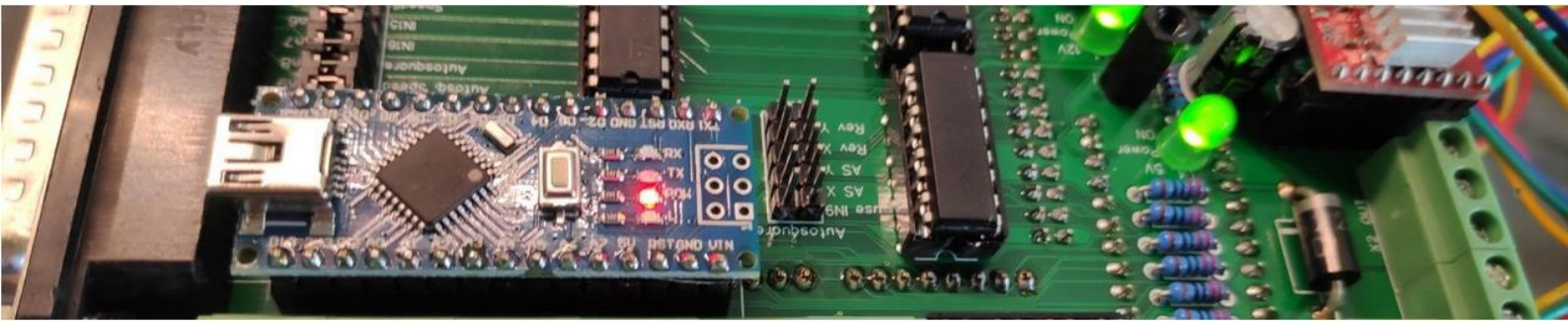
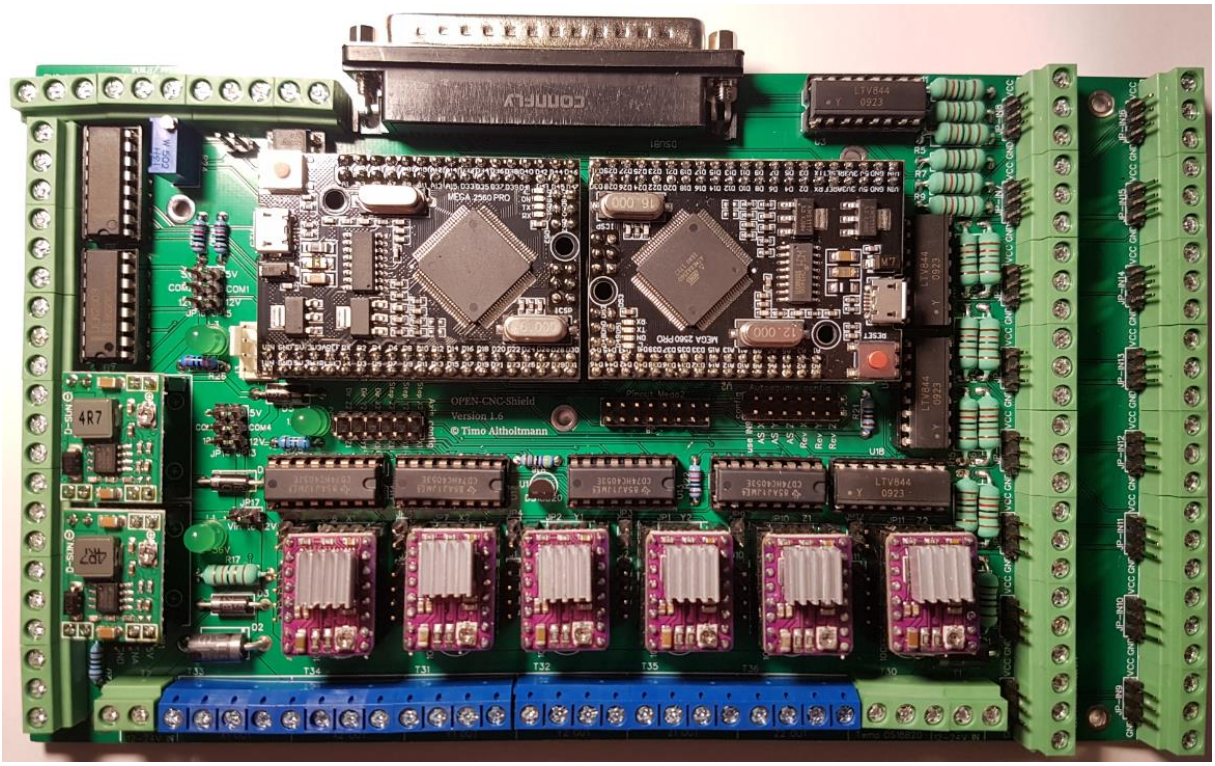
Timos Werkstatt



OPEN-CNC-Shield Dokumentation

V21-3



Timos Werkstatt / Timo Altholtmann
www.altholtmann.com /

Inhaltsverzeichnis

1 Jumper / Terminals / Anschlüsse

Beschriftung	Funktion	Beschreibung
JP6, JP7, JP2, JP1, JP10, JP11	Mikroschritte der Aufstecktreiber	Hier werden die Mikroschritte der Aufstecktreiber eingestellt. Falls externe Treiber verwendet werden können diese Jumper frei gelassen werden. Mehr dazu unter Mikroschritte
JP5, JP8, JP4, JP3, JP9, JP12	Fault Jumper der Aufstecktreiber	Muss bei A4988 Treiber gesetzt sein. Bei DRV8825 bleiben die Jumper frei. Mehr dazu unter Fault Jumper
J4, J7, J1, J2, J10, J11	Motoranschluss bei Aufstecktreibern	Hier werden die Motoren angeschlossen, wenn die Aufstecktreiber verwendet werden. Schema: 1A 1B 2A 2B. Drehen die Motoren falsch herum einfach den Stecker einmal drehen. Mehr dazu unter Anschließen der Motoren
JP19	Konfiguration der Achsen	Hier können die verschiedenen Achsen miteinander verbunden werden. Wird Estlcam als Software eingesetzt sollten hier alle Jumper gesetzt sein. Mehr dazu unter Konfiguration der Achsen
JP20	Autosquaring Konfiguration	Hier kann der Autosquaring Modus aktiviert werden und die Drehrichtung der Motoren, während Autosquaring festgelegt werden. Mehr dazu unter Autosquare Konfiguration
JP17	Verbindet „Vin“ mit dem 12V Netz	Wenn das Shield mit 12V betrieben wird, sollte dieser Jumper gesetzt werden. Mehr dazu unter Stromversorgung
JP15 und JP16	Jumper für die Spannung der Ausgänge	Hier kann die Spannung für die Ausgänge 1-8 festgelegt werden. Jeder Jumper stellt die Spannung für 4 Ausgänge ein. Mehr dazu unter Ausgänge
JP13 und JP14 (ab Version 1.5)	Jumper für die Spannung an den Eingängen	Hier kann die Spannung, welche an den Terminals der Eingänge anliegt, festgelegt werden. Jeder Jumper zählt dabei für 8 Eingänge. Mehr dazu im Kapitel Eingänge.
JP18	Spannung der Drehzahlsteuerung	Ist dieser Jumper gesetzt, liegt die Spannung des analogen Ausgangs für die Drehzahl zwischen 0 und 5V. Ansonsten liegt sie zwischen 0 und 10V. Mehr dazu unter Spindelsteuerung
JP21 (ab Version 1.6)	ENA Einstellung	Hier können die ENA Treiber standardmäßig eingestellt werden. Mit einem 10K Widerstand kann ENA hier entweder mit GND oder mit 5V verbunden werden. Wird dies nicht benötigt kann der Jumper auch weggelassen werden.

Tabelle 1: Jumber

2 Stromversorgung

Das OPEN-CNC-Shield kann mit 12-24V betrieben werden. Ich empfehle 24V.
Wenn das Shield mit 12V versorgt wird, kann es sein, dass im 12V Netz keine 12V ankommen, da durch den Spannungswandler ein geringer Spannungsverlust entsteht. Für diesen Fall ist der Jumper JP17 vorgesehen. Wenn dieser gesetzt ist, wird die Eingangsspannung direkt in das 12V Netz weitergeleitet.

> Achtung, der Jumper JP17 darf auf keinen Fall gesetzt werden, wenn mehr als 12V am Terminal T2 (12-24V in) anliegen. Das kann diverse Bauteile auf dem Board zerstören!

2.1 Benötigte Stromstärke

Wenn **keine** Aufstecktreiber verwendet werden, wird ein 24V 5A Netzteil völlig ausreichen und bietet auch noch Reserven. Der zweite 12-24V Vin ist in diesem Fall unnötig. Hier ist ein Beispiel Netzteil:

Sollten Aufstecktreiber verwendet werden sollte ein stärkeres Netzteil verwendet werden. Generell kann pro Treiber mit etwa 2A gerechnet werden. Hier ist zum Beispiel eins mit **24V und 15A**, damit können ohne Probleme alle 6 Aufstecktreiber verwendet werden und es bleibt auch noch genug

Reserve:
Die Netzteile mit hohen Strömen bieten meist mehrere Anschlüsse. Hierfür kann der zweite 12-24V Vin genutzt werden. Dieser dient der Sicherheit und soll der Wärmeentwicklung bei hohen Strömen entgegenwirken.

3 Eingänge

Es gibt 16 Eingänge auf dem OPEN-CNC-Shield. Diese können entweder mit positiver Spannung von 5-24V geschaltet werden oder mit GND. Wählbar ist dies über einen Jumper an dem jeweiligen Eingang. Jeder Eingang besitzt seinen eigenen Jumper.

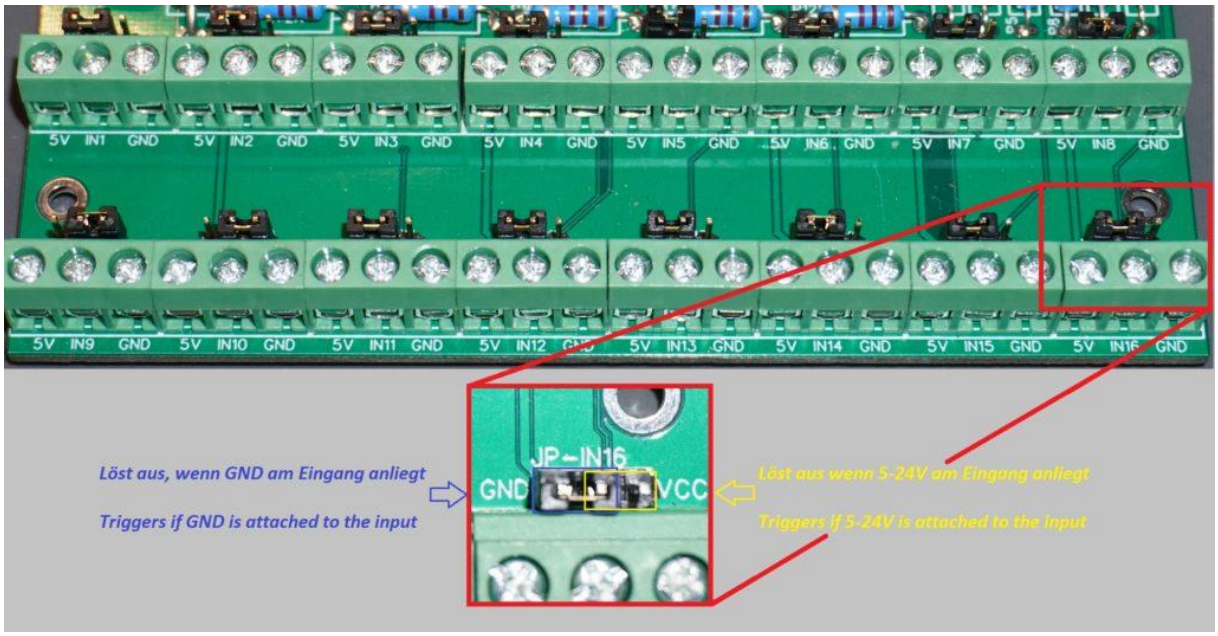


Abbildung 1: Eingänge

> Falls Autosquaring genutzt werden soll, braucht jeder Motor seinen eigenen Endstopp. Mehr dazu unter Autosquaring.

