BOM

1. Arduino / Funduino UNO
2. Zasilacz 12-24V
3. Przewód USB
4. CNC-Shield V3
5. 3-4szt. stepstick A4988 lub kompatybilny
6. 3 lub 4 silniki krokowe bipoplarne lub unipolarne 6 - przewodowe (5 przewodowe odpadają)
7. 3szt lub 6szt EndStop
8. przewody połączeniowe do zasilacza, endstopów i silników

Przygotowanie sprzętu:

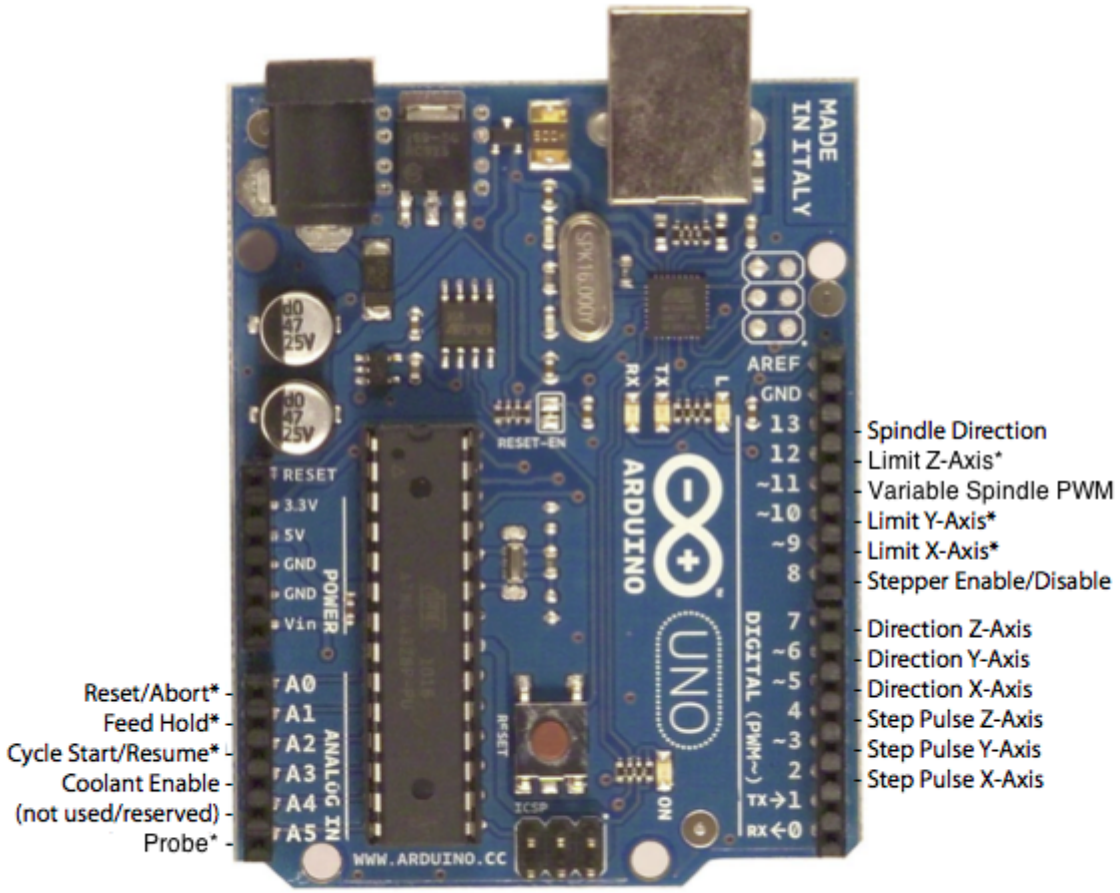
Pobierz i wgraj do Arduino UNO poniższy plik:
https://github.com/gnea/grbl/releases/download/v1.1h.20190825/grbl_v1.1h.20190825.hex

Dla windowsa najsympatyczniejszą opcją na wgranie pliku hex jest X-Loader:
https://www.hobbytronics.co.uk/download/XLoader.zip

Pobierz i zainstaluj na komputerze UGS:
https://winder.github.io/ugs_website/

Pobierz i zainstaluj najnowszy postprocesor do Fusion 360:
https://cam.autodesk.com/posts/download.php?name=grbl&type=post

Pierwsze kroki



* - Indicates input pins. Held high with internal pull-up resistors.

Najpierw połącz się z Grbl za pomocą wybranego terminala szeregowego.

Ustaw szybkość transmisji na 115200 jako 8-N-1 (8-bitów, bez parzystości i 1 bit stopu).

Wejścia i wyjścia GRBL v1.1 w połączeniu z Arduino Uno:	
Analog 0	STOP - awaryjny
Analog 1	Pauza / Drzwi otwarte
Analog 2	Start / Wznów
Analog 3	Chłodzenie
Analog 4	Mgła olejowa / ALARM
Analog 5	Wejście Sondy pomiarowej
Digital 13	Kierunek wrzeciona
Digital 12	Z - End Stop
Digital 11	Spindle / Laser Enable PWM
Digital 10	Y - End Stop
Digital 9	X - End Stop
Digital 8	Sygnał Enable / Disable dla Sterowników (stepsticków)
Digital 7	Kierunek Z
Digital 6	Kierunek Y
Digital 5	Kierunek Y
Digital 4	Krok Z
Digital 3	Krok Y
Digital 2	Krok X

Po połączeniu powinieneś otrzymać komunikat Grbl, który wygląda tak:

Grbl 0.9i ['$' for help]
Wpisz $ i naciśnij enter, aby Grbl wydrukował komunikat pomocy. Nie powinieneś widzieć żadnego lokalnego echa znaku $ i wprowadź. Grbl powinien odpowiedzieć:

$$ (view Grbl settings)
$# (view # parameters)
$G (view parser state)
$I (view build info)
$N (view startup blocks)
$x=value (save Grbl setting)
$Nx=line (save startup block)
$C (check gcode mode)
$X (kill alarm lock)
$H (run homing cycle)
~ (cycle start)
! (feed hold)
? (current status)
ctrl-x (reset Grbl)
Polecenia '$' to polecenia systemowe Grbl używane do dostrajania ustawień, przeglądania lub zmiany stanów i trybów działania Grbl oraz rozpoczynania cyklu bazowania. Ostatnie cztery polecenia inne niż -'$' to polecenia sterujące w czasie rzeczywistym, które można wysłać w dowolnym momencie, bez względu na to, co robi Grbl. Te albo natychmiast zmieniają zachowanie Grbl w biegu, albo natychmiast drukują raport ważnych danych w czasie rzeczywistym, takich jak aktualna pozycja (aka DRO).

Ustawienia Grbl

$$ - Wyświetl ustawienia Grbl
Aby wyświetlić ustawienia, wpisz $$ i naciśnij enter po połączeniu z Grbl. Grbl powinien odpowiedzieć listą aktualnych ustawień systemu, jak pokazano w poniższym przykładzie. Wszystkie te ustawienia są trwałe i przechowywane w pamięci EEPROM, więc jeśli wyłączysz zasilanie, zostaną one ponownie załadowane przy następnym uruchomieniu Arduino.

