Configurando Grbl v0.8

Grbl tiene un comando '$' del sistema para modificar la configuración y desencadenar comandos de tiempo de ejecución. Conéctese a Grbl utilizando el terminal en serie de su elección (velocidad en baudios 9600, a menos que haya cambiado eso en config.h) como 8-N-1 (8 bits, sin paridad y bit de 1 parada). Una vez conectado, debe obtener el Grbl-prompt, que se ve así:

Grbl 0.8c ['$' for help]

Escribe $ y presiona enter. No debería ver ningún eco local de $ e ingrese, pero Grbl debería responder con un mensaje de ayuda:

$$ (view Grbl settings)

$# (view # parameters)

$G (view parser state)

$N (view startup blocks)

$x=value (save Grbl setting)

$Nx=line (save startup block)

$C (check gcode mode)

$X (kill alarm lock)

$H (run homing cycle)

~ (cycle start)

! (feed hold)

? (current status)

ctrl-x (reset Grbl)

De estos comandos, solo cuatro de ellos se utilizan para configurar Grbl [$$, $x=value, $N, $Nx=line], mientras que el resto son comandos en tiempo de ejecución que informan el estado actual en Grbl o modifican el comportamiento.

# Los cuatro comandos de configuración: $$, $x=val, $N, $Nx=line

### $$ - Ver configuración de Grbl, $ x = val - Guardar configuración de Grbl

Cuando esté conectado a Grbl, escriba $$y presione enter. Grbl debe responder con una lista de la configuración actual del sistema, similar a la que se muestra a continuación. Todos estos ajustes son persistentes y se mantienen en EEPROM, por lo que si se apaga, estos se volverán a cargar la próxima vez que encienda su Arduino.

$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)

Para cambiar, por ejemplo, la opción de pulso de paso de microsegundos a 10us, debe escribir esto, seguido de una entrada:

$3=10

Si todo salió bien, Grbl responderá con un 'ok' y esta configuración se almacenará en EEPROM y se mantendrá para siempre o hasta que los cambie. Puede verificar si Grbl ha recibido y almacenado su configuración correctamente escribiendo $$para ver la configuración del sistema nuevamente.

### $ N - Ver bloques de inicio, $ Nx = línea - Guardar bloque de inicio

$Nxson los bloques de inicio que Grbl ejecuta cada vez que enciendes Grbl o reinicias Grbl. En otras palabras, un bloque de inicio es una línea de código g que puede hacer que Grbl ejecute de forma automática y mágica para establecer sus valores predeterminados de código g, o cualquier otra cosa que necesite que haga Grbl cada vez que encienda su máquina. En este momento, Grbl almacenará dos bloques de g-code como valor predeterminado del sistema, pero puede almacenar de 1 a 5, como una opción de tiempo de compilación (cambio de config.h y re-compilación).

Entonces, cuando esté conectado a Grbl, escriba $Ny luego ingrese. Grbl debería responder con algo corto como:

$N0=

$N1=

ok

No hay mucho que seguir, pero esto solo significa que no hay un bloque de código g almacenado en línea $N0para que Grbl se ejecute al inicio. $N1Es la siguiente línea a ejecutar. (Si vuelve a compilar para tener más, se ejecutará en orden a $N4).

Para configurar un bloque de inicio, escriba $N0=seguido de un bloque de g-code válido y una entrada. Grbl ejecutará el bloque para verificar si es válido y luego responderá con una oko una error:para decirle si tuvo éxito o si algo salió mal.

Por ejemplo, supongamos que desea utilizar su primer bloque de inicio $N0para configurar los modos del analizador de código g como la coordenada de trabajo G54, el modo G20 pulgadas, el plano G17 XY. Escribiría $N0=G20 G54 G17con una entrada y debería ver una respuesta 'ok'. Luego puede verificar si se almacenó escribiendo $Ny ahora debería ver una respuesta como $N0=G20G54G17.

Una vez que tenga un bloque de inicio almacenado en la EEPROM de Grbl, cada vez que inicie o reinicie verá que se le vuelve a imprimir el bloque de inicio y una respuesta de Grbl para indicar si se ejecutó correctamente. Así que para el ejemplo anterior, verás:

Grbl 0.8c ['$' for help]

G20G54G17ok

Si tiene varios bloques de inicio de código g, se imprimirán de nuevo a usted en orden en cada inicio. Y si desea borrar uno de los bloques de inicio, escriba $N0=sin nada siguiendo el signo igual.

Además, si tiene habilitado el inicio, los bloques de inicio se ejecutarán inmediatamente después del ciclo de inicio, no al inicio.

