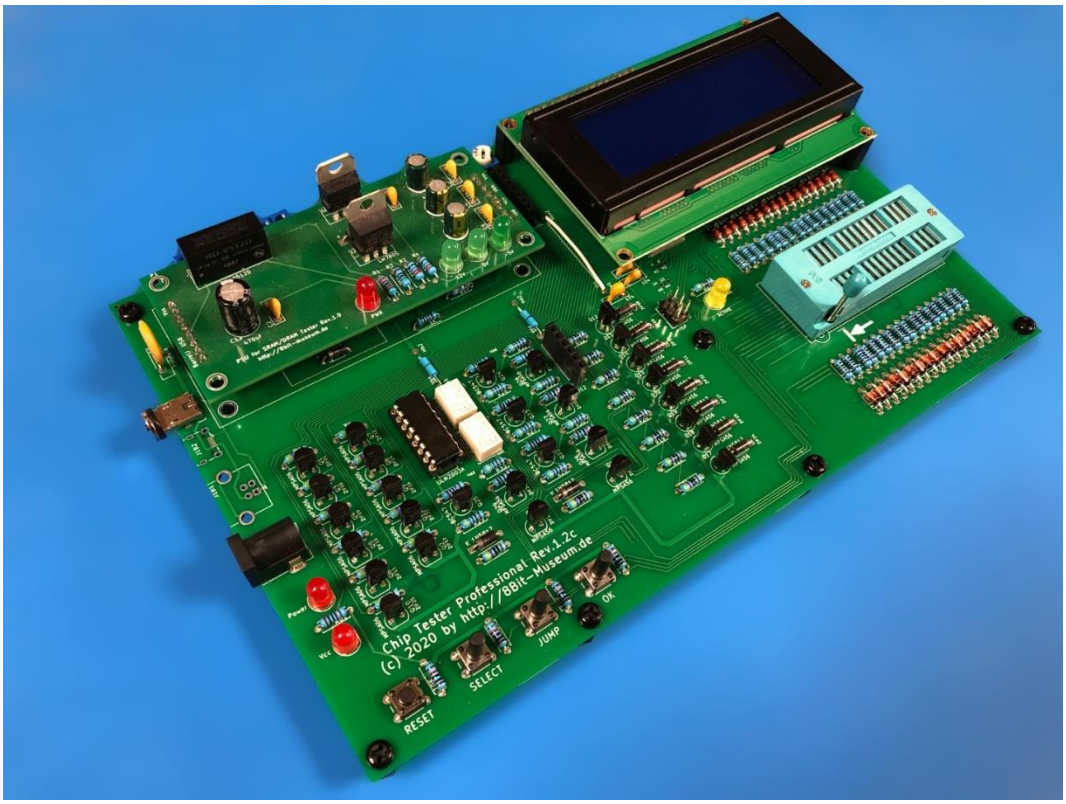
Version: 2024-09-21

Retro Chip Tester Professional

AUFBAU, PROGRAMMIERUNG UND FEHLERSUCHE



Rev.1.2

Hinweis:

Dieses Handbuch enthält bereits Erklärungen zu Features, die erst mit dem nächsten Firmware-Release verfügbar sein werden.

Diese Seite wurde absichtlich leer gelassen

Inhaltsverzeichnis

Sicherheitshinweise:

Bitte beachten sie folgende Hinweise zum Aufbau:

- Beim Löten werden Lötkolben, Lötzinn und auch Bauteile, die gelötet werden, heiß.
- Da beim Löten giftige Dämpfe entstehen, sorgen sie für eine gute Belüftung.
- Beachten sie die Warnhinweise in der Anleitung ihrer Lötstation bzw. Lötkolbens.
- Achten Sie beim Zusammenbau und Verschrauben der Einzelteile auf scharfe Kanten an Werkzeugen, Schrauben und Bauteilen.
- Der Aufbau des RCT mit seinen Erweiterungen ist sehr komplex und nicht für Kinder geeignet.

Bitte beachten sie folgende Hinweise zum Betrieb:

- Lassen Sie das Gerät im Betriebszustand niemals unbeaufsichtigt.
- Wird das Gerät nicht verwendet, lassen sie es nicht an der Spannungsversorgung angeschlossen.
- Wenn eine ungewöhnliche Geruchsentwicklung oder Hitzeentwicklung am Gerät festzustellen ist, schalten sie das Gerät sofort aus.
- Die Linearregler (7805, 7905) können je nach Belastung bei längerem Betrieb heiß werden.
- Stellen Sie sicher, dass das Gerät an einem Ort außerhalb der Reichweite von direkten Wärmequellen, offenen Flammen und brennbaren Materialien und Flüssigkeiten aufgestellt wird. Das Gerät ist für den Betrieb bei Raumtemperatur vorgesehen.
- Das Gerät ist für den USB-Betrieb und einer stabilisierten Versorgungsspannung von maximal 9V vorgesehen.

Der Speichertester wurde bereits mehrfach aufgebaut und funktioniert beim Entwickler. Das heißt aber nicht unbedingt, dass das automatisch auch bei einem Nachbau der Fall ist. Das Dokument soll helfen den Aufbau und die Funktionswiese zu erklären und gibt Hinweise, wie der Speichertester nachgebaut werden kann. Es werden oft zusätzliche, technische Informationen gegeben. Nicht alles ist für das Verständnis unbedingt notwendig. Bei Problemen können diese aber hilfreich sein.

Wer den Tester nachbaut, der ist selbst für die richtige Auswahl der Bauteile und den Aufbau verantwortlich. Eine Garantie für eine korrekte Funktion wird nicht gegeben. Es wird keine Haftung dafür übernommen, wenn beteiligte Geräte durch den Einsatz des Speichertesters Schaden nehmen. In dem Zusammenhang bitte unbedingt auch die Warnhinweise im Anwenderhandbuch beachten.

Einige Abschnitte sind mit einem „DANGER“ gekennzeichnet. Das heißt jetzt nicht wirklich, dass der Inhalt „gefährlich“ ist, sondern, dass hier der Tester beschädigt werden kann.

Warnhinweise sind entsprechend gekennzeichnet.

Diese Seite wurde absichtlich leer gelassen

Unbedingt benötigtes Werkzeug und Materialien

Um den Speichertester aufbauen zu können, wird benötigt:

- Lötkolben / Lötstation
- Lötzinn und geeignetes Flussmittel
- Dünne Entlötlitze
- Ein 6 pol. ISP-Programmierer (z.B. „Diamex USB-ISP-Programmer für Atmel AVR, Rev.2“ oder „Diamex USB-ISP-Programmer für AVR, STM32, LPC-Cortex (Prog-S2)“) zum Programmieren der Firmware. Kosten: unter 20 EUR. Der bekannte USBASP wird nicht empfohlen!

Sollte der ATmega2560 nicht bereits vormontiert sein, gibt es zum Löten des Chips im TQFN-100 Gehäuse einige gute „Lernvideos“:

- https://www.youtube.com/watch?v=5uiroWBkdFY
- https://www.youtube.com/watch?v=nyele3CIs-U
- https://www.youtube.com/watch?v=YUryJOAiPa4
- https://www.youtube.com/watch?v=IlPAJLaG1BQ
- https://www.youtube.com/watch?v=BvhE16vBfX4
- https://www.youtube.com/watch?v=Cww1ZGKClXw

Urheberrecht und Gewährleistung

Die RCT-Leiterplatte (bestückt mit dem ATmega2560 oder unbestückt ohne Bauteile („Blank Board“)), die Firmware und alle zusätzlichen Leiterplatten (z. B. Adapterboards) werden im Folgenden als „Produkte“ bezeichnet.

Die Verwendung der Hard- und Software geschieht auf eigene Gefahr. Es wird keine Haftung für etwaige Schäden übernommen, die beim Einsatz der vorliegenden Soft- und Hardware entstehen könnten. Es wird keine Gewährleistung für die korrekte Funktionsfähigkeit der bereitgestellten Funktionen übernommen. Es besteht auch kein Anspruch auf eine Fehlerbehebung, auch wenn der Entwickler bemüht ist Fehler schnellstmöglich zu beseitigen.

Wenn die Produkte außerhalb der EU gekauft werden, gilt diese Gewährleistung für ein Jahr ab Verkaufsdatum. Wenn die Produkte in der EU gekauft werden, gilt die Gewährleistung zwei Jahre ab Verkaufsdatum. Alle Komponenten, die vom Benutzer dazugekauft werden müssen, um ein funktionierendes RCT zu bauen, sind nicht von dieser Gewährleistung abgedeckt. Ebenso wird keine Gewährleistung für ein fertig montiertes RCT übernommen, mit Ausnahme der gelieferten Leiterplatten.

Der Entwickler haftet nicht für Mängel, die durch Nichteinhaltung von Vorgaben, Missbrauch oder falsche Bedienung durch den Benutzer verursacht wurden, einschließlich unsachgemäßer Installation oder Prüfungen oder wenn das Produkt in irgendeiner Weise geändert oder modifiziert wurde. Diese eingeschränkte Gewährleistung ist das einzige und ausschließliche Rechtsmittel des Endbenutzers gegenüber dem Entwickler, sofern gesetzlich zulässig. Die Produkte werden „wie besehen“ und „mit allen Fehlern“ bereitgestellt. Der Entwickler lehnt alle anderen ausdrücklichen oder stillschweigenden Gewährleistungen der Marktgängigkeit für einen bestimmten Zweck ab.

