventilador de capa con ABS. Parámetro *ABS_PREHEAT_FAN_SPEED* en el archivo Configuration.h. Modificable desde el LCD.
- Si tenemos definido la auto nivelación de la cama guarda el offset entre la punta del extrusor y la sonda de medida. Parámetro *Z_PROBE_OFFSET_FROM*_EXTRUDER en el archivo Configuration.h. Modificable con el Gcode personalizado *CUSTOM_M_CODE_SET_Z_PROBE_OFFSET.*
- Contraste del LCD gráfico. Parámetro *DEFAULT_LCD_CONTRAST* en el archivo Configuration.h. Modificable con el Gcode M250.
- Si tenemos una impresora tipo Delta, guarda el ajuste de los EndStop de todos los ejes. Parámetro solo modificable con el Gcode M666.
- Si tenemos una impresora tipo Delta, guarda el radio Delta. Parámetro *DELTA_RADIUS* del archivo Configuration.h especifico de impresoras Delta. Modificable con el Gcode

M665.
- Si tenemos una impresora tipo Delta, guarda la distancia entre centros de las barras diagonales. Parámetro *DELTA_DIAGONAL_ROD* del archivo Configuration.h especifico de impresoras Delta. Modificable con el Gcode M665.
- Si tenemos una impresora tipo Delta, guarda el numero de segmentos por segundo. Parámetro *DELTA_SEGMENTS_PER_SECOND* del archivo Configuration.h especifico de impresoras Delta. Modificable con el Gcode M665.
- Factor Kp del control PID del Hotend. Parámetro *DEFAULT_Kp* del archivo Configuration.h. Modificable con el Gcode M301.
- Factor Ki del control PID del Hotend. Parámetro *DEFAULT_Ki* del archivo Configuration.h. Modificable con el Gcode M301.
- Factor Kd del control PID del Hotend. Parámetro *DEFAULT_Kd* del archivo Configuration.h. Modificable con el Gcode M301.
- Si tenemos una impresora tipo Scara, guarda el factor de escala para todos los ejes. Parámetro solo modificable con el Gcode M365.

Uso

M500

Parámetros

No lleva parámetros.

Ejemplos

M500 (Guarda los parámetros actuales en la Eeprom)

5.11.2 M501: Revierte los parámetros a los valores guardado en la EEPROM

Establece los parámetros activos a los almacenados en la EEPROM. Esto es útil para revertir parámetros después de experimentar con ellos.

Uso

M501

Parámetros

No lleva parámetros.

Ejemplos

M501 (Revierte los parámetros con los valores guardados en la Eeprom)

5.11.3 M502: Restablece los parámetros a los predeterminados o "ajustes de fábrica".

Restablece todos los parámetros ajustables a sus valores predeterminados tal como se han fijado en Marlin en el momento de compilarlo. Esto no restablece los valores de los parámetros almacenados en la EEPROM, solo modifica los valores almacenados en la memoria RAM, por lo que se debe seguir con M500 si quieres cambiar definitivamente los valores almacenados en la memoria EEPROM.

Uso

M502

Parámetros

No lleva parámetros.

Ejemplos

M502 (Revierte los valores con los escritos en Marlin cuando se compilo)

5.11.4 M503: Saca por la consola serie los valores de los parámetros

Imprime vía consola serie o Pronterface los valores actuales de los parámetros en memoria RAM, no los almacenados en la EEPROM. Si queremos ver los guardados en la EEPROM, es necesario revertir primero los valores con un comando M501.

Uso

M503

Parámetros

No lleva parámetros.

Ejemplos

M503 (Saca por la consola serie o Pronterface los valores almacenados en la memoria, no en la Eeprom)

5.12 Gcodes de uso de la tarjeta SD

Comandos de control de la tarjeta SD cuando tenemos disponible un lector en la placa controladora o en el LCD. Es preciso haber habilitado el parámetro *SDSUPPORT* en el archivo Configuration.h para que se tengan en cuenta todos estos comandos, de otra forma no tendrán efecto.

5.12.1 M20: Lista de archivos en la tarjeta SD

Lista por el puerto serie (consola o Pronterface) los archivos disponibles en la carpeta raíz de la tarjeta SD.

Uso

M20

Parámetros

No lleva parámetros.

Ejemplos

M20 (Saca por la consola serie o Pronterface la lista de archivos disponibles en la SD)

5.12.2 M21: Inicializa la tarjeta SD

Se inicializa la tarjeta SD, no es un formateo de la tarjeta, solo hace que este disponible para su uso. Si una tarjeta SD se carga cuando la impresora está encendida, esto sucederá de forma predeterminada, en caso contrario habrá que usar este comando.

Uso

M21

Parámetros

No lleva parámetros.

Ejemplos

M21 (Inicializa la tarjeta SD)

5.12.3 M22: Expulsa la tarjeta SD

Expulsa la tarjeta SD del sistema, a continuación se podrá quitar físicamente de forma segura, esto es necesario para evitar bloqueos o "cuelgues" del sistema.

Uso

M22

Parámetros

No lleva parámetros.

Ejemplos

M22 (Expulsa la tarjeta SD)

5.12.4 M23: Selecciona un archivo en la SD

Selecciona el archivo pasado como parámetro y se queda listo para imprimir.

Uso

M23 filename.gco

Parámetros

Filename.gco Nombre del archivo

Ejemplos

M23 circulos.gco (Abre el archivo círculos.gco y lo deja listo para imprimir)

5.12.5 M24: Inicia / reanuda la impresión desde un archivo en la SD

Inicia o reanuda la impresión desde un archivo seleccionado en la SD con el Gcode M23.

Uso

M24

Parámetros

No lleva parámetros.

Ejemplos

M23 circulos.gco
M24 (Inicia la impresión del archivo círculos.gco seleccionado en el comando anterior)

5.12.6 M25: Pausa la impresión desde un archivo en la SD

Pausa la impresión en la posición actual dentro del archivo seleccionado con el Gcode M23.

Uso

M25

Parámetros

No lleva parámetros.

Ejemplos

M25 (Pausa la impresión)

5.12.7 M26: Se sitúa en la posición en bits del archivo seleccionado en la SD

Se sitúa en la posición en bits (pasada como parámetro) en el archivo previamente seleccionado con el Gcode M23.

Uso

M26 Snnn

Parámetros

Snnn Posición en bits del archivo.

Ejemplos

M26 S12052 (Se sitúa en el bit 12.052 del archivo previamente abierto)

5.12.8 M27: Informe de estado de la impresión desde la SD

Imprime por consola o Pronterface el informe de estado de la impresión desde la SD.

Uso

M27

Parámetros

No lleva parámetros.

Ejemplos

M27 (Saca por la consola serie o Pronterface el informe de estado de la impresión)

5.12.9 M28: Comienza la escritura de un archivo en la tarjeta SD

Se crea el archivo especificado pasado como parámetro (o se sobrescribe si existe) en la tarjeta SD y todos los comandos siguientes enviados a la impresora se escriben en ese archivo. Es necesario terminarlo con un Gcode M29 para cerrar el archivo.