IMPORTANTE: tenga mucho cuidado al almacenar cualquier comando de movimiento (G0 / 1, G2 / 3, G28 / 30) en los bloques de inicio. Estos comandos de movimiento se ejecutarán cada vez que reinicie o encienda Grbl, por lo que si tiene una situación de emergencia y tiene que detener y reiniciar, un movimiento de bloqueo de inicio puede y probablemente empeorará las cosas rápidamente.

# La $x=valconfiguración de Grbl y su significado.

### $ 0, $ 1 y $ 2 - XYZ, pasos / mm

Grbl necesita saber hasta qué punto cada paso llevará la herramienta a la realidad. Para calcular pasos / mm para un eje de su máquina, debe saber:

* El mm por revolución del tornillo de avance.
* Los pasos completos por revolución de tus steppers (normalmente 200)
* Los microsteps por paso de su controlador (típicamente 1, 2, 4, 8 o 16). Consejo: el uso de valores altos de microstep (p. Ej., 16) puede reducir el par de su motor paso a paso, así que use el más bajo que le ofrezca la resolución de ejes deseada y las propiedades de marcha cómodas.

Los pasos / mm se pueden calcular así: steps\_per\_mm = (steps\_per\_revolution\*microsteps)/mm\_per\_rev

Calcule este valor para cada eje y escriba esta configuración en Grbl.

### $ 3 - Paso de pulso, microsegundos

Los controladores paso a paso están clasificados para una cierta longitud mínima de impulso de paso. Revisa la hoja de datos o simplemente prueba algunos números. Usted desea pulsos tan cortos como los controladores paso a paso puedan reconocer de manera confiable. Si los pulsos son demasiado largos, puede tener problemas para ejecutar el sistema a altas velocidades de alimentación. En general, algo entre 5 y 50 microsegundos funciona bien.

### $ 4 y $ 5 - Tasas de alimentación y búsqueda predeterminadas, mm / min

Esta configuración establece la búsqueda predeterminada (G0) y ​​las velocidades de alimentación (G1, G2, G3) después de que Grbl se encienda y se inicialice. La tasa de búsqueda (también conocida como rápidos) se usa para moverse del punto A al punto B lo más rápido posible, generalmente para desplazarse a la posición. La tasa de búsqueda debe establecerse a la velocidad máxima que su máquina puede realizar en cualquier movimiento de los ejes. La velocidad de alimentación predeterminada generalmente no entra en la imagen, ya que las velocidades de alimentación generalmente se especificarán en el programa de código g, pero si no, se utilizará esta velocidad de alimentación predeterminada.

NOTA: En futuras versiones, se eliminará la velocidad de alimentación predeterminada. Los estándares de código G no tienen una velocidad de alimentación predeterminada (G1, G2, G3), pero se produce un error si no se ha especificado un comando de velocidad de alimentación 'F'. La tasa de búsqueda (también conocida como rápidos) se actualizará para que los usuarios puedan especificar la velocidad máxima que cada eje puede mover, en lugar de tener una sola velocidad que los limite a todos.

### $ 6 - Máscara de inversión de puerto de paso, int: binario

Algunos controladores cnc-stepper necesitan sus entradas alto-bajo invertidas tanto en dirección como en pasos. Las líneas de señal normalmente se mantienen altas o bajas en la dirección de la señal o se mantienen altas y baja durante un par de microsegundos para señalar un evento de paso. Para lograr esto, Grbl puede invertir los bits de salida para satisfacer necesidades particulares. El valor de la máscara de inversión es un byte que se almacena con los datos de pasos y direcciones antes de que se envíe por el puerto de pasos. De esa manera, puede usar esto tanto para invertir los pulsos de pasos como para invertir una o más de las direcciones de los ejes. Los bits en este byte corresponden a los pines asignados a escalonar en config.h. Tenga en cuenta que los bits 0 y 1 no se utilizan para la inversión. Los bits por defecto se asignan de esta manera:

#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

Si quisiera invertir las direcciones X e Y en esta configuración, calcularía un valor de desplazamiento de bits como este (en su entorno de cálculo favorito):

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

Que es igual a 96, por lo que emitir este comando los invertiría:

$6=96

Ahora, cuando vea la configuración actual, debería ver esto en su línea de máscara de inversión con la representación binaria del número (los bits 5 y 6 ahora deberían mostrar un 1 para indicar la inversión).