3.1 Spannung an den Eingängen

Ab Platinen Version 1.5: Die Spannung, welche an den Terminals für die Eingänge anliegt, kann mit den Jumper JP13 und JP14 festgelegt werden. Man hat die Wahl zwischen 5V und 12V. Es ist ebenfalls möglich selbst eine Spannung anzulegen. Dafür müssen die Jumper frei gelassen werden und die Spannung kann dann an den jeweiligen COM-Anschluss gelegt werden.

3.2 Endstopps

3.2.1 Normally closed / normally open

Es können entweder normally closed (NC = im Normalzustand geschlossen / ausgelöst) oder normally open (NO= im Normalzustand geöffnet / nicht ausgelöst) Endstopps angeschlossen werden. Diese können entweder mit mehreren an einem Eingang angeschlossen werden oder es kann jeder Endstopp an seinen eigenen Eingang angeschlossen werden.
Ich würde eine normally closed Verkabelung bevorzugen, da hierbei auch Kabelbrüche direkt erkannt werden können. Eingänge, an denen normally closed Sensoren angebracht sind müssen in der Software vermutlich invertiert werden.

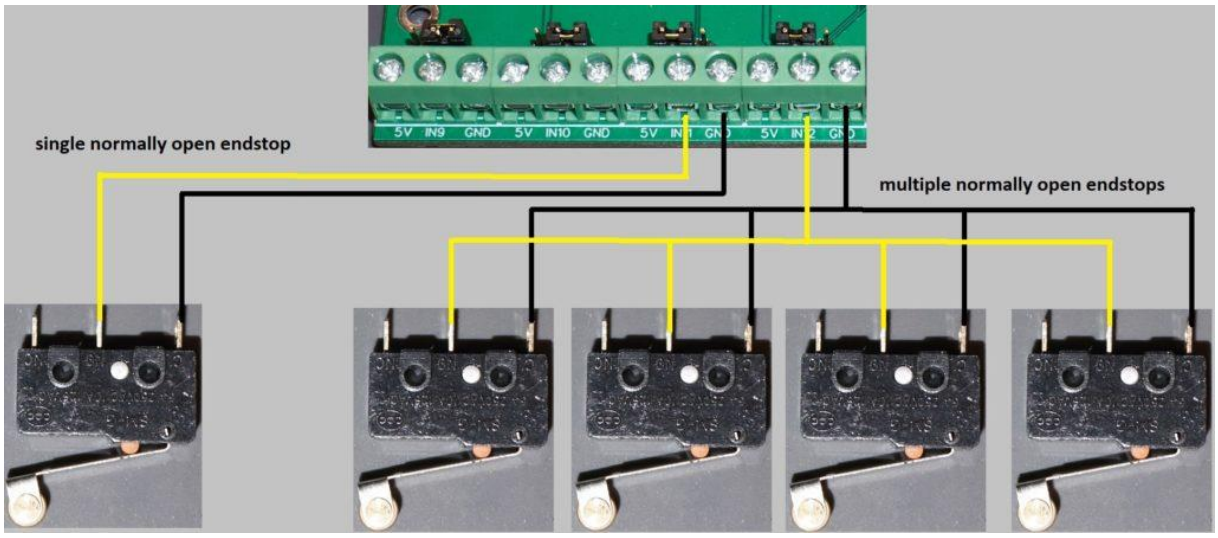


Abbildung 2: normal offen Verkabelung

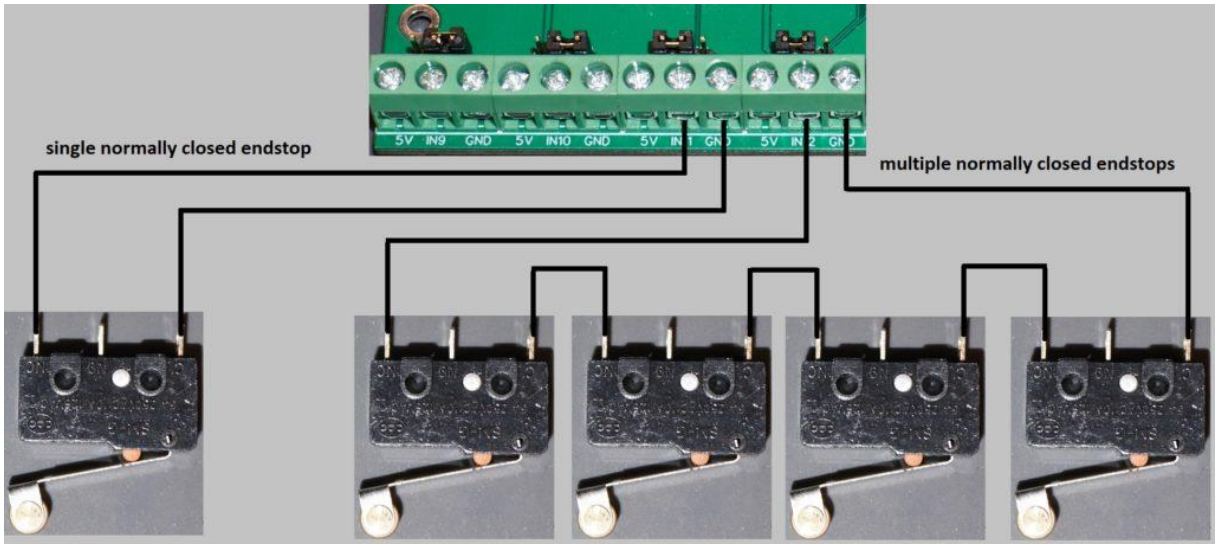


Abbildung 3: normal geschlossen Verkabelung

3.2.2 Näherungssensoren PNP oder NPN

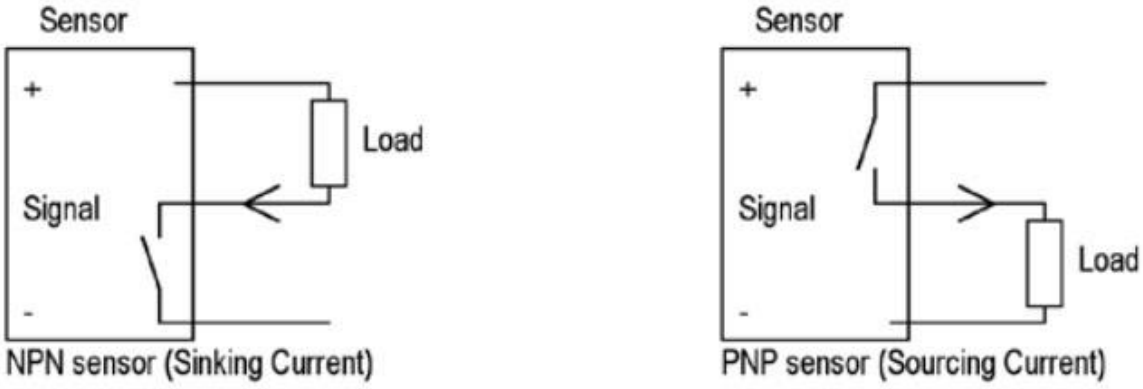


Abbildung 4: Näherungssensoren

Es können sowohl PNP Sensoren als auch NPN Sensoren angeschlossen werden. Hierzu muss lediglich die passende Jumper Einstellung an dem Eingang vorgenommen werden. Diese gibt es ebenfalls als NC (normally closed) oder NO (normally open) Variante. Siehe oben für die Verkabelung.

NPN NC Sensor:

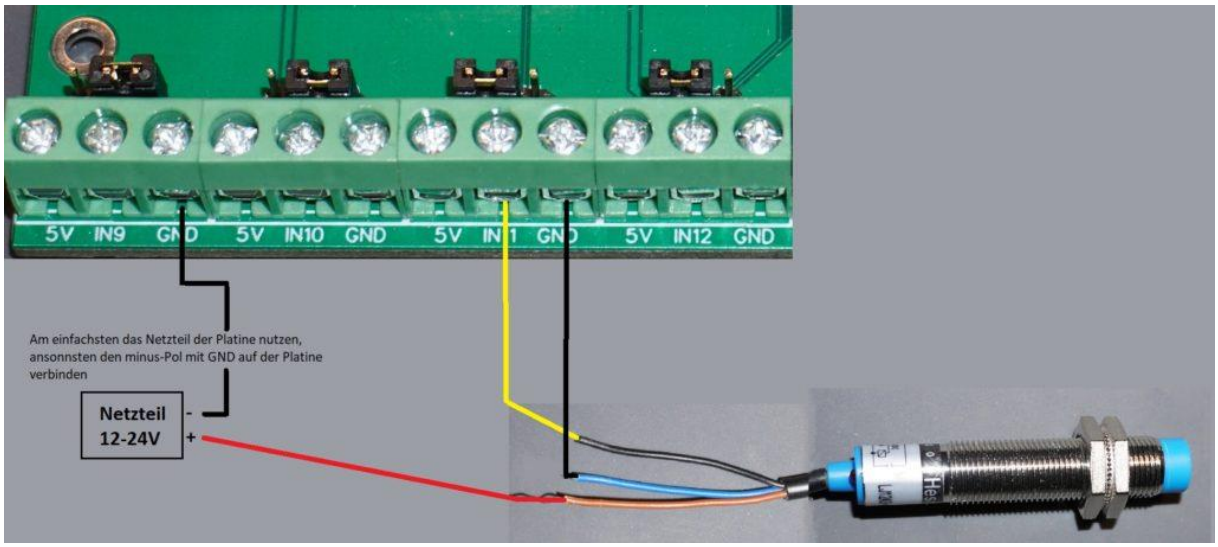


Abbildung 5: Sensor Anschliessen

3.3 LED als Signal für Eingänge

Möchte man eine LED verbauen, welche anzeigt, ob ein Eingang ausgelöst wurde, gibt es verschiedene Möglichkeiten dieses zu realisieren.

3.3.1 Direktes Anlöten an die Arduinos

Vorteil:	Unabhängig von der Jumper-Einstellung und Stromspannung der Eingänge.
Nachteil:	Es müssen Kabel verlötet werden.