Die Produkte wurden entwickelt, um unterstützte ICs in alten Heimcomputern, Synthesizern, Arcade- und Flipperautomaten zu testen. Es ist nur die nicht-kommerzielle Nutzung im privaten Kontext vorgesehen. Die Produkte sind nicht für die Verwendung an sicherheitskritischen Stellen zugelassen, in denen bereits erwartet werden kann, dass ein Versagen des Produkts schwere Verletzungen oder den Tod in irgendeiner Form verursacht. Der Benutzer erkennt an und stimmt zu, dass jede andere Verwendung dieser Produkte ausschließlich auf Risiko des Benutzers erfolgt und dass der Benutzer allein für die Einhaltung aller gesetzlichen und behördlichen Anforderungen in Verbindung mit einer solchen Verwendung verantwortlich ist.

Verzicht auf Folgeschäden. In keinem Fall haftet der Entwickler gegenüber dem Benutzer oder Dritten für besondere, begleitende, indirekte, strafbare, zufällige Folgeschäden oder Schadensersatz im Zusammenhang mit oder aus den hier bereitgestellten Produkten, unabhängig davon, ob der Entwickler auf die Möglichkeit solcher Schäden hingewiesen hatte oder nicht. Dieser Abschnitt gilt auch nach Ablauf der Gewährleistungsfrist.

1 Allgemeines zum Retro Chip Tester Professional

Wer den Tester aufbaut, sollte sich zuerst einen Überblick über die inzwischen recht umfangreiche Dokumentation verschaffen.

DE-1 – Aufbau, Programmierung und Fehlersuche

Das Dokument beschreibt Aufbau, Programmierung und Fehlersuche.

Hierzu gibt es weitere Dateien:

- Ordner „Programming Step 1 - Set Fuses“
- Ordner „Programming Step 2 - Flash Firmware“

Es ist alles vorhanden, um den ATmega zu programmieren. Es ist i.d.R. nicht notwendig weitere Software auf dem Internet zu laden.

DE-2 – Anwenderhandbuch

Das Dokument beschreibt die Anwendung des RCT.

Es befinden sich weiterhin

- CRC32-Listen (zur Identifizierung von ROMs),
- Eigene IC-Definitionen (zur Definition sehr spezieller ICs),
- Vergleichslisten (damit die richtigen Testeinstellungen verwendet werden)

in den Unterordnern, in denen dieses Dokument gespeichert ist.

DE-3 – Stückliste (BOM)

Das Dokument (nicht öffentlich verfügbar) enthält eine Übersicht der benötigten Bauteile und Links auf vorbereitete Warenkörbe. Im selben Verzeichnis befinden sich auch Bauteillisten für Reichelt Elektronik und Digikey. Weiterhin sind dort interaktive Bestückungslayouts vorhanden.

2 Bestückung der Platine

Die Stücklisten der einzelnen Versionen der Platinen befinden sich in einem separaten Dokument. Die Bauteile sind alle beschriftet:
z.B. „104“ = 100nF, „224“ = 220nF, „22“ = 22pF, „4K7“ = 4700 Ohm, „4.7“ = 4.7 Ohm.

Je nach Version der Platine kann die Bestückung leicht unterschiedlich sein.

Nehmen sie sich Zeit die Platine zu bestücken!

Planen sie mindestens 4 bis 6 Stunden ein!

Wenn sie die Platine bestücken, sorgen sie für eine gute Belüftung

bzw. verwenden sie einen Absauger für Lötdämpfe.

Tragen sie bitte eine Schutzbrille, wenn sie Drahtenden abschneiden.

Der folgende Schritt entfällt, wenn sie eine vorbestückte Platine besitzen:

Der ATmega2560 sollte als erstes bestückt werden. Eine eindeutige Empfehlung gibt es hierzu nicht: Einige verwenden einen Ofen oder Heißluftfön mit Lötpaste, andere viel Flux und eine Hohlkegellötspitze (meine bevorzugte Variante). Überflüssiges Lot und Kurzschlüsse werden mit Flux und SMD-Entlötlitze beseitigt. Unbedingt alle Pins „durchklingeln“, ob ggf. ein Kurzschluss vorliegt.

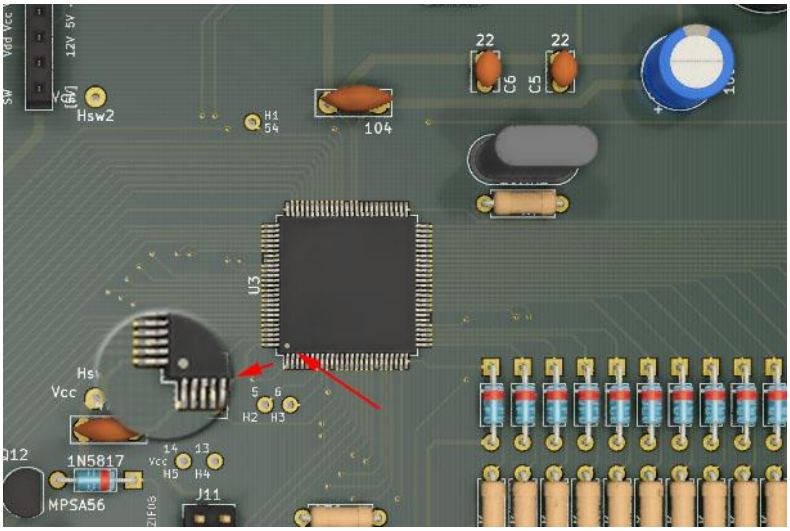


Abbildung 2.1: Ausrichtung des ATmega2560

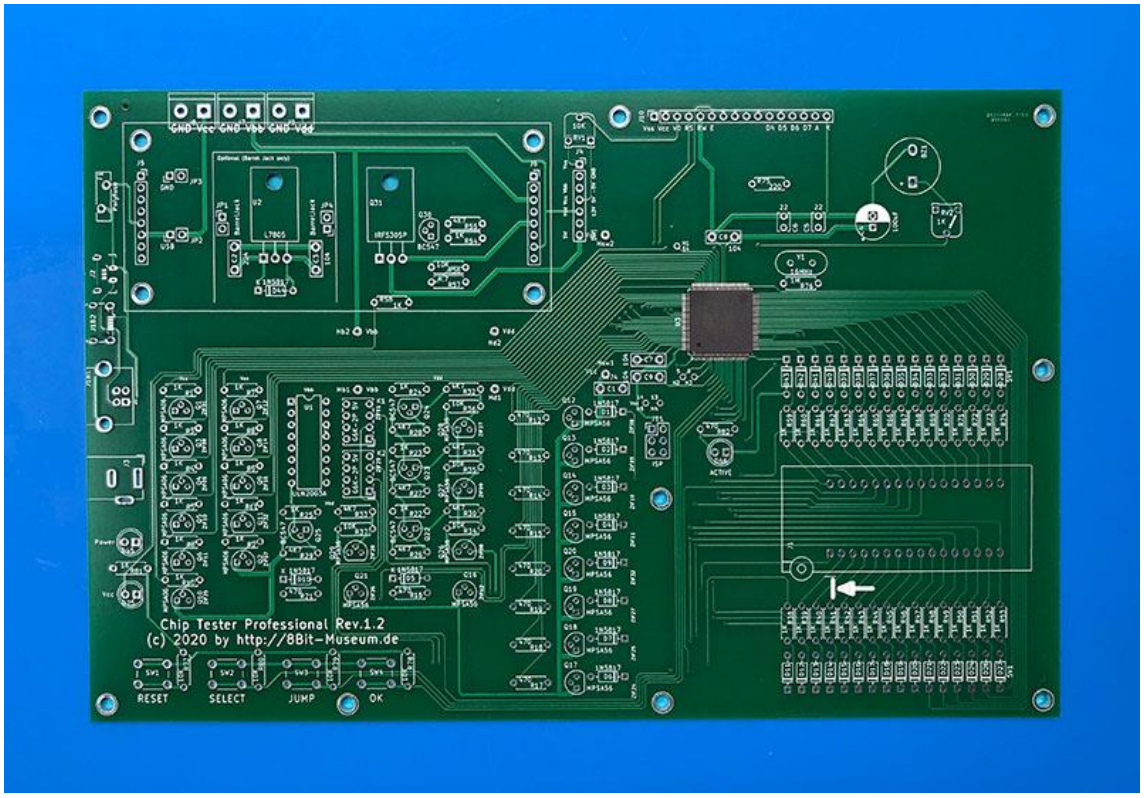


Abbildung 2.2: Bestückung des ATmega2560

Danach sollten alle Widerstände, die USB-Buchse, die Dioden und Z-Dioden bestückt werden.

Nicht versuchen alle Widerstände bzw. Dioden auf einmal zu bestücken, da diese sehr dicht zusammen liegen und man noch Platz für den Lötkolben braucht. Am besten drei oder vier Durchgänge einplanen.