$6=96 (step port invert mask. int:1100000)

### $ 7 - Step idle delay, msec

Cada vez que sus pasos completan un movimiento y se detienen, Grbl desactivará los pasos de forma predeterminada. El tiempo de bloqueo del ralentí por pasos es la duración de tiempo que Grbl mantendrá bloqueados antes de deshabilitarse. Dependiendo del sistema, puede establecerlo en cero y desactivarlo. En otros, es posible que necesite 25-50 milisegundos para asegurarse de que sus ejes se detengan por completo antes de deshabilitarlos. (Mi máquina tiende a desviarse solo un poco si no lo tengo habilitado). O, siempre puede mantener sus ejes habilitados en todo momento configurando este valor en el máximo de 255 milisegundos. Nuevamente, solo para repetir, puede mantener todos los ejes siempre habilitados mediante el ajuste $7=255.

### $ 8 - Aceleración, mm / seg ^ 2

Esta es la aceleración en mm / segundo / segundo. No tiene que entender lo que eso significa, baste decir que un valor más bajo proporciona una aceleración más suave, mientras que un valor más alto produce movimientos más estrictos y alcanza las velocidades de avance deseadas mucho más rápido. En términos técnicos, esta es la aceleración punto a punto de su máquina, independientemente de los ejes. Establezca este valor de aceleración tan alto como su eje más limitado le permita sin perder CUALQUIER paso. Por lo general, te gustaría darte un poco de búfer, porque si pierdes pasos, Grbl no tiene idea de que esto haya sucedido (los pasos a paso son control de bucle abierto) y continuarán.

### $ 9 - Desviación de unión, mm

El administrador de aceleración utiliza la desviación de la unión en la esquina para determinar qué tan rápido puede moverse a través de un camino. La matemática es un poco complicada, pero en general, los valores más altos generalmente dan un movimiento más rápido y posiblemente más brusco. Los valores más bajos hacen que el administrador de aceleración sea más cuidadoso y dará lugar a curvas más cuidadosas y lentas. Entonces, si se encuentra con problemas en los que su máquina intenta tomar una curva demasiado rápido, disminuya este valor para disminuir la velocidad. Si desea que su máquina se mueva más rápido a través de las uniones, aumente este valor para acelerarlo. Para los técnicos, haga clic en este [enlace](http://t.co/KQ5BvueY) para leer sobre el algoritmo de curvas de Grbl, que representa tanto la velocidad como el ángulo de unión con un método muy simple, eficiente y robusto.

### $ 10 - Arco, mm / segmento

Grbl hace círculos y arcos subdividiéndolos en pequeñas líneas diminutas. Probablemente nunca necesitará ajustar este valor, pero si encuentra que sus círculos son demasiado toscos (en realidad, ¿una décima de milímetro no es lo suficientemente precisa para usted? ¿Está en nanotecnología?), Puede ajustar esto. Los valores más bajos proporcionan mayor precisión, pero pueden llevar a problemas de rendimiento.

### $ 11 - Corrección de arco N, int

Esta es una configuración avanzada que no debe cambiarse a menos que existan circunstancias en las que deba hacerlo. Para hacer posibles los arcos G02 / 03 en Grbl, Grbl aproxima la ubicación del siguiente segmento de arco en una pequeña aproximación de ángulo. La corrección de arco N es el número de segmentos de arco aproximado realizados antes de que Grbl calcule un segmento de arco exacto para corregir la desviación del error de aproximación. El cómputo de estas ubicaciones exactas es computacionalmente costoso, pero hay algunos casos extremos en los que las aproximaciones de ángulos pequeños pueden introducir el error suficiente como para ser notables, como arcos muy muy pequeños con una longitud de segmento de arco grande. Cambie esta configuración solo si encuentra problemas extraños con los arcos, pero no se recomienda ir por debajo de 3, ya que esto puede provocar una inanición del búfer, donde los ejes se vuelven más lentos y más pequeños. Pero, los resultados pueden variar.

### $ 12 - N-decimal, int

Establezca cuántos decimales todos los valores de punto flotante que Grbl reporta. No es mucho más complicado que eso.

### $ 13 - Informe pulgadas, bool

Grbl v0.8 tiene una función de informe de posicionamiento en tiempo real para proporcionar comentarios de los usuarios sobre dónde se encuentra exactamente la máquina en ese momento. De forma predeterminada, está configurado para informar en mm, pero al enviar un $13=1comando, envía este indicador booleano a verdadero y la función de informe de estado ahora se informará en pulgadas. $13=0para volver a mm.

