ANEJO IV. COMANDOS DE GRBL

GRBL tiene un sistema de comandos $. Si se pulsa en la ventana de comandos el comando "$", GRBL responde con la lista de comandos que se pueden introducir y una pequeña explicación. A continuación se muestran esos comandos traducidos.

Comando	Explicación
$$	Ver configuración de GRBL.
$#	Ver parámetros #.
$G	Ver estado de configuración de cabecera.
$N	Ver bloques de inicio.
$x=valor	Guardar configuración de GRBL.
$Nx=línea	Guardar bloque de inicio.
$C	Comprobar el modo de código G.
$X	Quitar alarma de bloqueo por final de carrera.
$H	Ejecutar ciclo a casa.
~	Comenzar ciclo.
!	Mantener alimentación.
?	Estado actual.
Ctrl-x	Reiniciar GRBL.

De todos estos comandos, realmente los que se usan para configurar GRBL son $$, $x=valor, $N y $Nx=línea.

$$ Ver configuración de GRBL / $x=valor Guardar configuración de GRBL

Cuando GRBL está conectado, al escribir el comando $$ y pulsar intro se debe de recibir una respuesta con la lista de configuraciones. Toda esa configuración es persistente ya que se almacena en la memoria EEPROM. La respuesta tiene aproximadamente la siguiente forma (configuración que no corresponde con la llevada a cabo en este proyecto. En este momento solamente se pretende explicar cómo funciona. Más adelante se presentará la usada).

$0=755.906 (x, step/mm)
$1=755.906 (y, step/mm)
$2=755.906 (z, step/mm)
$3=30 (step pulse, usec)
$4=500.000 (default feed, mm/min)
$5=500.000 (default seek, mm/min)
$6=28 (step port invert mask, int:00011100)
$7=25 (step idle delay, msec)
$8=50.000 (acceleration, mm/sec^2)
$9=0.050 (junction deviation, mm)
$10=0.100 (arc, mm/segment)
$11=25 (n-arc correction, int)
$12=3 (n-decimals, int)
$13=0 (report inches, bool)
$14=1 (auto start, bool)
$15=0 (invert step enable, bool)
$16=0 (hard limits, bool)
$17=0 (homing cycle, bool)
$18=0 (homing dir invert mask, int:00000000)
$19=25.000 (homing feed, mm/min)
$20=250.000 (homing seek, mm/min)
$21=100 (homing debounce, msec)
$22=1.000 (homing pull-off, mm)

Por ejemplo, si se quieren cambiar los microsegundos del pulso STEP a 10µs, entonces hay que escribir $3=10 y pulsar intro.

A continuación se muestran todos los comandos $x=valor con una explicación de cuál es su significado.

- **$0, $1 y $2 – XYZ, pasos/mm**

GRBL necesita saber cuánto va a avanzar la máquina por cada paso. Los pasos por milímetro se pueden calcular mediante la siguiente ecuación:

$$pasos/mm = \frac{pasos\ por\ vuelta * micropasos}{mm\ por\ revolución}$$

Hay que calcularlo para cada eje y escribir esta configuración en GRBL.

- **$3 Pulso STEP, microsegundos**

Los drivers están diseñados para diferentes longitudes del pulso STEP. Hay que intentar buscar el valor mínimo, porque si se pone a trabajar el motor a velocidades altas con un pulso STEP demasiado largo, probablemente surjan problemas. En general un valor entre 5 y 50 microsegundos funciona perfectamente.

- **$4 y $5 Movimiento rápido y de mecanizado, mm/min**

Permiten configurar la velocidad de movimiento rápido (G0) y las velocidades de mecanizado (G1, G2 y G3) después de que GRBL se active e inicialice. La velocidad de desplazamiento rápido se realiza en vacío y en general se emplea para desplazarse rápidamente hasta alcanzar la posición desde la que empezará el mecanizado. En general esta velocidad debe de ser la mayor que sea capaz de proporcionar la máquina.