Uso

M28 filename.gco

Parámetros

Filename.gco Nombre del archivo que se creara.

Ejemplos

M28 cuadrados.gco (crea el archivo cuadrados.gco (o lo sobrescribe) y se prepara para escribir en el)

5.12.10 M29: Deja de escribir en el archivo de la tarjeta SD

Cierra el archivo abierto con el Gcode M28 y todos los comandos siguientes serán enviados a la impresora que los ejecutara de forma normal.

Uso

M29

Parámetros

No lleva parámetros.

Ejemplos

M29 (Cierra el archivo abierto con M28)

Ejemplo ampliado

M28 prueba.gco (crea el archivo prueba.gco y se prepara para escribir en el)

G28 (Lo escribe en el archivo)
G1 X2 Y3 (Lo escribe en el archivo)
G1 X25 (Lo escribe en el archivo)
G1 Y27 (Lo escribe en el archivo)
M104 S200 (Lo escribe en el archivo)

M29 (Cierra el archivo prueba.gco)

G28 (La impresora hará un Homing)

5.12.11 M30: Elimina un archivo de la tarjeta SD

Elimina el archivo pasado como parámetro.

Uso

M30 filename.gco

Parámetros

Filename.gco Nombre del archivo que se borrara.

Ejemplos

M30 cuadrados.gco (Borra el archivo cuadrados.gco)

5.12.12 M32: Selecciona un archivo en la SD y empieza a imprimirlo

Selecciona el archivo pasado como parámetro y empieza a imprimirlo. Este comando es equivalente a hacer primero un Gcode M23 (o M26 Snnn) seguido de un M24. Si este comando se usa dentro de un archivo .gcode es necesario incluir después del nombre del archivo, el carácter "#", para que no siga almacenando comandos en el buffer, ejemplo M32 filename.gco#. También es posible indicar una posición en bits a partir de donde se empezara a imprimir, en este caso es equivalente al comando M26 Snnn seguido de un M24.

Otra posibilidad es que se use el Gcode M32 dentro de un archivo raíz.gcode, y necesitemos que una vez abierto e impreso el archivo que iba como parámetro, siga ejecutando los comandos dentro del archivo raíz, sería equivalente a hacer un #include. En este caso se usara el parámetro P que le dice al Gcode M32 que está en forma de "procedimiento" es decir, ejecuta y retorna. Con un ejemplo del uso del parámetro P lo veremos mas claro

Archivo "inicial.gcode"

G28 (Homing)
M109 S200 (Enciende el Hotend y fija la temperatura en 200 grados)
M32 P !circulos.gcode# (Abre e imprime el archivo circulos.gcode)
M109 S0 (Apaga el Hotend)
G28
…
…

En este caso si abrimos el archivo inicial.gco y lo imprimimos el resultado será que hará primero un G28 seguido del M109 S200, a continuación abrirá e imprimirá el archivo círculos.gco. Una vez a terminado de imprimir el archivo círculos.gco, retorna al archivo inicial.gco y ejecutara el comando siguiente que es M109 S0, seguido del G28 y consecutivos.

Usos

M32 filename.gco(#)
M32 Snnn !filename.gco(#)
M32 P !filename.gco#

Parámetros

Filename.gco Nombre del archivo que se abrirá e imprimirá. Si va dentro de un archivo .gco ira seguido del carácter "#". **Si va a continuación de otro parámetro, llevara el carácter "!" antes.**
Snnn Posición en bits del archivo a partir de donde se empezara a imprimir.
P Indica que es una llamada dentro de un archivo, y retornara al punto donde se inicio el comando M32

Ejemplos

M32 circulos.gco (Abre e imprime el archivo círculos.gco)
M32 S12034 !circulos.gco (Abre el archivo círculos.gco por el bit 12.034 y lo imprime a partir de ese punto)

5.12.13 M928: Comienza la escritura de un archivo en la tarjeta SD, ejecutando los comandos

Se crea el archivo especificado pasado como parámetro (o se sobrescribe si existe) en la tarjeta SD y todos los comandos siguientes enviados a la impresora se escriben en ese archivo y a la vez se ejecutan. Este comando es similar al Gcode M28, solo que en este caso además de escribir los Gcodes en el archivo los ira ejecutando. Es necesario terminarlo con un Gcode M29 para cerrar el archivo.

Uso

M928 filename.gco

Parámetros

Filename.gco Nombre del archivo que se creara.

Ejemplos

M928 cuadrados.gco (crea el archivo cuadrados.gco (o lo sobrescribe) y se prepara para escribir en el, conforme guardemos comandos los ira ejecutando)

5.13 Extrusor tipo Baricuda

Si hemos instalado un extrusor de pasta tipo BariCUDA, con estos comandos controlamos las válvulas de aire necesarias para su funcionamiento. Más información http://www.thingiverse.com/thing:26343 y en https://github.com/jmil/BariCUDA. Es preciso haber habilitado el parámetro *BARICUDA* en el archivo Configuration.h para que se tengan en cuenta todos estos comandos, de otra forma no tendrán efecto.

5.13.1 M126: Abre la válvula de extrusión

Abre la válvula neumática de presión que empuja el embolo del extrusor y comenzara a extruir azúcar, chocolate o lo que tenga cargado.

Uso

M126

Parámetros

No tiene parámetros.

Ejemplos

M126 (Abre la válvula del extrusor)

5.13.2 M127: Cierra la válvula de extrusión

Cierra la válvula neumática que empuja el embolo del extrusor.

Uso

M127

Parámetros

No tiene parámetros.

Ejemplos

M127 (Cierra la válvula del extrusor)

5.13.3 M128: Regula la presión de aire del extrusor

Valor PWM para controlar la presión neumática que se aplica al embolo del extrusor, usa un transductor de "electricidad a presión" para regular la presión del aire de alimentación. Lleva un parámetro para ajustar la presión entre 0 y 255, siendo 255 la máxima presión a la que este configurado el transductor.

Uso

M128 Snnn

Parámetros

Snnn Valor entre 0-255 para ajustar la presión del aire. Si no se incluye este parámetro, se ajustara al valor máximo 255

Ejemplos

M128 S162 (Se ajusta la presión a la máxima configurada en el transductor)

5.13.4 M129: Pone la presión de aire del extrusor a 0

Establece la presión neumática que se aplica al embolo del extrusor en 0, es decir, sin presión.

Uso

M129

Parámetros

No tiene parámetros.

Ejemplos

M129 (Pone la presión a 0)

5.14 Sensor de diámetro del filamento

Podemos instalar un sensor de diámetro para ajustar el flujo de material según varíe el diámetro. La medida que tiene que proporcionar el sensor será un voltaje de 0-5V equivalente al diámetro, por ejemplo con un filamento de 1,75mm de diámetro el sensor dará una lectura de 1,75V. Es preciso haber habilitado el parámetro *FILAMENT_SENSOR* en el archivo Configuration.h para que se tengan en cuenta todos estos comandos, de otra forma no tendrán efecto.