### $ 14 - inicio automático, bool

En un entorno CNC más profesional, los profesionales inician un trabajo cargando su programa y luego presionando el botón de "inicio de ciclo" en su máquina. Comienza el trabajo. Grbl hace lo mismo, pero no por defecto. Como herramienta de aprendizaje, "iniciamos el ciclo automáticamente" cualquier comando de código g que el usuario envíe a Grbl, para simplemente mover la máquina para ver si se mueve en la dirección que cree que debería ir, o simplemente para ver qué puede hacer su máquina . Esto hace que sea más fácil cargar su máquina con Grbl y aprender cómo funciona, en lugar de tener que presionar con diligencia el 'inicio del ciclo' cada vez que desee mover cualquiera de sus ejes. Una vez que aprenda su máquina y obtenga un buen manejo del código g, puede desactivar la función de 'inicio automático del ciclo' enviando el $14=0comando Grbl . (Es posible que necesite un reinicio suave o un ciclo de alimentación para cargar el cambio).

Otra forma de decir que podría ser:

Si $ 14 = 0, algunos comandos de gcode como Xn, Yn, etc. no sucederán cuando los ingrese en un terminal serial. Si $ 14 = 1 entonces ocurrirán los comandos de gcode motion.

Aparentemente, los cnc grandes no ejecutarán gcode hasta que el operador presione un botón de 'inicio de ciclo'. Se recibieron los comandos de gcode pero se pusieron en cola y se esperaba el botón de 'inicio del ciclo'. Tiene sentido si? Cuando $ 14 = 0, así es como actúa grbl también. Necesitas el botón! (En este caso, coloque el botón Feed-Hold también!).

Cuando $ 14 = 1, entonces el software grbl presiona automáticamente una versión de software del 'inicio del ciclo' cada vez que 'ingresa' una línea de comando de gcode a través de un terminal serial. Esto se hace para que sea más conveniente para usted ingresar comandos y ver que algo suceda sin tener que presionar el botón.

### $ 15 - Invertir paso habilitar, bool

Por defecto, el pin de habilitación paso a paso es alto para deshabilitar y bajo para habilitar. Si su configuración necesita lo contrario, simplemente invierta el pin de habilitación paso a paso escribiendo $15=1. Desactivar con $15=0. (Puede necesitar un ciclo de energía para cargar el cambio).

### $ 16 - límites duros, bool

Los interruptores de límite duro son una característica de seguridad para ayudar a evitar que su máquina se desplace demasiado lejos de los extremos del viaje y se estrelle o rompa algo caro. Básicamente, cablea algunos interruptores (mecánicos u ópticos) cerca del final del recorrido de cada eje, o donde quiera que sienta que podría haber problemas si su programa se mueve demasiado hacia donde no debería. Cuando el interruptor se dispara, detendrá inmediatamente todos los movimientos, apagará el refrigerante y el eje (si está conectado) y entrará en el modo de alarma, lo que lo obliga a revisar su máquina y reiniciar todo.

Para usar los límites duros con Grbl, los pasadores de límite se mantienen altos con una resistencia de pull-up interna, por lo que todo lo que tiene que hacer es cablear un interruptor normalmente abierto con el pasador y la tierra y habilitar los límites rígidos con $16=1. Eso es. (Deshabilite con $16=0) Si desea un límite para ambos extremos de viaje de un eje, solo conecte dos interruptores en paralelo con el pasador y la conexión a tierra, por lo que si alguno de ellos se dispara, se activa el límite duro.

Solo se debe saber que un evento de límite estricto se considera un evento crítico, en el que los steppers se detienen de inmediato y es probable que hayan perdido pasos. Grbl no tiene ningún comentario sobre la posición, por lo que no puede garantizar que tenga idea de dónde está. Por lo tanto, si se activa un límite rígido, Grbl entrará en un modo de ALARMA de bucle infinito, lo que le dará la oportunidad de revisar su máquina y forzarlo a reiniciar Grbl. Recuerde que es una característica puramente de seguridad.

Si tiene problemas con el interruptor de límite duro que se dispara constantemente después de reiniciar, un reinicio suave restablecerá Grbl en un estado de alarma, donde podrá acceder a la configuración y los comandos de Grbl, pero todos los códigos G y los bloques de inicio se bloquearán. Así que puedes deshabilitar la configuración de límites estrictos y luego $ X desbloquear la alarma. O bien, puede conectar un interruptor normalmente cerrado en serie con conexión a tierra a todos los interruptores de límite para desconectar los interruptores temporalmente, de modo que Grbl pueda mover sus ejes fuera de los interruptores.