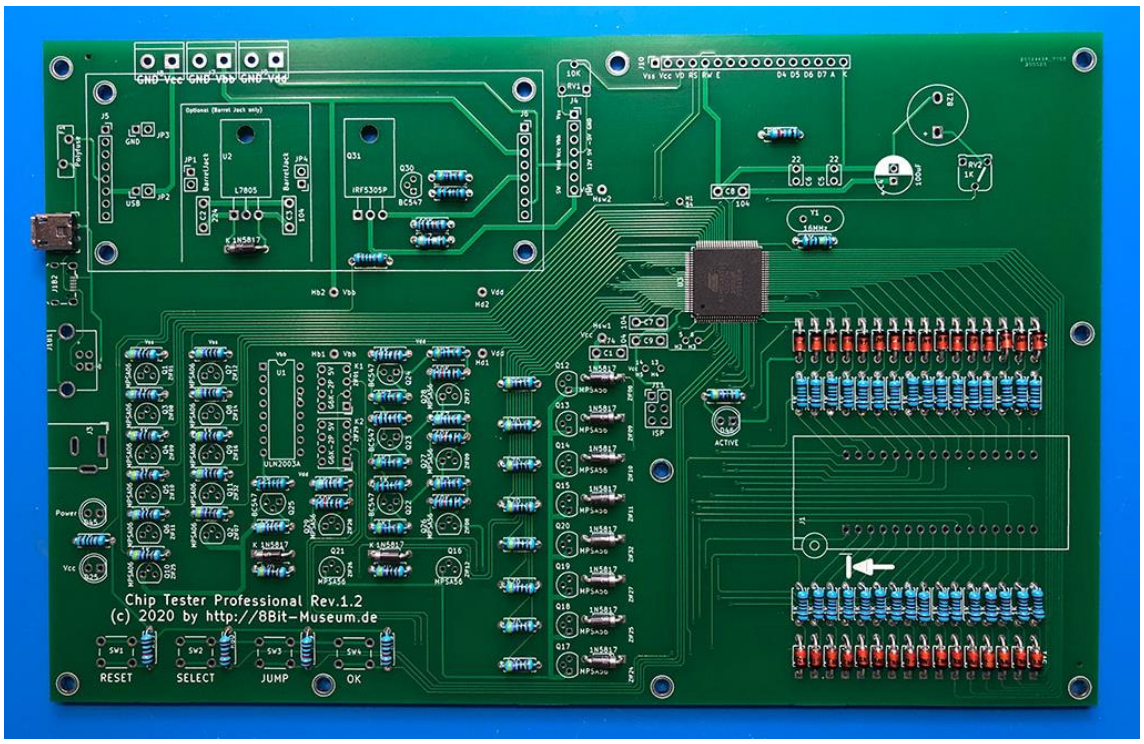


Abbildung 2.3: Bestückung der Zener-Dioden und Widerstände

Als nächstes folgen Spannungsregler, MOSFET, die zwei Relais, der 16-polige Sockel und die drei Drahtbrücken. Achtung: Rechts neben dem MOSFET (R57) wird ein 4.7 Ohm Widerstand bestückt, kein 4.7k Ohm Widerstand!

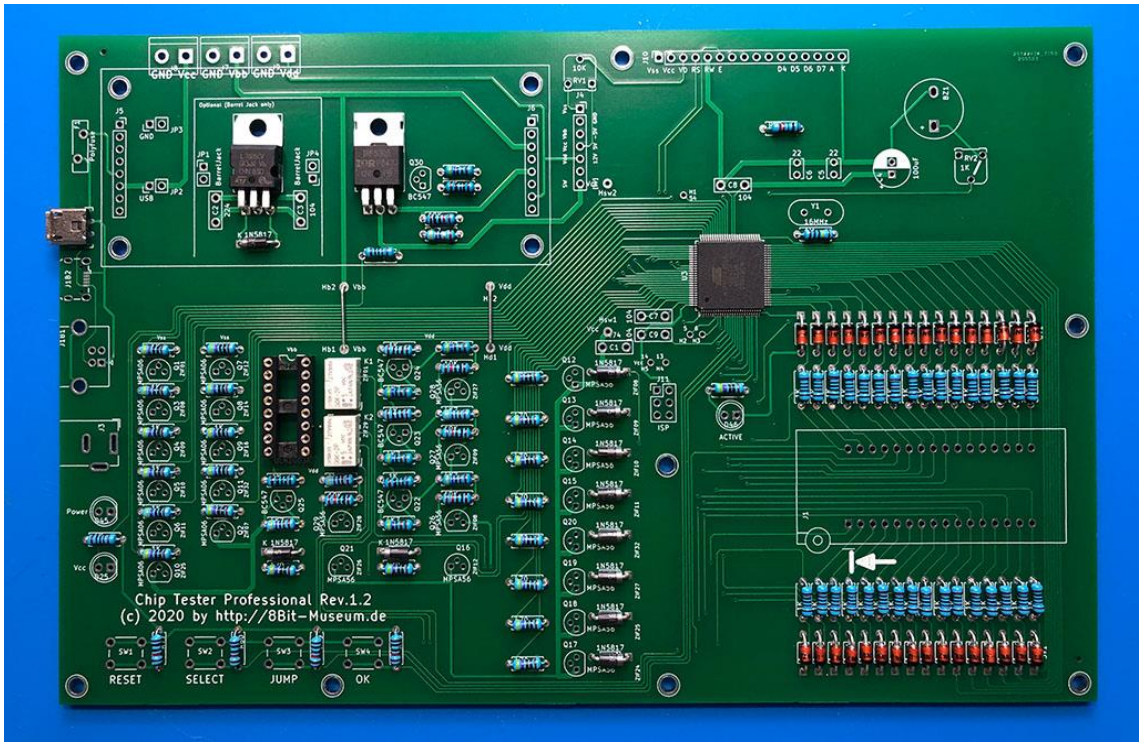


Abbildung 2.4: Bestückung der Relais und Spannungsregler

Bitte auf die Ausrichtung des Sockels und der Relais achten:

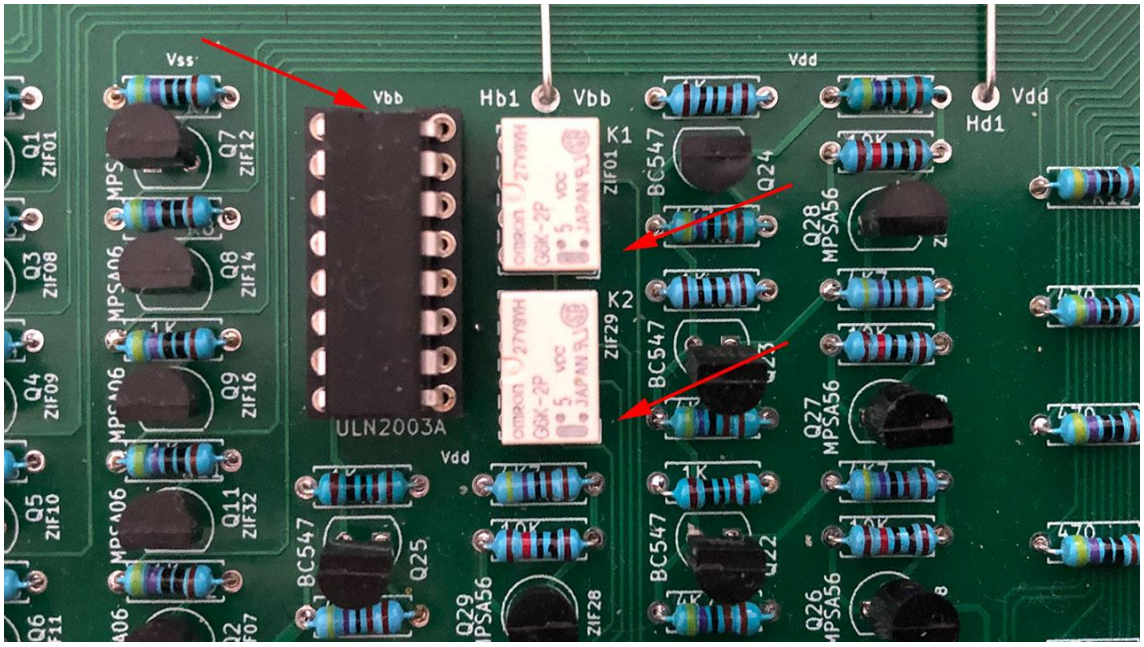


Abbildung 2.5: Bestückung der Relais und des ICs im Detail

Die Drahtbrücken noch einmal im Detail:

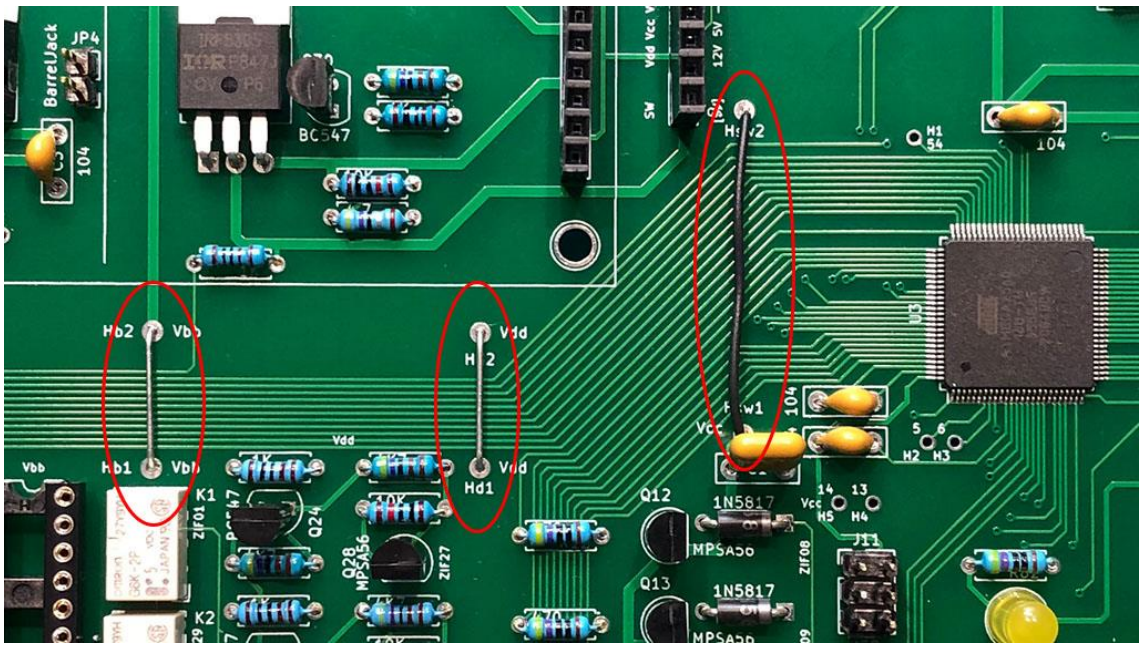


Abbildung 2.6: Bestückung der Drahtbrücken

Jetzt werden die Transistoren bestückt. Bitte nicht zu viel Lötzinn verwenden, die Anschlüsse liegen sehr dicht beieinander. Bitte unbedingt beim Abschneiden aufpassen, dass keine Kurzschlüsse entstehen!

