

\$\$ - Grbl-Einstellungen anzeigen

Um die Einstellungen anzuzeigen, geben Sie \$\$ ein und drücken Sie die Eingabetaste, nachdem Sie eine Verbindung zu Grbl hergestellt haben. Grbl sollte mit einer Liste der aktuellen Systemeinstellungen antworten, wie im folgenden Beispiel gezeigt. Alle diese Einstellungen sind dauerhaft und werden im EEPROM gespeichert. Wenn Sie sie also ausschalten, werden sie beim nächsten Einschalten Ihres Arduino wieder geladen.

Das x von \$ x = val gibt eine bestimmte Einstellung an, während val der Einstellungswert ist. In früheren Versionen von Grbl stand neben jeder Einstellung eine Beschreibung in () Klammern, die in Grbl v1.1 + leider nicht mehr enthalten ist. Dies wurde durchgeführt, um wertvollen Flash-Speicher freizugeben und die neuen Funktionen in Version 1.1 hinzuzufügen. Die meisten guten GUIs helfen jedoch, indem sie Beschreibungen für Sie anhängen, damit Sie wissen, was Sie suchen.

Einstellungen und Beispielwerte Beschreibung

\$ 0 = 10	Schrittpuls, Mikrosekunden
\$ 1 = 25	Schritt Leerlaufverzögerung, Millisekunden
\$ 2 = 0	Schritt Port invertieren, maskieren
\$ 3 = 0	Richtungsport invertieren, maskieren
\$ 4 = 0	Schritt aktivieren invertieren, boolesch
\$ 5 = 0	Limit Pins invertieren, boolesch
\$ 6 = 0	Sondenstift invertieren, boolesch
\$ 10 = 1	Statusbericht, Maske
\$ 11 = 0,010	Verbindungsabweichung, mm
\$ 12 = 0,002	Lichtbogentoleranz, mm
\$ 13 = 0	Zoll melden, boolesch
\$ 20 = 0	Weiche Grenzen, boolesch
\$ 21 = 0	Harte Grenzen, boolesch
\$ 22 = 1	Referenzierzyklus, boolesch
\$ 23 = 0	Homing dir invertieren, maskieren
\$ 24 = 25.000	Homing-Vorschub, mm / min
\$ 25 = 500.000	Homing-Suche, mm / min
\$ 26 = 250	Homing Debounce, Millisekunden
\$ 27 = 1.000	Homing-Abzug, mm
30 \$ = 1000.	Maximale Spindeldrehzahl, U / min
\$ 31 = 0.	Min. Spindeldrehzahl, U / min
\$ 32 = 0	Lasermodus, boolesch
\$ 100 = 250.000	x Schritte / mm
\$ 101 = 250.000	Y-Schritte / mm
\$ 102 = 250.000	Z Schritte / mm
\$ 110 = 500.000	X Max. Rate, mm / min
\$ 111 = 500.000	Y Maximale Rate, mm / min
\$ 112 = 500.000	Z Maximale Rate, mm / min
\$ 120 = 10.000	X Beschleunigung, mm / s ^ 2
\$ 121 = 10.000	Y Beschleunigung, mm / s ^ 2
\$ 122 = 10.000	Z Beschleunigung, mm / s ^ 2
\$ 130 = 200.000	X Max. Federweg, mm
\$ 131 = 200.000	Y Max. Federweg, mm
\$ 132 = 200.000	Z Max. Federweg, mm

\$ x = val - Grbl-Einstellung speichern

Der Befehl \$ x = val speichert oder ändert eine Grbl-Einstellung. Dies kann manuell erfolgen, indem dieser Befehl gesendet wird, wenn eine Verbindung über ein serielles Terminalprogramm mit Grbl hergestellt wird. Die meisten Grbl-GUIs erledigen dies jedoch als benutzerfreundliche Funktion für Sie.

Um z. B. mit der Mikrosekunden-Schritimpulsoption auf 10us würden Sie dies eingeben, gefolgt von einer Eingabe:

```
$ 0 = 10
```

Wenn alles gut gegangen ist, antwortet Grbl mit einem "OK" und diese Einstellung wird im EEPROM gespeichert und bleibt für immer oder bis Sie sie ändern. Sie können überprüfen, ob Grbl Ihre Einstellung korrekt empfangen und gespeichert hat, indem Sie \$\$ eingeben, um die Systemeinstellungen erneut anzuzeigen.

Grbls \$ x = val Einstellungen und was sie bedeuten

HINWEIS: Von Grbl v0.9 auf Grbl v1.1 wurden nur 10-Dollar-Statusberichte geändert und neue Einstellungen für die maximale Spindeldrehzahl von 30 US-Dollar / 31 US-Dollar und 32 US-Dollar für den Lasermodus hinzugefügt. Alles andere ist das gleiche.

\$ 0 - Schritimpuls, Mikrosekunden

Schrittreiber sind für eine bestimmte minimale Schritimpulslänge ausgelegt. Überprüfen Sie das Datenblatt oder probieren Sie einfach einige Zahlen aus. Sie möchten die kürzesten Impulse, die die Schrittreiber zuverlässig erkennen können. Wenn die Impulse zu lang sind, können Probleme auftreten, wenn Sie das System mit sehr hohen Vorschub- und Impulsraten betreiben, da sich die Schritimpulse überlappen können. Wir empfehlen etwa 10 Mikrosekunden, was der Standardwert ist.

\$ 1 - Schritt Leerlaufverzögerung, Millisekunden

Jedes Mal, wenn Ihre Stepper eine Bewegung ausführen und zum Stillstand kommen, verzögert Grbl das Deaktivieren der Stepper um diesen Wert. ODER Sie können Ihre Achsen jederzeit aktiviert lassen (um die Position zu halten), indem Sie diesen Wert auf maximal 255 Millisekunden einstellen. Um es noch einmal zu wiederholen, können Sie alle Achsen immer aktiviert lassen, indem Sie \$ 1 = 255 setzen.

Die Stepper-Leerlaufsperrzeit ist die Zeitdauer, die Grbl die Stepper vor dem Deaktivieren gesperrt hält. Je nach System können Sie dies auf Null setzen und deaktivieren. Bei anderen benötigen Sie möglicherweise 25 bis 50 Millisekunden, um sicherzustellen, dass Ihre Achsen vollständig zum Stillstand kommen, bevor Sie sie deaktivieren. Dies soll dazu beitragen, Maschinenmotoren zu berücksichtigen, die nicht gerne längere Zeit eingeschaltet bleiben, ohne etwas zu tun. Denken Sie auch daran, dass sich einige Steppertreiber nicht daran erinnern, bei welchem Mikroschritt sie angehalten haben. Wenn Sie sie also wieder aktivieren, können einige "verlorene" Schritte auftreten. In diesem Fall lassen Sie einfach Ihre Stepper über \$ 1 = 255 aktiviert.

\$ 2 - Schritt Port invertieren, maskieren

Diese Einstellung invertiert das Schritimpulssignal. Standardmäßig beginnt ein Schrittsignal bei normal-niedrig und geht bei einem Schritimpulsereignis hoch. Nach einer durch \$ 0 eingestellten Schritimpulszeit wird der Pin bis zum nächsten Schritimpulsereignis auf niedrig zurückgesetzt. Invertiert wechselt das Schritimpulsverhalten während des Impulses von normal-hoch zu niedrig

und zurück zu hoch. Die meisten Benutzer müssen diese Einstellung nicht verwenden, dies kann jedoch für bestimmte CNC-Schrittreiber mit besonderen Anforderungen hilfreich sein. Beispielsweise kann eine künstliche Verzögerung zwischen dem Richtungsstift und dem Schrittpuls durch Invertieren des Stufenstifts erzeugt werden.