5.14.1 M404: Diámetro nominal del filamento

Si se acompaña del parámetro Nnnn, fija el diámetro del filamento en mm. En caso de ir sin parámetro, muestra por consola o Pronterface el valor del diámetro actual. Modifica el parámetro *DEFAULT_NOMINAL_FILAMENT_DIA* configurado en el archivo Configuration.h

Uso

M404 Nnnn

Parámetros

No hace falta que lleve todos los parámetros.

Nnnn Diámetro del filamento

Ejemplos

M404 N1.75 (Fija el diámetro del filamento en 1,75 mm)

M404 (Saca por la consola el valor del diámetro actual)

5.14.2 M405: Activa el control del diámetro del filamento mediante el sensor

Enciende a activa el control del diámetro del filamento mediante el sensor. A partir de este momento tendrá en cuenta el diámetro del filamento para ajustar el flujo de extrusión. Se puede adjuntar un parámetro para fijar la distancia en cm que hay entre el sensor que mide el diámetro y la mitad del Hotend. Esto lo emplea Marlin para saber en qué momento justo llega al extrusor el filamento medido y poder ajustar el flujo de material.

Uso

No hace falta que lleve todos los parámetros.

M405 Dnnn

Parámetros

Dnnn Distancia en cm desde el sensor al centro del Hotend. Modifica el parámetro *MEASUREMENT_DELAY_CM* en el archivo Configuration.h.

Ejemplos

M405 D15 (Activa el control del diámetro y fija la distancia entre el sensor y el Hotend en 14 centímetros)

M405 (Activa el control del diámetro)

5.14.3 M406: Apaga el control del diámetro del filamento mediante el sensor

Apaga o desactiva el control del diámetro del filamento mediante el sensor.

Uso

M406

Parámetros

No tiene parámetros.

Ejemplos

M406 (Apaga el control del diámetro del filamento)

5.14.4 M407: Muestra el diámetro medido por el sensor

Saca o imprime por la consola o Pronterface el diámetro actual medido por el sensor.

Uso

M407

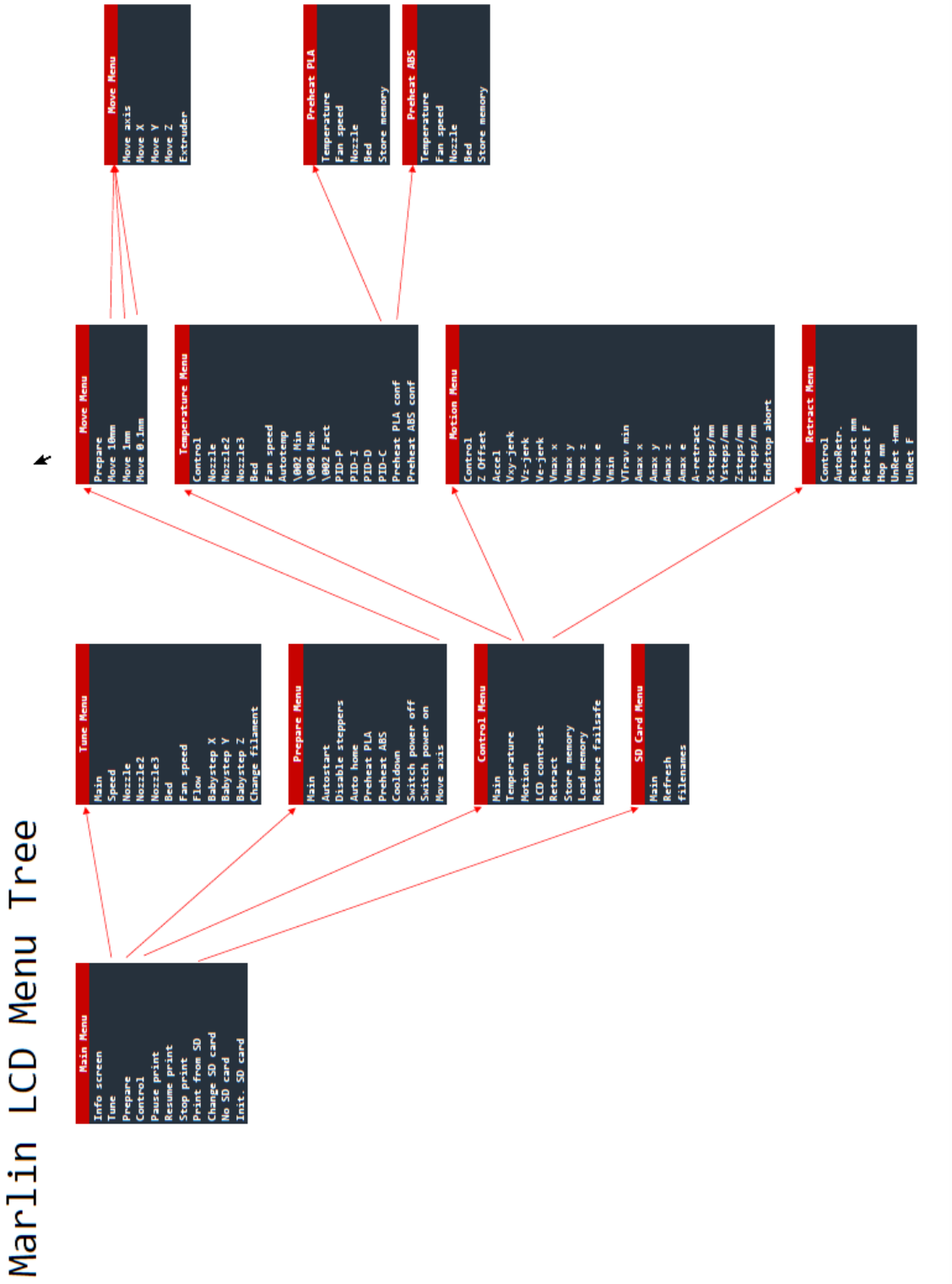
Parámetros

No tiene parámetros.

Ejemplos

M407 (Saca por consola el diámetro actual medido por el sensor)

6. Menus del LCD



7 Glosario

Este glosario es una transcripción del que se encuentra en la página http://www.reprap.org/wiki/Clone_wars:_Glosario/es

7.1 Glosario General

RepRap

Es un acrónimo que significa ***Rep****licating* ***Ra****pid* ***P****rototyper*, esto es, prototipador replicante rápido. Es un conjunto de impresoras 3D cuyas piezas en principio se pueden imprimir utilizando otra impresora 3D. Es una iniciativa Open Hardware para la creación de máquinas manufacturadas libres y que se pueden construir en casa. Por lo general cuando nos referimos a una RepRap nos referimos a una impresora RepRap.

El Logotipo de la RepRap es un cilindro con forma de gota. Este cilindro tiene esta forma ya que la impresora puede imprimir pendientes sobresalientes de como máximo 45º, y esta es la forma más fácil, y sin apoyos de imprimir un hueco cilíndrico horizontal en una pieza.

CNC (Control Numérico por Computadora)

Se refiere a cualquier sistema de control de máquinas herramienta por ordenador. Una RepRap es una máquina CNC cuya herramienta es un extrusor de plástico. Con las debidas adaptaciones se puede hacer que la RepRap en vez de extruir plástico, tenga una fresadora, un láser, o incluso que extruya chocolate o azúcar.