Die Kabel für die LEDs können entweder von unten an die Platine gelötet werden oder von oben direkt an die Arduino Megas. Im unteren Bild ist ein Beispiel für Eingang 1. GND von der LED wird mit dem Arduino Pin des Eingangs verbunden und der Pluspol der LED wird über einen 220 Ohm Widerstand mit 5V (Diese kann man sich irgendwo am Board abgreifen, zum Beispiel an einigen Schraubterminals) verbunden. Welcher Arduino dabei genommen wird, spielt keine Rolle. Eine Pinbelegung ist dem Schema zu entnehmen → Gitlab – Timos Werkstatt

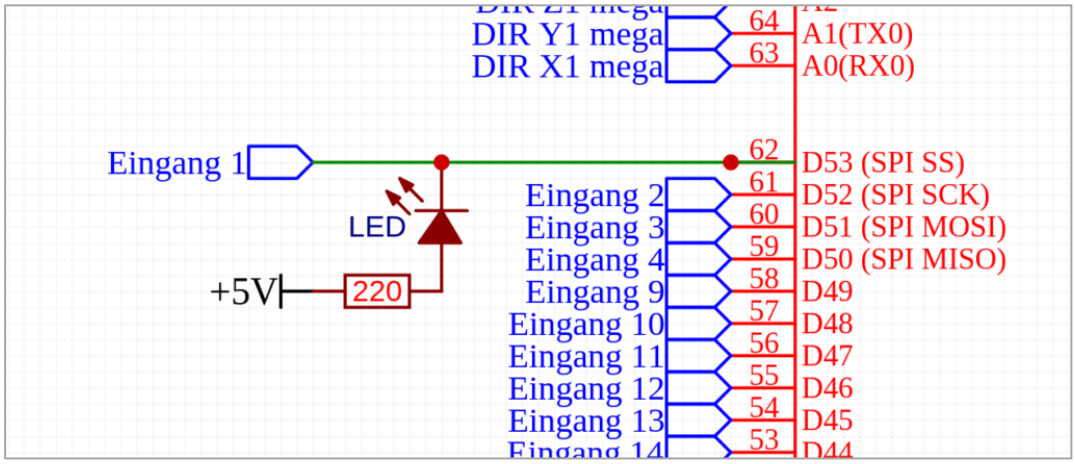


Abbildung 6: Schema Pinbelegung

3.3.2 LED ebenfalls am Eingang anschließen

Vorteil: Einfach anzuschließen, kein Löten
Nachteil: Abhängig von der Jumper-Einstellung und der Spannung des Eingangs

Jumper steht auf GND:
In diesem Fall wird die LED mit dem Minuspol an „IN“ vom Eingang angeschlossen und mit dem Pluspol mit einem Widerstand in Reihe an z.B. COM, oder direkt mit einem 220 Ohm Widerstand an 5V. Je, nachdem wie viel Spannung an COM anliegt, werden verschiedene Widerstände benötigt. Siehe dazu die untere Tabelle.

Jumper steht auf VCC
Der Minuspol der LED wird an GND angeschlossen und der Pluspol in Reihe mit einem passenden Widerstand an IN. Welcher Widerstand benötigt wird kann der unteren Tabelle entnommen werden:

Spannung	Widerstand
5V	220 Ohm
12V	680 Ohm
24V	1,2K Ohm

3.3.3 . Arduino Mega Ausgang für LED programmieren

Vorteil:	Es muss weder gelötet werden, noch ist man abhängig von der Jumper-Einstellung oder Spannung
Nachteil:	Es muss ein wenig Arduino Code geschrieben werden.

Bei dieser Methode muss der Code von dem zweiten Arduino ein wenig erweitert werden. In der Setup Methode müssen die Pins für die Eingänge auf „INPUT_PULLUP“ gestellt werden. Dazu einfach kurz googlen, wie man Eingänge bei Arduinos setzt. Die benötigten Pins findet man ebenfalls im Schema ⇾ Gitlab – Timos Werkstatt

Danach erstellt man in der Loop-Methode eine Prüfung, ob der als „INPUT“ deklarierte Pin ausgelöst wurde (Google ⇾ Arduino digitalRead). Und falls dem so ist, setzt man einen der freien Pins am Pinout **J8** entsprechend. Die LED wird dann mit dem Pluspol in Reihe mit einem 220 Ohm Widerstand an dem Pinout Pin angeschlossen und mit dem Minuspol irgendwo an GND.

Hier ein wenig Beispielcode (bisher ungetestet) für Eingang 1:

```
const byte input1 = 15; // Digital Pin for Input1
const byte output1 = A1; // Output Pin on Pinout J8 on the Shield

setup() {
  ...
  pinMode(input1, INPUT_PULLUP); // Set pinmode for input pin

  pinMode(output1, OUTPUT); // Set pinmode as output
  digitalWrite(output1, LOW); // Initial state for the output LED
  ...
}

loop() {
  ...
  bool input1State = digitalRead(input1); // Read input 1 pin
  // If input1State is LOW (which means it is triggered) write HIGH
  // to the output pin and vice versa
  digitalWrite(output1, !input1State);
...
}
```

4 Ausgänge

Das OPEN-CNC-Shield bietet 8 Ausgänge, welche in Estlcam nach Belieben geschaltet werden können. Es sind zwei Relaistreiber verbaut. Für jeden Relaistreiber kann die Spannung über die Jumper JP15 und JP16 gesetzt werden. OUT1-4 sind an COM1 angeschlossen und werden somit mit dem Jumper JP15 mit einer Spannung verbunden.

Achtung: COM1 wird ebenfalls für das Spindel Relais genutzt, mehr dazu unter Spindel Steuerung. OUT5-8 sind an COM2 angeschlossen und werden mit dem Jumper JP16 konfiguriert.

So ist es zum Beispiel möglich, dass die Ausgänge 1-4 mit 5V arbeiten und die Ausgänge 5-8 mit 12V. Möchte man an eine andere Spannung kann man die jeweiligen Jumper offenlassen und diese Spannung direkt an COM1 oder COM2 anlegen. Diese ist dann an allen gleichnamigen COM Anschlüssen verfügbar, darf aber 50V nicht überschreiten.

> Pro Ausgang darf maximal 500mA Strom fließen. Daher immer mit Relais arbeiten, sobald die Grenze eventuell überschritten wird. Auch auf die maximalen Ströme der Netze achten. Im 5V und 12V Netz stehen jeweils max. 3A zur Verfügung. Die Ausgänge sind hier nicht die einzigen Abnehmer – Lüfter, Arduinos usw. ziehen auch Strom!

In der nachfolgenden Bild ist beispielhaft der Anschluss eines 5V Relais. COM2 wird hierbei auf 5V gesetzt. Dann wird VCC mit COM2 verbunden (Also 5V an VCC) und GND mit GND. Geschaltet wird dann mit OUT5 und OUT6. Sobald diese in Estlcam ausgelöst werden, liegt dort GND an. Sollte das Relais nicht schalten muss der Jumper auf dem Relais einmal umgesetzt werden. Dieser ist auf diesem Bild Gelb und unter den Anschlüssen für GND, IN1, IN2, VCC.

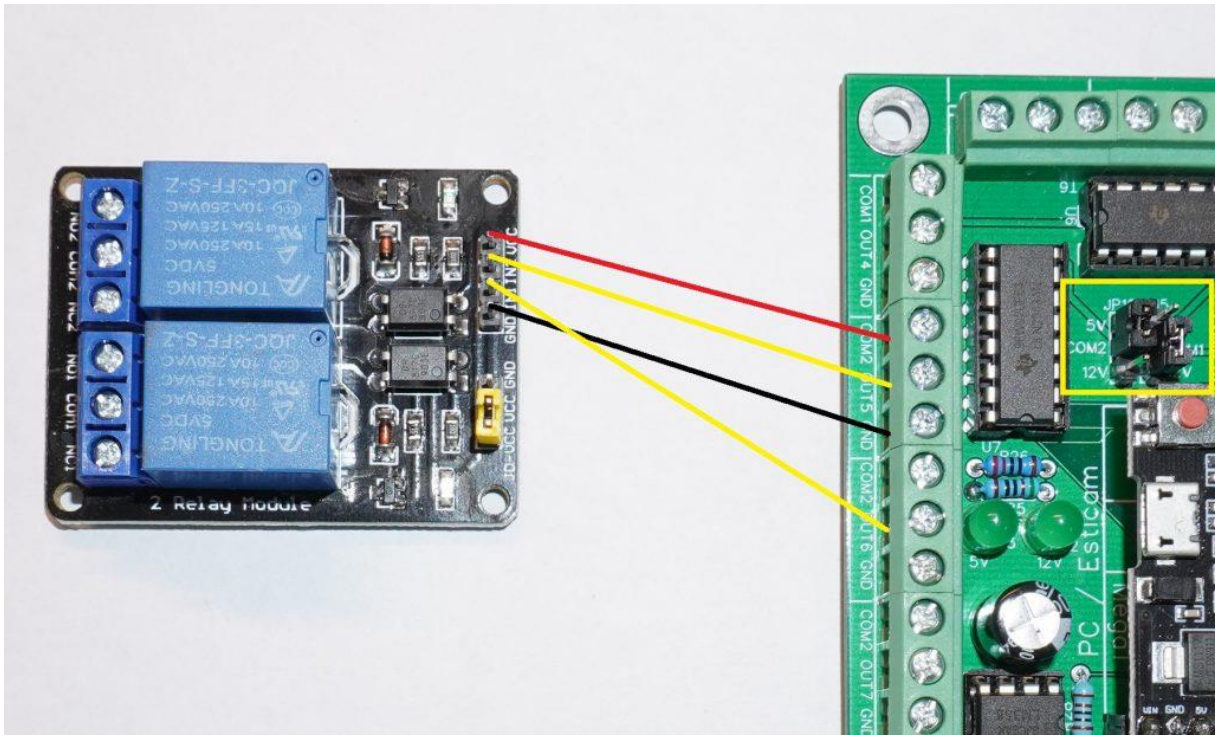


Abbildung 7: Anschluss 5V Relais

5 Aufstecktreiber / Externe Motortreiber

Das OPEN-CNC-Shield unterstützt sowohl Aufstecktreiber wie z.B. den A4988 oder DRV8825 als auch den Anschluss externer Motortreiber. Für was man sich entscheidet, bleibt einem selbst überlassen. Generell kann man sagen, dass externe Treiber einen deutlich höheren Kostenaufwand haben, aber dafür auch höhere Ströme unterstützen und nicht so eine gute Kühlung benötigen. Für Nema 17 Motoren mit etwa 2A Leistung sollten die Aufstecktreiber bei guter Kühlung vollkommen ausreichen. Möchte man größere/stärkere Motoren/Servos/“Closed-Loop Motoren“ sollte man sich eher mit externen Treibern beschäftigen. Man kann auch erstmal klein anfangen und später auf externe Treiber aufrüsten.

5.1 ENA Einstellung

Ab Platinen Version 1.6: Die Einstellung welchen Zustand der ENA (Enable) Anschluss im Standard hat kann über den Jumper JP21 getätigt werden. Dies gilt sowohl für die Aufstecktreiber als auch für externe Treiber. Besitzen die Treiber selbst schon entsprechende Pullup- / oder Pulldown Widerstände kann der Jumper frei gelassen werden.

Platinen Version v1.6 und v1.7: Der Widerstand R20 ist leider falsch beschriftet. Dort muss anstatt eines 10K Widerstands ein 1K Widerstand eingesetzt werden. Ist dort der 10K Widerstand verbaut geht auch nichts kaputt, allerdings funktioniert dann der Jumper für den ENA State nicht (JP21).

5.2 Aufstecktreiber

Alle Änderungen sollten immer im Stromlosen zustand erfolgen. Bitte auch unbedingt darauf achten die Treiber richtig herum einzustecken! Hier gibt es Unterschiede zwischen A4988 und DRV8825 Treiber! Wurde ein Treiber einmal falsch herum eingesteckt und das Board unter Strom gesetzt ist der Treiber sehr wahrscheinlich defekt.

5.2.1 Microstepping

Das Microstepping der Aufstecktreiber wird mit Jumpern eingestellt. Diese befinden sich auf der Platine genau unter den jeweiligen Treibern. Diese Einstellung kann demnach nur erfolgen, wenn die Treiber nicht eingesteckt sind.

Mit Microstepping kann man die Schritte des Motors weiter verkleinern. Ein typischer Schrittmotor hat z.B. 200 Schritte für eine volle Umdrehung. Stellt man das Mircostepping nun etwa auf 1/8 Schritt bedeutet dies: 200 × 8 = 1600 – Es werden nun 1600 Schritte für eine volle Drehung benötigt. Je höher die Microstepping Einstellung, desto geringer wird der Haltemoment der Motoren. Generell kann man mit diesen Werten experimentieren, um zu schauen, welche sich am besten für die eigene Maschine eignen. Oft geben Hersteller auch an, wie die Mikroschritte eingestellt werden sollten. Hat man hier eine Einstellung gewählt muss, man diese auch unbedingt der Software mitteilen.