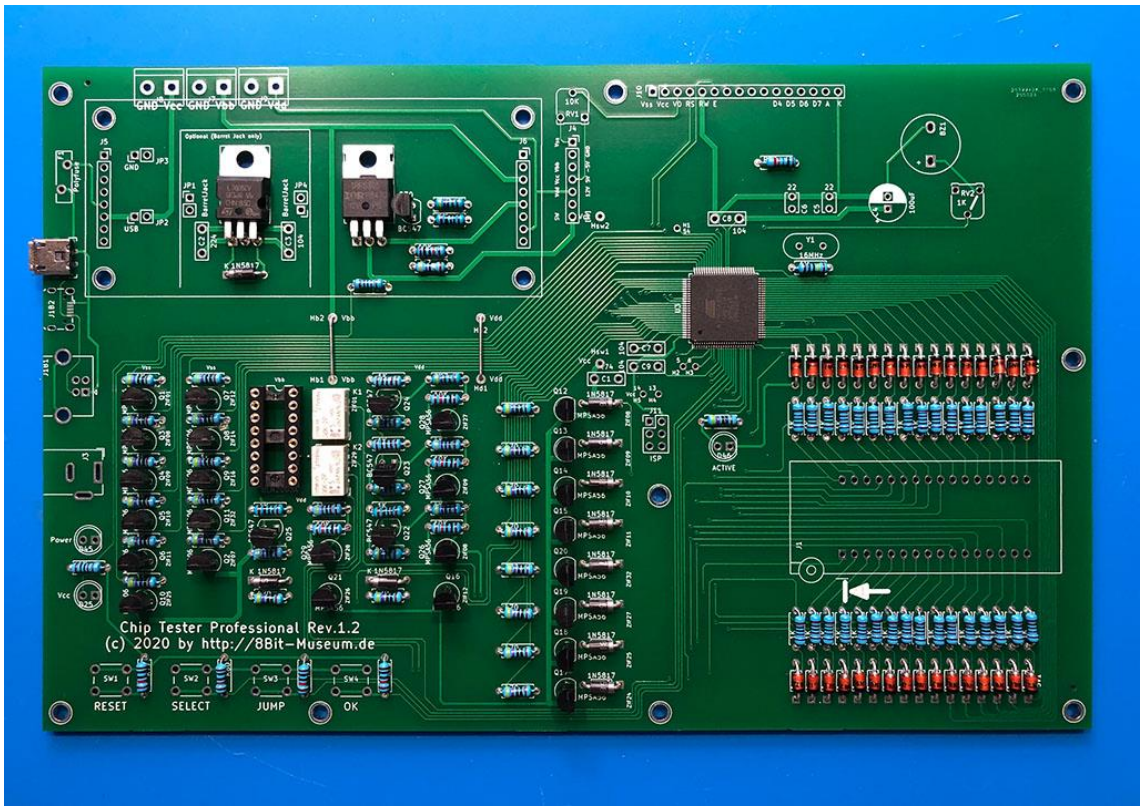


Abbildung 2.7: Bestückung der Transistoren

Zum Schluss werden die übrigen Bauelemente bestückt. Als Reihenfolge bietet sich folgende an: LEDs, Pfostenleisten und Pinheader, Potis, Elko (horizontal einbauen) und Kondensatoren, Polyfuse, Quarz, Abstandshalter, der Piezo-Lautsprecher, Tastschalter und der ZIF-Sockel. Optional können noch die Terminal-Blöcke, zusätzliche USB-Header und die Hohlstecker-Buchse bestückt werden.

Wird ein Wannenstecker verwendet, so ist auf die Orientierung zu achten:

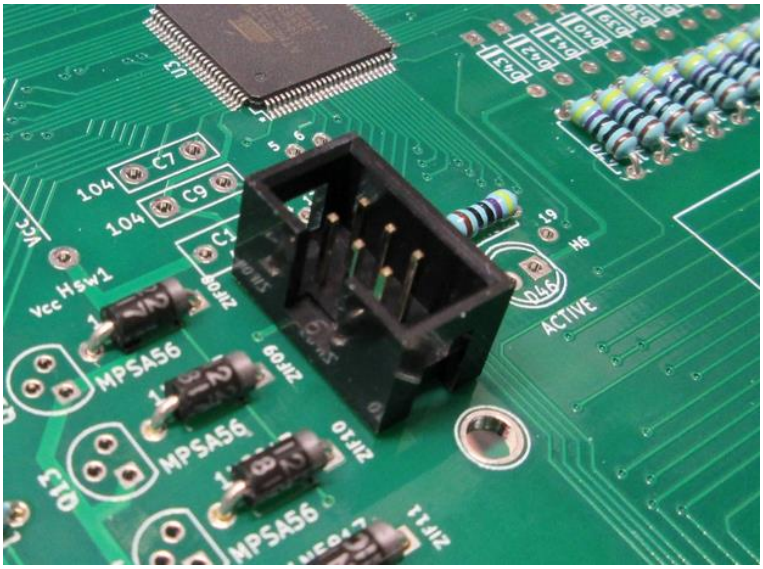


Abbildung 2.8: Wannenstecker (Orientierung)

Die bestückte Platine sieht wie folgt aus:

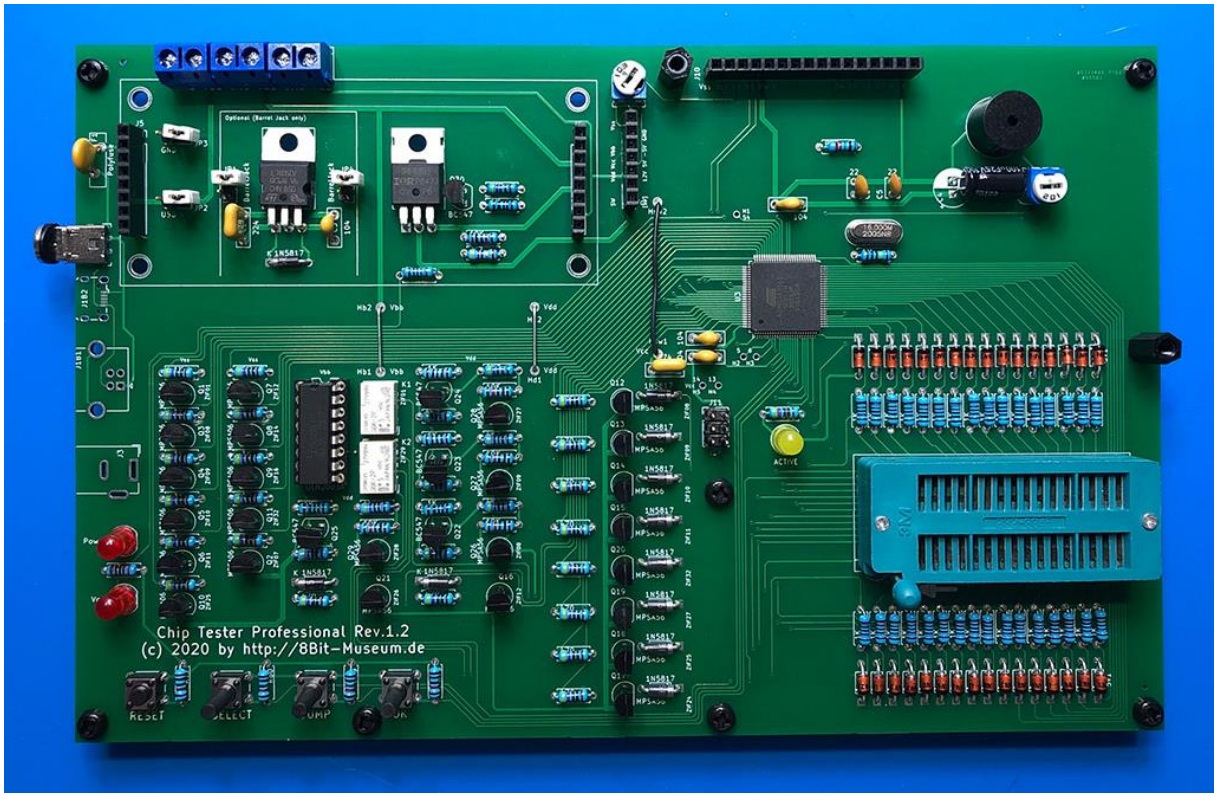


Abbildung 2.9: Fertig bestückte Platine

Die vollständig aufgebaute Platine mit Display und DC/DC-Modul sieht wie folgt aus:

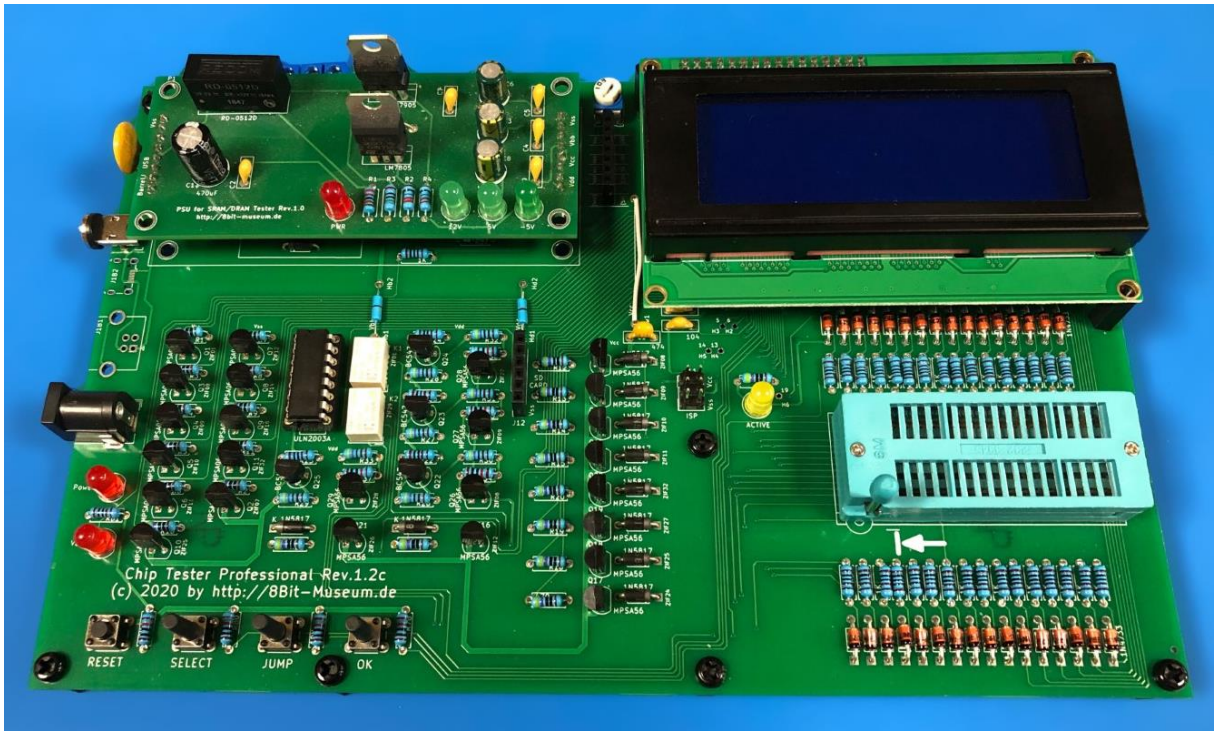


Abbildung 2.10: Vollständig aufgebaute Platine

Je nach Version der Platine kann die Bestückung leicht unterschiedlich sein.

Ab der Revision 1.2h befindet sich ein weiterer Footprint (Q32, SOT-23) auf der Platine. Sollte der MOSFET im TO-220 Gehäuse (Q31) nicht lieferbar sein, kann alternativ Q32 im SOT-23 Gehäuse bestückt werden.

Damit die Platine einen guten Stand und das Display einen guten Halt hat, sollten Distanzhülsen eingesetzt werden. Die folgenden Bilder zeigen, wie die Hülsen montiert werden können.

Die Distanzhülsen für die Füße sollten zwischen 5-10mm lang sein:

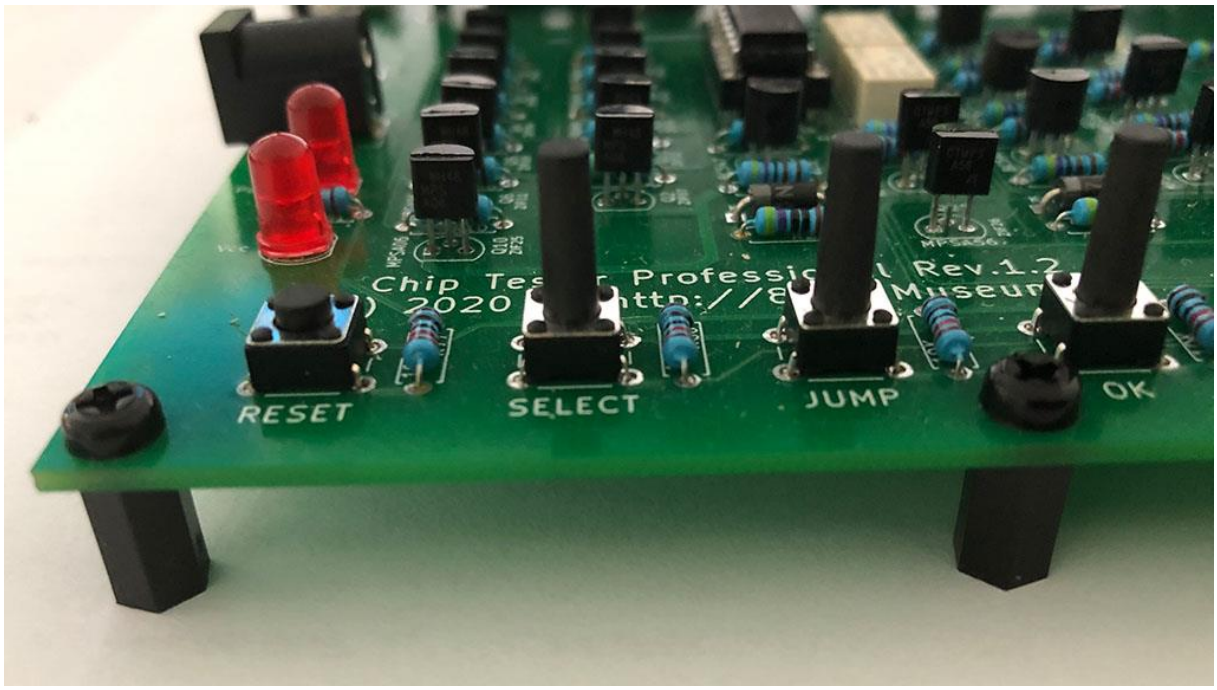


Abbildung 2.11: Montage von Standfüßen

Die Distanzhülsen für das Display sollten 11mm lang sein:

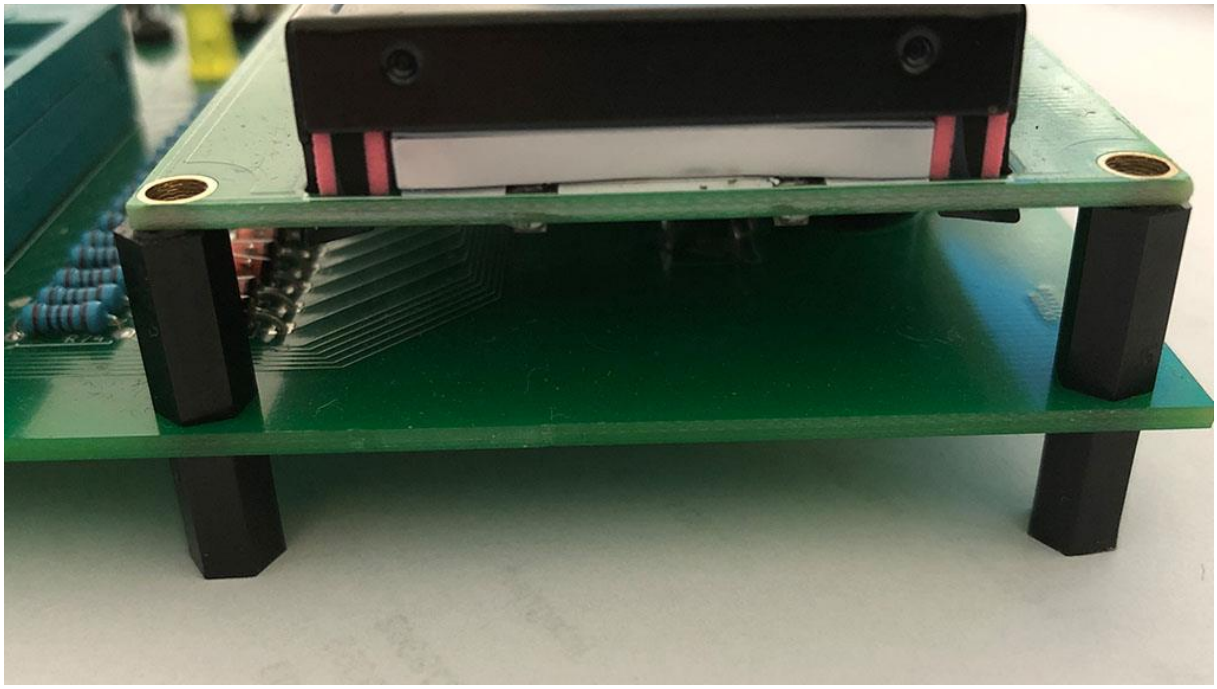


Abbildung 2.12: Montage des Displays

Die Bohrungen haben einen Durchmesser von etwas über 3mm (also passend für M3 Schrauben). Die Distanzhülsen sollten aus Kunststoff sein, damit keine Kratzer auf der Tischplatte entstehen können.