$0=10 (step pulse, usec)
$1=25 (step idle delay, msec)
$2=0 (step port invert mask:00000000)
$3=6 (dir port invert mask:00000110)
$4=0 (step enable invert, bool)
$5=0 (limit pins invert, bool)
$6=0 (probe pin invert, bool)
$10=3 (status report mask:00000011)
$11=0.020 (junction deviation, mm)
$12=0.002 (arc tolerance, mm)
$13=0 (report inches, bool)
$20=0 (soft limits, bool)
$21=0 (hard limits, bool)
$22=0 (homing cycle, bool)
$23=1 (homing dir invert mask:00000001)
$24=50.000 (homing feed, mm/min)
$25=635.000 (homing seek, mm/min)
$26=250 (homing debounce, msec)
$27=1.000 (homing pull-off, mm)
$100=314.961 (x, step/mm)
$101=314.961 (y, step/mm)
$102=314.961 (z, step/mm)
$110=635.000 (x max rate, mm/min)
$111=635.000 (y max rate, mm/min)
$112=635.000 (z max rate, mm/min)
$120=50.000 (x accel, mm/sec^2)
$121=50.000 (y accel, mm/sec^2)
$122=50.000 (z accel, mm/sec^2)
$130=225.000 (x max travel, mm)
$131=125.000 (y max travel, mm)
$132=170.000 (z max travel, mm)
$x=val - Zapisz ustawienia Grbl

Polecenie $x=val zapisuje lub zmienia ustawienie Grbl, co można zrobić ręcznie, wysyłając to polecenie po połączeniu z Grbl za pomocą programu terminala szeregowego, ale większość interfejsów graficznych Grbl zrobi to za Ciebie jako funkcja przyjazna dla użytkownika.

Aby ręcznie zmienić np. opcję impulsów krokowych w mikrosekundach na 10us, należy wpisać to, a następnie wprowadzić:

$0=10
Jeśli wszystko poszło dobrze, Grbl odpowie „ok”, a to ustawienie zostanie zapisane w pamięci EEPROM i zostanie zachowane na zawsze lub do czasu ich zmiany. Możesz sprawdzić, czy Grbl odebrał i zapisał Twoje ustawienia poprawnie, wpisując $$, aby ponownie wyświetlić ustawienia systemowe.

Ustawienia Grbl $x=val i ich znaczenie
UWAGA: Numeracja ustawień zmieniła się od wersji 0.8c w celu zabezpieczenia na przyszłość.

$0 – Impuls krokowy, mikrosekundy

Sterowniki krokowe są oceniane na określoną minimalną długość impulsu kroku. Sprawdź arkusz danych lub po prostu wypróbuj kilka liczb. Potrzebujesz najkrótszych impulsów, które sterowniki krokowe mogą niezawodnie rozpoznać. Jeśli impulsy są zbyt długie, możesz napotkać problemy podczas pracy systemu z bardzo wysokimi wartościami posuwu i impulsów, ponieważ impulsy krokowe mogą zacząć nakładać się na siebie. Zalecamy około 10 mikrosekund, co jest wartością domyślną.

$1 — opóźnienie bezczynności kroku, ms

Za każdym razem, gdy steppery wykonają ruch i zatrzymają się, Grbl opóźni wyłączenie stepperów o tę wartość. LUB , zawsze możesz utrzymać włączone osie (zasilane tak, aby utrzymać pozycję), ustawiając tę wartość na maksymalnie 255 milisekund. Ponownie, powtórzę, możesz zawsze włączyć wszystkie osie, ustawiając $1=255.

Czas blokady steppera to czas, przez jaki Grbl będzie blokował steppery przed wyłączeniem. W zależności od systemu możesz ustawić to na zero i wyłączyć. W innych przypadkach możesz potrzebować 25-50 milisekund, aby upewnić się, że osie całkowicie się zatrzymają przed wyłączeniem. Ma to pomóc w wyjaśnieniu silników maszyn, które nie lubią być pozostawione włączone przez długi czas bez robienia czegoś. Należy również pamiętać, że niektóre sterowniki krokowe nie pamiętają, na którym mikrokroku się zatrzymały, więc po ponownym włączeniu możesz być świadkiem niektórych „zgubionych” kroków z tego powodu. W takim przypadku po prostu włącz steppery przez $1=255.

$2 – Maska odwrócenia portu krokowego: binarna

To ustawienie odwraca sygnał impulsu kroku. Domyślnie sygnał kroku zaczyna się od normalnego niskiego poziomu i przechodzi w stan wysoki po wystąpieniu impulsu kroku. Po upływie czasu impulsu krokowego ustawionego przez $0, pin resetuje się do stanu niskiego, aż do następnego zdarzenia impulsu krokowego. Po odwróceniu, zachowanie impulsu krokowego przełącza się z normalnego wysokiego na niski podczas impulsu iz powrotem do wysokiego. Większość użytkowników nie będzie musiała używać tego ustawienia, ale może to być przydatne w przypadku niektórych sterowników krokowych CNC, które mają

szczególne wymagania. Na przykład sztuczne opóźnienie między kołkiem kierunkowym a impulsem krokowym można wytworzyć przez odwrócenie kołka skokowego.

To ustawienie maski odwrócenia jest wartością, która przechowuje osie do odwrócenia jako flagi bitowe. Naprawdę nie musisz całkowicie rozumieć, jak to działa. Wystarczy wprowadzić wartość ustawień dla osi, które chcesz odwrócić. Na przykład, jeśli chcesz odwrócić osie X i Z, wyślij $2=5do Grbl, a ustawienie powinno teraz brzmieć $2=5 (step port invert mask:00000101).

Wartość ustawienia	Maska	Odwróć X	Odwróć Y	Odwróć Z
0	00000000	N	N	N
1	00000001	Tak	N	N
2	00000010	N	Tak	N
3	00000011	Tak	Tak	N
4	00000100	N	N	Tak
5	00000101	Tak	N	Tak
6	00000110	N	Tak	Tak
7	00000111	Tak	Tak	Tak

$3 – Maska odwrócenia portu kierunku: binarna

To ustawienie odwraca sygnał kierunku dla każdej osi. Domyślnie Grbl zakłada, że osie poruszają się w kierunku dodatnim, gdy sygnał pinu kierunkowego jest niski i w kierunku ujemnym, gdy pin jest wysoki. Często osie nie poruszają się w ten sposób w przypadku niektórych maszyn. To ustawienie odwróci sygnał kierunkowy dla tych osi, które poruszają się w przeciwnym kierunku.