### $ 17 - ciclo de inicio, bool

Ahh, homing. Algo que ha sido muy necesario en Grbl durante mucho tiempo. Ahora es totalmente compatible con v0.8. Para aquellos que acaban de iniciarse en CNC, el ciclo de inicio se utiliza para localizar de manera precisa y precisa la posición cero en una máquina (también conocida como máquina cero) cada vez que inicie su Grbl entre sesiones. En otras palabras, usted sabe exactamente dónde se encuentra en un momento dado, cada vez. Supongamos que comienza a mecanizar algo o está a punto de comenzar el siguiente paso en un trabajo y se apaga, reinicia Grbl y Grbl no tiene idea de dónde está. Te quedas con la tarea de averiguar dónde estás. Si tiene la orientación, siempre tiene el punto de referencia cero de la máquina desde donde ubicarse, por lo que todo lo que tiene que hacer es ejecutar el ciclo de referencia y continuar donde lo dejó.

Para configurar el ciclo de inicio para Grbl, necesita tener los interruptores de límite en una posición fija que no se muevan ni se muevan, o de lo contrario su punto de referencia se desordena. Por lo general, se configuran en el punto más alejado en + x, + y, + z de cada eje. Conecte sus interruptores de límite con los pines de límite y la conexión a tierra, al igual que con los límites rígidos, y habilite la orientación. Si tienes curiosidad, puedes usar tus interruptores de límite tanto para límites duros como para el inicio. Ellos juegan bien con los demás.

Por defecto, el ciclo de inicio de Grbl mueve el eje Z positivo primero para despejar el espacio de trabajo y luego mueve los ejes X e Y al mismo tiempo en la dirección positiva. Para configurar cómo se comporta su ciclo de inicio, hay más configuraciones de Grbl en la página que describen lo que hacen (y también las opciones de tiempo de compilación).

También, una cosa más a tener en cuenta, cuando el inicio está habilitado. Grbl bloqueará todos los comandos de código g hasta que realice un ciclo de inicio. Lo que significa que no hay movimientos de ejes, a menos que el bloqueo esté deshabilitado ($ X) pero más adelante. La mayoría, si no todos los controladores CNC, hacen algo similar, ya que es principalmente una función de seguridad para ayudar a los usuarios a cometer errores de posicionamiento, lo cual es muy fácil de hacer y entristece cuando un error arruina una parte. Si encuentra esto molesto o encuentra algún error extraño, háganoslo saber y trataremos de solucionarlo para que todos estén contentos. :)

NOTA: consulte config.h para obtener más opciones de inicio para usuarios avanzados. Puede desactivar el bloqueo de inicio en el inicio, configurar qué ejes se mueven primero durante un ciclo de inicio y en qué orden, y más.

### $ 18 - Homing dir invert mask, int: binario

De manera predeterminada, Grbl asume que los interruptores de límite de posición inicial están en la dirección positiva, primero moviendo el eje z positivo, luego los ejes xy positivos antes de tratar de ubicar con precisión la máquina cero yendo y viniendo lentamente alrededor del interruptor. Si su máquina tiene un interruptor de límite en la dirección negativa, la máscara de dirección de inicio puede invertir la dirección de los ejes. Funciona igual que la máscara invertida paso a paso, donde todo lo que tiene que hacer es establecer los pines de dirección del eje en 1 que desea invertir y los ejes buscarán ahora el pin de límite en la dirección negativa.

### $ 19 - alimentación Homing, mm / min

El ciclo de inicio primero busca los interruptores de límite a una tasa de búsqueda más alta y, una vez que los encuentra, se mueve a una velocidad de avance más lenta para ubicarse en la ubicación precisa de la máquina cero. La velocidad de avance de Homing es la velocidad de avance más lenta. Establezca este valor en cualquier valor de tasa que proporcione una ubicación de máquina cero precisa y repetible.

### $ 20 - Homing Seek, mm / min

La tasa de búsqueda de inicio es la tasa de búsqueda del ciclo de referencia, o la tasa a la que primero intenta encontrar los interruptores de límite. Ajústese a la velocidad que llegue a los interruptores de límite en un tiempo lo suficientemente corto sin chocar con los interruptores de límite si llegan demasiado rápido. Esta tasa de búsqueda se comporta de forma un poco diferente a la del controlador paso a paso principal. En lugar de la velocidad de un punto a otro, simplemente mueve todos los ejes a la misma velocidad individual, independientemente de cuántos ejes se mueven al mismo tiempo. Por lo tanto, el movimiento de búsqueda XY parecerá moverse un 41% más rápido que si lo moviera con un comando G1. (Puede deshabilitar esto en config.h si le molesta. Está ahí para acelerar el ciclo de inicio).