3 Spannungsversorgung

Es müssen nicht alle Bauteile bestückt werden. Die Tabelle zeigt die möglichen Bestückungsvarianten (rot = Bestückung erforderlich, grün = Bestückung kann entfallen). Natürlich kann die Platine auch komplett bestückt werden, per Jumper wird festgelegt, wie diese eingesetzt werden kann.

In den kommenden Abschnitten werden die einzelnen Möglichkeiten noch einmal detailliert gezeigt.

Zu prüfende Chips	Spannungsversorgung über	Bestückung
keine 4116/4108 (= keine -5V/12V erforderlich)	**nur USB (5V)** JP2 „USB“ gesetzt JP3 „GND“ gesetzt	DC/DC-Konverter Modul nicht erforderlich Hauptplatine („Option“) kann entfallen (1x 7805, 2x Kondensator, 1x Diode)
	Hohlstecker (7-9V) oder USB (5V) JP2 „USB“ gesetzt JP3 „GND“ gesetzt JP1/JP4 „BarrelJack“ gesetzt	DC/DC- Konverter Modul nicht erforderlich Hauptplatine („Option“) muss bestückt werden (1x 7805, 2x Kondensator, 1x Diode)
alle Chips (inkl. -5V/12V)	**Hohlstecker (6-12V) oder USB (5V)** JP1, JP2, JP3, JP4 offen	DC/DC-Konverter Modul erforderlich Hauptplatine („Option“) kann entfallen (1x 7805, 2x Kondensator, 1x Diode)
	nur Schraubterminal für 5V, 12V und -5V JP1, JP2, JP3, JP4 offen	DC/DC- Konverter Modul nicht erforderlich Hauptplatine („Option“) kann entfallen (1x 7805, 2x Kondensator, 1x Diode)

Abbildung 3.1: Übersicht der möglichen Spannungsversorgung

Wird ein DC/DC-Konverter Modul verwendet, darf kein Jumper gesetzt werden!

Wird ein Netzteil mit Hohlstecker verwendet, so sollte das Netzteil zwischen 7V bis 9V liefern. Der Betrieb mit 12V ist durchaus möglich, dann sollte der 7805 aber ggf. mit einem Kühlkörper versehen werden. Ein Betrieb über 12V wird nicht empfohlen.

Wichtiger Hinweis beim Betrieb über USB:

Die Versorgungsspannung Vcc (+5V) kommt direkt aus der USB-Versorgung. Leider sind einige USB-Netzteile sehr tolerant und liefern auch schon einmal 4.8V oder weniger. Durch den Spannungsabfall (Transistoren, Dioden, etc.) von ca. 0.3V liegt die bei 4.8V im noch akzeptablen Bereich (meistens +/- 10% Toleranz bei Vcc). Wird es weniger, kann das Auslesen von z.B. ROMs fehlschlagen. Insbesondere 6540 ROMs sind hier zickig.

Wenn ein ROM nicht ausgelesen werden kann (Symptom: mehrmaliges Auslesen liefert unterschiedliche CRC32), dann den Tester probeweise per Netzteil über den Hohlstecker betreiben. Die Linearregler (egal ob mit oder ohne DC/DC-Modul) liefert exakt 5V für Vcc.

Betrieb an einem USB-Anschluss:

Der Betrieb an einem normalen USB 2.0/USB 3.0 Port ist möglich. Allerdings findet keine Stromaushandlung, wie bei vielen anderen Geräten mit USB-Spannungsversorgung, statt. Der Tester geht davon aus, dass 500 mA zur Verfügung stehen. I.d.R. führt das zu keinen Problemen. Im „worst case“ stellt der USB-Port nur 100 mA zur Verfügung, was für den Tester zu wenig ist. In diesem Fall sollte der Tester an ein USB-Netzteil oder einen USB-Hub mit eigener Spannungsversorgung angeschlossen werden.

Warnung:

Niemals den RCT per USB und Hohlstecker gleichzeitig mit Spannung versorgen!

3.1 … über USB oder Hohlstecker mit DC/DC-Modul (empfohlen)

Mit eingesetztem DC/DC-Modul können alle Chips getestet werden. Das Modul stellt die notwendigen Versorgungsspannungen von -5V, 5V und 12V zur Verfügung. Die Spannungsversorgung kann wahlweise über USB oder den Hohlstecker (7 - 9V) erfolgen.

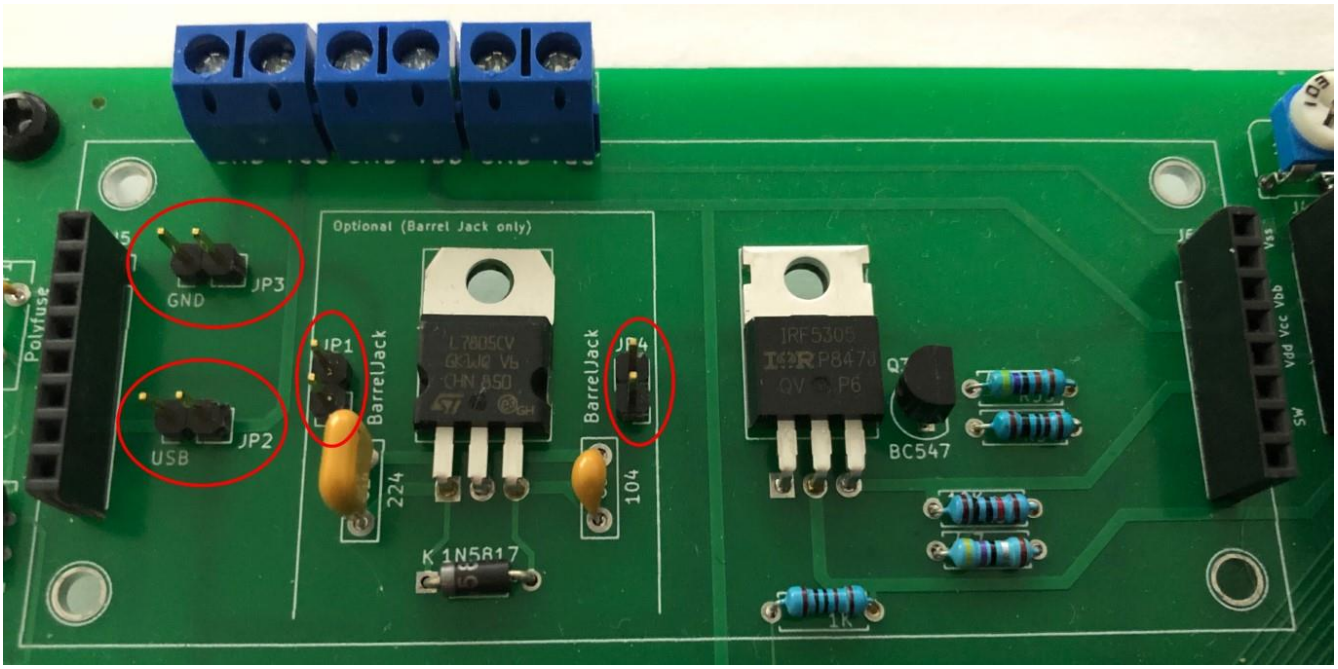


Abbildung 3.2: Alle Jumper offen

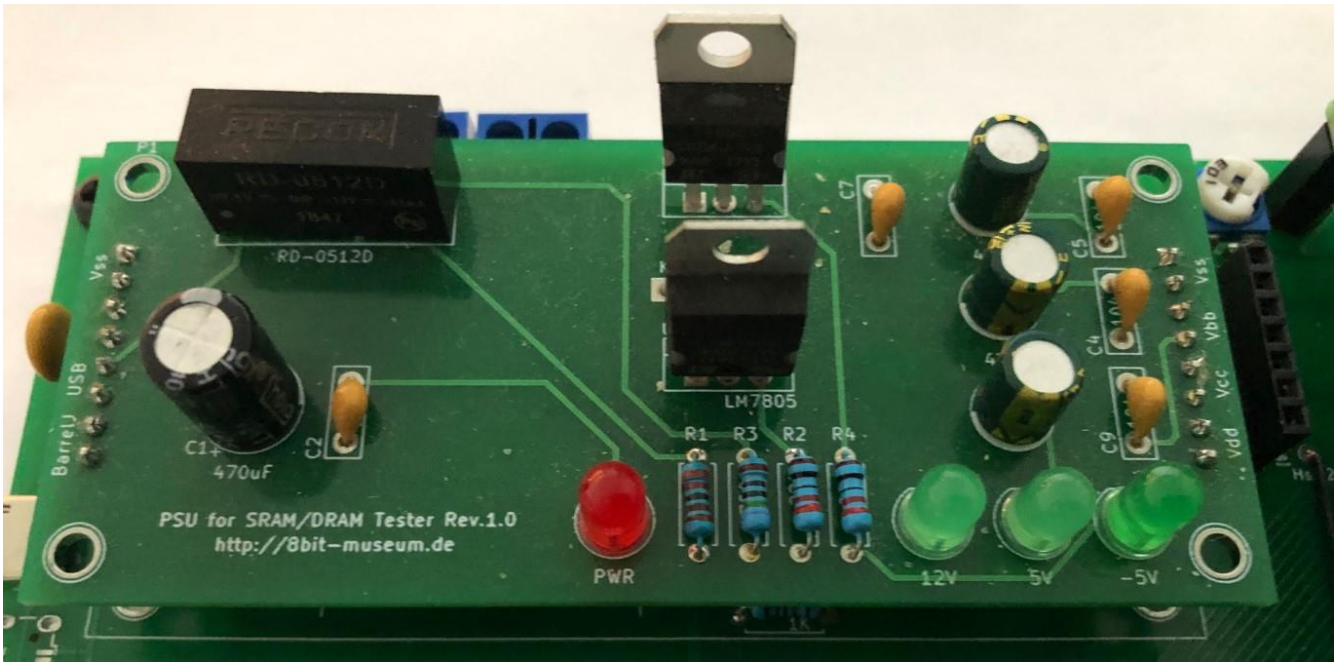


Abbildung 3.3: Gestecktes DC/DC-Modul

Wird das DC/DC-Modul zur Spannungsversorgung gewählt, so kann auf die Bestückung des Spannungsreglers auf der Hauptplatine („Optional (Barrel Jack only)“) entfallen. Es dürfen keine Jumper gesteckt werden.