To ustawienie maski odwrócenia działa dokładnie tak samo, jak maska odwrócenia portu step i przechowuje, które osie mają być odwrócone jako flagi bitowe. Aby skonfigurować to ustawienie, wystarczy wysłać wartość dla osi, które chcesz odwrócić. Skorzystaj z powyższej tabeli. Na przykład, jeśli chcesz odwrócić tylko kierunek osi Y, wyślij $3=2do Grbl, a ustawienie powinno teraz czytać$3=2 (dir port invert mask:00000010)

$4 - Włącz odwrócenie kroku, bool

Domyślnie pin włączania krokowego jest wysoki, aby wyłączyć i niski, aby włączyć. Jeśli twoja konfiguracja wymaga czegoś przeciwnego, po prostu odwróć pin włączania krokowego, wpisując $4=1. Wyłącz za pomocą $4=0. (Może wymagać cyklu zasilania, aby załadować zmianę.)

$5 - Odwrócone sygnały endstopów, bool

Domyślnie piny ograniczające są utrzymywane normalnie na wysokim poziomie za pomocą wewnętrznego rezystora podciągającego Arduino. Gdy limit pin jest niski, Grbl interpretuje to jako wyzwolenie. Aby uzyskać odwrotne zachowanie, po prostu odwróć piny ograniczające, wpisując $5=1. Wyłącz za pomocą $5=0. Do załadowania zmiany może być potrzebny cykl zasilania.

UWAGA: Jeśli odwrócisz szpilki limitu, będziesz potrzebować zewnętrznego rezystora ściągającego podłączonego do wszystkich szpilek limitu, aby zapobiec przeciążeniu pinów prądem i smażeniu ich.

$6 - Odwrócony pin sondy, bool

Domyślnie pin sondy jest utrzymywany normalnie wysoko za pomocą wewnętrznego rezystora podciągającego Arduino. Gdy pin sondy jest niski, Grbl interpretuje to jako wyzwolenie. Aby uzyskać odwrotne zachowanie, po prostu odwróć pin sondy, wpisując $6=1. Wyłącz za pomocą $6=0. Do załadowania zmiany może być potrzebny cykl zasilania.

UWAGA: Jeśli odwrócisz szpilkę sondy, będziesz potrzebować zewnętrznego rezystora ściągającego podłączonego do szpilki sondy, aby zapobiec przeciążeniu jej prądem i smażeniu.

$10 — maska raportu o stanie: binarna

To ustawienie określa, jakie dane Grbl w czasie rzeczywistym zgłasza użytkownikowi, gdy pojawi się znak '?' wysyłany jest raport o stanie. Domyślnie Grbl odeśle swój stan pracy (nie można go wyłączyć), pozycję maszyny i pozycję roboczą (położenie maszyny z przesunięciami współrzędnych i innymi zastosowanymi przesunięciami). Dostępne są trzy dodatkowe funkcje raportowania, które są przydatne dla interfejsów lub użytkowników konfigurujących swoje maszyny, które obejmują bufor szeregowego RX, wykorzystanie bufora bloku planera i stany pinów limitu (wysokie lub niskie, pokazane w kolejności ZYX).

Aby je ustawić, skorzystaj z poniższej tabeli, aby określić, jakie dane chcesz odesłać Grbl. Wybierz typy raportów, które chcesz widzieć w raportach o stanie, i dodaj ich wartości. Jest to wartość, której używasz do wysyłania do Grbl. Na przykład, jeśli potrzebujesz pozycji maszyny i pracy, dodaj wartości 1 i 2 i wyślij Grbl $10=3, aby je ustawić. Lub, jeśli potrzebujesz tylko pozycji maszyny i stanu sworznia granicznego, dodaj wartości 1 i 16 i wyślij Grbl $10=17.

Ogólnie rzecz biorąc, należy ograniczyć do minimum te dane stanu w czasie rzeczywistym, ponieważ drukowanie i wysyłanie tych danych z dużą szybkością wymaga zasobów. Na przykład raportowanie limitów pinów jest generalnie potrzebne tylko wtedy, gdy użytkownicy konfigurują swój komputer. Następnie zaleca się wyłączenie go, ponieważ nie jest to zbyt przydatne, gdy wszystko już się zorientujesz.

Typ raportu	Wartość
Pozycja maszyny	1
Stanowisko pracy	2
Bufor planera	4
Bufor RX	8
Limit pinów	16

$11 - Odchylenie złącza, mm

Odchylenie skrzyżowania jest używane przez menedżera przyspieszenia do określenia, jak szybko może poruszać się przez skrzyżowania segmentów linii ścieżki programu kodu G. Na przykład, jeśli ścieżka kodu G zbliża się ostro o 10 stopni, a maszyna porusza się z pełną prędkością, to ustawienie pomaga określić, jak bardzo maszyna musi zwolnić, aby bezpiecznie przejść przez róg bez utraty kroków.

Sposób, w jaki to obliczamy, jest nieco skomplikowany, ale ogólnie wyższe wartości dają szybszy ruch w zakrętach, jednocześnie zwiększając ryzyko utraty kroków i pozycjonowania. Niższe wartości sprawiają, że kierownik ds. przyspieszenia jest bardziej ostrożny i prowadzi do ostrożnego i wolniejszego pokonywania zakrętów. Jeśli więc napotkasz problemy, w których twoja maszyna zbyt szybko próbuje pokonywać zakręt, zmniejsz tę wartość, aby zwolnić podczas wchodzenia w zakręt. Jeśli chcesz, aby Twoja maszyna poruszała się szybciej przez skrzyżowania, zwiększ tę wartość, aby przyspieszyć. Dla ciekawskich, kliknij ten link , aby przeczytać o algorytmie pokonywania zakrętów Grbl, który uwzględnia zarówno prędkość, jak i kąt skrzyżowania za pomocą bardzo prostej, wydajnej i niezawodnej metody.

$12 – Tolerancja łuku, mm

Grbl renderuje okręgi, łuki i helisy G2/G3, dzieląc je na maleńkie linie, tak aby dokładność śledzenia łuku nigdy nie była niższa od tej wartości. Prawdopodobnie nigdy nie będziesz musiał zmieniać tego ustawienia, ponieważ 0.002mmjest znacznie poniżej dokładności większości maszyn CNC. Jeśli jednak okaże się, że Twoje kręgi są zbyt prymitywne lub śledzenie łuku działa wolno, dostosuj to ustawienie. Niższe wartości zapewniają większą precyzję, ale mogą prowadzić do problemów z wydajnością przez przeciążenie Grbl zbyt wieloma małymi wierszami. Alternatywnie, wyższe wartości śledzą z mniejszą precyzją, ale mogą przyspieszyć działanie łuku, ponieważ Grbl ma mniej linii, z którymi trzeba się uporać.

Dla ciekawskich, tolerancja łuku jest definiowana jako maksymalna prostopadła odległość od odcinka linii, którego punkty końcowe leżą na łuku, czyli cięciwie. W przypadku niektórych podstawowych geometrii ustalamy długość segmentów linii, aby prześledzić łuk, który spełnia to ustawienie. Modelowanie łuków w ten sposób jest świetne, ponieważ segmenty linii łuku automatycznie dostosowują się i skalują wraz z długością, aby zapewnić optymalną wydajność śledzenia łuku, nie tracąc przy tym dokładności.