Diese Invertierungsmaskeneinstellung ist ein Wert, der die zu invertierenden Achsen als Bitflags speichert. Sie müssen wirklich nicht vollständig verstehen, wie es funktioniert. Sie müssen lediglich den Einstellungswert für die Achsen eingeben, die Sie invertieren möchten. Wenn Sie beispielsweise die X- und Z-Achse invertieren möchten, senden Sie \$ 2 = 5 an Grbl, und die Einstellung sollte jetzt \$ 2 = 5 lauten (Schrittport-Invertierungsmaske: 00000101).

Einstellwert	Maske	Invertieren X	Invertieren Y	Invertieren Z
0	00000000	N	N	N
1	00000001	J	N	N
2	00000010	N	Y	N
3	00000011	J	J	N
4	00000100	N	N	Y
5	00000101	Y	N	Y
6	00000110	N	Y	Y
7	00000111	Y	Y	Y

\$ 3 - Richtungsport invertieren, maskieren

Diese Einstellung invertiert das Richtungssignal für jede Achse. Standardmäßig geht Grbl davon aus, dass sich die Achsen in eine positive Richtung bewegen, wenn das Richtungsstift-Signal niedrig ist, und in eine negative Richtung, wenn der Stift hoch ist. Bei einigen Maschinen bewegen sich Achsen häufig nicht so. Diese Einstellung invertiert das Richtungsstift-Signal für die Achsen, die sich in die entgegengesetzte Richtung bewegen.

Diese Einstellung zum Umkehren der Maske funktioniert genau wie die Umkehrmaske des Schrittports und speichert die zu invertierenden Achsen als Bitflags. Um diese Einstellung zu konfigurieren, müssen Sie lediglich den Wert für die Achsen senden, die Sie invertieren möchten. Verwenden Sie die obige Tabelle. Wenn Sie beispielsweise nur die Richtung der Y-Achse invertieren möchten, senden Sie \$ 3 = 2 an Grbl, und die Einstellung sollte jetzt \$ 3 = 2 lauten (dir port invert mask: 00000010).

\$ 4 - Schritt aktivieren invertieren, boolesch

Standardmäßig ist der Stepper-Aktivierungsstift zum Deaktivieren hoch und zum Aktivieren niedrig. Wenn Ihr Setup das Gegenteil benötigt, invertieren Sie einfach den Stepper-Aktivierungsstift, indem Sie \$ 4 = 1 eingeben. Deaktivieren Sie mit \$ 4 = 0. (Möglicherweise ist ein Aus- und Wiedereinschalten erforderlich, um die Änderung zu laden.)

\$ 5 - Limit Pins invertieren, boolesch

Standardmäßig werden die Begrenzungsstifte mit dem internen Pull-up-Widerstand des Arduino normal hoch gehalten. Wenn ein Grenzwert niedrig ist, interpretiert Grbl dies als ausgelöst. Für das umgekehrte Verhalten invertieren Sie einfach die Grenzwertstifte, indem Sie \$ 5 = 1 eingeben. Deaktivieren Sie mit \$ 5 = 0. Möglicherweise benötigen Sie einen Aus- und Wiedereinschaltvorgang, um die Änderung zu laden.

HINWEIS: Für eine erweiterte Verwendung ist der interne Pull-up-Widerstand an den Begrenzungsstiften in config.h möglicherweise deaktiviert.

\$ 6 – Limit Pins invertieren, boolesch

Standardmäßig wird der Sondenstift mit dem internen Pull-up-Widerstand des Arduino normal hoch

gehalten. Wenn der Sondenstift niedrig ist, interpretiert Grbl dies als ausgelöst. Für das umgekehrte Verhalten invertieren Sie einfach den Sondenstift, indem Sie \$ 6 = 1 eingeben. Deaktivieren Sie mit \$ 6 = 0. Möglicherweise benötigen Sie einen Aus- und Wiedereinschaltvorgang, um die Änderung zu laden.

\$ 10 - Statusbericht, Maske

Diese Einstellung bestimmt, welche Grbl-Echtzeitdaten an den Benutzer zurückgemeldet werden, wenn ein '?' Statusbericht wird gesendet. Diese Daten umfassen den aktuellen Ausführungsstatus, die Echtzeitposition, die Echtzeit-Vorschubrate, die Pin-Zustände, die aktuellen Überschreibungswerte, die Pufferzustände und die aktuell ausgeführte G-Code-Zeilenummer (sofern über Kompilierungsoptionen aktiviert).

Standardmäßig enthält die neue Berichtsimplementierung in Grbl v1.1 + nahezu alles im Standardstatusbericht. Viele Daten sind ausgeblendet und werden nur angezeigt, wenn sie sich ändern. Dies erhöht die Effizienz gegenüber dem alten Berichtsstil erheblich und ermöglicht es Ihnen, schnellere Updates zu erhalten und dennoch mehr Daten über Ihren Computer zu erhalten. Die Schnittstellendokumentation beschreibt, wie es funktioniert, und das meiste davon gilt nur für GUI-Entwickler oder Neugierige.

Um die Dinge einfach und konsistent zu halten, bietet Grbl v1.1 nur zwei Berichtsoptionen. Diese sind in erster Linie nur für Benutzer und Entwickler gedacht, um die Einrichtung zu erleichtern.

Der Positionstyp kann angegeben werden, um entweder die Maschinenposition (MPos :) oder die Arbeitsposition (WPos :) anzuzeigen, jedoch nicht mehr beide gleichzeitig. Das Aktivieren der Arbeitsposition ist in bestimmten Szenarien nützlich, wenn mit Grbl über ein serielles Terminal direkt interagiert wird. Die Berichterstellung zur Maschinenposition sollte jedoch standardmäßig verwendet werden.

Verwendungsdaten von Grbls Planer- und seriellen Empfangspuffern können aktiviert werden. Dies zeigt die Anzahl der Blöcke oder Bytes, die in den jeweiligen Puffern verfügbar sind. Dies wird im Allgemeinen verwendet, um die Leistung von Grbl beim Testen einer Streaming-Schnittstelle zu bestimmen. Dies sollte standardmäßig deaktiviert sein.

Verwenden Sie die folgende Tabelle, um Berichtsoptionen zu aktivieren und zu deaktivieren. Fügen Sie einfach die aufgelisteten Werte hinzu, die Sie aktivieren möchten, und speichern Sie sie, indem Sie Grbl Ihren Einstellungswert senden. Beispielsweise ist der Standardbericht mit Maschinenposition und ohne Einstellung für Pufferdatenberichte \$ 10 = 1. Wenn Arbeitsposition und Pufferdaten gewünscht werden, beträgt die Einstellung \$ 10 = 2.

Berichtstyp Wert Beschreibung

Positionstyp 0	WPos aktivieren:	MPos deaktivieren:.
Positionstyp 1	MPos aktivieren:.	WPos deaktivieren:.
Buffer Data 2	Enabled Buf:	Das Feld wird mit dem verfügbaren Planer und dem seriellen RX-Puffer angezeigt.

\$ 11 - Verbindungsabweichung, mm

Die Junction-Abweichung wird vom Beschleunigungsmanager verwendet, um zu bestimmen, wie schnell er sich durch Liniensegment-Junctions eines G-Code-Programmpfads bewegen kann. Wenn der G-Code-Pfad beispielsweise eine scharfe 10-Grad-Drehung aufweist und sich die Maschine mit voller Geschwindigkeit bewegt, hilft diese Einstellung zu bestimmen, wie viel die Maschine verlangsamen muss, um sicher durch die Kurve zu fahren, ohne Schritte zu verlieren.