MS0	MS1	MS2	Mikroschritte
0	0	0	1 – Vollschritt
1	0	0	1/2 Schritt
0	1	0	1/4 Schritt
1	1	0	1/8 Schritt
0	0	1	1/16 Schritt
1	0	1	1/32 Schritt
0	1	1	1/32 Schritt
1	1	1	1/32 Schritt

Tabelle 2: Mikroschritte

0 bedeutet Jumper ist nicht gesetzt, 1 bedeutet Jumper ist gesetzt

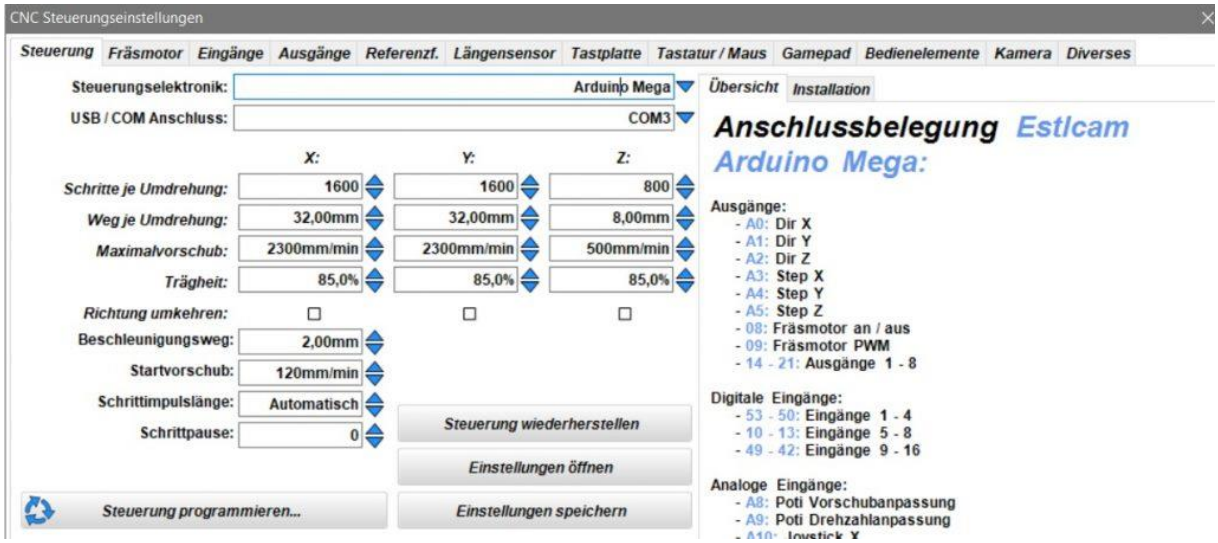


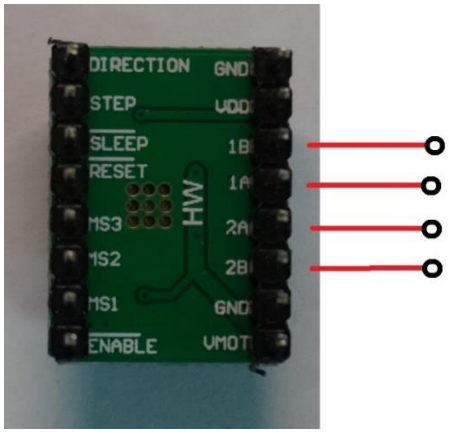
Abbildung 8: Estlcam Microschritte einstellen

Hier ist ein Beispiel für Estlcam. Die Mikroschritte sollten so für eine MPCNC-Fräse funktionieren. Die Jumper sind in diesem Beispiel für die x- und y-Achse auf 1/8 Schritt und für die z-Achse auf 1/4 Schritt eingestellt.

5.2.2 Einsetzen der Treiber

> Bitte nur im Stromlosen zustand!

Hier ist es wichtig auf die richtige Orientierung zu achten, welche sich durchaus zwischen den verschiedenen Treiber Typen unterscheiden kann.



Ich mache es immer so, dass ich unter die Treiber schaue und die Position der Motoranschlüsse suche (1B, 1A, 2A, 2B) und darauf achte, diese dann in Richtung des 4Pin Headers für den Motor auszurichten.

Abbildung 9: Treiber von unten

So sieht das Ganze dann von oben aus. Die Motoranschlüsse gehen in Richtung des Pin-Headers. ***Achtung, das Bild eines DRV8825 Treibers unterscheidet sich! Nur auf die Motoranschlüsse achten!***

Abbildung 10: Treiber von oben

5.2.3 Fault Jumper

Dieser Jumper ist neben jedem Treiber zu finden. Bezeichnungen JP5, JP8, JP4, JP3, JP9, JP12. Falls man die DRV8825 Treiber nutzt, bleibt dieser Jumper offen bzw. wird nicht gesetzt. Die Treiber geben hierüber ein Signal, wenn diese überlastet sind, welches man abgreifen und nutzen könnte. **Bei den A4988 Treibern muss dieser Jumper gesetzt werden!** Hiermit wird der Treiber mit 5V versorgt.

5.2.4 Treiberstrom einstellen

Die Treiber besitzen ein Potentiometer, mit welchem der Strom bzw. die Spannung eingestellt werden kann. Dies wird folgendermaßen berechnet: Maximalstrom der Motoren geteilt durch 2. Also für Standard Nema 17 Schrittmotoren mit 2A ergibt sich eine Referenzspannung von 1V (2A / 2 = 1). Bei dieser Spannung brauchen die Treiber allerdings eine sehr gute Kühlung. Bei der MPCNC werden die Treiber meist auf 0,7V eingestellt. Kühlen muss man die Treiber aber trotzdem ;-).
Das Einstellen der Spannung muss im bestromten Zustand erfolgen. Daher muss man hier sehr vorsichtig sein, nicht abzurutschen und somit einen Kurzschluss zu verursachen. In dem folgenden Video habe ich kurz gezeigt, wie man die Motoren einstellen kann:

Film 1: Aufstecktreiber Strom/Spannung einstellen

5.2.5 Anschließen der Motoren

Die Motoren werden an den Anschlüssen J4, J7, J1, J2, J10 und J11 angeschlossen. Hierfür kann man

Dupont Stecker verwenden. Hier zum Beispiel ein Set mit Zange:
Folgendes Anschlussschema sollte helfen:

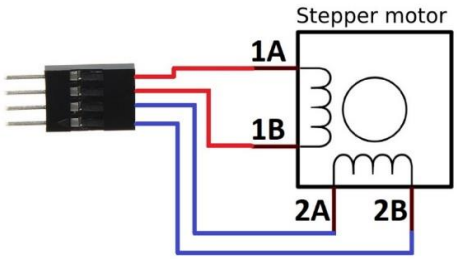


Abbildung 11: Motor anschliessen

Um herauszufinden welche Kabel zusammengehören kann man einfach eine Durchgangsmessung machen. Die Anschlüsse 1A und 1B sollten einen Durchgang haben. Genauso wie 2A und 2B
.

5.3 Externe Motortreiber

Es gibt eine Vielzahl unterschiedlicher externer Treiber. Nutzt man externe Treiber, wird die Microstepp Konfiguration direkt an dem Treiber vorgenommen und nicht mehr auf der Platine. Die Jumperstellung spielt also keine Rolle mehr auf der Platine. Ebenfalls braucht die Platine nun nur noch einen Bruchteil des Stroms, da die Treiber direkt mit dem Motorstrom betrieben werden. Hier ist ein Beispiel für TB6600 Treiber:

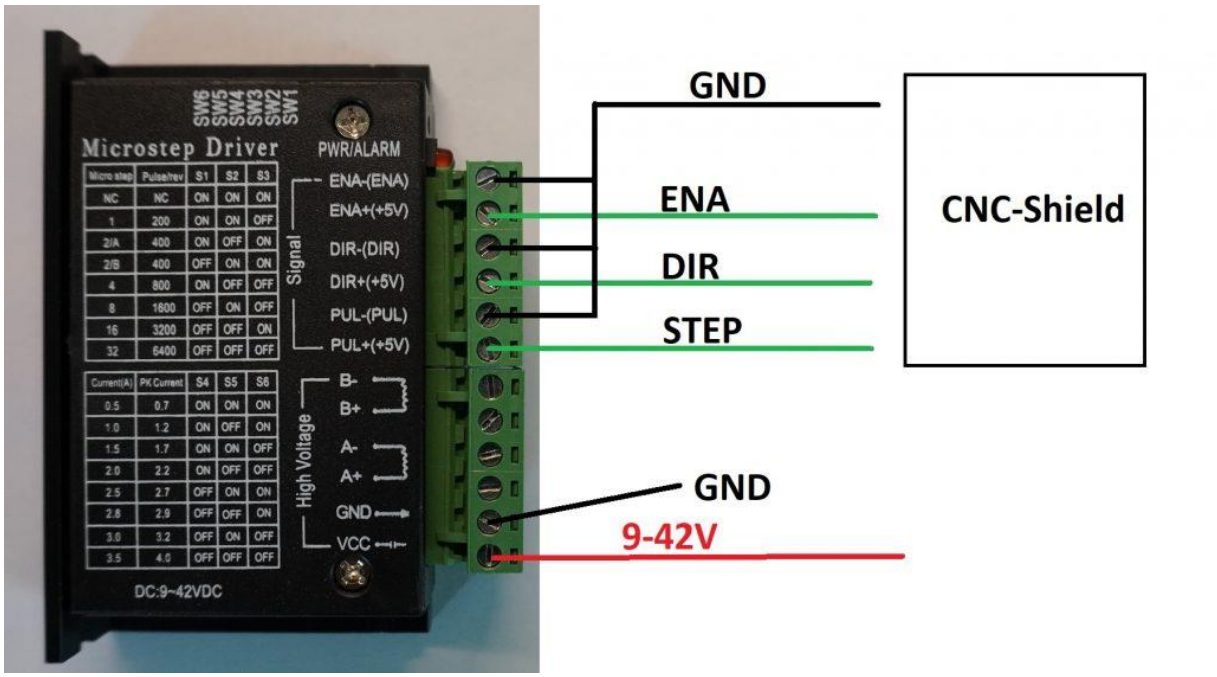
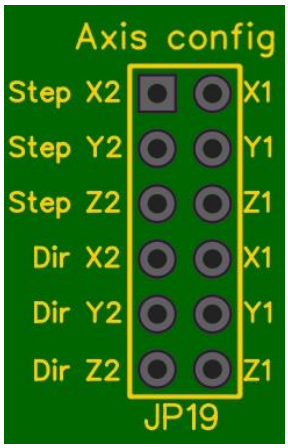


Abbildung 12: Externer Motorentreiber

Auf dem OPEN-CNC-Shield sind für jeden Treiber 4 Ausgänge. Einmal Enable(ENA) zum stromlos Schalten der Motoren, dann DIR für die Drehrichtung und STEP für die Schritte. Als letztes wird GND an allen Minus Polen aufgelegt. Im „High Voltage“ Bereich wird der Schrittmotor und das Netzteil angeschlossen.

6 Konfiguration der Achsen



Hier kann man die Achsen miteinander verbinden. Je nachdem wofür man das Shield verwendet, können die Achsen konfiguriert werden. Setzt man beispielsweise einen Jumper bei „Step X2“ und „X1“ sind die Steps der beiden Achsen verbunden / gleich. Gleiches gilt für die anderen Jumper.

Abbildung 13: Achsen Konfigurieren

6.1 Achsen in Estlcam

Bei der Verwendung von Estlcam sollten alle Jumper gesetzt sein. Estlcam unterstützt nur drei Achsen (x, y und z). Durch Setzen der Jumper Verfahren die Motoren für x1 und x2 dieselben Schritte und drehen in dieselbe Richtung. Gleiches gilt für die anderen Achsen.