$13 - Zgłoś cale, bool

Grbl posiada funkcję raportowania pozycjonowania w czasie rzeczywistym, aby zapewnić użytkownikowi informację zwrotną o tym, gdzie dokładnie znajduje się maszyna w tym czasie, a także o parametrach przesunięcia współrzędnych i sondowania. Domyślnie jest ustawiony na raport w mm, ale wysyłając **$13=1** polecenie, wysyłasz tę flagę logiczną do wartości true, a te funkcje raportowania będą teraz raportować w calach. **$13=0** ustawić z powrotem na mm.

$20 - Limity miękkie, bool

Limity miękkie to funkcja bezpieczeństwa, która zapobiega zbyt dużemu przejechaniu maszyny i przekraczaniu jej granic, awariom lub zepsuciu czegoś drogiego. Działa dzięki znajomości maksymalnych limitów ruchu dla każdej osi i położenia Grbl we współrzędnych maszyny. Za każdym razem, gdy nowy ruch kodu G jest wysyłany do Grbl, sprawdza, czy przypadkowo przekroczyłeś przestrzeń maszyny. Jeśli to zrobisz, Grbl wyda natychmiastowe wstrzymanie posuwu, gdziekolwiek się znajduje, wyłączy wrzeciono i chłodziwo, a następnie ustawi alarm systemowy wskazujący problem. Pozycja maszyny zostanie później

zachowana, ponieważ nie jest to spowodowane natychmiastowym wymuszonym zatrzymaniem, takim jak sztywne limity.

UWAGA: Limity miękkie wymagają włączenia bazowania i dokładnych ustawień maksymalnego przesunięcia osi, ponieważ Grbl musi wiedzieć, gdzie się znajduje. $20=1włączyć i $20=0wyłączyć.

$21 - limity sprzętowe, bool

Hard limit działa w zasadzie tak samo jak miękkie limity, ale zamiast tego używaj fizycznych przełączników. Zasadniczo podłączasz kilka krańcówek (mechanicznych, magnetycznych lub optycznych) w pobliżu końca ruchu każdej osi lub w dowolnym miejscu, w którym czujesz, że mogą wystąpić problemy, jeśli twój program przesunie się zbyt daleko tam, gdzie nie powinien. Gdy przełącznik zostanie uruchomiony, natychmiast zatrzyma cały ruch, wyłączy chłodziwo i wrzeciono (jeśli są podłączone) i przejdzie w tryb alarmu, który zmusza cię do sprawdzenia maszyny i zresetowania wszystkiego.

Aby użyć sprzętowych limitów z Grbl, piny limitów są utrzymywane wysoko za pomocą wewnętrznego rezystora podciągającego, więc wszystko, co musisz zrobić, to podłączyć normalnie otwarty przełącznik z pinem i uziemieniem i włączyć twarde limity za pomocą $21=1. (Wyłącz za pomocą $21=0.) Zdecydowanie zalecamy podjęcie środków zapobiegających zakłóceniom elektrycznym. Jeśli chcesz mieć limit dla obu końców ruchu jednej osi, po prostu podłącz dwa przełączniki równolegle do pinu i masy, więc jeśli któryś z nich wyłączy się, wyzwala twardy limit.

Należy pamiętać, że wydarzenie ze sprzętowym limitem jest uważane za wydarzenie krytyczne, w którym steppery natychmiast się zatrzymują i prawdopodobnie zgubią kroki. Grbl nie ma żadnych informacji zwrotnych na temat pozycji, więc nie może zagwarantować, że ma pojęcie, gdzie się znajduje. Tak więc, jeśli zostanie uruchomiony twardy limit, Grbl przejdzie w tryb ALARMU z nieskończoną pętlą, dając Ci szansę na sprawdzenie komputera i zmuszenie do zresetowania Grbl. Pamiętaj, że to wyłącznie funkcja bezpieczeństwa.

$22 - Cykl bazowania, bool

Dla tych, którzy dopiero co rozpoczęli pracę w CNC, cykl bazowania służy do dokładnego i precyzyjnego zlokalizowania znanej i spójnej pozycji na maszynie za każdym razem, gdy uruchamiasz Grbl między sesjami. Innymi słowy, za każdym razem wiesz dokładnie, gdzie jesteś w danym momencie. Powiedzmy, że zaczynasz coś obrabiać lub masz zamiar rozpocząć kolejny krok w pracy, a moc gaśnie, ponownie uruchamiasz Grbl, a Grbl nie ma pojęcia, gdzie to jest. Pozostaje Ci zadanie dowiedzieć się, gdzie jesteś. Jeśli masz naprowadzanie, zawsze masz zerowy punkt odniesienia maszyny, z którego możesz zlokalizować, więc wszystko, co musisz zrobić, to uruchomić cykl naprowadzania i wznowić od miejsca, w którym zostało przerwane.

Aby skonfigurować cykl bazowania dla Grbl, musisz mieć wyłączniki krańcowe w ustalonej pozycji, które nie zostaną uderzone ani przesunięte, w przeciwnym razie twój punkt

odniesienia zostanie pomieszany. Zazwyczaj są one ustawiane w najdalszym punkcie w +x, +y, +z każdej osi. Podłącz swoje wyłączniki krańcowe za pomocą kołków granicznych i uziemienia, tak jak w przypadku twardych limitów, i włącz bazowanie. Jeśli jesteś ciekawy, możesz użyć swoich wyłączników krańcowych zarówno dla twardych limitów, jak i bazowania. Dobrze się ze sobą bawią.

Domyślnie cykl bazowania Grbl przesuwa najpierw dodatnią oś Z, aby oczyścić obszar roboczy, a następnie jednocześnie przesuwa obie osie X i Y w dodatnim kierunku. Aby skonfigurować zachowanie cyklu bazowania, na stronie znajduje się więcej ustawień Grbl opisujących, co robią (oraz opcje czasu kompilacji).

Jeszcze jedna rzecz do zapamiętania, gdy włączone jest bazowanie. Grbl zablokuje wszystkie polecenia kodu G, dopóki nie wykonasz cyklu bazowania. Oznacza to, że żadne osie się nie poruszają, chyba że blokada jest wyłączona ($X), ale o tym później. Większość, jeśli nie wszystkie sterowniki CNC, robią coś podobnego, ponieważ jest to głównie funkcja bezpieczeństwa zapobiegająca popełnieniu przez użytkownika błędu pozycjonowania, co jest bardzo łatwe do zrobienia i jest smutne, gdy błąd niszczy część. Jeśli uznasz to za irytujące lub znajdziesz jakieś dziwne błędy, daj nam znać, a postaramy się nad tym popracować, aby wszyscy byli zadowoleni. :)

UWAGA: Sprawdź config.h, aby uzyskać więcej opcji bazowania dla zaawansowanych użytkowników. Możesz wyłączyć blokadę bazowania przy starcie, skonfigurować, które osie poruszają się jako pierwsze podczas cyklu bazowania iw jakiej kolejności, i nie tylko.