Wie wir es berechnen, ist etwas kompliziert, aber im Allgemeinen führen höhere Werte zu einer schnelleren Bewegung durch Ecken und erhöhen gleichzeitig das Risiko, Schritte und Positionierungen zu verlieren. Niedrigere Werte machen den Beschleunigungsmanager

vorsichtiger und führen zu vorsichtigen und langsameren Kurvenfahrten. Wenn Sie also auf Probleme stoßen, bei denen Ihre Maschine versucht, eine Kurve zu schnell zu fahren, verringern Sie diesen Wert, um sie beim Betreten von Kurven zu verlangsamen. Wenn Sie möchten, dass sich Ihre Maschine schneller durch Kreuzungen bewegt, erhöhen Sie diesen Wert, um sie zu beschleunigen. Wenn Sie neugierig sind, klicken Sie auf diesen Link, um mehr über den Kurvenalgorithmus von Grbl zu erfahren, der sowohl Geschwindigkeit als auch Verbindungswinkel mit einer sehr einfachen, effizienten und robusten Methode berücksichtigt.

\$ 12 – ARC toleranz, mm

Grbl rendert G2 / G3-Kreise, -Bögen und -Helices, indem es sie in winzige Linien unterteilt, sodass die Genauigkeit der Bogenverfolgung niemals unter diesem Wert liegt. Sie müssen diese Einstellung wahrscheinlich nie anpassen, da 0,002 mm weit unter der Genauigkeit der meisten CNC-Maschinen liegen. Wenn Sie jedoch feststellen, dass Ihre Kreise zu grob sind oder die Bogenverfolgung langsam ausgeführt wird, passen Sie diese Einstellung an. Niedrigere Werte bieten eine höhere Genauigkeit, können jedoch zu Leistungsproblemen führen, indem Grbl mit zu vielen kleinen Linien überladen wird. Alternativ führen höhere Werte zu einer geringeren Genauigkeit, können jedoch die Lichtbogenleistung beschleunigen, da Grbl weniger Linien zu verarbeiten hat.

Für Neugierige ist die Bogentoleranz definiert als der maximale senkrechte Abstand von einem Liniensegment, dessen Endpunkte auf dem Bogen liegen, auch bekannt als Akkord. Mit einigen grundlegenden Geometrien lösen wir die Länge der Liniensegmente auf, um den Bogen zu verfolgen, der diese Einstellung erfüllt. Das Modellieren von Bögen auf diese Weise ist großartig, da die Bogenliniensegmente automatisch angepasst und mit der Länge skaliert werden, um eine optimale Leistung bei der Bogenverfolgung zu gewährleisten, ohne dabei an Genauigkeit zu verlieren.

\$ 13 - Zoll melden, Boolescher Wert

Grbl verfügt über eine Echtzeit-Positionierungsberichts-funktion, die dem Benutzer ein Feedback darüber gibt, wo sich die Maschine genau zu diesem Zeitpunkt befindet, sowie über Parameter für Koordinatenversätze und -prüfungen. Standardmäßig ist die Berichterstellung in mm festgelegt. Wenn Sie jedoch einen Befehl \$ 13 = 1 senden, senden Sie dieses boolesche Flag an true, und diese Berichtsfunktionen werden jetzt in Zoll gemeldet. **\$ 13 = 0, um auf mm zurückzusetzen.**

\$ 20 - Weiche Limits, boolesch

Weiche Grenzwerte sind eine Sicherheitsfunktion, die verhindert, dass Ihre Maschine zu weit und über die Grenzen des Fahrens hinaus fährt, abstürzt oder etwas Teureres kaputt macht. Es funktioniert, indem man die maximalen Verfahrgrenzen für jede Achse kennt und weiß, wo sich Grbl in Maschinenkoordinaten befindet. Immer wenn eine neue G-Code-Bewegung an Grbl gesendet wird, wird überprüft, ob Sie versehentlich Ihren Maschinenraum überschritten haben oder nicht. Wenn Sie dies tun, gibt Grbl überall einen sofortigen Vorschubstopp aus, schaltet die Spindel und das Kühlmittel ab und stellt dann den Systemalarm ein, der auf das Problem hinweist. Die Maschinenposition bleibt danach erhalten, da dies nicht auf einen sofortigen erzwungenen Stopp wie bei harten Grenzen zurückzuführen ist.

HINWEIS: Für weiche Grenzwerte müssen die Referenzfahrt aktiviert und die Einstellungen für den maximalen Hub der Achse genau sein, da Grbl wissen muss, wo sie sich befinden. \$ 20 = 1 zum Aktivieren und \$ 20 = 0 zum Deaktivieren.

\$ 21 - Harte Grenzen, boolesch

Harte Limits funktionieren im Grunde genauso wie weiche Limits, verwenden jedoch stattdessen

physische Schalter. Grundsätzlich verdrahten Sie einige Schalter (mechanisch, magnetisch oder optisch) gegen Ende der Bewegung jeder Achse oder wo immer Sie das Gefühl haben, dass es Probleme geben könnte, wenn sich Ihr Programm zu weit dorthin bewegt, wo es nicht sein sollte. Wenn der Schalter ausgelöst wird, stoppt er sofort alle Bewegungen, schaltet das Kühlmittel und die Spindel (falls angeschlossen) ab und wechselt in den Alarmmodus, wodurch Sie gezwungen sind, Ihre Maschine zu überprüfen und alles zurückzusetzen.

Um mit Grbl harte Grenzwerte zu verwenden, werden die Grenzwertstifte mit einem internen Pull-up-Widerstand hoch gehalten. Sie müssen also nur einen normalerweise offenen Schalter mit Stift und Masse anschließen und harte Grenzwerte mit 21 USD = 1 aktivieren. (Deaktivieren Sie mit \$ 21 = 0.) Wir empfehlen dringend, Maßnahmen zur Verhinderung elektrischer Störungen zu ergreifen. Wenn Sie eine Begrenzung für beide Verfahrenen einer Achse wünschen, verdrahten Sie einfach zwei Schalter parallel zu Stift und Masse. Wenn einer von beiden auslöst, wird die harte Begrenzung ausgelöst.

Beachten Sie, dass ein Hard-Limit-Ereignis als kritisches Ereignis angesehen wird, bei dem Stepper sofort anhalten und wahrscheinlich Schritte verloren haben. Grbl hat kein Feedback zur Position, kann also nicht garantieren, dass es keine Ahnung hat, wo es sich befindet. Wenn also ein hartes Limit ausgelöst wird, wechselt Grbl in einen Endlosschleifen-ALARM-Modus, sodass Sie Ihre Maschine überprüfen und Grbl zurücksetzen können. Denken Sie daran, es ist eine reine Sicherheitsfunktion.

\$ 22 - Homing-Zyklus, boolesch

Für diejenigen, die gerade erst in die CNC eingeführt wurden, wird der Referenzierzyklus verwendet, um bei jedem Start Ihres Grbl zwischen den Sitzungen eine bekannte und konsistente Position auf einer Maschine genau und präzise zu lokalisieren. Mit anderen Worten, Sie wissen jedes Mal genau, wo Sie sich zu einem bestimmten Zeitpunkt befinden. Angenommen, Sie beginnen mit der Bearbeitung von etwas oder stehen kurz vor dem nächsten Schritt in einem Job, und der Strom geht aus. Sie starten Grbl neu und Grbl hat keine Ahnung, wo dies daran liegt, dass Stepper eine Regelung sind. Sie müssen herausfinden, wo Sie sich befinden. Wenn Sie eine Referenzfahrt haben, haben Sie immer den Nullpunkt-Referenzpunkt der Maschine, von dem aus Sie suchen können. Sie müssen also nur den Referenzierzyklus ausführen und dort weitermachen, wo Sie aufgehört haben.