6.2 Achsen in LinuxCNC

6.2.1 LinuxCNC Adapters v1.0

Hier kann der Jumper für die z-Achse sowohl bei „Step“ als auch bei „Dir“ freigelassen werden. Hierdurch kann der Motor Z2 für eine weitere Achse verwendet werden. Demnach könnten man dadurch in LinuxCNC 4 Achsen verfahren.

6.2.2 LinuxCNC Adapter 6Achs v1.0

Hier sollten keine Jumper gesetzt werden. Dadurch können in LinuxCNC 6 Achsen verfahren werden.

7 Spindel Steuerung

Das OPEN-CNC-Shield bietet vielfältige Möglichkeiten der Spindel Steuerung. Unter anderem ist es möglich diese ein- und auszuschalten oder die Drehzahl zu regulieren.

> Achtung! Lebensgefahr! Lasse den Fräsmotor von einem Elektriker anschließen!

Auf dem untenstehenden Bild sind die Terminals zum Anschluss einer Spindel abgebildet.

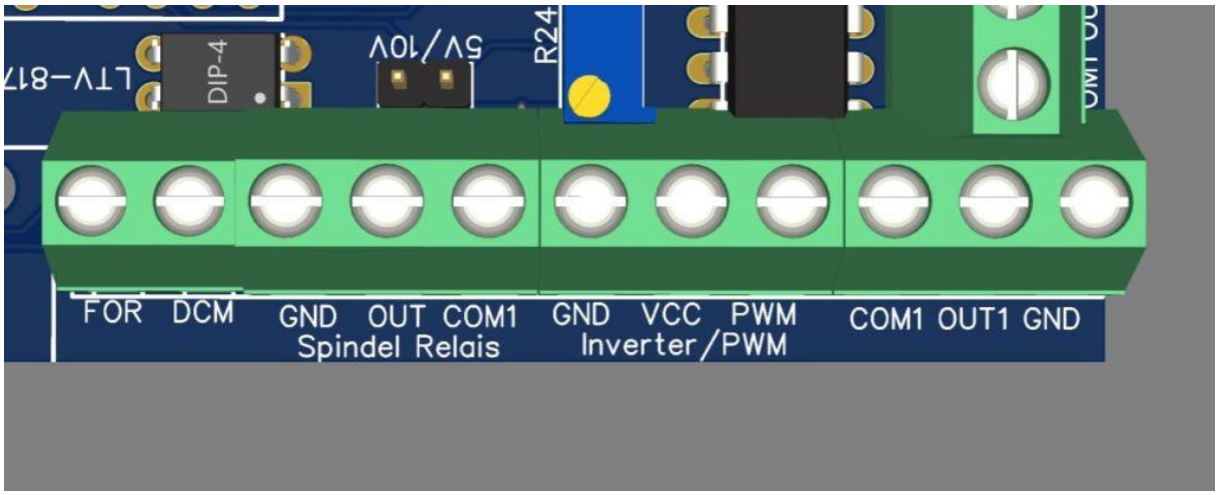


Abbildung 14: Spindel anschliessen

Beschriftung	Funktion
FOR / DCM	Dieser Anschluss ist für die Steuerung eines Frequenzumrichters (FU) z.B. bei der typischen Chinaspindel. Hiermit wird der FU an- und ausgeschaltet. Genaueres hierzu unter Anschluss eines Frequenzumrichters.
Spindel Relais GND/OUT/COM1	Hier kann ein Relais zum Einschalten einer Spindel angeschlossen werden. Es kann hier auch sonstige Hardware welche gleichzeitig mit der Spindel eingeschaltet werden soll über ein Relais geschaltet werden (zum Beispiel die Pumpe bei einer Wassergekühlten Spindel). An dem Ausgang OUT liegt im geschalteten Zustand GND(Masse) an. COM1 wird über die Jumper auf der Platine gesetzt und kann entweder 5V, 12V oder eine selbst angelegte Spannung sein. **NIEMALS 230V hier anlegen!!!** Mehr dazu unter Anschluss einer Fräse.
Inverter/PWM GND/VCC/PWM	Diese Anschlüsse dienen der Drehzahlsteuerung. Zwischen GND und VCC wird ein analoges Signal zwischen 0-10V ausgegeben. Ist der Jumper **JP18(5V/10V)** gesetzt wird hier ein analoges Signal zwischen 0-5V ausgegeben. Dieses kann über den Einstellbaren Widerstand R24 justiert werden. Siehe dazu Analoges Signal kalibrieren. Zwischen PWM und GND wird ein PWM Signal zwischen 0 und 5V ausgegeben. Dieses kann z.B. für den Anschluss eines Lasers verwendet werden, siehe dazu Anschluss eines Lasers.

Tabelle 3: Beschreibung Terminals Anschlüsse

7.1 Analoges Signal kalibrieren

Wie der analoge Ausgang auf 10V kalibriert wird, zeigt das folgende Video. Dieser Schritt ist übrigens nur notwendig, wenn auch eine Spindel mit externer Drehzahlsteuerung angeschlossen wird.

Film 2: Analoge Steuerung Drehzahlsteuerung kalibrieren

Hier nochmal kurz zusammengefasst:

1.	Platine mit Strom versorgen.
2.	Arduino an Computer mit Estlcam anschließen
3.	Multimeter auf Gleichstrommessung stellen und an den Terminal *Inverter/PWM* anschließen. Dabei kommt der Minus-Pol an GND und der Plus-Pol an VCC.
4.	In der CNC Steuerung von Estlcam den Arduino Mega mit richtigem Port auswählen.
5.	Unter dem Reiter *Fräsmotor* die Einstellungen für die China Spindel laden.
6.	Die Steuerung programmieren.
7.	Den Button drücken, um die Spindel zu starten und den Befehl „pwm100“ in der Kommandozeile eintippen.
8.	Sicherstellen, dass der Schieberegler für die Spindel Geschwindigkeit über 100 % steht und dann so lange an der Stellschraube auf dem Potentiometer drehen, bis das Multimeter etwa 10V anzeigt.
9.	Fertig.

7.2 Anschuss einer Fräse (Ein/Aus)

Abbildung 15: Fräse Ein/Aus

Normale luftgekühlte Fräse. Drehzahlregelung an der Fräse selbst. Achtung, laut.

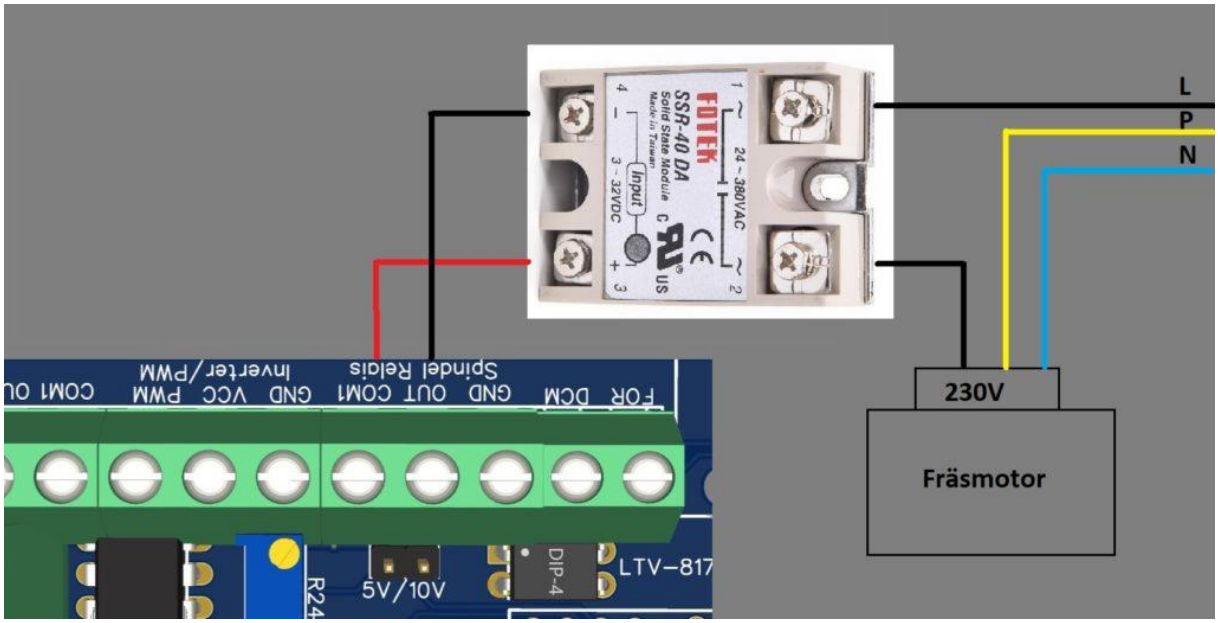


Abbildung 16: Anschluss Fräse

Der Fräsmotor wird über ein Relais mit der Steuerung verbunden. Es sollte ein Solid State mit Nulldurchgangsschaltung verwendet werden. Hier ist ein Beispiel: Ebay – RM1A23D25. Achtung, je nach Stromentnahme kann das Relais heiß werden und muss gekühlt werden. Hier ist zum Beispiel ein passender Kühlkörper: Amazon – Kühlkörper.
Die typischen mechanischen 5V Relais für z.B. Arduinos sind nicht geeignet. Diese können Störungen verursachen und geben auch des Öfteren mal den Geist auf.

7.3 Anschluss Fräse mit externer Drehzahlsteuerung

Abbildung 17: Fräse externe Drehzahlregelung

Fräsen mit externer Drehzahlsteuerung wie zum Beispiel eine AMB 1050 FME oder auch Fräsen von Mafell