Open Hardware

Se refiere a hardware libre (al igual que Open Software se refiere a software libre). Se refieren a piezas de maquinaria, electrónica, mecánica que no están sujetas por un copyright, patente, o registro de cualquier tipo sino por una licencia de uso, distribución y modificación libre. Esto implica que los productos Open Hardware tendrán planos, diagramas a disposición del usuario, y que se podrán replicar sin violar ninguna ley, al igual que modificar o mejorar.

Logotipo del movimiento Open Hardware

Arduino y Arduino Mega

Es una placa de circuito para la realización de prototipos basada en hardware libre. El Arduino Mega es una versión del Arduino con más pines (esto es más salidas digitales), mas memoria... que el Arduino estándar (Arduino UNO). El Arduino Mega se puede utilizar para manejar la impresora RepRap.

Arduino Uno

Shield (referido a un Arduino)

Es una placa de circuito diseñada para ponerse encima de un Arduino o Arduino Mega (o alguna otra variante). Se llaman *shields* (escudo) porque cubren la Arduino, aunque no sea esa su función.

La utilidad de estas placas es extender las capacidades del Arduino de forma sólida y permitiendo que esta funcionalidad extra se pueda poner y quitar con libertad y así utilizar el Arduino para más proyectos. Normalmente estas placas permiten conectarse unas encima de otras, aunque no todas tienen esta capacidad.

Atmel

Es una empresa que fabrica los microprocesadores que, seguramente, manejarán nuestras impresoras. Es también la empresa que fabrica el microprocesador de las placas Arduino.

Firmware

Es el programa que se instala en el microprocesador y permite la comunicación de la impresora con el ordenador y también controlan los procesos de la impresora.

Host

Es el programa que se instala en el ordenador y que permite la comunicación con la impresora y enviar los datos a la misma para que esta imprima las piezas.

Gcode

Es el lenguaje estándar para las máquinas CNC y que también comparten las RepRap. Es lo que se manda a la impresora para que esta realice los movimientos necesarios e imprima las piezas.

Laminador (SkeinForge / Slicer / Cura)

Son programas que convierten el modelo 3D (en formato STL) en código Gcode para que la impresora la imprima. Su funcionamiento se basa en hacer rodajas de la pieza y, sobre estas rodajas, calcular los movimientos de la impresora para que imprima cada una de ellas.

En la wiki hay una guía sobre los numerosos parámetros de Skeinforge. También existe una guía sobre el uso de Cura.

Aquí tenéis un manual completo de Cura.

7.2 Tipos de Impresora

Prusa Mendel

Son modificaciones sobre la impresora Mendel original llevadas a cabo por Joseph Prusa, cuyo objetivo era la generación de una impresora estable, barata y muy fácil de construir. Existen dos iteraciones (la Prusa Mendel original y la Prusa Mendel Iteración II) y se está desarrollando una tercera (la Prusa Mendel Iteración III). Las dos primeras iteraciones tienen forma triangular, aunque la tercera es más recta.

Existe información sobre los 3 modelos existentes en la wiki:

- Prusa Mendel Original
- Prusa Mendel Iteración 2
- Prusa Mendel Iteración 3 (en desarrollo)
- Prusa Mendel Iteración 2 Mold (en desarrollo)

Clon R2D2, modelo de la Prusa Mendel It1

PrintrBot

PrintrBot es una impresora que también sigue la filosofía RepRap, y diseñada por Brook Drumm y financiada con un proyecto en Kickstarter. Es una impresora cuadrada que es completamente escalable. Esto significa que si compras barras más grandes puedes hacer una impresora capaz de imprimir piezas más grandes (ya que aumentas el área de impresión).

Se puede ver más información sobre la PrintrBot en esta wiki.

Adicionalmente, no se debe confundir con PrintBot (sin la r) también puede referirse a un robot compuesto de piezas impresas (Juan González suele referirse a ellos así).

Impresora PrintrBot Lucy

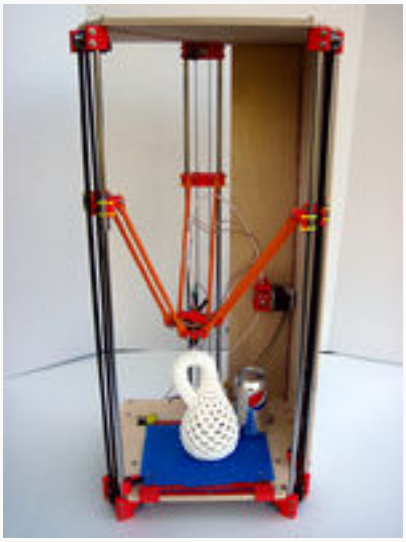
Rostock (Delta Printer)

Es una impresora también de 3 ejes (más extrusor) creada por Johann, pero con la propiedad de que los 3 ejes son verticales y están separados 120 grados cada uno, dando un sistema de coordenadas distinto, pero un resultado muy interesante. Es una impresora muy alta (pero su

espacio de impresión no llega a ser tan alto), y requiere tener el mecanismo del extrusor separado del HotEnd para evitar problemas con la inercia del soporte (el motor pesa algo menos de medio kilo, y desestabilizaría la impresión).

Es una impresora en desarrollo y no está del todo terminada. Se puede ver más información en su blog aquí.

Aviso: tal y como sugiere el autor, esta impresora no es para principiantes (especialmente porque está en desarrollo) y si se quiere montar es muy recomendable tener una impresora ya montada (ya que se requerirán cambios de piezas debidos a mejoras en el diseño del autor). El propio autor ofrece su ayuda a la hora de encontrar los componentes para montar la impresora.



Prototipo original de Rostock

Modelos no RepRap (Generación 0)

Son modelos construidos con una carcasa de madera contrachapada cortada con láser y que los venden las empresas *MakerBot*, *MakeGrear*... por algo más de 1000 euros. Son modelos con piezas no imprimibles y por tanto no son replicables ni pertenecen a las RepRap, pero nos permiten empezar a construir las mismas ya que nos imprimen las piezas de estas.

7.3 Problemas en la impresión

Backslash (histéresis)

Es un problema que puede ocurrir en una impresora RepRap (y en general en máquinas aunque es signo de que algo no funciona) producido cuando, al cambiar de sentido el giro del motor (paso a paso) el motor gira un poco antes de que la correa que este mueve reaccione (se tense) y, por tanto, se pierden pasos (la máquina cree que está en una posición en la que no está).

Esto se soluciona tensando las correas (no demasiado, pero si lo justo), y utilizando poleas de metal o de SLS.

Pérdida de pasos (en un motor)

Esto sucede cuando el motor trata de girar, pero se encuentra con mucha resistencia y no termina de dar el paso volviendo al punto de partida. Esto puede suceder por atascos (por ejemplo, si los finales de carrera no están operativos y se llega al final de las barras) o porque el motor no dispone de la corriente necesaria para vencer el par resistente. En el primer caso el motor hará un ruido fuerte y repetitivo, bastante alarmante. En el segundo el ruido será menor porque la fuerza implicada es menor.