### $ 21 - Homing debounce, ms

Cada vez que se dispara un interruptor, algunos de ellos pueden tener un ruido eléctrico / mecánico que en realidad "rebota" la señal alta y baja durante unos pocos milisegundos antes de establecerse. Para resolver esto, debe rebotar la señal, ya sea por hardware con algún tipo de Acondicionador de señal o por software con un breve retraso para que la señal termine de rebotar. Grbl realiza una breve demora solo en la localización al ubicar la máquina en cero. Establezca este valor de retardo en lo que necesite su conmutador para obtener un inicio repetible. En la mayoría de los casos, 5-25 milisegundos está bien.

### $ 22 - Homing pull-off, mm

Para jugar bien con la función de límites duros, donde el homing puede compartir los mismos interruptores de límite, el ciclo de homing se moverá de todos los finales de límite por este recorrido de arranque una vez que se complete. En otras palabras, ayuda a prevenir el disparo accidental del límite duro después de un ciclo de inicio.

La configuración de velocidad de búsqueda de inicio controla qué tan rápido se mueve la maniobra de extracción, como un comando G1.

# Comandos internos de Grbl:

Técnicamente, los comandos de Grbl restantes no son comandos de configuración, pero los explicaremos aquí sin otra razón que no sea completa.

## Los tres comandos del estado actual: $G, $#, ?

Grbl proporciona tres comandos para informar su estado actual y dónde se encuentra. De los tres, uno de ellos es en tiempo real, respondiendo con la posición actual. Los otros dos no lo son, pero responden con cómo se procesarán los bloques entrantes según los estados establecidos por el código g, la configuración o los interruptores. La razón por la que no están en tiempo real es que Grbl tiene un búfer interno que almacena y planifica los movimientos del búfer. Los bloques de código de entrada simplemente se insertan en un nuevo movimiento al final del búfer cuando tiene espacio. Básicamente, el búfer garantiza el movimiento de movimiento de Grbl a las velocidades más rápidas y seguras posibles, recalculando constantemente en tiempo real a medida que los movimientos finalizan y entran nuevos.

### $G - Ver el estado del analizador gcode

Este comando imprime todos los modos de gcode activos que el analizador interpretará cualquier comando entrante. Estos modos incluyen el modo G20 / G21 pulgadas / mm, el sistema de coordenadas de trabajo activo G54-G59, el modo de movimiento activo G0 / G1 / G2 / G3, la selección del plano activo G17 / G18 / G19, el modo absoluto G90 / G91, la alimentación inversa G93 / G94 modo de velocidad, flujo de programa M0 / M1 / ​​M2, estado de husillo M3 / M4 / M5, estado de refrigerante M8 / M9, número de herramienta T y velocidad de alimentación activa F. También mostrará una lista de los $Sxconmutadores Grbl activos , como la $S1eliminación de bloques. Cuando se le llame, Grbl devolverá una línea como esta:

[G0 G54 G17 G21 G90 G94 M0 M5 M9 T0 F500.000]

### $# - Ver los parámetros de gcode.

Los parámetros de código G generalmente almacenan las compensaciones de coordenadas de trabajo G54-G59 y las posiciones iniciales de G28 / G30 (no se deben confundir con la orientación y el cero de la máquina. Estas posiciones de inicio se pueden configurar en cualquier lugar del espacio de la máquina mediante el G28.1 / G30.1 comandos). La mayoría de estos parámetros se escriben directamente en la EEPROM cada vez que se modifican y son persistentes. Lo que significa que seguirán siendo los mismos, independientemente del apagado, hasta que se cambien explícitamente.

Las coordenadas de trabajo de G54-G59 se pueden cambiar mediante el comando G10 L2 Pxo G10 L20 Pxdefinido por el estándar NIST gcode y el estándar EMC2 (linuxcnc.org). Las posiciones iniciales de G28 / G30 se pueden cambiar a través de G28.1los G30.1comandos y, respectivamente. Tenga en cuenta que G92 no es persistente o se mantiene en EEPROM según los estándares de g-code, y se restablecerá a cero cuando se reinicie Grbl. Por favor lea este estándar de g-code para entender cómo se usan.

Cuando $#se llama, Grbl responderá con las compensaciones almacenadas de las coordenadas de la máquina para cada sistema de la siguiente manera.

[G54:4.000,0.000,0.000]

[G55:4.000,6.000,7.000]

[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]

[G28:1.000,2.000,0.000]

[G30:4.000,6.000,0.000]

[G92:0.000,0.000,0.000]

En su mayor parte, estos desplazamientos no son particularmente útiles a menos que tenga habilitado el inicio. Si tiene el modo de inicio habilitado, estos son maravillosos, porque una vez que está en casa, siempre puede regresar a la misma posición almacenada utilizando los sistemas de coordenadas de trabajo con la precisión de su máquina. O si está creando varias partes y tiene una ruta de herramienta para una sola parte, todo lo que tiene que hacer es configurar sus sistemas de coordenadas de trabajo en la ubicación donde se realizará la próxima parte y volver a ejecutar ese mismo código.