Amazon – AMB 1050
Amazon – Mafell FM 1000

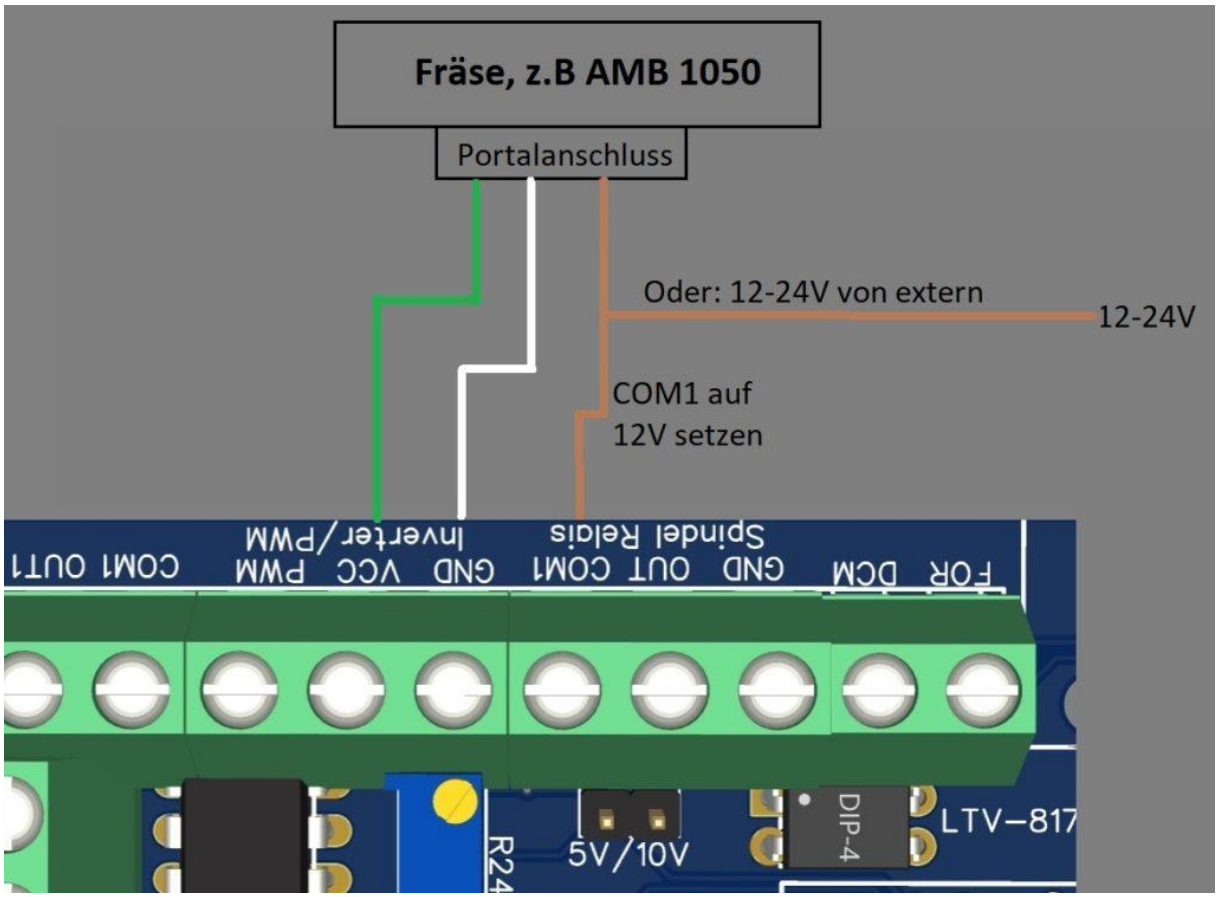


Abbildung 18: Anschluss Fräse mit externer Drehzahlsteuerung

Die Fräse kann hier ohne ein Relais direkt an dem OPEN-CNC-Shield angeschlossen werden. Das braune Kabel kann entweder an COM1 angeschlossen werden, oder direkt an das Netzteil zur Stromversorgung der Platine angeschlossen werden.

7.4 Anschluss eines Frequenzumrichters (China Spindel)

Typische wassergekühlte China Spindel. Besitze ich selbst auch. Wenn die einmal läuft, läuft sie

Abbildung 19: Frequenzumrichter Set

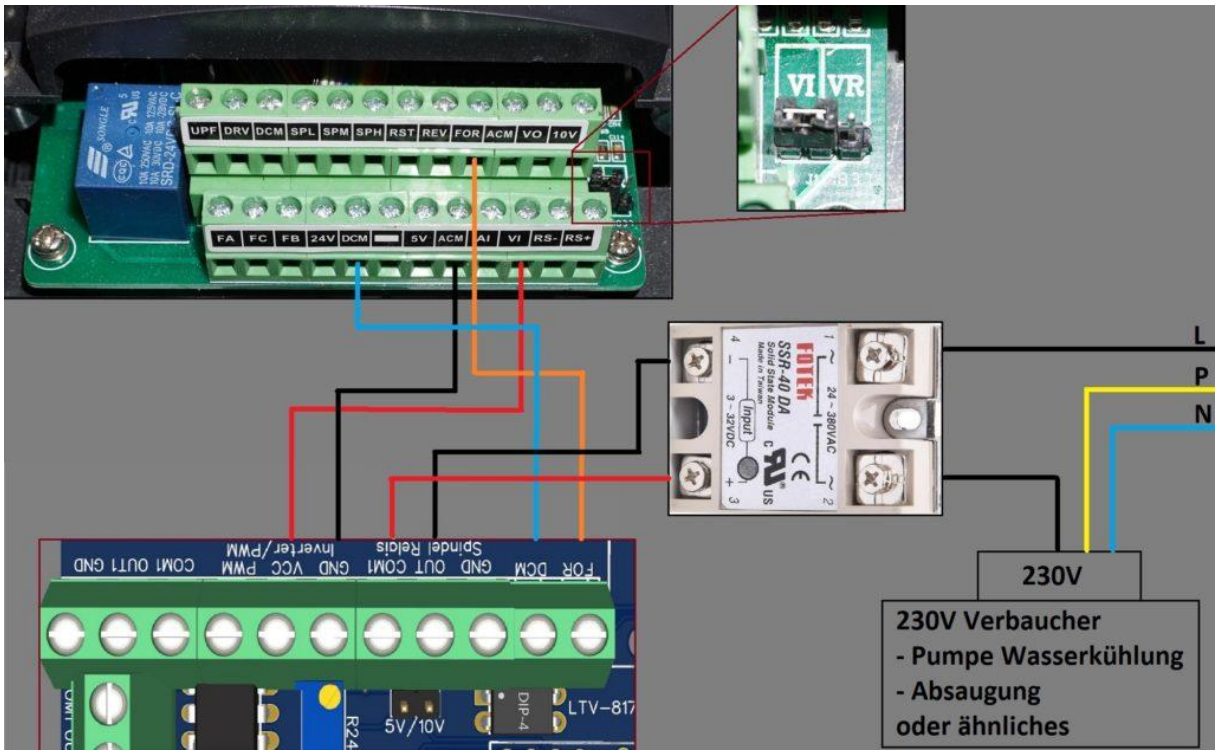


Abbildung 20: Anschluss eines Frequenzumrichters

Verbindungstabelle:

OPEN-CNC-Shield	Frequenzumrichter	Erklärung
FOR / DCM	FOR / DCM	An- und ausschalten des Frequenzumrichters. Strom fließt von FOR nach DCM. Hier ist die Fließrichtung entscheidend. Sollte die Spindel nicht schalten, obwohl sie eigentlich „müsste“ einfach mal die Kabel vertauschen und ausprobieren, ob es dann funktioniert.
Inverter/PWM GND und VCC	ACM(GND) und VI(VCC)	Diese Verbindung ist für die Drehzahlsteuerung. Normalerweise 0-10V. Über den Jumper JP18 auf 0-5V einstellbar. Bei meinem FU konnte man auch einstellen, wie viel Volt der Eingang funktionieren soll. Dafür bitte in die Bedienungsanleitung des FU schauen.
Spindel Relais OUT / COM1		Optional: Hier kann man zum Beispiel ein Relais anschließen, um Pumpen oder Absaugung etc. zu schalten. Achtung: Je nach Höhe der Stromabnahme kann das Relais heiß werden und muss gekühlt werden!

Tabelle 4: Verbindung Frequenzumrichter

Achtung, damit die **Drehzahlsteuerung** funktioniert muss der Jumper bei **VI/VR** wie im obigen Bild gesetzt sein. Ansonsten wird die Drehzahl weiterhin über den Poti am FU gesteuert.

Weitere Informationen zu Einstellen und Anschließen des Frequenzumrichters findet man auch auf der Estlcam Seite. Dort ist dies noch einmal sehr gut beschrieben: Estlcam – Chinaspindel

7.5 Anschluss eines Lasers

Hier einmal beispielhaft der Anschluss eines Lasers. Ich habe folgenden Laser verwendet: ebay

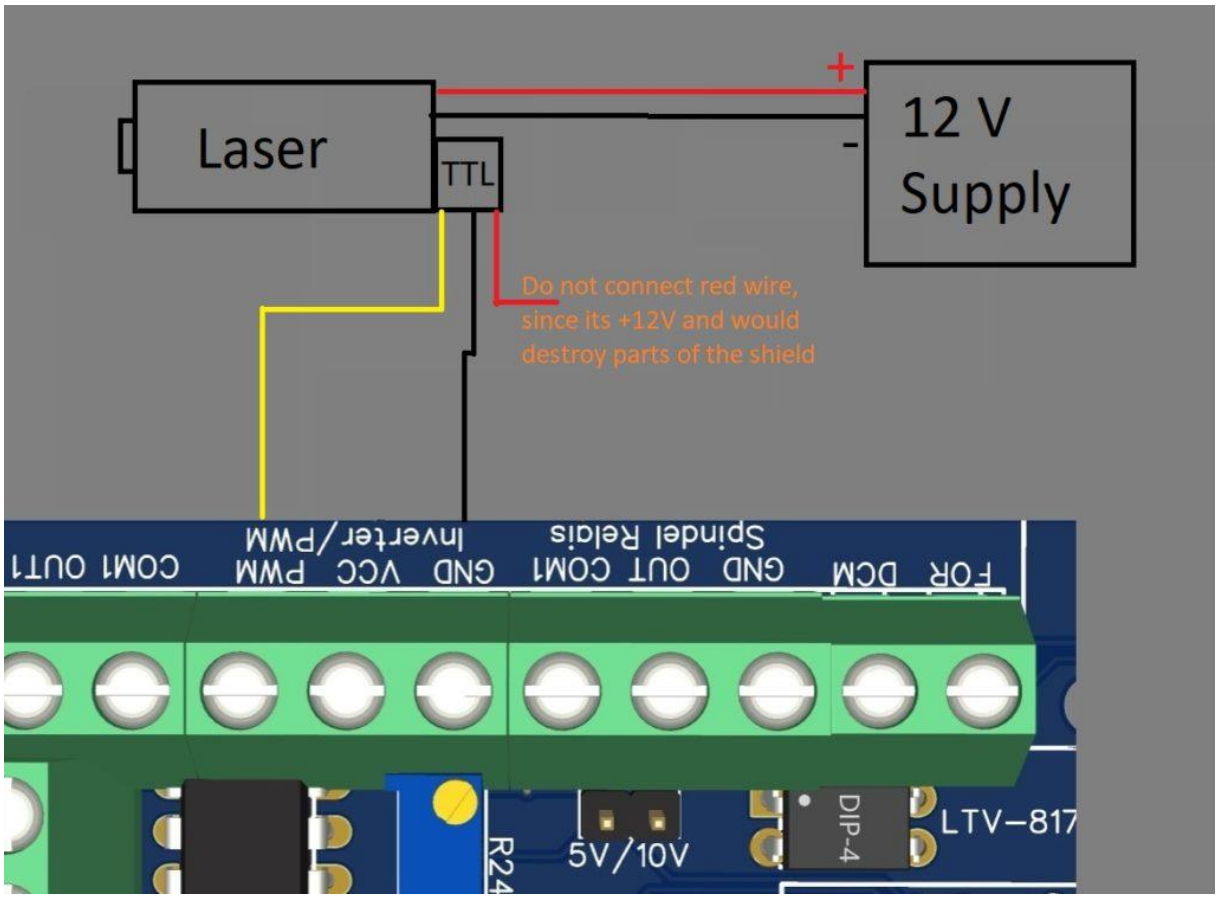


Abbildung 21: Laser anschliessen

Der Laser, den ich verwende, braucht 12V, diese hole ich von einem externen Netzteil. Da der Laser nur läuft, wenn über PWM ein Signal herausgegeben wird, spare ich es mir, den Laser über ein Relais mit Strom zu versorgen. Das wäre aber auch möglich. Das Relais würde dann genauso wie die Spindeln an den Anschluss „Spindel Relais“ auf der Platine angeschlossen werden.
Damit TTL des Lasers funktioniert braucht dieser ein sauberes PWM Signal von 0 oder 5V. Dieses findet man auf der Platine an dem Anschluss „PWM“. Der Anschluss „VCC“ funktioniert auch dann nicht, wenn der Jumper für 5V gesetzt ist. VCC gibt lediglich eine Analoge Spannung heraus.

> Wichtig, bei meinem Laser lagen an dem roten Kabel von TTL 12V an. Diese keinesfalls mit der Platine verbinden. Dabei können Bauteile zerstört werden. Es reicht das gelbe Signal Kabel und das schwarze GND Kabel anzuschließen.