$23 - Odwrócona maska bazowania, int:binary

Domyślnie Grbl zakłada, że wyłączniki krańcowe naprowadzania są w dodatnim kierunku, najpierw przesuwając dodatnią oś z, a następnie dodatnią oś xy, zanim spróbujesz precyzyjnie zlokalizować zero maszyny, poruszając się powoli tam i z powrotem wokół przełącznika. Jeśli twoja maszyna ma wyłącznik krańcowy w kierunku ujemnym, maska kierunku bazowania może odwrócić kierunek osi. Działa podobnie jak maski odwrócenia portu krokowego i odwrócenia portu kierunkowego, w których wszystko, co musisz zrobić, to wysłać wartość w tabeli, aby wskazać, które osie chcesz odwrócić i szukać w przeciwnym kierunku.

$24 — Prędkość posuwu bazowania, mm/min

Cykl bazowania najpierw szuka wyłączników krańcowych z większą szybkością wyszukiwania, a po ich znalezieniu porusza się z mniejszą szybkością posuwu, aby powrócić do dokładnej lokalizacji zerowej maszyny. Szybkość posuwu bazowania to wolniejsza szybkość posuwu. Ustaw tę wartość na dowolną wartość szybkości, która zapewnia powtarzalne i precyzyjne lokalizowanie zera maszyny.

$25 — prędkość wyszukiwania bazy, mm/min

Szybkość wyszukiwania bazowania to szybkość wyszukiwania cyklu bazowania lub szybkość, z jaką po raz pierwszy próbuje znaleźć wyłączniki krańcowe. Dostosuj się do

szybkości docierania do wyłączników krańcowych w wystarczająco krótkim czasie bez zderzenia z wyłącznikami krańcowymi, jeśli wejdą zbyt szybko.

$26 — Odbicie naprowadzania, ms

Za każdym razem, gdy przełącznik jest wyzwalany, niektóre z nich mogą mieć szum elektryczny/mechaniczny, który faktycznie „odbija” sygnał na wysokim i niskim poziomie przez kilka milisekund przed ustawieniem się. Aby rozwiązać ten problem, musisz odbić sygnał, albo sprzętowo z jakimś rodzajem kondycjoner sygnału lub przez oprogramowanie z krótkim opóźnieniem, aby sygnał zakończył się odbijaniem. Grbl wykonuje krótkie opóźnienie, bazuje tylko na miejscu zerowym maszyny. Ustaw tę wartość opóźnienia na taką, jakiej potrzebuje twój przełącznik, aby uzyskać powtarzalne naprowadzanie. W większości przypadków wystarczy 5-25 milisekund.

$27 - Odciąganie naprowadzania, mm

Aby dobrze bawić się z funkcją twardych limitów, w której naprowadzanie może współdzielić te same wyłączniki krańcowe, cykl naprowadzania przesunie się ze wszystkich wyłączników krańcowych o ten ruch odciągania po jego zakończeniu. Innymi słowy, pomaga zapobiegać przypadkowemu uruchomieniu twardego limitu po cyklu bazowania.

$100, $101 i $102 – [X,Y,Z] kroki/mm

Grbl musi wiedzieć, jak daleko zajdzie każdy krok w rzeczywistości. Aby obliczyć kroki/mm dla osi swojej maszyny, musisz wiedzieć:

Przebyty w mm na obrót silnika krokowego. Zależy to od kół zębatych napędu pasowego lub skoku śruby pociągowej.
Pełne kroki na obrót Twoich stepperów (zwykle 200)
Mikrokroki na krok kontrolera (zwykle 1, 2, 4, 8 lub 16). Wskazówka: Używanie wysokich wartości mikrokroków (np. 16) może zmniejszyć moment obrotowy silnika krokowego, więc używaj najniższej, która zapewnia pożądaną rozdzielczość osi i komfortowe właściwości podczas pracy.
Kroki/mm można następnie obliczyć w następujący sposób:steps_per_mm = (steps_per_revolution*microsteps)/mm_per_rev

Oblicz tę wartość dla każdej osi i zapisz te ustawienia do Grbl.

$110, $111 i $112 – [X,Y,Z] Maksymalna prędkość posuwu, mm/min

Ustawia to maksymalną prędkość, jaką może poruszać się każda oś. Za każdym razem, gdy Grbl planuje ruch, sprawdza, czy ruch powoduje, że którakolwiek z tych indywidualnych osi przekracza maksymalną prędkość. Jeśli tak, spowolni ruch, aby żadna z osi nie przekroczyła limitów maksymalnej szybkości. Oznacza to, że każda oś ma własną niezależną prędkość, co jest niezwykle przydatne w ograniczaniu zwykle wolniejszej osi Z.

Najprostszym sposobem określenia tych wartości jest przetestowanie każdej osi pojedynczo, powoli zwiększając ustawienia maksymalnej szybkości i przesuwając ją. Na przykład, aby przetestować oś X, wyślij coś podobnego do Grbl G0 X50z wystarczającą odległością przesuwu, aby oś przyspieszyła do maksymalnej prędkości. Będziesz wiedział, że osiągnąłeś próg maksymalnej szybkości, gdy steppery przestaną działać. Będzie trochę hałasować, ale nie powinno uszkodzić silników. Wprowadź ustawienie o 10-20% poniżej tej wartości, aby uwzględnić zużycie, tarcie i masę obrabianego przedmiotu/narzędzia. Następnie powtórz dla pozostałych osi.

UWAGA: To ustawienie maksymalnej szybkości określa również szybkości wyszukiwania G0.

$120, $121, $122 – [X,Y,Z] Przyspieszenie, mm/s^2

Ustawia to parametry przyspieszenia osi w mm/sekundę/sekundę. W uproszczeniu, niższa wartość sprawia, że Grbl zwalnia wolniej w ruchu, podczas gdy wyższa wartość daje

ciaśniejsze ruchy i znacznie szybciej osiąga pożądane prędkości posuwu. Podobnie jak ustawienie maksymalnej szybkości, każda oś ma swoją własną wartość przyspieszenia i są od siebie niezależne. Oznacza to, że ruch wieloosiowy przyspieszy tylko tak szybko, jak tylko może to zrobić najniższa oś przyczyniająca się.

Podobnie jak w przypadku ustawienia maksymalnej szybkości, najprostszym sposobem określenia wartości dla tego ustawienia jest indywidualne testowanie każdej osi z powoli rosnącymi wartościami, aż do zatrzymania silnika. Następnie sfinalizuj ustawienie przyspieszenia z wartością 10-20% poniżej tej bezwzględnej wartości maksymalnej. Powinno to uwzględniać zużycie, tarcie i bezwładność masy. Zdecydowanie zalecamy przetestowanie na sucho niektórych programów z kodem G z nowymi ustawieniami przed ich zastosowaniem. Czasami obciążenie maszyny jest inne podczas poruszania się we wszystkich osiach razem.