### ? - Estado actual

El ?comando devuelve inmediatamente el estado activo de Grbl y la posición actual en tiempo real, tanto en las coordenadas de la máquina como en las coordenadas de trabajo. Esto se puede enviar en cualquier momento y funciona de forma asíncrona con todos los demás procesos que Grbl está realizando. La $13configuración de Grbl determina si informa milímetros o pulgadas. Cuando ?se presiona, Grbl responderá inmediatamente con algo como lo siguiente:

<Idle,MPos:5.529,0.560,7.000,WPos:1.529,-5.440,-0.000>

Los estados activos en los que Grbl puede estar son: **Inactivo, Cola, Ejecutar, Retener, Inicio, Alarma, Verificar**

* **Inactivo** : todos los sistemas están listos y están listos para cualquier cosa.
* **Cola** : los movimientos se ponen en cola en el búfer del planificador a la espera de que se ejecute un comando de inicio de ciclo. Ciertos procesos como el modo de verificación de código g no se pueden ejecutar mientras algo está en cola. Restablecer para borrar la cola.
* **Ejecutar** : indica que un ciclo se está ejecutando.
* **Retención** : una retención de alimentación está en proceso de ejecución o de desaceleración hasta detenerse. Una vez finalizada la retención, Grbl entrará en un estado de Cola, esperando un inicio de ciclo para reanudar el programa.
* **Hogar** : En medio de un ciclo de homing. NOTA: Las posiciones no se actualizan en vivo durante el ciclo de inicio, pero se establecerán en [0,0,0] una vez que estén listas.
* **Alarma** : Esto indica que algo salió mal o Grbl no conoce su posición. Este estado bloquea todos los comandos de código g, pero le permite interactuar con la configuración de Grbl si lo necesita. El bloqueo de alarma de "$ X" anula este estado y pone a Grbl en estado inactivo, lo que te permitirá mover las cosas de nuevo. Como se dijo antes, tenga cuidado con lo que está haciendo después de una alarma.
* **Compruebe** : Grbl está en modo de código g de verificación. Procesará y responderá a todos los comandos de código g, pero no a movimiento ni a nada. Una vez que se apaga con otro comando '$ C', Grbl se restablecerá.

## Otros comandos $C $X $H ~ ! Ctrl-X

### $C - Compruebe el modo gcode

Esto alterna el analizador gcode de Grbl para que todos los bloques entrantes los procesen completamente, como lo haría en la operación normal, pero no mueve ninguno de los ejes, ignora las paradas y apaga el eje y el refrigerante. Esto pretende ser una forma de proporcionar al usuario una forma de verificar cómo su nuevo programa de código g se adapta al analizador y monitor de Grbl para detectar cualquier error. (Y, finalmente, esto también comprobará si hay infracciones de límite suave).

Cuando se desactiva, Grbl realizará un restablecimiento automático (^ X). Esto es para dos propósitos. Simplifica un poco la gestión del código. Pero, también evita que los usuarios comiencen un trabajo cuando sus modos de código g no son lo que creen que son. Un reinicio del sistema siempre le da al usuario un comienzo nuevo y consistente.

NOTA: Eventualmente, el modo de chequeo de código de código podría volver a ser factorizado para permitir una función de "reanudación de programa" en Grbl. Esto significa que Grbl podría iniciar un programa de código g en cualquier lugar. Pasaría internamente por todo el programa de código g hasta el punto de reanudación central del programa deseado para establecer todos los estados y ubicaciones del analizador, moverse a ese punto de inicio y comenzar a ejecutar / moverse desde ese punto en adelante. Por ejemplo, supongamos que tuvo que hacer una parada en línea en medio de un programa porque olvidó algo o tiene la herramienta incorrecta en el eje. Tu parte está bien y necesitas reiniciar el programa. En este momento, tendrás que iniciar el programa desde el principio y dejar que se mueva físicamente y se ejecute hasta el punto en el que te detuviste. Si tienes un programa largo, esto podría tomar un tiempo. En su lugar, un "programa de reanudación"

### $X - Matar el bloqueo de alarma

El modo de alarma de Grbl es un estado cuando algo ha salido gravemente mal, como un límite duro o un aborto durante un ciclo, o si Grbl no conoce su posición. De forma predeterminada, si tiene habilitada la función de inicio y enciende el Arduino, Grbl entra en estado de alarma porque no conoce su posición. El modo de alarma bloqueará todos los bloques de código g hasta que se haya realizado el ciclo de inicio '$ H'. O si un usuario necesita anular el bloqueo de alarma para mover sus ejes fuera de sus interruptores de límite, por ejemplo, el bloqueo de alarma de "$ X" anulará los bloqueos y permitirá que las funciones de código G funcionen nuevamente.