Es curioso pero en realidad la velocidad que se establece por defecto de mecanizado no tiene sentido, ya que no existe ningún estándar de fichero de control numérico que requiera de una velocidad por defecto para G1, G2 y G3. En futuras versiones, esta configuración se eliminará.

- **$6 Máscara invertir sentido ejes, int: binario**

Si por ejemplo se efectúa un movimiento positivo en el eje X, y la máquina responde de forma opuesta, es decir moviéndolo en sentido negativo, se debe de actuar sobre esta máscara. Esto puede deberse a multitud de factores constructivos del CNC.

Esta máscara es un byte con el que se realiza la operación xor con el byte de STEP y DIR. Esto permite configurar cada eje de forma independiente. Los bits de este byte corresponden con los pines asignados en el fichero config.h. Por defecto estos pines están asignados como:

```
#define X_STEP_BIT 2
#define Y_STEP_BIT 3
#define Z_STEP_BIT 4
#define X_DIRECTION_BIT 5
#define Y_DIRECTION_BIT 6
#define Z_DIRECTION_BIT 7
```

Por ejemplo, si se quiere invertir el sentido del eje X e Y entonces el valor entero se debe de calcular de la siguiente forma:

$$> (1 << X\_DIRECTION\_BIT)|(1 << Y\_DIRECTION\_BIT)$$

Para realizar esta operación lo más recomendable es emplear una calculadora. Este cálculo en este ejemplo arroja el valor de 96, por lo que si se hace $6=96, se habrá realizado la operación con éxito. Para asegurarse de que todo ha ido bien, se puede acceder de nuevo a la configuración de GRBL y los bits 5 y 6 deben de estar a 1 (1100000).

- **$7 Tiempo motores parados, ms**

Cada vez que los motores ejecutan un movimiento, GRBL desactiva los motores por defecto. Esta variable controla el tiempo que los motores van a estar bloqueados (recibiendo intensidad) antes de ser desactivados. Este parámetro depende en gran medida de los motores empleados. Si se configura a 0, entonces se apagan los motores directamente y no se bloquean. Si se configura a 255 nunca se desactivan (siempre recibirán intensidad, por lo que al terminar de moverse se bloquearán instantáneamente). En otras máquinas lo recomendable es emplear un valor entre 25 y 50 milisegundos, para asegurar que los motores están totalmente parados antes de desactivarlos.

Hay que tener en cuenta que un motor que siempre está bloqueado, es un motor que siempre está trabajando, consumiendo intensidad y calentándose.

- **$8 Aceleración, mm/seg^2**

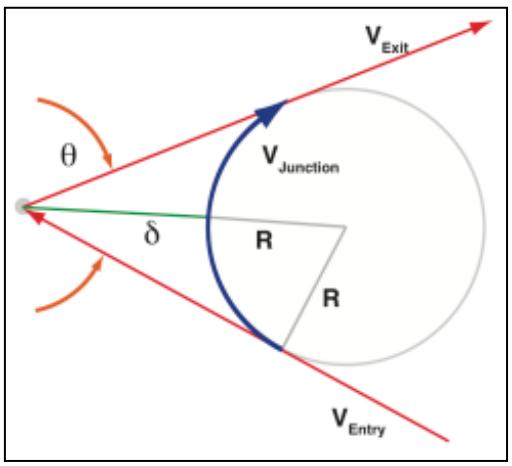
La curva de aceleración permite realizar movimientos suaves y rápidos pero con un par inicial muy grande.

En realidad, lo único que hay que saber es que un valor bajo, proporciona una curva de aceleración tremendamente suave (sistema de primer orden con una constante de tiempo baja). Si el valor es muy elevado, el movimiento del motor es más brusco, pareciéndose su curva de velocidad a un escalón.

Para configurar este parámetro, hay que realizar el método de ensayo/error hasta dar con el valor máximo que hace que no se pierda ningún paso.