Warping (curvatura de la piezas)

Problema en el que las piezas se despegan de la cama/base de impresión y se curvan hacia arriba. El material ABS es más susceptible a éste problema. La heatbed (cama caliente) es una mejora para evitar éste problema manteniendo las piezas pegadas a la base por viscosidad del ABS a una temperatura alta pero menor que la temperatura de extrusión. El ABS requiere mayor temperatura para la cama que el PLA. Hay otros métodos (adicionales) para minimizar éste efecto utilizando superficies más rugosas en la cama o elementos viscosos como laca o "ABS juice".

7.4 Material de impresión

ABS (Acrilonitrilo Butadieno eStireno)

Es un tipo de plástico que funde a 215ºC más o menos. Es flexible (hasta cierto punto) y puede venir en diferentes colores. Se fabrica a partir del petróleo y es seguro para la manipulación de alimentos (con ellos se hacen las piezas de LEGO). Sin embargo, los fabricantes no garantizan que los colorantes que usan en sus productos sean seguros para dicha manipulación de alimentos por lo que no es recomendable usar este plástico para ese fin a no ser que no tenga colorante.

PLA (PolyLactic Acid, Ácido Poliláctico)

Es otro tipo de plástico usado en las RepRap que funde a 185ºC (mas o menos). Es menos flexible, pero desprende menos olores. Además, su origen no es el petróleo y es biodegradable. También es seguro para la manipulación de alimentos pero no se recomienda su uso para dicho fin si el PLA posee colorantes.

7.5 Procesos de impresión

Sliceing o Laminado

Por el momento es un paso previo a la impresión que todas las impresoras 3D siguen, que consiste en laminar la pieza (cortarla en rodajas) las cuales, apiladas, formarán la pieza impresa final.

FDM (Fused Deposition Modeling)

Modelado por deposiciones fundidas, es la forma que tienen la mayoría de las RepRap de imprimir. Funden plástico y lo depositan capa a capa hasta formar la pieza. Modelado por deposición fundida en wikipedia

SLS (Selective Laser Sintering)

Síntesis Selectiva Láser, es un proceso similar al FDM de las RepRap pero usado a un nivel más profesional que utiliza un láser de alta potencia para fundir polvo de un material (como plástico, metal, cristal o cerámica) a una masa que formará el modelo 3D deseado.

DLP (Digital Light Processing)

Es un método alternativo al de las RepRap utilizado en impresoras como la B9Creator de Michael Joyce que hace que una resina líquida se endurezca y solidifique mediante la proyección de un haz de luz. Esto permite mucha más precisión, pero se reduce notablemente el tamaño posible de la pieza y es mucho más caro de construir.

7.6 Otros aspectos de la impresión

Raft (balsa)

Es una opción de Skeinforge por la cual se trazan líneas paralelas separadas en la placa caliente y sobre estas otras perpendiculares (como si fuera una balsa de un náufrago o una malla para filtrar) y sobre la cual se imprime la pieza. Sirve para facilitar la separación entre la pieza y la placa caliente (aunque cuesta separar la balsa de la pieza) y para evitar problemas de combado (que la base de la pieza se arrugue).

Skirt (falda)

Antes de imprimir la pieza (o la balsa) imprime un perímetro al rededor de la misma. La única utilidad de esto es darle tiempo al extrusor de que empiece a extruir correctamente antes de empezar a imprimir la pieza (o la balsa).

ABS juice

Solución de ABS con acetona que aplicada sobre la heatbed crea una película que mejora la adhesión a la cama caliente de la pieza a imprimir. Utilizado para minimizar el temido "warping".

7.7 Medidas

M8, M4, M3 (Rosca)

Se leen métrica 8, métrica 4... se refieren al estándar ISO con el mismo nombre. A efectos prácticos, una varilla M8 tendrá 8mm de diámetro (sin contar los dientes) y 1.25mm de paso (la distancia entre dos picos de la rosca).

NEMA 17 (Dimensiones de la base del motor)

Es un estándar (NEMA significa National Electrical Manufacturers Association, Asociación de Fabricantes Nacionales Eléctricos, de EEUU) que indica el tamaño de la carcasa y disposición de los agujeros de un motor (paso a paso). Referirse a un motor NEMA 17 es referirse a un motor que tiene 42 milímetros de base y tiene los agujeros (M3) sobre esta base separados 31mm.

El número 17 viene de que 42mm en pulgadas equivalen a 1.7".

En nuestras RepRap usaremos sobre todo motores NEMA 17, aunque, adaptando las piezas de plástico, se pueden usar motores de mayores tamaños, como los motores NEMA 22.

T5, T2.5 (Correas de sincronización y poleas)

Es un estándar de correas de sincronización (timing belt). Son correas con dientes cuya separación en milímetros es la indicada por el numero que indica su referencia (la T5 tiene 5mm entre un punto de un diente y el mismo punto del siguiente diente). Están diseñadas para realizar movimientos constantes en una dirección (y la RepRap suele cambiar bastante de dirección), pero funcionan suficientemente bien. La forma de los dientes es cuadrada.

A su vez si se usa un estándar de correas, se ha de usar ese mismo estándar para las poleas que muevan dichas correas, de esta forma garantizamos el engrane.

GT2 (Correas de sincronización y poleas)

Es otro estándar de correas de sincronización (timing belt) pero cuyos dientes son redondos en vez de rectos permitiendo el ajuste a movimientos no constantes. Tienen mejor precisión que las T2.5 para movimientos en ambas direcciones. Son mejores para las RepRap, pero son más caras.

7.8 Piezas de la impresora

Vitaminas

Se refiere a cualquier pieza necesaria en la impresora pero que no es imprimible. Esto es los motores, varillas de metal... El objetivo de RepRap es reducir al máximo las vitaminas (aunque a veces conviene añadir alguna, como los rodamientos lineales en la Prusa).

Motor Paso a Paso (Bipolar)

Es un tipo de motor eléctrico en cuyo interior existen dos bobinas eléctricas que conectadas alternativamente producen el giro de un rotor (el movimiento del eje). Tienen la propiedad de que al activar una bobina el rotor sólo gira un ángulo muy pequeño llamado paso o *step*. Al activar la otra bobina (desactivando la anterior) se mueve otro paso, y así avanzan. Permiten controlar la posición ya que puedes controlar el número de pasos que quieres que avancen los motores. Estos motores son usados en aparatos que requieren control de la posición, como las impresoras de inyección de tinta, o nuestras RepRap.

Adicionalmente conectando las bobinas a la vez se consigue que el rotor se quede en una posición intermedia. Si variamos la intensidad de una bobina frente a la otra podemos conseguir más posiciones intermedias. A esto se le llama microsteping.

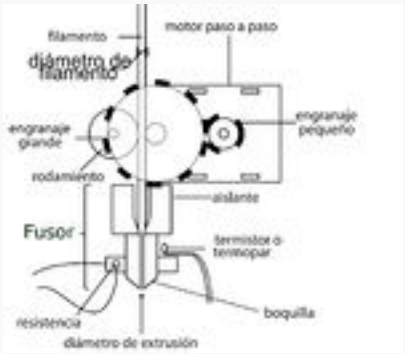
Existen motores paso a paso monopolares (de él solo salen 2 cables, aunque tienen todavía dos bobinas, pero se alternan internamente), o con mas polos (bobinas), cada par de cables que salen de un motor se llaman polo).