Um den Referenzierzyklus für Grbl einzurichten, müssen sich die Endschalter in einer festen Position befinden, die nicht gestoßen oder bewegt wird, da sonst Ihr Referenzpunkt durcheinander gerät. Normalerweise werden sie am entferntesten Punkt in + x, + y, + z jeder Achse eingerichtet. Verdrahten Sie Ihre Endschalter mit den Endstiften, fügen Sie einen empfohlenen RC-Filter hinzu, um elektrische Störungen zu reduzieren, und aktivieren Sie die Referenzfahrt. Wenn Sie neugierig sind, können Sie Ihre Endschalter sowohl für harte Grenzwerte als auch für Referenzfahrten verwenden. Sie spielen gut miteinander.

Stellen Sie vor dem ersten Versuch des Referenzierzyklus sicher, dass Sie alles richtig eingerichtet haben, da sich das Referenzieren sonst möglicherweise seltsam verhält. Stellen Sie zunächst sicher, dass sich Ihre Maschinenachsen gemäß den kartesischen Koordinaten in die richtige Richtung bewegen (rechte Regel). Wenn nicht, beheben Sie es mit der Einstellung \$ 3 Richtungsumkehr. Stellen Sie zweitens sicher, dass Ihre Endschalterstifte in den Statusberichten von Grbl nicht als "ausgelöst" angezeigt werden. Wenn ja, überprüfen Sie Ihre Verkabelung und Einstellungen. Stellen Sie schließlich sicher, dass Ihre maximalen Reiseeinstellungen für 13x USD ziemlich genau sind (innerhalb von 20%), da Grbl diese Werte verwendet, um zu bestimmen, wie weit es nach den Referenzschaltern suchen soll.

Standardmäßig verschiebt der Referenzierzyklus von Grbl zuerst die Z-Achse positiv, um den Arbeitsbereich freizugeben, und bewegt dann gleichzeitig die X- und Y-Achse in die positive Richtung. Um festzulegen, wie sich Ihr Referenzierzyklus verhält, finden Sie auf der Seite weitere Grbl-Einstellungen, die beschreiben, was sie tun (und auch Optionen zur Kompilierungszeit).

Noch eine Sache zu beachten, wenn Homing aktiviert ist. Grbl sperrt alle G-Code-Befehle, bis Sie einen Referenzierzyklus durchführen. Das heißt, keine Achsenbewegungen, es sei denn, die Sperre ist deaktiviert (\$ X), aber dazu später mehr. Die meisten, wenn nicht alle CNC-Steuerungen, tun etwas Ähnliches, da dies hauptsächlich eine Sicherheitsfunktion ist, um zu verhindern, dass Benutzer einen Positionierungsfehler machen. Dies ist sehr einfach und traurig, wenn ein Fehler ein Teil ruiniert. Wenn Sie dies ärgerlich finden oder seltsame Fehler finden, lassen Sie es uns bitte wissen und wir werden versuchen, daran zu arbeiten, damit alle glücklich sind. :) :)

HINWEIS: Weitere Konfigurationsoptionen für fortgeschrittene Benutzer finden Sie in config.h. Sie können die Referenzierungssperre beim Start deaktivieren, konfigurieren, welche Achsen sich während eines Referenzierzyklus zuerst bewegen und in welcher Reihenfolge, und vieles mehr.

\$ 23 - Homing dir invertieren, Maske

Standardmäßig geht Grbl davon aus, dass sich Ihre Zielsuchschalter in der positiven Richtung befinden. Bewegen Sie zuerst die positive Z-Achse und dann die positive x-y-Achse, bevor Sie versuchen, die Maschinennull genau zu lokalisieren, indem Sie langsam um den Schalter hin und her gehen. Wenn Ihre Maschine einen Endschalter in negativer Richtung hat, kann die Referenzrichtungsmaske die Richtung der Achsen umkehren. Dies funktioniert genau wie die Masken für Step-Port-Invertierung und Richtungs-Port-Invertierung. Sie müssen lediglich den Wert in der Tabelle senden, um anzugeben, welche Achsen Sie in die entgegengesetzte Richtung invertieren und suchen möchten.

\$ 24 - Homing Feed, mm / min

Der Referenzierzyklus sucht zuerst mit einer höheren Suchrate nach den Endschaltern, und nachdem er sie gefunden hat, bewegt er sich mit einer langsameren Vorschubgeschwindigkeit nach Hause an den genauen Ort der Maschinennull. Die Referenziervorschubrate ist die langsamere Vorschubrate. Stellen Sie diesen Wert auf einen beliebigen Ratenwert ein, der eine wiederholbare und präzise Lokalisierung der Maschinennull ermöglicht.

\$ 25 - Homing-Suche, mm / min

Die Referenzierungssuchrate ist die Suchrate des Referenzierzyklus oder die Rate, mit der zuerst versucht wird, die Endschalter zu finden. Stellen Sie in kürzester Zeit die Geschwindigkeit ein, mit der die Endschalter erreicht werden, ohne gegen Ihre Endschalter zu stoßen, wenn diese zu schnell eingehen.

\$ 26 - Homing Schalter Prellen /Debounce, Millisekunden

Immer wenn ein Schalter ausgelöst wird, können einige von ihnen elektrisches / mechanisches Rauschen aufweisen, das das Signal einige Millisekunden lang hoch und niedrig "abprallt", bevor es sich einstellt. Um dies zu lösen, müssen Sie das Signal entweder durch Hardware mit einer Art von Signal entprellen Signalaufbereiter oder per Software mit einer kurzen Verzögerung, damit das Signal abprallt. Grbl führt eine kurze Verzögerung durch, die nur beim Lokalisieren der Maschine Null erfolgt. Stellen Sie diesen Verzögerungswert auf den Wert ein, den Ihr Switch benötigt, um eine wiederholbare Referenzfahrt zu erhalten. In den meisten Fällen sind 5-25 Millisekunden in Ordnung.

\$ 27 - Homing-Abzug, mm

Um mit der Funktion für harte Grenzwerte, bei der die Referenzfahrt dieselben Endschalter verwenden kann, gut zu spielen, werden alle Endschalter durch diesen Abfahrweg nach Abschluss des Referenzfahrvorgangs abgeschaltet. Mit anderen Worten, es hilft, ein versehentliches Auslösen des harten Grenzwerts nach einem Referenzierzyklus zu verhindern. Stellen Sie sicher,

dass dieser Wert groß genug ist, um den Endschalter zu löschen. Wenn nicht, gibt Grbl einen Alarmfehler aus, wenn er nicht gelöscht werden kann.

\$ 30 - Maximale Spindeldrehzahl, U / min

Hiermit wird die Spindeldrehzahl für den maximalen 5-V-PWM-Pin-Ausgang eingestellt. Wenn Sie beispielsweise 10000 U / min auf 5 V einstellen möchten, programmieren Sie \$ 30 = 10000. Programmieren Sie für 255 U / min bei 5 V \$ 30 = 255. Wenn ein Programm versucht, eine höhere Spindeldrehzahl einzustellen, die über der maximalen Spindeldrehzahl von 30 USD liegt, gibt Grbl nur die maximalen 5 V aus, da es nicht schneller gehen kann. Standardmäßig bezieht Grbl die max-min-Drehzahlen linear auf 5V-0,02V PWM-Pin-Ausgang in 255 gleich abgestandenen Schritten. Wenn der PWM-Pin 0 V anzeigt, bedeutet dies, dass die Spindel deaktiviert ist. Beachten Sie, dass in config.h zusätzliche Konfigurationsoptionen verfügbar sind, um die Funktionsweise zu optimieren.