Wie das Ganze dann angesteuert wird und Softwareseitig funktioniert zeige ich in diesem Blog Beitrag: LaserGRBL mit dem OPEN-CNC-Shield

Link zur angepassten GRBL Firmware: Gitlab – grbl

8 SUB-D 37 Anschluss

Dieser Anschluss führt viele der Arduino Pins nach außen und dient als Schnittstelle für externe Bedienelemente wie zum Beispiel Handräder, Joystick, Buttons etc.

8.1 Externes Funk Panel

Dieses Bedienpanel ist kabellos. Weitere Informationen gibt es auf dieser Seite: Wireless Control.

Abbildung 22: Funk Panel

8.2 Extension Panel (Anschluss per Kabel)

Das Extension-Panel wird per Kabel angeschlossen und bietet eine Menge Möglichkeiten für externe Bedienelemente. Hierzu habe ich eine Seite mit weiteren Informationen erstellt:

Extension Panel
Bild von *Carsten Schröder*

Abbildung 23: Panel mit Kabel Anschluss

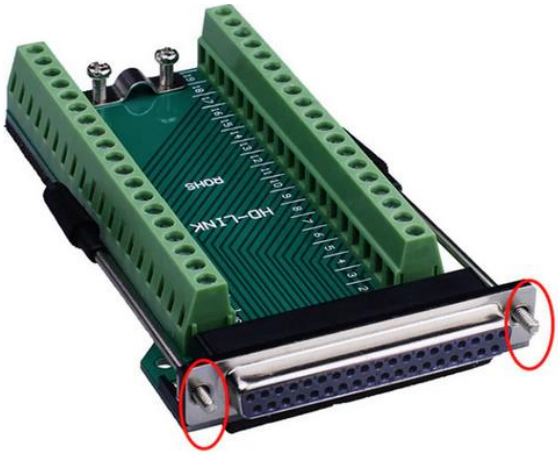
8.3 Handrad

Dieses Handrad ist schön klein und bietet die wichtigsten Bedienelemente. Es wird per Kabel angeschlossen. Hierzu gibt es einen Beitrag mit einer Zusammenbau-Anleitung: DIY Handrad unter 70€

Abbildung 24: Handrad

8.4 Breakout Board / Eigene Entwicklung

Es soll lieber etwas Eigenes sein? Das ist auch kein Problem. Dafür kann man sich so einen Adapter bestellen und anschließen:



SUB-D 37 Breakout Board

Abbildung 25: Breakout Bord

Dort lassen sich dann alle Bedienelemente anschließen und man kann z.B. der Anleitung auf der Estlcam Homepage folgen: Estlcam Handrad – Dazu einfach nach dem unten Abgebildetem Pin out anschließen.

8.5 SUB-D 37 Pinout

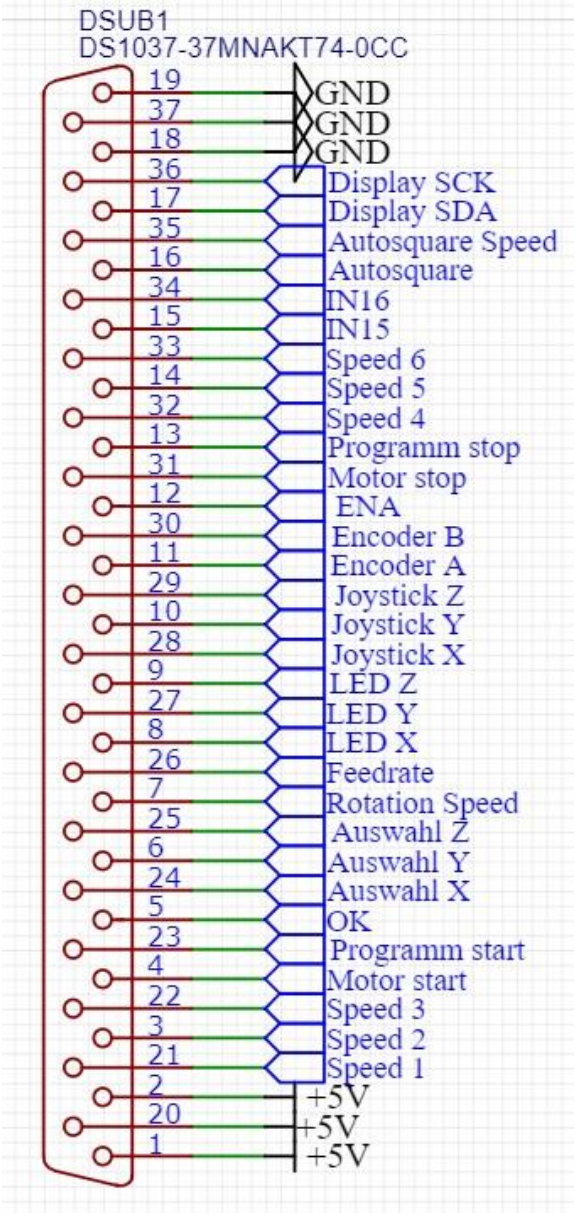


Abbildung 26: SUB-D Anschluss

9 Lüfteranschluss

Es ist möglich an dem 4Pin Lüfter Anschluss auf dem Board auch einen 2Pin Lüfter oder einen 3Pin Lüfter mit 12V anzuschließen. Dabei bitte auf die richtige Verpolung von GND und +12V achten. Diese sind auf der Platine gekennzeichnet.

> Die Steuerung eines 4Pin Lüfters mit Drehzahlsteuerung funktioniert nur mit dem zweiten Arduino Mega! Ohne den zweiten Arduino dreht ein PWM-Lüfter vermutlich gar nicht, oder nur sehr langsam.

10 Zweiter Arduino Mega

Der zweite Arduino Mega mini pro ist optional und kann eingesetzt werden, um weitere Funktionalitäten hinzuzufügen. Dazu gehört Autosquaring, temperaturabhängige Lüfter Steuerung, Ansteuerung eines Displays usw.

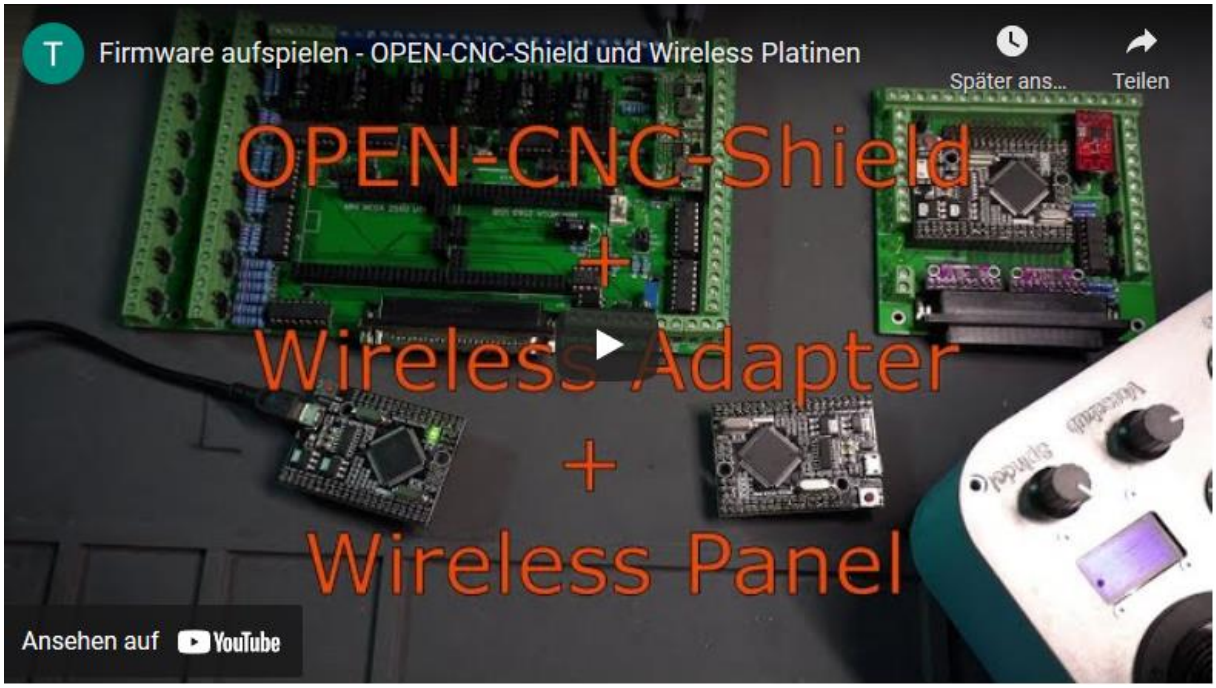
10.1 Software aufspielen / konfigurieren

Die Software für den Arduino liegt im Gitlab – Timos Werkstatt – Repository: open-cnc-shieldSketch

Mögliche Display Layouts (hier unbedingt darauf achten, dass der Displaytyp übereinstimmt):

```
Layout_128X32_Fan_and_Temp,
Layout_128X32_Temp_and_Autosquare,
Layout_128X32_Autosquare,
Layout_128X64_1
```

10.2 Erklärvideo



Film 3: Firmware aufspielen

10.3 Autosquaring

Autosquaring der Achsen ist vor allem bei DIY Fräse nützlich und soll für die Rechtwinkligkeit der Achsen zueinander sorgen. Gerade bei der MPCNC oder Artisan, welche größtenteils aus dem 3D-Drucker kommen kann es Probleme bei der Rechtwinkligkeit geben, welche hierdurch hoffentlich behoben werden. Beim Autosquaring werden alle Motoren, für welche Autosquaring aktiviert ist gegen ihre jeweiligen Endstops gefahren. Damit die Achsen dann im rechten Winkel sind, müssen die Endstops entsprechend ausgerichtet werden.

> Achtung, um Autosquaring zu nutzen, muss ein Taster / Schalter an den passenden Pin des SUB-D 37 Anschlusses angeschlossen werden. Nutzt man das Extension-Panel oder Funk Panel sind hierfür bereits Anschlüsse vorgesehen und herausgeführt.

In diesem Video zeige ich wie man Autosquaring nachrüstet bzw. nutzt und was das ist:

Film 4: MPCNC Autosquare nachrüsten

10.3.1 Autosquare Konfiguration

Das Autosquaring wird über Jumper konfiguriert und aktiviert.

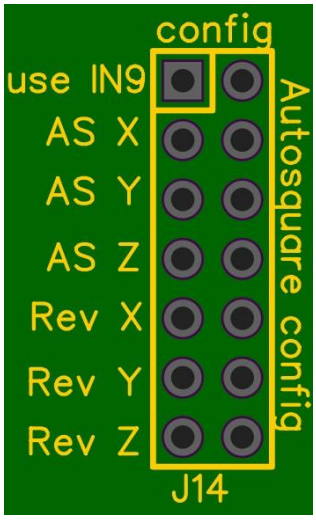


Abbildung 27: Autosquare

Hier kann für jede Achse einzeln festgelegt werden, ob diese aktiv ist oder nicht. Ebenso kann die Richtung geändert werden. Sollten die Motoren also in die falsche Richtung nach dem Start des Autosquaring laufen sorgt ein Jumper bei z.B. „Rev X“ dafür, dass die Motoren der x-Achse in die andere Richtung laufen. **Diese Einstellung gilt nur für den Autosquaring Modus!**

Wenn der Jumper bei „use IN9“ gesetzt ist, wird beim Starten von Autosquaring der Eingang 9 ausgelöst und kann in Estlcam zum Beispiel als Endstopp konfiguriert werden. Damit denkt Estlcam, dass ein Endstopp ausgelöst wurde und versucht nicht selbst die Fräse zu verfahren.

Autosquaring für eine Achse ergibt nur Sinn, wenn es auch zwei Motoren für diese Achse gibt.