130 $, 131 $, 132 $ – [X,Y,Z] Wielkość pola roboczego, mm

Ustawia to maksymalny przesuw od końca do końca dla każdej osi w mm. Jest to przydatne tylko wtedy, gdy masz włączone limity miękkie (i bazowanie), ponieważ jest używane tylko przez funkcję limitów miękkich Grbl, aby sprawdzić, czy przekroczyłeś limity maszyny za pomocą polecenia ruchu.

Inne polecenia „$” Grbl
Inne $ polecenia zapewniają użytkownikowi dodatkowe elementy sterujące, takie jak drukowanie informacji zwrotnej na temat bieżącego stanu modalnego parsera kodu G lub uruchamianie cyklu bazowania. W tej sekcji wyjaśniono, czym są te polecenia i jak ich używać.

$# - Wyświetl parametry gcode

Parametry kodu G przechowują wartości korekcji współrzędnych dla współrzędnych roboczych G54-G59, wstępnie zdefiniowanych pozycji G28/G30, korekcji współrzędnych G92, korekcji długości narzędzia i sondowania (nieoficjalnie, ale i tak dodaliśmy tutaj). Większość z tych parametrów jest zapisywana bezpośrednio w pamięci EEPROM po ich zmianie i jest trwała. Oznacza to, że pozostaną takie same, niezależnie od wyłączenia zasilania, dopóki nie zostaną wyraźnie zmienione. Nietrwałe parametry, które nie zostaną zachowane po zresetowaniu lub przełączeniu zasilania, to korekcje długości narzędzia G92, G43.1 oraz dane sondowania G38.2.

Współrzędne robocze G54-G59 można zmienić za pomocą polecenia G10 L2 Pxlub G10 L20 Pxzdefiniowanego przez standard gcode NIST i standard EMC2 (linuxcnc.org). Predefiniowane pozycje G28/G30 można zmienić odpowiednio za pomocą poleceń i G28.1.G30.1

Gdy $# zostanie wywołany, Grbl odpowie z zapamiętanymi przesunięciami ze współrzędnych maszyny dla każdego systemu w następujący sposób. To ooznacza korekcję długości narzędzia i PRB oznacza współrzędne ostatniego cyklu próbkowania.

[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:0.000,0.000,0.000]
[PRB:0.000,0.000,0.000]

$G- Zobacz stan parsera gcode0

To polecenie drukuje wszystkie aktywne tryby gcode w parserze G-code Grbl. Wysyłając to polecenie do Grbl, odpowie coś w stylu:

[G0 G54 G17 G21 G90 G94 M0 M5 M9 T0 S0.0 F500.0]
Te aktywne tryby określają, w jaki sposób następny blok lub polecenie kodu G będzie interpretowany przez parser kodu G Grbl. Dla tych, którzy są nowicjuszami w obróbce kodu G i CNC, tryby ustawiają parser w określonym stanie, dzięki czemu nie musisz ciągle mówić parserowi, jak go przeanalizować. Te tryby są zorganizowane w zestawy zwane „grupami modalnymi", które nie mogą być logicznie aktywne w tym samym czasie. Na przykład grupa modalna jednostek określa, czy program w kodzie G jest interpretowany w calach czy w milimetrach.

Krótka lista grup modalnych obsługiwanych przez Grbl jest pokazana poniżej, ale pełniejsze i bardziej szczegółowe opisy można znaleźć na stronie LinuxCNC . Polecenia G-code wytłuszczone wskazują domyślne tryby po włączeniu zasilania Grbl lub zresetowaniu go.

Znaczenie grupy modalnej Słowa członków
Tryb ruchu G0 , G1, G2, G3, G38.2, G38.3, G38.4, G38.5, G80
Wybór układu współrzędnych G54 , G55, G56, G57, G58, G59
Wybór samolotu G17 , G18, G19
Tryb odległości G90 , G91
Tryb odległości łuku IJK G91.1
Tryb prędkości posuwu G93, G94
Tryb jednostekG20, G21
Kompensacja promienia frezu G40
Przesunięcie długości narzędzia G43.1, G49
Tryb programuM0 , M1, M2, M30
Stan wrzeciona M3, M4, M5
Stan chłodziwa M7, M8, M9
Oprócz trybów analizatora kodu G, Grbl zgłosi numer aktywnego Tnarzędzia, Sprędkość wrzeciona i prędkość Fposuwu, które po zresetowaniu mają wartość domyślną 0. Dla tych, którzy są ciekawi, nie pasują one do ładnych grup modalnych, ale są równie ważne dla określenia stanu parsera.

$I- Wyświetl informacje o kompilacji

Spowoduje to wydrukowanie informacji zwrotnej dla użytkownika o wersji Grbl i dacie kompilacji kodu źródłowego. Opcjonalnie $Imoże również przechowywać krótki ciąg znaków, aby pomóc zidentyfikować maszynę CNC, z którą się komunikujesz, jeśli masz więcej niż maszynę używającą Grbl. Aby ustawić ten ciąg, wyślij Grbl $I=xxx, gdzie xxxjest ciąg dostosowywania, który ma mniej niż 80 znaków. Następnym razem, gdy wyślesz zapytanie do Grbl z $Iwidokiem informacji o kompilacji, Grbl wydrukuje ten ciąg po wersji i dacie kompilacji.

$N — Wyświetl bloki startowe

$Nxto bloki startowe, które Grbl uruchamia za każdym razem, gdy włączasz Grbl lub resetujesz Grbl. Innymi słowy, blok startowy to wiersz kodu G, który możesz uruchomić automatycznie w Grbl, aby ustawić domyślne ustawienia modalne kodu G, lub cokolwiek innego, co Grbl musi zrobić za każdym razem, gdy uruchamiasz swoją maszynę. Grbl może domyślnie przechowywać dwa bloki kodu G.

Tak więc po połączeniu z Grbl wpisz, $N a następnie enter. Grbl powinien odpowiedzieć czymś krótkim, takim jak:

$N0=
$N1=
ok
Nic do zrobienia, ale oznacza to po prostu, że nie ma zapisanego bloku kodu G,
$N0 aby Grbl mógł uruchomić się po uruchomieniu. $N1 to kolejna linia do uruchomienia.

$Nx = line - Zapisz blok startowy

WAŻNE: Zachowaj szczególną ostrożność podczas zapisywania poleceń ruchu (G0/1,G2/3,G28/30) w blokach startowych. Te polecenia ruchu będą uruchamiane za każdym razem, gdy zresetujesz lub włączysz Grbl, więc jeśli masz sytuację awaryjną i musisz zatrzymać e-stop i zresetować, ruch bloku startowego może i prawdopodobnie szybko pogorszy sytuację. Nie należy również umieszczać żadnych komend, które zapisują dane w EEPROM, takich jak G10/G28.1/G30.1. Spowoduje to, że Grbl będzie stale ponownie zapisywał te dane przy każdym uruchomieniu i zresetowaniu, co ostatecznie doprowadzi do zużycia pamięci EEPROM Arduino.