\$ 31 - Min. Spindeldrehzahl, U / min

Hiermit wird die Spindeldrehzahl für den minimalen PWM-Pin-Ausgang von 0,02 V eingestellt (0 V sind deaktiviert). Niedrigere Drehzahlwerte werden von Grbl akzeptiert, aber der PWM-Ausgang wird 0,02 V nicht unterschreiten, außer wenn die Drehzahl Null ist. Bei Null ist die Spindel deaktiviert und der PWM-Ausgang ist 0V.

\$ 32 - Lasermodus, boolesch

Wenn aktiviert, bewegt sich Grbl kontinuierlich durch aufeinanderfolgende G1-, G2- oder G3-Bewegungsbefehle, wenn es mit einer S-Spindeldrehzahl (Laserleistung) programmiert wird. Der PWM-Pin der Spindel wird bei jeder Bewegung sofort aktualisiert, ohne anzuhalten. Bitte lesen Sie die GRBL-Laserdokumentation und die Dokumentation Ihres Lasergeräts, bevor Sie diesen Modus verwenden. Laser sind sehr gefährlich. Sie können Ihre Sicht sofort dauerhaft schädigen und Brände verursachen. Grbl übernimmt keine Verantwortung für Probleme, die die Firmware gemäß der GPL-Lizenz verursachen kann.

Wenn deaktiviert, arbeitet Grbl wie immer und stoppt die Bewegung mit jedem S-Spindeldrehzahlbefehl. Dies ist der Standardbetrieb einer Fräsmaschine, um eine Pause zu ermöglichen, damit die Spindel die Drehzahl ändern kann.

\$ 100, \$ 101 und \$ 102 - [X, Y, Z] Schritte / mm

Grbl muss wissen, wie weit jeder Schritt das Tool in der Realität bringen wird. Um Schritte / mm für eine Achse Ihrer Maschine zu berechnen, müssen Sie Folgendes wissen:

Der pro Umdrehung Ihres Schrittmotors zurückgelegte mm. Dies hängt von Ihren Riemenantriebsrädern oder der Steigung der Leitspindel ab.

Die vollen Schritte pro Umdrehung Ihrer Stepper (normalerweise 200)

Die Mikroschritte pro Schritt Ihres Controllers (normalerweise 1, 2, 4, 8 oder 16). Tipp: Die

Verwendung hoher Mikroschrittweite (z. B. 16) kann das Drehmoment Ihres Schrittmotors verringern. Verwenden Sie daher den niedrigsten Wert, der Ihnen die gewünschte Achsauflösung und komfortable Laufeigenschaften bietet.

Die Schritte / mm können dann wie folgt berechnet werden: Schritte_per_mm = (Schritte_per_revolution * Mikroschritte) / mm_per_rev

Berechnen Sie diesen Wert für jede Achse und schreiben Sie diese Einstellungen in Grbl.

\$ 110, \$ 111 und \$ 112 - [X, Y, Z] Maximale Rate, mm / min

Hiermit wird die maximale Geschwindigkeit festgelegt, mit der sich jede Achse bewegen kann. Wann immer Grbl eine Bewegung plant, prüft es, ob die Bewegung dazu führt, dass eine dieser einzelnen Achsen ihre maximale Rate überschreitet. In diesem Fall wird die Bewegung verlangsamt, um sicherzustellen, dass keine der Achsen ihre maximalen Ratengrenzen überschreitet. Dies bedeutet, dass jede Achse ihre eigene unabhängige Geschwindigkeit hat, was äußerst nützlich ist, um die typischerweise langsamere Z-Achse zu begrenzen.

Der einfachste Weg, diese Werte zu bestimmen, besteht darin, jede Achse einzeln zu testen, indem Sie die Einstellungen für die maximale Rate langsam erhöhen und verschieben. Um beispielsweise die X-Achse zu testen, senden Sie Grbl so etwas wie G0 X50 mit einer ausreichenden Verfahrstrecke, damit die Achse auf ihre maximale Geschwindigkeit beschleunigt. Sie werden wissen, dass Sie die maximale Rate erreicht haben, wenn Ihre Stepper stehen bleiben. Es macht ein bisschen Geräusche, sollte aber Ihre Motoren nicht verletzen. Geben Sie eine Einstellung ein, die 10 bis 20% unter diesem Wert liegt, damit Sie Verschleiß, Reibung und die Masse Ihres Werkstücks / Werkzeugs berücksichtigen können. Wiederholen Sie dies für Ihre anderen Achsen.

HINWEIS: Mit dieser Einstellung für die maximale Rate werden auch die G0-Suchraten festgelegt.

\$ 120, \$ 121, \$ 122 - [X, Y, Z] Beschleunigung, mm / s ²

Dadurch werden die Achsenbeschleunigungsparameter in mm / Sekunde / Sekunde eingestellt. Vereinfacht gesagt lässt ein niedrigerer Wert die Bewegung von Grbl langsamer werden, während ein höherer Wert engere Bewegungen ergibt und die gewünschten Vorschubgeschwindigkeiten viel schneller erreicht. Ähnlich wie bei der Einstellung der Maximalrate hat jede Achse ihren eigenen Beschleunigungswert und ist unabhängig voneinander. Dies bedeutet, dass eine mehrachsige Bewegung nur so schnell beschleunigt wird, wie es die Achse mit dem niedrigsten Beitrag kann.

Wie bei der Einstellung der maximalen Rate besteht die einfachste Möglichkeit, die Werte für diese Einstellung zu bestimmen, darin, jede Achse einzeln mit langsam ansteigenden Werten zu testen, bis der Motor abgewürgt ist. Schließen Sie dann Ihre Beschleunigungseinstellung mit einem Wert ab, der 10 bis 20% unter diesem absoluten Maximalwert liegt. Dies sollte Verschleiß, Reibung und Massenträgheit berücksichtigen. Wir empfehlen dringend, dass Sie einige G-Code-Programme mit Ihren neuen Einstellungen trocken testen, bevor Sie sie festlegen. Manchmal ist die Belastung Ihrer Maschine unterschiedlich, wenn Sie sich in allen Achsen zusammen bewegen.

\$ 130, \$ 131, \$ 132 - [X, Y, Z] Max. Federweg, mm

Dies legt den maximalen Federweg von Ende zu Ende für jede Achse in mm fest. Dies ist nur nützlich, wenn Sie Soft Limits (und Homing) aktiviert haben, da dies nur von der Soft Limit-Funktion von Grbl verwendet wird, um zu überprüfen, ob Sie Ihre Maschinenlimits mit einem Bewegungsbefehl überschritten haben.

Grbl '\$' Befehle

Die \$ system-Befehle bieten dem Benutzer zusätzliche Steuerelemente, z. B. das Drucken von Rückmeldungen zum aktuellen modalen Status des G-Code-Parsers oder das Ausführen des Referenzierzyklus. In diesem Abschnitt wird erläutert, was diese Befehle sind und wie sie verwendet werden.

\$\$ und \$ x = val - Grbl-Einstellungen anzeigen und schreiben

Weitere Informationen zum Anzeigen und Schreiben von Einstellungen sowie zum Erlernen der Einstellungen finden Sie unter Grbl v1.1-Konfiguration.