- **$9 Desviación entre uniones, mm**

Este parámetro es usado por el gestor de aceleración para determinar lo rápido que se mueve entre una trayectoria. En general, valores bajos permiten que la máquina alcance las esquinas de una forma lenta. Si se incrementa, se moverá más rápida entre las uniones. Aunque no se pretende entrar en términos matemáticos difíciles, la idea se puede entender desde el siguiente dibujo:



El segmento circular crea una desviación en la trayectoria de valor δ. Este valor define la distancia desde la unión hasta el lado del segmento circular (definido por el usuario), por lo que indirectamente se define el radio del círculo.

- **$10 Arco, mm/segmento**

GRBL divide los arcos y curvas en pequeñas líneas. Se puede aumentar la precisión disminuyendo este valor, pero puede llevar a problemas de rendimiento. En general este parámetro no se debe modificar.

- **$11 N-arco corrección, int**

De nuevo se trata de un parámetro de configuración avanzado que en general no hay que tocar. Para que GRBL pueda trazar arcos mediante G02 y G03, aproxima la localización del siguiente arco mediante un pequeño ángulo de aproximación.

Este parámetro determina el número de segmentos de arcos a aproximar antes de que GRBL calcule el arco exacto para eliminar el error de acercamiento. Calcular estas posiciones óptimas es computacionalmente complejo, pero hay algunos casos en los que este parámetro es importante. No obstante, no se recomienda bajar este valor de 3, ya que puede sobrecargar el buffer de GRBL.

- **$12 N-decimales, int**

Este número determina el número de cifras decimales que GRBL va a tener en cuenta en el fichero de control numérico que recibe.

- **$13 Respuesta en pulgadas, booleano**

GRBL tiene capacidad para enviar en tiempo real al usuario la posición en la que se encuentra la máquina en cualquier momento. Por defecto este valor lo envía en milímetros, pero si se hace $13=1, entonces este valor lo envía en pulgadas.

- **$14 Comienzo automático, booleano**

En un entorno profesional, los técnicos comienzan un trabajo cuando se ha cargado el programa y tras pulsar el botón de comienzo de ciclo. Solo cuando se pulsa es cuando comienza el trabajo. GRBL también puede hacer esto pero no por defecto. Como herramienta de aprendizaje, por defecto se comienza directamente el trabajo tras enviar el código, así se puede ver rápidamente como reacciona la máquina sin necesidad de pulsar una y otra vez el botón de comienzo. Una vez que el usuario aprenda a manejar correctamente la máquina, es correcto desactivar el comienzo automático del ciclo haciendo $14=0 en la consola de comandos.

- **$15 Invertir enable, booleano**

Por defecto, en el pin enable se envía 1 para desactivar los drivers y 0 para activarlos. Si los drivers que se están empleando funcionan con lógica inversa, entonces hay que establecer este parámetro en 1.

- **$16 Finales de carrera, booleano**

El uso de finales de carrera es crucial para evitar romper algo en caso de mal funcionamiento. Cuando se activa el final de carrera, se paran todos los motores, se apaga la fresadora y se deja de aportar refrigerante. La máquina entra en un estado de alarma que obliga a reiniciar el proceso.

Para usar los finales de carrera, hay que establecer este parámetro en 1, y para no usarlos, en 0. Recordar que simplemente los finales de carrera son interruptores normalmente abiertos.

Cuando se acciona un final de carrera se considera un evento crítico, donde los motores se paran inmediatamente. GRBL no tiene información a cerca de la posición sobre la que ocurrió la alarma, por lo que si se acciona el final de carrera, hay que revisar manualmente qué ha ocurrido, y comenzar de nuevo el proceso.