Motor Paso a Paso Nema 17 (12V, 2.5A)

Extrusor

Mecanismo que se encarga de extruir el filamento de plástico con el que se imprimen las piezas, esto es, introducir un volumen con una determinada geometría (en este caso filamento de 3mm de diámetro o de 1.75mm de diámetro) y sacar un volumen (el mismo mas la dilatación térmica) con una geometría distinta (en este caso un filamento derretido de 0.5mm de diámetro o menos). Las extrusiones pueden hacerse en frío (sin el material fundido) pero en el caso de la RepRap (y en general de plásticos) se ha de calentar hasta que fluya.

El extrusor de una RepRap se compone de un motor (paso a paso) que, mediante unos engranajes, fuerza al filamento de 3mm (o 1.75mm) a entrar a un fusor (el *hotend*) que derrite el plástico y, por la propia presión que ejerce, es forzado a fluir por una abertura en el fusor (*nozzle*, o aguja) que permite que salga esa "extrusión".



Modelo de Extrusor

Hobbed Bolt (Tornillo Moleteado)

Hobbed Bolt (bolt es tornillo en inglés) es un tornillo M8 que tiene una muesca en el centro que presenta unos dientes paralelos al eje del tornillo. Este tornillo sirve para empujar el filamento hacia el mecanismo del extrusor (es lo que, a través de los engranajes, hace girar el motor).

Hobbed Bolt

Hobbed Pulley (Polea Moleteada)

Hobbed Pulley (puley es polea en inglés) Se trata de una polea dentada con un tornillo que la sujeta al eje del motor. Se utiliza para empujar el filamento hacia el extrusor. Este sistema de empuje recibe el nombre de Direct Drive.

Hobbed Pulley

HotEnd (Fusor)

Literalmente Final Caliente en inglés. Se refiere a la pieza que sobresale hacia abajo del extrusor y cuya finalidad es fundir el filamento y terminar de extruirlo. El Hotend se compone de un tubo de latón hueco recubierto de un tubo de aislante térmico (PFTE por lo general), un bloque de latón o aluminio con dos agujeros, uno para la resistencia que calentará el sistema, y el otro para poner un termistor para medir la temperatura, y finalmente una rosca (generalmente) donde se introduce el *nozzle* o boquilla. El bloque de material almacena el calor generado por la resistencia para fundir el plástico.

Hotend J-Head

Nozzle (Boquilla)

Significa boquilla (aunque también se le puede llamar aguja), es una pieza situada en el extremo inferior del *HotEnd* que tiene forma cónica por lo general y tiene un agujero pequeño que determina el diámetro del plástico extruido.

Este agujero suele ser de entre 0.5mm y 0.25mm.

HotEnd Budas (Budaschnozzle)

Es un tipo de HotEnd de mucha calidad vendido por LulzBot. Tiene un alto precio, pero pagas por la calidad del producto.

Se puede comprar en la tienda de LulzBot directamente Aquí (USA) o en 3dprintershop (Europa).

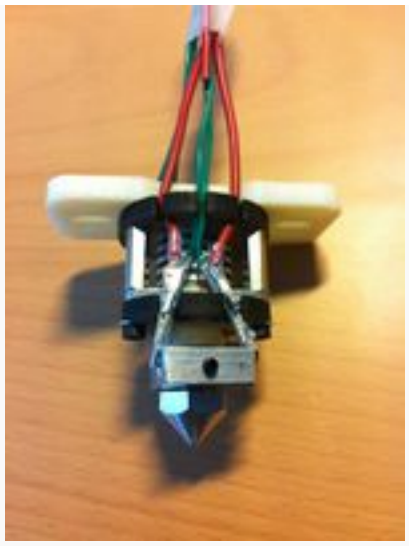
HotEnd Farynozzle

El Farynozzle es un HotEnd derivado del Budaschnozzle 1.2 de Lulzbot, al cual se le han realizado mejoras para aumentar su calidad, vida útil y solventar uno de los problemas más comunes en los hotends, el sobrecalentamiento. Esto se evitó mediante la inclusión de un disipador de aletas de aluminio, esto es una característica peculiar de este diseño.

Está fabricado con materiales de alta calidad tales como PEEK, PTFE y aluminio 2030. La fabricación se realiza íntegramente con la ayuda de empresas españolas. Esta iniciativa surge a petición de la comunidad para no depender de proveedores de piezas extranjeros evitando los molestos costes de aduanas, y fomentar la actividad local.

Hay más información disponible sobre este hotend en Farynozzle.com

Si quieres fabricar uno, tienes los planos disponibles libremente en github



Hotend Farynozzle

Heatbed (Cama o base caliente)

Significa placa (cama) caliente. En ella reposan las piezas al ser impresas. Tiene que estar caliente (100ºC para el plástico ABS, 60ºC para el plástico PLA) para evitar que la pieza tenga un gradiente de temperaturas muy elevado durante la impresión (parte de abajo fría parte de arriba caliente) que generaría deformaciones en la pieza. Es posible que no se necesite para imprimir PLA (porque es térmicamente estable, esto es, no varía mucho su dimensión con la temperatura), pero ayuda en cualquier caso.

Heated Bed

Pololu (Controlador de los motores)

Es un controlador (también llamado *driver*) en forma de chip pequeñito que controla el funcionamiento de los motores paso a paso. Utiliza un chip Allegro A4988, que permite controlar el paso de los motores (paso a paso) y también permite un microstepping de 1/2, 1/4, 1/8 o 1/16.

StepStick (Controlador de los motores)

Es un controlador de los motores paso a paso totalmente compatible con los Pololu. Esto implica que donde pones un Pololu puedes poner con facilidad un StepStick. La diferencia que tienen son el precio (el StepStick es más barato) y el grosor de la capa de cobre (en el Pololu es mayor, permitiendo que pasen más amperios sin calentarse tanto la placa, pero bien configurado no es necesario, y se puede poner un ventilador). Los StepStick funcionan, en definitiva, muy bien.

Aviso: A veces venden unos controladores llamados *Pololu/StepStick Compatible* (RepRap.me). Esto puede ser una pequeña estafa. Los controladores de los motores paso a paso que utilizamos permiten hacer microstepping, esto es que se muevan pasos no enteros (1/2, 1/8). Los Pololu y StepStick permiten un máximo de 1/16 (que está muy bien), pero algunos de estos "clones" solo permiten un máximo de 1/8. En principio se pueden usar (la diferencia son del orden de 100 micras), pero tienen mayor precisión los otros.

Controlador de voltaje/intensidad (referido a los Pololu/StepStick)

Es un pequeño potenciómetro (resistencia variable) situado en el chip del controlador que permite controlar la intensidad que irá a los motores. Esto permite no calentar demasiado los controladores y los motores, y reducir el gasto energético de la máquina. Una buena configuración de intensidad nos ahorrará energía y sudores en verano, además de ser más seguro. Por lo general, y según Obijuan, la configuración óptima para motores de 2.5A de intensidad nominal (esto es máxima), es de 0.4A por motor, exceptuando el extrusor que tiene que ir al doble (0.8A). De esta forma conseguiremos que se caliente muy poco la placa y que ahorremos energía.