\$ # - Gcode-Parameter anzeigen

G-Code-Parameter speichern die Koordinatenversatzwerte für G54-G59-Arbeitskoordinaten, vordefinierte G28 / G30-Positionen, G92-Koordinatenversatz, Werkzeuglängenversätze und Abtastung (nicht offiziell, aber wir haben sie trotzdem hinzugefügt). Die meisten dieser Parameter werden bei jeder Änderung direkt in das EEPROM geschrieben und sind dauerhaft. Dies bedeutet, dass sie unabhängig vom Ausschalten gleich bleiben, bis sie explizit geändert werden. Die nicht persistenten Parameter, die beim Zurücksetzen oder Aus- und Wiedereinschalten nicht beibehalten werden, sind G92, G43.1-Werkzeuglängenkorrekturen und die G38.2-Prüfdaten.

Die Arbeitskoordinaten G54-G59 können über den Befehl G10 L2 Px oder G10 L20 Px geändert werden, der durch den NIST-Gcode-Standard und den EMC2-Standard (linuxcnc.org) definiert ist. Vordefinierte Positionen von G28 / G30 können über die Befehle G28.1 bzw. G30.1 geändert werden.

Wenn \$ # aufgerufen wird, antwortet Grbl mit den gespeicherten Offsets von den Maschinenkoordinaten für jedes System wie folgt. TLO bezeichnet den Werkzeuglängenversatz (für die Standard-Z-Achse) und PRB bezeichnet die Koordinaten des letzten Prüfzyklus, wobei das Suffix: 1 angibt, ob die letzte Prüfung erfolgreich war, und: 0 als nicht erfolgreich.

```
[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]
[TLO:0.000]
[PRB:0.000,0.000,0.000:0]
```

\$ G - Status des Gcode-Parsers anzeigen

Dieser Befehl druckt alle aktiven Gcode-Modi im G-Code-Parser von Grbl. Wenn dieser Befehl an Grbl gesendet wird, antwortet er mit einer Nachricht, die mit einem [GC: Indikator wie folgt beginnt:

```
[GC: G0 G54 G17 G21 G90 G94 M0 M5 M9 T0 S0.0 F500.0]
```

Diese aktiven Modi bestimmen, wie der nächste G-Code-Block oder Befehl vom G-Code-Parser von Grbl interpretiert wird. Für diejenigen, die noch keine Erfahrung mit G-Code- und CNC-Bearbeitung haben, versetzt der Modus den Parser in einen bestimmten Zustand, sodass Sie dem Parser nicht ständig mitteilen müssen, wie er ihn analysieren soll. Diese Modi sind in Gruppen organisiert, die als "modale Gruppen" bezeichnet werden und nicht gleichzeitig logisch aktiv sein können. Beispielsweise legt die Modalgruppe Einheiten fest, ob Ihr G-Code-Programm in Zoll oder in Millimetern interpretiert wird.

Eine kurze Liste der von Grbl unterstützten Modalgruppen finden Sie unten. Vollständigere und detailliertere Beschreibungen finden Sie jedoch auf der LinuxCNC-Website. Die fett gedruckten G-Code-Befehle geben die Standardmodi beim Einschalten oder Zurücksetzen von Grbl an.

Modal Group	Member Words
Motion Mode	G0 , G1, G2, G3, G38.2, G38.3, G38.4, G38.5, G80
Coordinate System Select	G54 , G55, G56, G57, G58, G59
Plane Select	G17 , G18, G19
Distance Mode	G90 , G91
Arc IJK Distance Mode	G91.1
Feed Rate Mode	G93, G94
Units Mode	G20, G21
Cutter Radius Compensation	G40
Tool Length Offset	G43.1, G49
Program Mode	M0 , M1, M2, M30
Spindle State	M3, M4, M5
Coolant State	M7, M8, M9

Zusätzlich zu den G-Code-Parser-Modi meldet Grbl die aktive T-Werkzeugnummer, die S-Spindeldrehzahl und die F-Vorschubgeschwindigkeit, die beim Zurücksetzen standardmäßig auf 0 gesetzt sind. Für diejenigen, die neugierig sind, passen diese nicht ganz in nette Modalgruppen, sind aber genauso wichtig für die Bestimmung des Parser-Status.

Beachten Sie, dass diese Liste nicht die nicht-modale G-Code-Befehlsgruppe enthält und nicht im \$ G-Parser-Bericht aufgeführt ist, da sie nur die aktuelle Zeile betreffen, in der sie befohlen werden. Der Vollständigkeit halber sind hier die nicht-modalen Befehle aufgeführt unterstützt von Grbl:

Unterstützte nichtmodale Befehle

G4, G10 L2, G10 L20, G28, G30, G28.1, G30.1, G53, G92, G92.1

\$ I - Build-Informationen anzeigen

Dies gibt dem Benutzer eine Rückmeldung über die Grbl-Version und das Erstellungsdatum des Quellcodes. Optional kann \$ I auch eine kurze Zeichenfolge speichern, um festzustellen, mit welcher CNC-Maschine Sie kommunizieren, wenn Sie mehr als eine Maschine mit Grbl haben. Um diese Zeichenfolge festzulegen, senden Sie Grbl \$ I = xxx, wobei xxx Ihre Anpassungszeichenfolge mit weniger als 80 Zeichen ist. Diese Zeichenfolge wird als Großbuchstaben gespeichert, Leerzeichen entfernt und kann nur alphanumerische Zeichen enthalten. Wenn Sie Grbl das nächste Mal mit einer Build-Information von \$ I abfragen, druckt Grbl diese Zeichenfolge nach der Version und dem Erstellungsdatum.

HINWEIS: Einige OEMs blockieren möglicherweise den Zugriff auf das Überschreiben der Build-Info-Zeichenfolge, damit sie dort Produktinformationen und Codes speichern können.

\$ N - Startblöcke anzeigen

\$ Nx sind die Startblöcke, die Grbl jedes Mal ausführt, wenn Sie Grbl einschalten oder Grbl zurücksetzen. Mit anderen Worten, ein Startblock ist eine Zeile mit G-Code, die Grbl automatisch ausführen kann, um die Standardeinstellungen für den G-Code festzulegen, oder alles andere, was Grbl bei jedem Start Ihres Computers ausführen muss. Grbl kann standardmäßig zwei G-Code-Blöcke speichern.

Wenn Sie mit Grbl verbunden sind, geben Sie \$ N ein und geben Sie dann ein. Grbl sollte mit etwas Kurzem antworten wie:

\$ N0 =
\$ N1 =
in Ordnung

Es ist nicht viel zu tun, aber dies bedeutet nur, dass in Zeile \$ N0 kein G-Code-Block gespeichert ist, damit Grbl beim Start ausgeführt werden kann. \$ N1 ist die nächste auszuführende Zeile.

\$ Nx = Zeile - Startblock speichern

WICHTIG: Seien Sie sehr vorsichtig, wenn Sie Bewegungsbefehle (G0 / 1, G2 / 3, G28 / 30) in den Startblöcken speichern. Diese Bewegungsbefehle werden jedes Mal ausgeführt, wenn Sie Grbl zurücksetzen oder einschalten. Wenn Sie also eine Notsituation haben und E-Stop und Reset durchführen müssen, kann und wird eine Bewegung des Startblocks die Situation schnell verschlimmern. Platzieren Sie auch keine Befehle, die Daten im EEPROM speichern, wie z. B. G10 / G28.1 / G30.1. Dies führt dazu, dass Grbl diese Daten bei jedem Start und Zurücksetzen ständig neu schreibt, wodurch das EEPROM Ihres Arduino möglicherweise abgenutzt wird.