10.3.2 Endstopps anschließen

Die Endstopps müssen an den richtigen Eingängen angeschlossen sein, damit Autosquaring funktioniert. Hier ist eine Anschlusstabelle:

Motor	Endstopps
X1	Eingang 5
X2	Eingang 6
Y1	Eingang 7
Y2	Eingang 8
Z1	Eingang 3
Z2	Eingang 4

Tabelle 5: Endstopps

Es wird im Standard von Programm von NO (= normally open) Endstopps ausgegangen. Falls die Endstopps als NC angeschlossen werden, muss dies im Arduino Code geändert werden. Siehe dazu folgende Zeile am Anfang des Arduino Codes:

```
// Use HIGH for normally open endstops and LOW for normally closed endstops
const uint8_t endstopStateTriggered = HIGH;
```

10.4 Lüfter Steuerung mit PWM

Auf der Platine ist ein 4Pin Lüfter Anschluss. Hier können alle möglichen Arten von 12V Lüfter angeschlossen werden. Darunter ein 2Pin Lüfter (nur mit GND und +12V) ein 3Pin Lüfter oder ein 4Pin Lüfter mit Drehzahlsteuerung.
Auf dem zweiten Arduino wird standardmäßig alle 10 Sekunden die Temperatur des Onboard Temperaturfühlers ausgelesen und die Drehzahl des Lüfters entsprechend bei Temperaturanstieg erhöht. Dies funktioniert nur mit einem 4Pin PWM Lüfter, und auch dort gibt es Unterschiede in der Frequenz. Ich habe die Platine mit folgendem Lüfter getestet:

12V PWM Lüfter:
Dieser Lüfter ist angenehm leise und funktioniert super mit der Drehzahlsteuerung.
Wobei sicherlich auch andere Funktionieren. Ich habe halt nur diesen einen getestet.

Abbildung 28: Lüfter

10.5 Externe Temperatur Sensoren

Es ist möglich weitere externe Temperatur Sensoren anzuschließen. Dafür gibt es auf der Platine einen Terminal mit der Beschriftung „Temp DS18B20“. Hier kann ein oder auch mehrere Sensoren vom Typ DS18B20 angeschlossen werden. Ich weiß gerade nicht genau wie viele, aber sicherlich mehr als man braucht. Diese gibt es unter anderem auch in einer wasserdichten Variante. Da bietet es sich doch an, auch z.B. die Kühlwassertemperatur im Auge zu behalten :-).

DS18B20:

Abbildung 29: Temperatur Sensor

Standardmäßig wird nur ein Sensor dort ausgelesen und auf dem Display des Bedienpanels ausgegeben. Möchte man mehrere Sensoren auslesen und verarbeiten oder auf irgendeine Art auf die Temperatur reagieren, muss man selbst Hand anlegen und die Funktion selbst auf dem Arduino programmieren. Beizeiten mache ich hierzu mal einen Beitrag.

11 Estlcam

Folgt…

12 LinuxCNC

Folgt…

12.1 Anschluss des Adapters

Den Estlcam Arduino abziehen und dafür einen der LinuxCNC Adapter aufstecken.

12.2 LPT / Parallel Port verbinden

Hier einmal die Pinbelegung der LPT Anschlüsse:

12.2.1 LinuxCNC Adapter v1.0

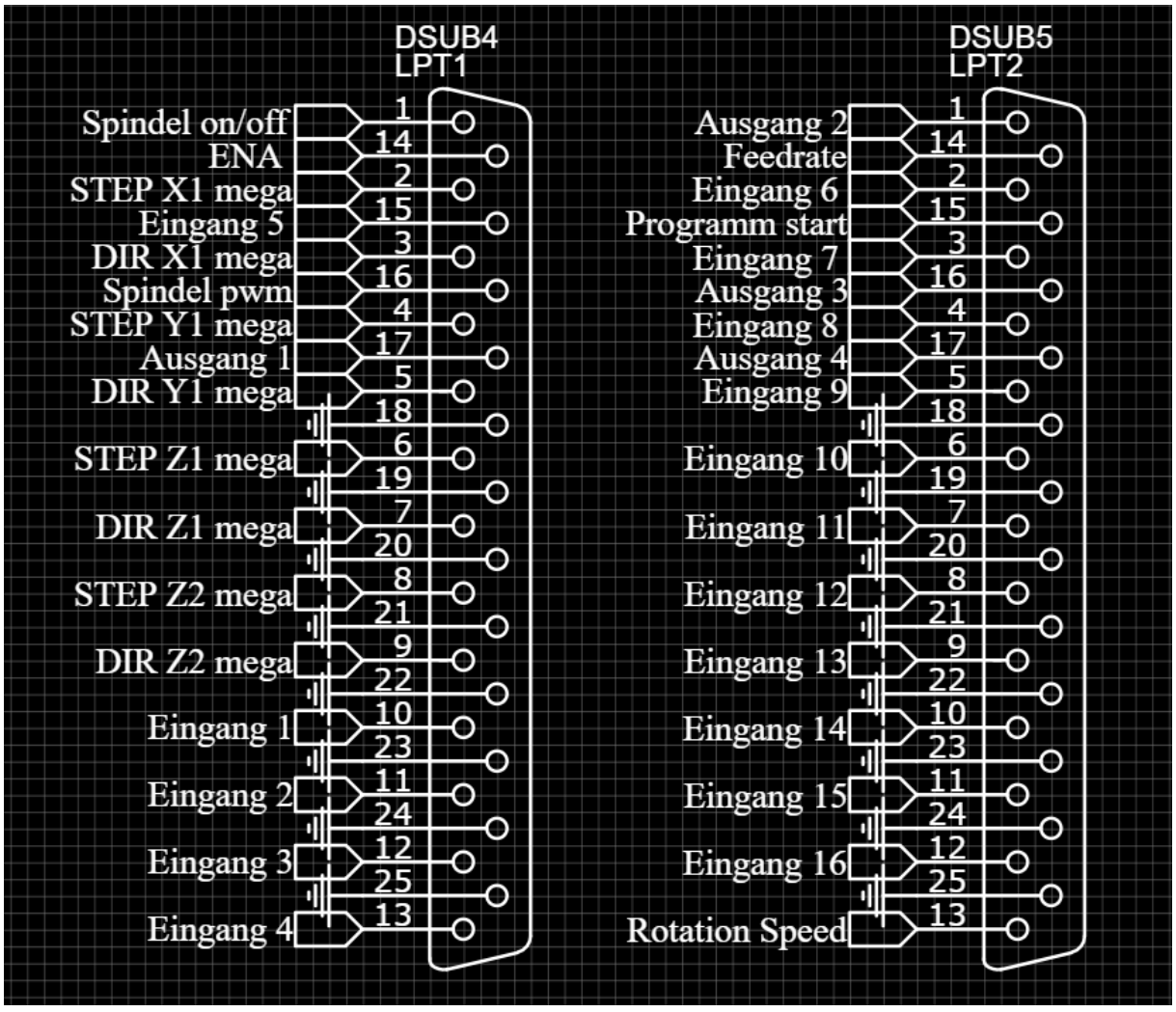


Abbildung 30: D-SUB Anschluss LinuxCNC v1.0

12.2.2 LinuxCNC Adapter 6Achs v1.0

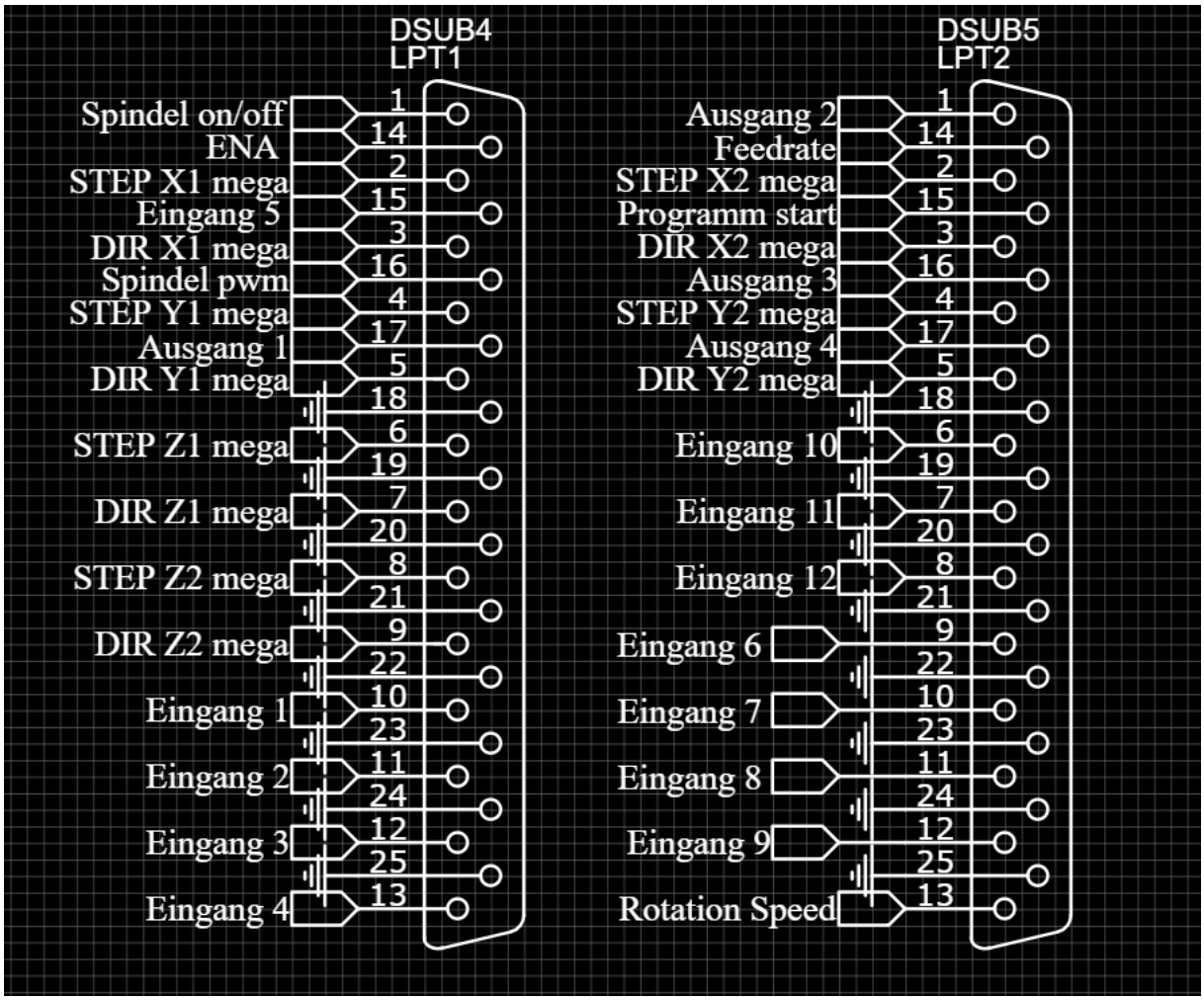


Abbildung 31: D-SUB Anschluss LinuxCNC 6-Achse v1.0

12.3 Konfiguration der CNC

Folgt

13 LaserGRBL

Hierzu habe ich einen Beitrag erstellt: LaserGRBL mit dem OPEN-CNC-Shield

14 FAQ – Häufig gestellte Fragen

14.1 Wieso zeigt Eingang 8 in Estlcam immer ausgelöst an?

Das liegt daran, dass Estlcam den Arduino Pin D13 für den Eingang 8 nutzt und dieser leider bei den Arduinos meist mit der internen LED verbunden ist. Zum Glück ist diese überflüssig und kann einfach entfernt werden. Ich mache das immer mit einem Cuttermesser und hebel die aus.
Man kann selbstverständlich auch vorsichtig Vorgehen und dies auslöten. Dann hat man die Möglichkeit diese später wieder anzulöten. In diesem kurzen Video zeige ich, wie ich das immer mache:

Film 5: Arduino Mega mini Onboard LED entfernen