Es importante tener en cuenta que los StepStick, al tener solamente 2 capas de cobre (respecto a las 4 del Pololu) disipan menos calor, y por ello están diseñados para limitar la corriente a un máximo de 1 amperio. Esto no es debido al chip en si (que es el mismo en StepStick y Pololu) sino a las dos resistencias S1 y S2 que miden la corriente y al potenciómetro instalado [Esquemático de la PCB]. No es un problema para la RepRap ya que los Pololu se suelen regular a media potencia, es decir 1A y los StepStick a tope (1A). Es posible, sin embargo, hacer trabajar a un Stepstick por encima de 1A, pero siempre con refrigeración por ventilador.

Los Pololu y los StepStick vienen con su regulador de corriente, pero los *compatibles* no tienen porque.

Controladores de los motores paso a paso Stepstick con sus correspondientes disipadores de calor. Se pueden ver unos circulitos redondos a un lado del disipador que son los reguladores de voltaje/intensidad

EndStop (Final de carrera)

En español se llaman interruptor final de carrera. Hay dos tipos principales, los mecánicos que son un interruptor con una palanquita, y los ópticos que tienes que meter una banderilla (una tirita de metal) en una ranura para indicar que hay pieza. Existe un tercer tipo que es magnético. Se usan para indicar que una pieza ha llegado al final de su recorrido (y por tanto no debería avanzar más).

En la RepRap se ponen por lo general 3 finales de carrera (aunque se pueden poner 6) para marcar el origen de los ejes X, Y y Z. Los más fáciles de usar son los mecánicos, ya que requieren menos componentes (y menos consumo energético) a parte de que funcionan muy bien. En general la desventaja de los finales de carrera mecánicos es que por las tensiones en la palanca estos se dañan antes, pero, dado el poco uso que se hace de ellos generalmente en la RepRap, es muy poco probable que fallen.

Final de carrera mecánico.

Rodamientos Radiales de Bolas (o *bearings*)

Son aparatos que permiten el acoplamiento de un eje móvil (que gira) con una pieza fija (un soporte) tal que se minimice el rozamiento producido. Los rodamientos radiales están compuestos por dos cilindros concéntricos con bolas entre ambos que ruedan permitiendo un giro de un cilindro con respecto a otro con poco rozamiento.

Rodamientos Lineales

Son rodamientos que, de forma similar a los rodamientos radiales, evitan la fricción, pero esta vez al trasladar el rodamiento en la dirección longitudinal del eje (en la dirección del eje). Permiten que una barra lisa se comporte como una guía. Son imprimibles.

Rodamientos radiales (izqda.) y lineales (dcha.)

Varillas lisas

Se utilizan en la mayoría de impresoras RepRap como sistema de guiado lineal (los llamados carros) en combinación con rodamientos lineales. Pueden ser de acero inoxidable o de aluminio, pero es importante que el diámetro sea el adecuado (por lo general de 8mm) y sea constante, y que la varilla esté bien recta.

Cuando la compras, para comprobar que las dimensiones son las adecuadas, puedes hacer pasar un rodamiento por ella para comprobar el diámetro, y puedes hacerla rodar por una superficie plana para comprobar que está recta. Para saber donde adquirir este material visita la sección de tiendas.

Varillas roscadas

Se utilizan en la mayoría de impresoras RepRap como elemento estructurales y como "husillos" para el desplazamiento del eje Z. Son varillas de acero inoxidable con una rosca métrica mecanizada en su cara exterior. Las más utilizadas son M8 aun que también se están empezando a utilizar M5 para la prusa 3 ya que dan mayor precisión en el posicionamiento en el caso de los husillos.

Puedes comprarlas en grandes ferreterías (Leroy Merlin, Brico Depot, AKI) a buen precio, pero asegúrate de que están rectas y de que la rosca no está dañada.

Husillo

Un husillo es un tipo de varilla roscada que tiene una rosca trapezoidal (en vez de triangular como la de la M8). Tienen la ventaja con respecto a las varillas roscadas normales de que su diseño está orientado a generar desplazamientos como los que se realizan en una RepRap, pero también son mucho más caros.

Husillos compatibles con las piezas ya existentes en las impresoras se pueden comprar (no muy caros) en http://techpaladin.com/ (USA)

Estos husillos, si bien pueden tener el mismo diámetro de las varillas roscadas (que son M8, por lo que su diámetro es 8mm) no son compatibles con las tuercas de métrica 8 (M8) ya que, entre otras cosas tienen una rosca trapezoidal (en vez de triangular) y un paso posiblemente distinto. Por tanto habrá que comprar tuercas especiales para estos husillos.

Cinta Kapton (Cinta de Poliamida)

Es un tipo de cinta aislante de color naranja semitransparente y brillante que puede resistir muy bien altas temperaturas. En una RepRap se utiliza, además de para proteger zonas que pueden entrar en contacto y están a altas temperaturas, para cubrir la placa caliente y permitir que el ABS derretido se adhiera a la placa.

Cinta Kapton

Tubo termo retráctil (Heatshrink)

Es un tubo flexible hecho de un plástico especial que al calentarse se comprime sobre si mismo y se vuelve rígido. Se utiliza como aislante ya que se puede colocar con facilidad y al calentarse se adapta a la forma de la conexión protegiéndola.

Termistor

Es un tipo de resistencia eléctrica que cambia el valor de su resistencia con la temperatura de forma muy acusada, por lo que se puede medir la temperatura de un objeto, midiendo la resistencia que presenta el termistor a la corriente eléctrica.

Termopar (Termocouple)

Es una alternativa al uso de termistores. En la unión de dos metales diferentes al variar la temperatura de esta unión, se produce un desequilibrio eléctrico que se traduce en la aparición de un potencial (o voltaje) entre los dos metales. Este potencial se puede medir y tiene relación con la temperatura de la unión.

Bowden

Es un mecanismo de transmisión del movimiento en el que el movimiento de un núcleo (en este caso filamento) se transmite a través de un tubo exterior al mismo. La ventaja que tiene para las impresoras 3D es que gracias a él se puede colocar el extrusor fuera del carro del eje X, reduciendo la masa e inercia de éste, y con ella, las vibraciones, permitiendo obtener velocidades de impresión más altas sin perder precisión.

7.9 Electrónica

Se refiere a los componentes electrónicos que controlan el funcionamiento de la RepRap (el chip). Hay 3 principales, pero existen mas: RAMPS (con su correspondiente Arduino Mega), Sanguinololu y Gen7.

RAMPS

Literalmente **R**epRap **A**rduino **M**ega **P**ololu **S**hield. Es un shield de la Arduino Mega que permite poner hasta 5 controladores Pololu de los motores paso a paso. Adicionalmente tiene la ventaja de que, como la Arduino Mega tiene muchos pines, se le pueden hacer muchas expansiones (como por ejemplo, una pantalla LCD, un lector de tarjetas SD, un panel de control con botones...).