Die typische Verwendung für einen Startblock besteht einfach darin, Ihre bevorzugten Modalzustände, wie z. B. den G20-Zoll-Modus, immer standardmäßig auf ein anderes Arbeitskoordinatensystem festzulegen oder einem Benutzer die Möglichkeit zu geben, eine vom Benutzer geschriebene einzigartige Funktion auszuführen, die er benötigt für ihr verrücktes Projekt.

Geben Sie zum Setzen eines Startblocks \$ N0 = gefolgt von einem gültigen G-Code-Block und einer Eingabe ein. Grbl führt den Block aus, um zu überprüfen, ob er gültig ist, und antwortet dann mit einem OK oder einem Fehler: um Ihnen mitzuteilen, ob er erfolgreich ist oder etwas schief gelaufen ist. Wenn ein Fehler auftritt, wird er von Grbl nicht gespeichert.

Angenommen, Sie möchten Ihren ersten Startblock \$ N0 verwenden, um Ihre G-Code-Parser-Modi wie G54-Arbeitskoordinate, G20-Zoll-Modus, G17-XY-Ebene festzulegen. Sie würden \$ N0 = G20 G54 G17 mit einer Eingabe eingeben und sollten eine gute Antwort sehen. Sie können dann überprüfen, ob es gespeichert wurde, indem Sie \$ N eingeben. Jetzt sollte eine Antwort wie \$ N0 = G20G54G17 angezeigt werden.

Sobald Sie einen Startblock im EEPROM von Grbl gespeichert haben, wird bei jedem Start oder Zurücksetzen Ihr Startblock zurückgedruckt, beginnend mit einem Open-Chevron> und einer: OK-Antwort von Grbl, um anzuzeigen, ob er in Ordnung war. Im vorherigen Beispiel sehen Sie also:

```
rbl 1.1d ['$' für Hilfe]  
> G20G54G17: ok
```

Wenn Sie mehrere G-Code-Startblöcke haben, werden diese bei jedem Start in der richtigen Reihenfolge an Sie zurückgedruckt. Und wenn Sie einen der Startblöcke löschen möchten (z. B. Block 0), geben Sie \$ N0 = ein, ohne dass dem Gleichheitszeichen etwas folgt.

HINWEIS: Es gibt zwei Varianten, wann Startblöcke nicht ausgeführt werden. Erstens wird es nicht ausgeführt, wenn Grbl aus Sicherheitsgründen in einem ALARM-Status initialisiert oder einen ALARM-Status über eine \$ X-Entsperrung beendet. Adressieren und löschen Sie den ALARM immer und beenden Sie ihn mit einem Reset, bei dem die Startblöcke bei der Initialisierung ausgeführt werden. Zweitens, wenn Sie das Referenzieren aktiviert haben, werden die Startblöcke unmittelbar nach einem erfolgreichen Referenzierungszyklus ausgeführt, nicht beim Start.

\$ C - Überprüfen Sie den Gcode-Modus

Dadurch wird der Gcode-Parser des Grbl so umgeschaltet, dass alle eingehenden Blöcke wie im normalen Betrieb vollständig verarbeitet werden. Es bewegt jedoch keine der Achsen, ignoriert Verweilzeiten und schaltet die Spindel und das Kühlmittel aus. Dies soll dem Benutzer eine Möglichkeit bieten, zu überprüfen, wie sein neues G-Code-Programm mit dem Parser von Grbl abschneidet, und auf Fehler zu überwachen (und auf Soft-Limit-Verstöße zu prüfen, falls aktiviert).

Nach dem Ausschalten führt Grbl einen automatischen Soft-Reset durch (^ X). Dies dient zwei Zwecken. Dies vereinfacht die Codeverwaltung ein wenig. Es verhindert jedoch auch, dass Benutzer einen Job starten, wenn ihre G-Code-Modi nicht so sind, wie sie glauben. Ein System-Reset gibt dem Benutzer immer einen neuen, konsistenten Start.

\$ X - Alarmsperre beenden

Der Alarmmodus von Grbl ist ein Zustand, in dem ein kritischer Fehler aufgetreten ist, z. B. ein hartes Limit oder ein Abbruch während eines Zyklus, oder wenn Grbl seine Position nicht kennt. Wenn Sie das Referenzieren aktiviert und das Arduino eingeschaltet haben, wechselt Grbl standardmäßig in den Alarmstatus, da es seine Position nicht kennt. Der Alarmmodus sperrt alle G-Code-Befehle, bis der Referenzierzyklus '\$ H' ausgeführt wurde. Wenn ein Benutzer die Alarmsperre überschreiben muss, um seine Achsen von den Endschaltern zu entfernen, überschreibt beispielsweise die Alarmsperre '\$ X' die Sperren und ermöglicht, dass die G-Code-Funktionen wieder funktionieren.

Aber vorsichtig sein !! Dies sollte nur in Notsituationen verwendet werden. Die Position ist wahrscheinlich verloren gegangen, und Grbl ist möglicherweise nicht dort, wo Sie denken, dass es ist. Es wird daher empfohlen, den inkrementellen G91-Modus zu verwenden, um kurze Züge auszuführen. Führen Sie dann einen Referenzierzyklus durch oder setzen Sie ihn unmittelbar danach zurück.

Wie bereits erwähnt, werden Startzeilen nach einem \$ X-Befehl nicht ausgeführt. Immer zurücksetzen, wenn Sie den Alarm gelöscht und das Szenario behoben haben, das ihn verursacht hat. Wenn Grbl auf Leerlauf zurückgesetzt wird, werden die Startleitungen wie gewohnt ausgeführt.

\$ H - Referenzfahrtzyklus ausführen

Dieser Befehl ist die einzige Möglichkeit, den Referenzierzyklus in Grbl durchzuführen. Einige andere Motion Controller geben einen speziellen G-Code-Befehl zum Ausführen eines Referenzierzyklus an. Dies ist jedoch gemäß den G-Code-Standards falsch. Homing ist ein völlig separater Befehl, der von der Steuerung ausgeführt wird.

TIPP: Nach dem Ausführen eines Referenzierzyklus müssen Sie die ganze Zeit manuell zu einer Position in der Mitte Ihres Arbeitsbereichsvolumens joggen. Sie können eine vordefinierte G28- oder G30-Position als Post-Homing-Position festlegen, die näher an der Stelle liegt, an der Sie die Bearbeitung durchführen. Um diese einzustellen, müssen Sie Ihre Maschine zuerst an die Stelle bewegen, an die sie nach dem Referenzieren verschoben werden soll. Geben Sie G28.1 (oder G30.1) ein, damit Grbl diese Position speichert. Nach dem Referenzieren von '\$ H' können Sie einfach 'G28' (oder 'G30') eingeben und es bewegt sich automatisch magisch dorthin. Im Allgemeinen würde ich einfach die XY-Achse in die Mitte verschieben und die Z-Achse oben lassen. Dies stellt sicher, dass keine Gefahr besteht, dass das Werkzeug in der Spindel stört, und dass es nichts erfasst.

\$ J = line - Laufbewegung ausführen

Neu in Grbl v1.1 führt dieser Befehl eine spezielle Tippbewegung aus. Es gibt drei Hauptunterschiede zwischen einer Tippbewegung und einer Bewegung, die von einer G-Code-Zeile befohlen wird.