3.2 … über Schraubterminals ohne DC/DC-Modul

Auch ohne gestecktes DC/DC-Modul können alle Chips getestet werden, wenn die Spannungsversorgung über die Schraubterminals erfolgt.

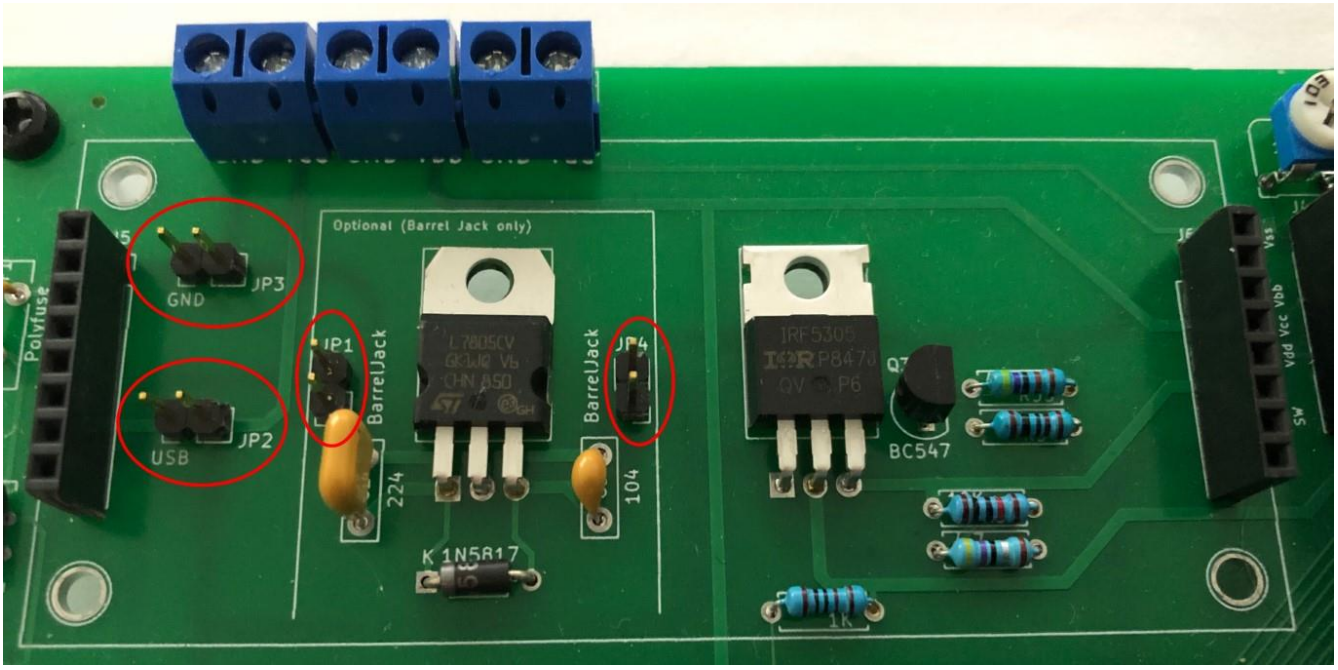


Abbildung 3.4: Alle Jumper offen

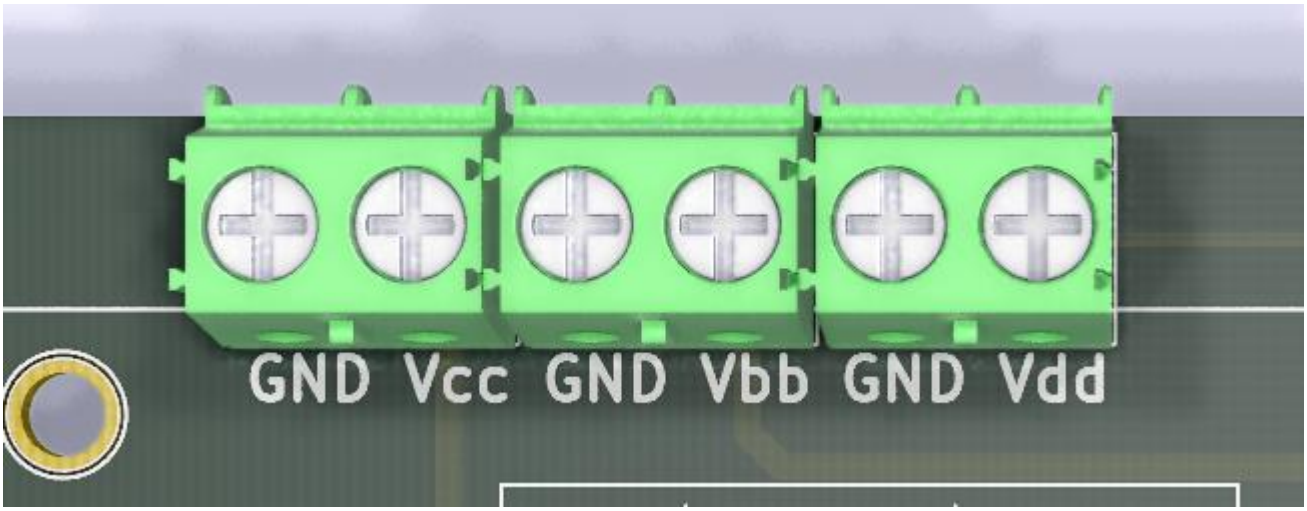


Abbildung 3.5: Belegung der Schraubterminals (Vcc = 5V, Vbb = -5V, Vdd = 12V)

Wenn die Schraubterminals verwendet werden, dürfen keine Jumper gesteckt werden. Die Spannung muss stabilisiert sein. Am besten wird ein PC-Netzteil zur Spannungsversorgung verwendet. Unbedingt auf die Polung und die korrekten Spannungen achten!

3.3 … über USB oder Hohlstecker ohne DC/DC-Modul

Ohne gestecktes DC/DC-Modul können nicht alle Chips getestet werden: Chips, die eine negative Versorgungsspannung von -5V benötigen und/oder 12V, können so nicht getestet werden.

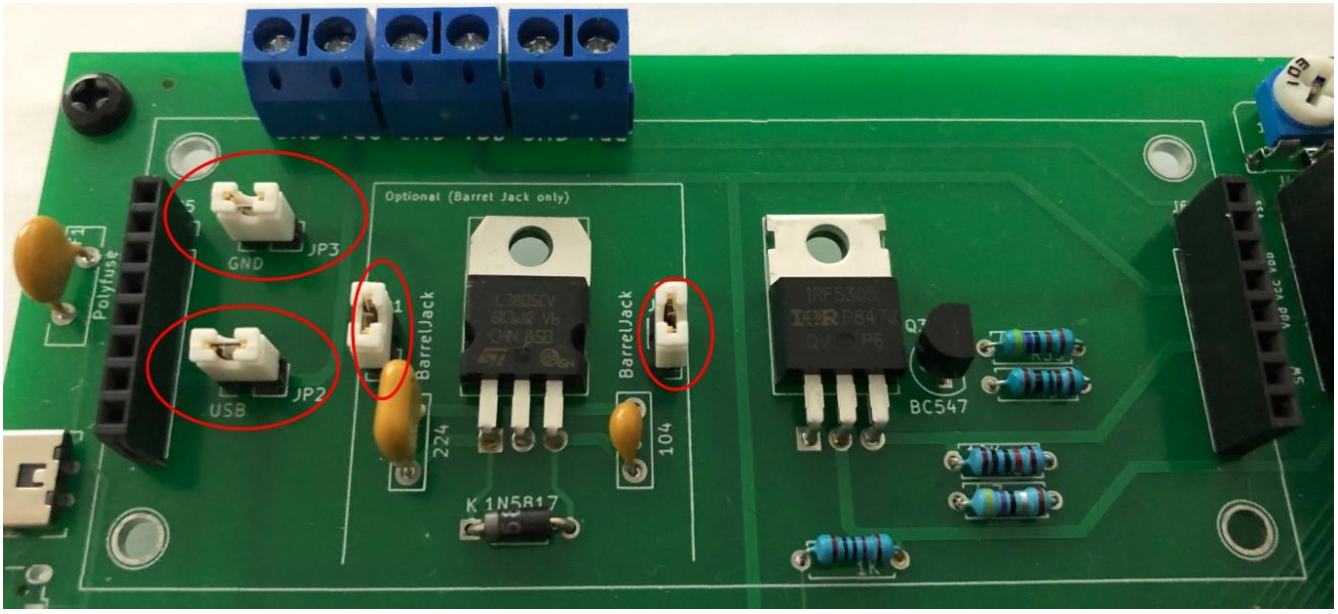


Abbildung 3.6: Alle Jumper geschlossen

Werden die Jumper JP2 und JP3 geschlossen, wird der Tester direkt über die USB-Versorgungsspannung versorgt.