Typowym zastosowaniem bloku startowego jest po prostu ustawienie preferowanych stanów modalnych, takich jak tryb cali G20, zawsze domyślnie na inny układ współrzędnych roboczych lub zapewnienie użytkownikowi sposobu na uruchomienie niektórych unikalnych funkcji napisanych przez użytkownika, których potrzebuje za ich szalony projekt.

Aby ustawić blok startowy, wpisz, $N0=a następnie poprawny blok kodu G i enter. Grbl uruchomi blok, aby sprawdzić, czy jest prawidłowy, a następnie odpowie za pomocą oklub an, error:aby powiedzieć, czy się powiodło, czy coś poszło nie tak. Jeśli wystąpi błąd, Grbl go nie zapisze.

Załóżmy na przykład, że chcesz użyć pierwszego bloku startowego $N0 do ustawienia trybów analizatora kodu G, takich jak współrzędna robocza G54, tryb cali G20, płaszczyzna XY G17. Wpisałbyś $N0=G20 G54 G17 z enterem i powinieneś zobaczyć odpowiedź „ok”. Następnie możesz sprawdzić, czy został zapisany, wpisując $N, a teraz powinieneś zobaczyć odpowiedź, taką jak $N0=G20G54G17.

Gdy już masz blok startowy przechowywany w pamięci EEPROM Grbl, za każdym razem, gdy uruchamiasz lub resetujesz, zobaczysz wydrukowany blok startowy i odpowiedź od Grbl, aby wskazać, czy wszystko działało poprawnie. Tak więc w poprzednim przykładzie zobaczysz:

Grbl 0.9i ['$' for help]
G20G54G17ok
Jeśli masz wiele bloków startowych kodu G, będą one drukowane z powrotem w kolejności przy każdym uruchomieniu. A jeśli chcesz wyczyścić jeden z bloków startowych (np. blok 0), wpisz $N0= bez niczego po znaku równości.

Ponadto, jeśli masz włączone bazowanie, bloki startowe będą wykonywane natychmiast po cyklu bazowania, a nie przy starcie.

$C - Sprawdź tryb gcode

To przełącza parser gcode Grbl, aby pobrać wszystkie przychodzące bloki i przetworzyć je całkowicie, tak jak w normalnej pracy, ale nie porusza żadnej z osi, ignoruje przerwy i wyłącza wrzeciono i chłodziwo. Ma to na celu zapewnienie użytkownikowi sposobu na sprawdzenie, jak ich nowy program G-code radzi sobie z parserem Grbl i monitorowaniem wszelkich błędów (oraz sprawdzanie naruszeń limitu miękkiego, jeśli jest włączony).

Po wyłączeniu Grbl wykona automatyczny miękki reset (^X). Ma to dwa cele. Upraszcza nieco zarządzanie kodem. Ale uniemożliwia to również użytkownikom rozpoczęcie pracy, gdy ich tryby G-kodu nie są takie, jak im się wydaje. Reset systemu zawsze daje użytkownikowi świeży, spójny start.

$X - Usuń blokadę alarmu

Tryb alarmu Grbl to stan, w którym coś poszło krytycznie nie tak, na przykład twardy limit lub przerwanie podczas cyklu, lub jeśli Grbl nie zna swojej pozycji. Domyślnie, jeśli masz włączone bazowanie i włączasz Arduino, Grbl wchodzi w stan alarmu, ponieważ nie zna swojej pozycji. W trybie alarmowym wszystkie polecenia G-code zostaną zablokowane do czasu wykonania cyklu bazowania „$H”. Lub jeśli użytkownik musi obejść blokadę alarmu, aby przesunąć swoje osie poza wyłączniki krańcowe, na przykład, blokada alarmu „$X” ominie blokady i umożliwi ponowne działanie funkcji kodu G.

Ostrożnie!! Powinno to być używane tylko w sytuacjach awaryjnych. Pozycja prawdopodobnie została utracona, a Grbl może nie być tam, gdzie myślisz, że jest. Dlatego zaleca się używanie trybu przyrostowego G91 do wykonywania krótkich ruchów. Następnie wykonaj cykl bazowania lub zresetuj natychmiast po tym.

$H - Uruchom cykl bazowania

To polecenie jest jedynym sposobem na wykonanie cyklu bazowania w Grbl. Niektóre inne kontrolery ruchu wyznaczają specjalne polecenie kodu G do uruchomienia cyklu bazowania, ale jest to nieprawidłowe zgodnie ze standardami kodu G. Homing to zupełnie osobna komenda obsługiwana przez kontroler.

WSKAZÓWKA: Po uruchomieniu cyklu bazowania, raczej ręcznie biegnij cały czas do pozycji w środku objętości obszaru roboczego. Możesz ustawić wstępnie zdefiniowaną pozycję G28 lub G30 jako pozycję post-homing, bliżej miejsca, w którym będziesz obrabiać. Aby je ustawić, musisz najpierw przesunąć maszynę do miejsca, w którym chcesz, aby przeniosła się po naprowadzeniu. Wpisz G28.1 (lub G30.1), aby Grbl zapisał tę pozycję. Więc po naprowadzeniu '$H', możesz po prostu wpisać 'G28' (lub 'G30') i przemieści się tam automagicznie. Ogólnie rzecz biorąc, po prostu przesunąłbym oś XY do środka, a oś Z opuściłbym w górę. Gwarantuje to, że narzędzie we wrzecionie nie będzie przeszkadzać i niczego się nie zaczepi.

$RST=$, $RST=#, i $RST=*- Przywróć ustawienia Grbl do wartości domyślnych

Komendy te nie są wymienione w głównym komunikacie $pomocy Grbl, ale są dostępne, aby umożliwić użytkownikom odtwarzanie części lub wszystkich danych EEPROM Grbl. Uwaga: Grbl zostanie automatycznie zresetowany po wykonaniu jednego z tych poleceń, aby zapewnić prawidłowe zainicjowanie systemu.

$RST = $: Usuwa i przywraca $$domyślne ustawienia Grbl, które są zdefiniowane przez plik ustawień domyślnych używany podczas kompilacji Grbl. Często producenci OEM budują oprogramowanie sprzętowe Grbl z zalecanymi ustawieniami specyficznymi dla maszyny. Zapewnia to użytkownikom i producentom OEM szybki powrót do punktu wyjścia, jeśli coś poszło nie tak lub jeśli użytkownik chce zacząć od nowa.
$RST=#: Kasuje i zeruje wszystkie przesunięcia współrzędnych roboczych G54-G59 i pozycje G28/30 zapisane w EEPROM. Są to na ogół wartości widoczne na $#wydruku parametrów. Zapewnia to łatwy sposób ich usunięcia bez konieczności robienia tego ręcznie dla każdego zestawu za pomocą polecenia G20 L2/20lub G28.1/30.1.
$RST=*: To czyści i przywraca wszystkie dane EEPROM używane przez Grbl. Obejmuje to $$ustawienia, $#parametry, $Nwiersze startowe i $Iciąg informacji o kompilacji. Zauważ, że nie powoduje to wyczyszczenia całej pamięci EEPROM, tylko obszary danych, których używa Grbl. Aby wykonać pełne czyszczenie, użyj przejrzystego przykładowego projektu EEPROM Arduino IDE.