Se ha de comprender que la RAMPS ha de ir montada sobre una Arduino Mega, y requiere de esta placa para funcionar (por tanto esta alternativa se compone de 2 placas, el shield RAMPS y la Arduino Mega).

Página de la wiki de RepRap sobre RAMPS

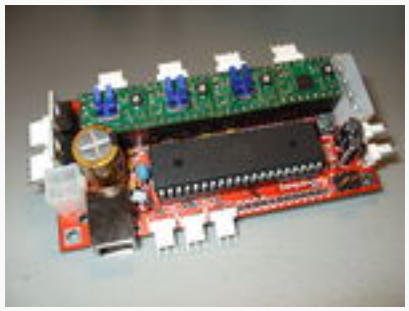
RAMPS, sin Arduino Mega (iría debajo)

Sanguinololu

Es una electrónica basada en la Sanguino (un clon de Arduino) que proporciona una alternativa compuesta de una sola placa a diferencia de la RAMPS. En esta placa se incluyen los controladores de los motores sólo pudiéndose conectar 4. El precio de la Sanguinololu es más barata que el precio conjunto de una RAMPS con su correspondiente Arduino Mega. Por otra parte la RAMPS sin el Arduino Mega, cuesta menos que la Sanguinololu (como es lógico).

Existen dos versiones equivalentes de la Sanguinololu en estos momentos, la 1.3a y la 1.3b. La 1.3b es una modificación de la 1.3a diseñada por la tienda de RepRap.me y basada en componentes SMT (componentes superficiales). Es una placa que ocupa menos. Los MOSFET que alimenta la base caliente y el extrusor soportan mucho mejor las altas corrientes que la equivalente 1.3a, por lo que no tendrás problemas de quemarlos aun sin usar ventilador. Conviene poner un disipador en el regulador de entrada. En cuanto a todo lo demás, es totalmente equivalente a la 1.3a. Tanto la 1.3a como la 1.3b las venden en España LeapTo3d.com

Página de la wiki de RepRap sobre Sanguinololu



Sanguinololu 1.3

Gen 7

Es la placa mas fiel al movimiento RepRap ya que es totalmente DIY (*do it yourself*, hazlo tu mismo). Es posiblemente la mas barata pero fuerza a buscar todos los componentes por separado y a grabar la PCB (la placa de circuito).

Página de la wiki de RepRap sobre la Gen7

Gen7

SAV MKI

La nueva placa creada en España.

Esta placa se ha diseñado y desarrollado usando las contribuciones y comentarios del Grupo Clone Wars de RepRap España. Se han intentado eliminar los defectos de placas similares intentando mantener un precio asequible pero con muchas características. La mejor placa diseñada en España (hasta la fecha) y todo de la mano de la comunidad Clone Wars

Página de la wiki de RepRap sobre la SAV MKI

SAV MKI

7.10 Terminología de motores

Intensidad nominal

En un motor eléctrico se refiere a la intensidad máxima a la que puede funcionar correctamente el motor. Someter al motor a intensidades superiores no tiene por qué suponer el deterioro inmediato del mismo, pero si la reducción de forma considerable de la vida media.

Tensión nominal

Es la tensión (potencial o voltaje) para la que está diseñado el motor. Es posible hacer trabajar a un motor a menores potenciales, pero no se asegura que el par generado pueda arrancar el motor. Es peligroso hacer trabajar al motor a tensiones superiores ya que los aislamientos internos no tienen porqué soportar dicha tensión.

Par motor (o torque)

Es la fuerza de giro que puede proporcionar el motor. Un par se componen de dos fuerzas que son de igual magnitud y sentido opuesto y que se aplican sobre puntos distintos tal que producen que un cuerpo gire, pero que no se traslade. Se mide en newton metro (Nm) en el sistema internacional, aunque es poco común encontrarlos en esta denominación y se usan mas los Ncm (100 veces más pequeño, pero más ajustado para los pares que estamos usando), los gcm (es aproximadamente 100 veces más que los Ncm) y las unidades del sistema anglosajón. Podemos utilizar herramientas online para hacer la conversión Aquí, y tener en cuenta que los motores tendrán que tener al menos 20Ncm para los ejes X, Y y Z, y como mínimo 40Ncm para el extrusor.

Velocidad nominal (no se aplica a los paso a paso)

Velocidad a la que gira un motor en condiciones nominales sin tener que soportar ninguna carga (par resistente).

Funcionamiento en vacío

Es el funcionamiento de un motor cuando gira sin soportar carga (solo la propia) y por lo general es cuando menos energía (intensidad) consume. En los motores paso a paso este funcionamiento no es común ya que el funcionamiento de un motor paso a paso lo lleva a muchos estados de corto (uno por paso), de allí el ruido que producen.

Funcionamiento en corto

Es el funcionamiento de un motor cuando el rotor no gira pero hay corriente en el bobinado. Por lo general es el momento en el que más energía consumen y los motores no suelen estar preparados para funcionar en este estado mucho tiempo. Sin embargo los motores paso a paso están preparados para funcionar en este estado, y de hecho es una de sus propiedades: la capacidad de aguantar en una posición resistiendo fuerzas moderadas sin moverse.

7.11 Enfermedades

SAV: Síndrome del Ansia Viva

definición: enfermedad o trastorno que sufren los 'makers' tras estar en contacto durante prolongados periodos de tiempo con las impresoras 3D. Este trastorno les lleva a querer probar todo tipo de modelos de impresoras, materiales para imprimir, hotends, etc. y a no poder resistirse. Existen diferentes teorías aún no probadas que exponen que el SAV se produce por respirar vapores de ABS o PLA. También hay quien opina que es producido hipnóticamente por el propio movimiento de los ejes de la impresora. También, hay quien sostiene que es debido a la respiración de los vapores de la laca Nelly y el tufo a peluquería que se forma. Esto está aún por comprobar. La enfermedad se detectó por primera vez entre los integrantes del proyecto Clone Wars

8 Fuentes de información

Fuentes de información empleadas para elaborar este documento.

Como no, hay que mencionar principalmente a

http://www.reprap.org/wiki/Proyecto_Clone_Wars

Y a

https://github.com/MarlinFirmware/Marlin

Otras fuentes de información consultadas:

http://3digitalcooks.com/2014/10/marlin-movement-101/

http://3digitalcooks.com/2014/12/marlin-planner-101/

http://www.extrudable.me/2013/04/02/the-myth-of-z-speed/

http://es.slideshare.net/roboard/3d-printer-marlin

http://3dprintboard.com/showthread.php?2802-Auto_Bed_Leveling-Z-Probe-Repeatability-code

http://blog.lincomatic.com/?p=773

http://hydraraptor.blogspot.ie/2010/12/frequency-limit.html

https://github.com/alexrj/Slic3r/issues/1494

http://www.makerbot.com/sailfish/tuning/#5

http://reprap.org/wiki/G-code

http://reprap.org/wiki/PID_Tuning

http://www.reprap.org/wiki/Clone_wars:_Glosario/es

9 Copyright

10 Revisiones

Edición 0	Versión de Marlin 1.0.2	20/03/2015