- **$17 Ciclo a casa, booleano**

El ciclo a casa se emplea para saber la posición con respecto al origen de la máquina cada vez que GRBL inicializa. En otras palabras, la máquina sabe en cada momento donde está. Si no se tiene en ciclo a casa activo, supongamos que se está mecanizando una pieza y se va a electricidad. Cuando GRBL vuelve a inicializar, no tiene ninguna idea de donde está. Si el ciclo a casa está activo, se tiene referencia con el origen, por lo que simplemente lo que hay que hacer es ejecutar un ciclo a casa y continuar el trabajo por donde se dejó.

Para emplear el ciclo a casa, hacen falta interruptores de límite situados en una posición estática. Normalmente la referencia se toma en la posición positiva máxima de cada eje (X+, Y+ y Z+). Los finales de carrera se pueden emplear tanto como finales de carrera como interruptores de límite para el ciclo a casa.

Por defecto, GRBL ejecuta el ciclo a casa primero desplazando el eje Z en una posición positiva con el fin de no interferir, y después los ejes X e Y al mismo tiempo en la posición positiva. Existen multitud de parámetros para configurar el ciclo a casa, que se irán describiendo a continuación.

- **$18 Ciclo a casa máscara invertir sentido ejes, booleano**

Por defecto, GRBL asume que los interruptores de límite se encuentran en la posición positiva de los ejes, moviendo primero el eje Z en el sentido positivo y luego el X e Y intentando ubicar la máquina de una forma precisa en la posición cero yendo hacia delante y hacia atrás suavemente. Si la máquina tiene los interruptores de límite en la posición negativa, hay que configurar la máscara tal y como se hizo en el apartado 6.3.4.

- **$19 Velocidad lenta ciclo a casa, mm/min**

El ciclo a casa primero busca los interruptores de límite a una velocidad elevada. Tras encontrarlos, se mueve a una velocidad mucho más lenta hasta ubicar la máquina de forma precisa en la posición cero. Este parámetro determina esa velocidad de desplazamiento precisa (lenta). Hay que emplear un valor que permita repetibilidad y precisión en la búsqueda del cero.

- **$20 Velocidad rápida ciclo a casa, mm/min**

Esta es la velocidad con la que la máquina busca los interruptores de límite. Hay que establecer una velocidad en la que la máquina llegue relativamente rápida sin que pueda romper los finales de carrera. Un valor óptimo es utilizar una velocidad un 41% más rápida que la velocidad que se emplea en G1.

- **$21 Antirebote ciclo a casa, ms**

Cuando un final de carrera se acciona, en muchas ocasiones se produce ruido eléctrico o mecánico debido al efecto del rebote. Esto se puede solucionar mediante hardware con un acondicionador de señal, o simplemente mediante software mediante un pequeño retraso para

dejar que la señal se estabilice. Este parámetro establece ese tiempo. En general, un valor entre 5 y 25ms funciona perfectamente.

- **$22 Ciclo a casa pull-off, mm**

Para que la funcione a la perfección un interruptor como final de carrera y como límite para el ciclo a casa, la máquina va a retroceder tantos milímetros como se establezcan en este parámetro después de que se accione el interruptor. Esto ayuda a prevenir accionamientos accidentales de los finales de carrera después de un ciclo a casa.

Esta velocidad de retroceso es controlada por el parámetro del apartado 6.3.18.

$N Ver bloques de inicio / $Nx=línea Guardar bloque de inicio

El comando $Nx lo que hace es enviar bloques de código G en el momento en el que GRBL inicializa (desde encendido o desde reset). En otras palabras, son segmentos de código G que se introducen directamente. Se pueden caracterizar como código de pre inicialización. El sketch de que se dispone permite emplear hasta dos bloques, pero el fichero config.h se puede modificar para emplear desde 1 hasta 5 (este proceso requiere compilación).

Inicialmente, si se introduce el comando $N, aparece la siguiente información:

```
$N0=
$N1=
```

Esto quiere decir que los bloques de pre inicialización están vacíos. GRBL no va a hacer nada cuando inicialice. Para configurar el bloque 0, basta con escribir $N0= “Bloque de código G válido”. En este ejemplo se va a establecer G54 (sistema de coordenadas), G20 (cotas en pulgadas) y G17 (plano XY). Entonces habría que escribir $N0=G20 G54 G17. Esta información se almacena en la memoria EEPROM, por lo que cada vez que inicialice GRBL se va a ejecutar este código.