Wie bei normalen G-Code-Befehlen können mehrere Tippbewegungen in den Planerpuffer eingereiht werden, aber das Tippen kann einfach durch einen Echtzeitbefehl zum Tippen oder Halten des Vorschubs abgebrochen werden. Grbl hält sofort den aktuellen Lauf und löscht dann automatisch die Puffer von verbleibenden Befehlen.

Tippbefehle sind völlig unabhängig vom Status des G-Code-Parsers. Es werden keine Modi wie der inkrementelle Distanzmodus G91 geändert. Sie müssen also nicht mehr sicherstellen, dass Sie es anschließend wieder in den absoluten Distanzmodus G90 ändern. Dies verringert die Wahrscheinlichkeit, mit den falschen aktivierten G-Code-Modi zu beginnen.

Wenn Soft-Limits aktiviert sind, gibt jeder Tippbefehl, der ein Soft-Limit überschreitet, einfach einen Fehler zurück. Es wird kein Alarm ausgelöst, wie dies bei einem normalen G-Code-Befehl der Fall wäre. Dies ermöglicht eine viel angenehmere und flüssigere GUI- oder Joystick-Interaktion.

Das Ausführen eines Tippvorgangs erfordert eine bestimmte Befehlsstruktur, wie unten beschrieben:

Die ersten drei Zeichen müssen '\$ J =' sein, um den Tipp anzuzeigen.

Der Tippbefehl folgt unmittelbar nach dem '=' und funktioniert wie ein normaler G1-Befehl.

Die Vorschubgeschwindigkeit wird nur in G94-Einheiten pro Minute interpretiert. Ein früherer G93-Zustand wird beim Tippen ignoriert.

Erforderliche Wörter:

XYZ: Ein oder mehrere Achsenwörter mit Zielwert.

F - Vorschubgeschwindigkeitswert. HINWEIS: Jeder Lauf erfordert diesen Wert und wird nicht als modal behandelt.

Optionale Wörter: Jog wird basierend auf dem aktuellen G20 / G21- und G90 / G91-G-Code-Parser-Status ausgeführt. Wenn eines der folgenden optionalen Wörter übergeben wird, wird dieser Status nur für einen Befehl überschrieben.

G20 oder G21 - Zoll- und Millimeter-Modus

G90 oder G91 - Absolute und inkrementelle Abstände

G53 - Maschinenkoordinaten einfahren

Alle anderen G-Codes, M-Codes und Wertwörter werden im Tippbefehl nicht akzeptiert.

Leerzeichen und Kommentare sind im Befehl zulässig. Diese werden vom Pre-Parser entfernt.

Beispiel: G21 und G90 sind vor dem Joggen aktive Modalzustände. Dies sind sequentielle Befehle.

\$ J = X10.0 Y-1.5 F100 bewegt sich im Arbeitskoordinatenrahmen (WPos) mit einer Vorschubgeschwindigkeit von 100 zu X = 10,0 mm und Y = -1,5 mm.

\$ J = G91 G20 X0.5 F10 bewegt sich mit einer Vorschubgeschwindigkeit von 10 um +0,5 Zoll (12,7 mm) auf X = 22,7 mm (WPos). Beachten Sie, dass G91 und G20 nur auf diesen Tippbefehl angewendet werden.

\$ J = G53 Y5.0 F10 bewegt die Maschine mit einer Vorschubgeschwindigkeit von 10 auf Y = 5,0 mm im Maschinenkoordinatenrahmen (MPos). Wenn der Arbeitskoordinatenversatz für die y-Achse 2,0 mm beträgt, beträgt Y 3,0 mm in (WPos).

Jog-Befehle verhalten sich fast identisch mit normalem G-Code-Streaming. Jeder Tippbefehl gibt

ein "OK" zurück, wenn die Tippbewegung analysiert wurde und für die Ausführung eingerichtet ist. Wenn ein Befehl ungültig ist oder ein Soft-Limit überschreitet, gibt Grbl einen 'Fehler:' zurück. Mehrere Tippbefehle können nacheinander in die Warteschlange gestellt werden.

\$ RST = \$, \$ RST = # und \$ RST = * - Grbl-Einstellungen und -Daten auf die Standardeinstellungen zurücksetzen

Diese Befehle sind nicht in der Haupt-Hilfemeldung von Grbl \$ aufgeführt, stehen jedoch zur Verfügung, damit Benutzer Teile oder alle EEPROM-Daten von Grbl wiederherstellen können. Hinweis: Grbl wird nach Ausführung eines dieser Befehle automatisch zurückgesetzt, um sicherzustellen, dass das System korrekt initialisiert wird.

\$ RST = \$: Löscht die \$\$ Grbl-Einstellungen und stellt sie auf die Standardeinstellungen zurück, die durch die Standardeinstellungsdatei definiert werden, die beim Kompilieren von Grbl verwendet wird. Oft bauen OEMs ihre Grbl-Firmwares mit den maschinenspezifischen empfohlenen Einstellungen. Dies bietet Benutzern und OEMs eine schnelle Möglichkeit, wieder auf den ersten Platz zurückzukehren, wenn etwas schief gelaufen ist oder wenn ein Benutzer von vorne beginnen möchte.

\$ RST = #: Löscht und setzt alle im EEPROM gespeicherten G54-G59-Arbeitskoordinaten-Offsets und G28 / 30-Positionen auf Null. Dies sind im Allgemeinen die Werte, die im Ausdruck der \$ # -Parameter angezeigt werden. Dies bietet eine einfache Möglichkeit, diese zu löschen, ohne dies manuell für jeden Satz mit einem Befehl G20 L2 / 20 oder G28.1 / 30.1 tun zu müssen.

\$ RST = *: Hiermit werden alle von Grbl verwendeten EEPROM-Daten gelöscht und wiederhergestellt. Dies umfasst \$\$ Einstellungen, \$ # Parameter, \$ N Startzeilen und \$ I Build Info String. Beachten Sie, dass dadurch nicht das gesamte EEPROM gelöscht wird, sondern nur die Datenbereiche, die Grbl verwendet. Verwenden Sie zum vollständigen Löschen das EEPROM-Beispielprojekt der Arduino IDE.

HINWEIS: Einige OEMs können einige oder alle dieser Befehle einschränken, um zu verhindern, dass bestimmte von ihnen verwendete Daten gelöscht werden.

\$ SLP - Aktiviert den Schlafmodus

Dieser Befehl versetzt Grbl in einen Ruhezustand ohne Stromversorgung, wodurch die Aktivierungsstifte für Spindel, Kühlmittel und Schritt abgeschaltet und alle Befehle blockiert werden. Es kann nur durch Soft-Reset oder Aus- und Wiedereinschalten beendet werden. Nach der erneuten Initialisierung wechselt Grbl automatisch in den ALARM-Status, da nicht sicher ist, wo er sich befindet, da die Stepper deaktiviert sind.

Diese Funktion ist nützlich, wenn Sie am Ende eines Jobs alles automatisch ausschalten müssen, indem Sie diesen Befehl am Ende Ihres G-Code-Programms hinzufügen. Es wird jedoch dringend empfohlen, Befehle hinzuzufügen, um Ihren Computer zuerst auf einen Computer zu verschieben sicherer Parkplatz vor diesem Schlafbefehl. Es sollte auch betont werden, dass Sie eine zuverlässige CNC-Maschine haben sollten, die alles deaktiviert, wenn es soll, wie Ihre Spindel. Grbl haftet nicht für Schäden, die dadurch entstehen können. **Es ist niemals eine gute Idee, Ihre Maschine unbeaufsichtigt zu lassen. Verwenden Sie diesen Befehl daher mit äußerster Vorsicht!**