Polecenia w czasie rzeczywistym:~, !, ?, and Ctrl-X

Ostatnie cztery polecenia Grbl są poleceniami czasu rzeczywistego. Oznacza to, że można je wysłać w dowolnym miejscu i czasie, a Grbl natychmiast zareaguje, bez względu na to, co robi. Dla tych, którzy są ciekawi, są to znaki specjalne, które są „wybierane" z przychodzącego strumienia szeregowego i informują Grbl, aby je wykonał, zwykle w ciągu kilku milisekund.

~ - Rozpoczęcie cyklu
Jest to polecenie rozpoczęcia lub wznowienia cyklu, które można wydać w dowolnym momencie, ponieważ jest to polecenie w czasie rzeczywistym. Gdy Grbl ma ruchy w kolejce w swoim buforze i jest gotowy do pracy, ~polecenie rozpoczęcia cyklu rozpocznie wykonywanie bufora, a Grbl rozpocznie przesuwanie osi. Jednak domyślnie automatyczne uruchamianie cyklu jest włączone, więc nowi użytkownicy nie będą potrzebować tego polecenia, chyba że zostanie wykonane wstrzymanie posuwu. Po wykonaniu wstrzymania posuwu, rozpoczęcie cyklu wznowi program. Rozpoczęcie cyklu będzie skuteczne tylko wtedy, gdy w buforze pojawią się ruchy gotowe do pracy i nie będzie działać z żadnym innym procesem, takim jak bazowanie.

! - Zatrzymanie pracy
Polecenie zatrzymania posuwu spowoduje zatrzymanie aktywnego cyklu poprzez kontrolowane hamowanie, aby nie stracić pozycji. Działa również w czasie rzeczywistym i może być aktywowany w dowolnym momencie. Po zakończeniu lub wstrzymaniu Grbl poczeka na wydanie polecenia rozpoczęcia cyklu, aby wznowić program. Zatrzymanie posuwu może tylko wstrzymać cykl i nie wpłynie na bazowanie ani żaden inny proces.

Jeśli musisz zatrzymać cykl w środku programu i nie możesz sobie pozwolić na utratę pozycji, wykonaj wstrzymanie posuwu, aby Grbl doprowadził wszystko do kontrolowanego zatrzymania. Po zakończeniu możesz wykonać reset. Zawsze staraj się wykonać zatrzymanie posuwu za każdym razem, gdy maszyna jest uruchomiona przed naciśnięciem przycisku resetowania, oczywiście z wyjątkiem sytuacji awaryjnych.

? - Aktualny stan
Polecenie ? natychmiast zwraca stan aktywny Grbl i bieżącą pozycję w czasie rzeczywistym, zarówno we współrzędnych maszyny, jak i współrzędnych roboczych. Opcjonalnie można również uzyskać odpowiedź Grbl z buforem szeregowym RX i użyciem bufora planowania za pośrednictwem ustawienia maski raportu o stanie. Polecenie ?można wysłać w dowolnym momencie i działa asynchronicznie ze wszystkimi innymi procesami, które wykonuje Grbl. Ustawienie $13Grbl określa, czy raportuje milimetry czy cale. Po ?naciśnięciu, Grbl natychmiast odpowie czymś podobnym do następującego:

<Idle,MPos:5.529,0.560,7.000,WPos:1.529,-5.440,-0.000>
Aktywne stany, w których może znajdować się Grbl, to: Bezczynny, Uruchomiony, Wstrzymany, Drzwi, Dom, Alarm, Sprawdź

Bezczynność : Wszystkie systemy są uruchomione, żadne ruchy nie są w kolejce i jest gotowe na wszystko.
Uruchom : wskazuje, że cykl jest uruchomiony.

Wstrzymaj : Wstrzymanie posuwu jest w trakcie wykonywania lub zwalnia aż do zatrzymania. Po zakończeniu wstrzymania Grbl pozostanie w trybie wstrzymania i czeka na rozpoczęcie cyklu, aby wznowić program.
Drzwi : (Nowość w wersji 0.9i) Ta opcja kompilacji powoduje, że Grbl zatrzymuje podawanie, wyłącza wrzeciono i chłodziwo i czeka, aż przełącznik drzwi zostanie zamknięty, a użytkownik uruchomi cykl. Przydatne dla producentów OEM, którzy potrzebują drzwi bezpieczeństwa.
Home : W środku cyklu bazowania. UWAGA: Pozycje nie są aktualizowane na żywo podczas cyklu bazowania, ale zostaną ustawione w pozycji wyjściowej po zakończeniu.
Alarm : Wskazuje, że coś poszło nie tak lub Grbl nie zna swojej pozycji. Ten stan blokuje wszystkie polecenia G-code, ale pozwala na interakcję z ustawieniami Grbl, jeśli zajdzie taka potrzeba. Blokada alarmu „$ X” zwalnia ten stan i umieszcza Grbl w stanie bezczynności, co pozwoli ci ponownie przenosić rzeczy. Jak wspomniano wcześniej, uważaj na to, co robisz po alarmie.
Sprawdź : Grbl jest w trybie sprawdzania kodu G. Będzie przetwarzał i odpowiadał na wszystkie polecenia G-kodu, ale nie poruszał ani niczego nie włączał. Po wyłączeniu innym poleceniem „$C”, Grbl zresetuje się.
Ctrl-x- Zresetuj Grbl
To jest polecenie miękkiego resetu Grbl. Odbywa się w czasie rzeczywistym i można go wysłać w dowolnym momencie. Jak sama nazwa wskazuje, resetuje Grbl, ale w kontrolowany sposób zachowuje pozycję maszyny, a wszystko odbywa się bez wyłączania Arduino. Miękki reset może stracić pozycję tylko wtedy, gdy pojawiają się problemy i steppery giną podczas ruchu. Jeśli tak, zgłosi, czy śledzenie pozycji maszyny przez Grbl zostało utracone. Dzieje się tak, ponieważ niekontrolowane zwalnianie może prowadzić do utraty kroków, a Grbl nie ma informacji zwrotnej na temat tego, ile stracił (to jest ogólnie problem ze stepperami). W przeciwnym razie Grbl po prostu ponownie się zainicjuje, uruchomi linie startowe i będzie kontynuował swoją wesołą drogę.

Należy pamiętać, że zaleca się wykonanie miękkiego resetu przed rozpoczęciem pracy. Gwarantuje to, że nie ma żadnych aktywnych trybów G-kodów, takich jak zabawa lub konfiguracja maszyny przed uruchomieniem zadania. Dzięki temu Twoja maszyna zawsze będzie uruchamiać się świeżo i konsekwentnie, a Twoja maszyna robi to, czego od niej oczekujesz.