Si se desea limpiar un bloque, por ejemplo el 0, basta con hacer $N0=. Es decir, no poner nada después del signo igual.

Además, si el ciclo a casa está activado, primero se ejecuta el ciclo a casa, y después el código de pre inicialización.

Es muy importante tener cuidado al usar funciones G0/1, G2/3 y G28/30 en los bloques de pre inicialización. Por ejemplo, supongamos que se produce una situación de emergencia y al activarse un final de carrera hay que reinicializar GRBL. Entonces este código se va a ejecutar, complicando aún más las cosas.

Otros comandos

Aunque ya se han explicado todos los comandos de configuración, existen otros que aunque no permitan configurar nada, realizan ciertas acciones fundamentales para el completo funcionamiento del CNC. Algunos de ellos, incluso disponen de botón propio en GRBL Controller.

- **Ctrl-x Reiniciar GRBL**

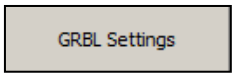
Esta función, como ya se indicó, reinicia GRBL. Es necesario pulsarlo para que se hagan vigentes la mayoría de modificaciones en la configuración. Se puede acceder a él desde la pestaña “Advanced”, pulsando sobre:

- **$X Quitar alarma de bloqueo por final de carrera**

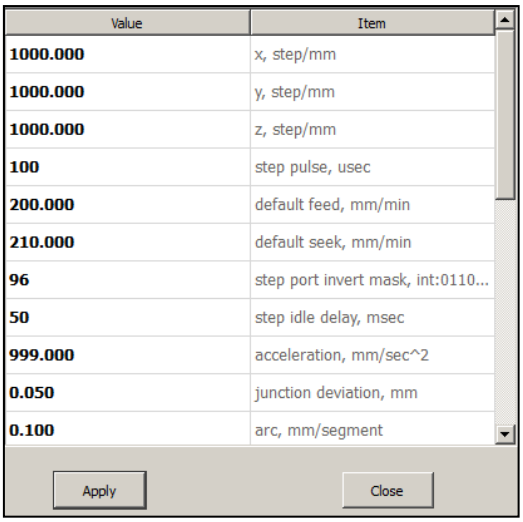
También llamado “Desbloquear GRBL”. Necesario pulsar cuando se produce el accionamiento de un final de carrera. También se puede accionar desde la pestaña “Advanced”:

- **$$ Configuración de GRBL**

Desde la pestaña “Advanced”:



Este botón tiene doble funcionalidad. En primer lugar ejecutar el comando $$ para ver toda la configuración de la máquina de tipo $n=valor, y además muestra una tabla sobre la que se pueden modificar los valores de los comandos $n=valor directamente. Se recomienda pulsar el botón “Soft Reset Grbl” o ejecutar el comando Ctrl-x para que surjan efecto estos cambios.



- **$# Sistemas de coordenadas máquina**

Al ejecutar este comando, GRBL muestra los sistemas de coordenadas máquina que dispone. La salida por consola es:

```
[G54:0.000,0.000,0.000]
[G55:0.000,0.000,0.000]
[G56:0.000,0.000,0.000]
[G57:0.000,0.000,0.000]
[G58:0.000,0.000,0.000]
[G59:0.000,0.000,0.000]
[G28:0.000,0.000,0.000]
[G30:0.000,0.000,0.000]
[G92:0.000,0.000,0.000]
```

Los parámetros de los sistemas de coordenadas suelen tener estos códigos:

Función	X	Y	Z
G54	5221	5222	5223
G55	5241	5242	5243
G56	5261	5262	5263
G57	5281	5282	5283
G58	5301	5302	5303
G59	5321	5322	5323