Pero, pise con cuidado !! Esto solo debe ser usado en situaciones de emergencia. Es probable que la posición se haya perdido y que Grbl no esté donde crees que está. Por lo tanto, se recomienda utilizar el modo incremental G91 para hacer movimientos cortos. Luego, realice un ciclo de inicio o reinicie inmediatamente después.

### $H - Ejecutar ciclo de homing

Este comando es la única forma de realizar el ciclo de inicio en Grbl. Anteriormente, G28 y G30 iniciarían automáticamente el ciclo de inicio, pero esto es incorrecto de acuerdo con los estándares de g-code. Homing es un comando completamente separado manejado por el controlador. G28 y G30 solo se mueven a una posición 'inicial' / predefinida que se almacena en los parámetros del código g, que se pueden ubicar en cualquier lugar de la máquina.

CONSEJO: después de ejecutar un ciclo de inicio, más bien trotar manualmente todo el tiempo a una posición en medio de su volumen de trabajo. Puede establecer una posición predefinida de G28 o G30 para que sea su posición posterior al giro, más cerca de donde estará mecanizando. Para configurar estos, primero deberá mover su máquina hacia donde desee que se mueva después de la orientación. Escriba G28.1 (o G30.1) para que Grbl guarde esa posición. Entonces, después de '$ H' homing, puedes simplemente ingresar 'G28' (o 'G30') y se moverá allí de forma automática. En general, solo movería el eje XY hacia el centro y dejaría el eje Z arriba. Esto asegura que no hay posibilidad de que la herramienta en el eje no se enganche en nada.

### ~ - Inicio del ciclo

Este es el comando de inicio o reanudación del ciclo que se puede emitir en cualquier momento, ya que es un comando en tiempo real. Cuando Grbl tiene movimientos en cola en su búfer y está listo para ~comenzar, el comando de inicio del ciclo comenzará a ejecutar el búfer y Grbl comenzará a mover los ejes. Sin embargo, de forma predeterminada, el inicio del ciclo automático está habilitado, por lo que los nuevos usuarios no necesitarán este comando a menos que se realice una retención de alimentación. Cuando se ejecuta una retención de alimentación, el inicio del ciclo reanudará el programa. El inicio del ciclo solo será efectivo cuando haya movimientos en el búfer listos para funcionar y no funcionará con ningún otro proceso como el inicio.

### ! - Mantener la alimentación

El comando de retención de alimentación detendrá el ciclo activo mediante una desaceleración controlada, para no perder la posición. También es en tiempo real y puede activarse en cualquier momento. Una vez finalizado o en pausa, Grbl esperará hasta que se emita un comando de inicio de ciclo para reanudar la programación. La retención de alimentación solo puede hacer una pausa en un ciclo y no afectará el inicio ni ningún otro proceso.

Si necesita detener un programa a mitad de ciclo y no puede permitirse perder la posición, realice una retención de alimentación para que Grbl lleve todo a una parada controlada. Una vez finalizado, puede emitir un reinicio. Siempre intente ejecutar una retención de alimentación cuando la máquina esté funcionando antes de reiniciar, excepto, por supuesto, si hay alguna situación de emergencia.

### Ctrl-x - Restablecer Grbl

Este es el comando de reinicio suave de Grbl. Es en tiempo real y puede ser enviado en cualquier momento. Como su nombre lo indica, restablece Grbl, pero de forma controlada, conserva la posición de su máquina y todo esto sin apagar su Arduino. Las únicas veces que un reinicio de software podría perder la posición es cuando hay problemas como si los steppers murieran mientras se movían. Si es así, informará si se ha perdido el seguimiento de Grbl de la posición de la máquina. Esto se debe a que una desaceleración descontrolada puede llevar a pasos perdidos, y Grbl no tiene comentarios sobre cuánto perdió (este es el problema con los motores paso a paso en general). De lo contrario, Grbl solo se reiniciará, ejecutará las líneas de inicio y continuará su camino alegre.

Tenga en cuenta que se recomienda hacer un reinicio por software antes de comenzar un trabajo. Esto garantiza que no haya modos g-code activos que deban ser de jugar o configurar su máquina. Por lo tanto, su máquina siempre comenzará de nuevo y de manera constante, y su máquina hará lo que usted espera.