Werden JP3, JP1 und JP4 geschlossen, wird die Spannung am Hohlstecker durch den Spannungsregler auf 5V reguliert.

Werden alle vier Jumper geschlossen, kann die Spannungsversorgung wahlweise per USB oder Hohlstecker erfolgen.

Der Hohlstecker besitzt positive Polarität.

Solange die Jumper gesteckt sind, darf das DC/DC-Modul nicht eingesetzt werden!

3.4 Spannungsversorgung per optionalen DC/DC-Modul

Die folgende Tabelle gibt eine Übersicht über die vorhandenen Optionen:

	Option #1 / Standard **RD-0512D**	**Option #2** **RD-0512D/LM317**	**Option #3** **RD-0512D/RS-0505S**
Versorgung per Hohlstecker	7.5V – 9V 7.5V empfohlen	7.5V – 9V	4.5V – 9V (funktioniert auch noch mit 4V)
Vcc bei USB	Vcc entspricht Eingangsspannung	Vcc entspricht Eingangsspannung	Vcc = 5V
Vcc bei Hohlstecker	Vcc = 5V	Vcc = 4.8V / 5V / 5.15V / 5.3V	Vcc = 5V
Besonderheit		Tests mit erhöhter Spannung	ab 4.5V Eingangsspannung
Kosten	günstig	günstig	teuer

Warnung:

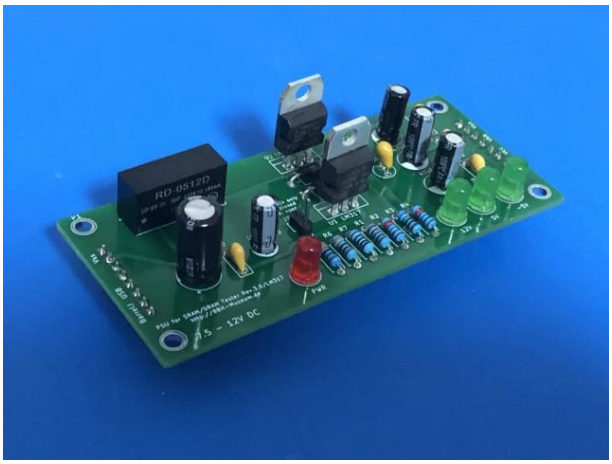
Niemals den RCT per USB und Hohlstecker gleichzeitig mit Spannung versorgen!

Option #1 (mit RD-0512D) - Standard

Diese Option verwendet einen DC/DC-Regler von RECOM. Es entsteht auch bei längerer Benutzung keine Wärme und die Ausgangsspannung ist sauber geregelt. Dieses Netzteil ist das Standardnetzteil für den Chip-Tester.

Auf der Platine kann R1 noch als „2.2k“ bezeichnet sein. Für Low-Power LEDs sind 2.2k ausreichend, der Wert für normale LEDs sollte aber 1k betragen damit die LED ausreichend hell leuchtet.

Option #2 (mit RD-0512D und LM317)



Die zweite Option verwendet wie die Option #1 einen RECOM-Regler, aber anstelle des Festspannungsreglers einen LM317 für die 5V Versorgungsspannung. Dadurch kann durch Setzen eines Jumpers die Versorgungsspannung von 5V auf bis zu 5.3V angehoben werden. Dieses kann für spezielle Tests notwendig sein, z.B.

- bei NVRAMs, die unter 4.8V in einen Read-Only Modus gehen (selten),
- bei gealterten Speicherbausteinen, deren Verhalten man bei erhöhter Spannung testen möchte.

Option #3 (mit RD-0512D und RS-0505S)

Die dritte Option verwendet zwei RECOM Regler. Falls der Chip Tester mit weniger als 5V versorgt wird, z.B. USB-Anschluss unter 5V (USB 1.0 erlaubt zwischen 4.25V und 5.25V) oder per Batterie, dann wird durch das zusätzliche Modul zunächst eine geregelte 5V Spannung erzeugt.

4 Programmieren des Testers (Fuses und Firmware)

Man benötigt einen 6-poligen ISP-Programmer (z.B. „Diamex USB ISP-Programmer für Atmel AVR, Rev.2“ oder „Diamex USB ISP-Programmer für AVR, STM32, LPC-Cortex (Prog-S2)“). Die Stromversorgung während des Programmiervorgangs läuft i.d.R. über den ISP-Programmer.

Die Programmierung erfolgt beim ersten Mal in zwei Schritten. Hierzu befinden sich in den mitgelieferten Archiven

- eine Batchdatei und AVRdude zum Setzen der Fuses, und
- eine Batchdatei und AVRdude zum Programmieren des ATmega.

Die Dateien sind teilweise vorkonfiguriert und müssen für den eigenen Programmer angepasst werden. I.d.R. wird keine weitere Software (außer dem Programmer und AVRdude) benötigt.

Die o.g. Aktionen können auf unterschiedlicher Art durchgeführt werden (bitte nach eigener Vorliebe auswählen). Der verwendete ISP-Programmer (Parameter „-c“) und ggf. der COM-Port (Parameter „-P“) müssen entsprechend der eigenen Hardware angepasst werden. Diese Parameter sind nachfolgend rot markiert.

Diamex	`-c stk500`	`-P COMx` angeben
Pololu	`-c stk500`	`-P COMx` angeben
avrisp mk2	`-c avrispmkii`	`-P` entfällt
Atmel Ice	`-c avrispmkii`	`-P` entfällt

Der Tester kann beim Programmieren mit einem der o.g. Programmer durch diese mit Spannung versorgt werden. Die meisten Programmer besitzen hierzu einen Dip-Schalter (z.B. Diamex) oder einen Jumper. Bitte darauf achten, dass der Programmer die Versorgungsspannung von 5V auch zur Verfügung stellt und damit den ATmega2560 versorgt (ggf. muss das Display entfernt werden).

In Kapitel 9 befinden sich weitere Anleitungen für einige Programmiergeräte. Geeignete Programmierer werden in Kapitel 8.1 aufgeführt.

Wenn der Tester per ISP mit Spannung (5V) versorgt wird (empfohlen), dann darf dieser nicht mehr per USB, Hohlstecker etc. mit Spannung versorgt werden!

Die Erfahrungen mit dem USBASP sind durchwachsen. Eine Empfehlung für diesen Programmer kann nicht gegeben werden. Bitte Kapitel 8.1 beachten!

Der SD-Kartenadapter darf während der Programmierung nicht eingesteckt sein, da dieser ansonsten den ISP-Port belegt.

Warnhinweise zur Programmierung des Speichertesters:

Niemals den Speichertester **gleichzeitig** mit dem ISP-Programmierer und per USB, Hohlstecker oder Schraubterminal mit Spannung versorgen.

Wenn mit dem gesteckten DC/DC-Konverter „Alternative #2“ (mit Recom-Wandler) der ATmega2560 per ISP programmiert wird, ist zu beachten, dass der DC/DC-Konverter bei einer Spannungsversorgung durch den Programmierer, die Versorgungsspannungen Vdd und Vbb erzeugt. Der Programmierer sollte diese zusätzliche Last vertragen können, ansonsten sollte der DC/DC-Konverter vorher entfernt werden. Bei dem oben empfohlenen Programmierer ist das der Fall und es können sogar Chips getestet werden.

4.1 Programmieren der Fuses

Möglichkeit 1: Per Kommandozeile

Zunächst sollte die Kommunikation getestet werden, kann dieses per

```
avrdude.exe -C"avrdude.conf" -v -patmega2560 -cstk500 -PCOM5
-U lfuse:r:-:i -U hfuse:r:-:i -U efuse:r:-:i
```

erfolgen. Es werden mit diesem Befehl nur die gesetzten Fuses angezeigt.

```
avrdude.exe: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.02s

avrdude.exe: Device signature = 0x1e9801 (probably m2560)
avrdude.exe: reading lfuse memory:

Reading | ################################################## | 100% 0.00s

avrdude.exe: writing output file "<stdout>"
:01000000629D
:00000001FF
avrdude.exe: reading hfuse memory:

Reading | ################################################## | 100% 0.00s

avrdude.exe: writing output file "<stdout>"
:010000009966
:00000001FF
avrdude.exe: reading efuse memory:

Reading | ################################################## | 100% 0.00s

avrdude.exe: writing output file "<stdout>"
:01000000FF00
:00000001FF

avrdude.exe: safemode: Fuses OK (E:FF, H:99, L:62)

avrdude.exe done.  Thank you.
```

Abbildung 4.1: unprogrammierter ATmega2560

Die Programmierung der Fuses erfolgt dann wie folgt:

```
avrdude.exe -C"avrdude.conf" -v -patmega2560 -cstk500 -PCOM5
-U lfuse:w:0xf7:m -U hfuse:w:0xd7:m -U efuse:w:0xff:m
```

Hinweis: Durch Setzten der h-Fuse auf "0xd7" wird die Konfiguration des Testers dauerhaft gespeichert und überlebt ein erneutes Programmieren der Firmware. Durch Ändern der h-Fuse von "0xd7" auf "0xdf" wird das EEPROM bei jedem Flash-Vorgang gelöscht und damit auch eine vorhandene Konfiguration.

Möglichkeit 2: Per Batchdatei

In dem Ordner „fuses“ befindet sich eine Batchdatei `flash_fuses_for_stk500-wiring-usbasp.bat`.

Diese Batchdatei fragt nacheinander die folgenden Punkte ab:

1. verwendeter Programmer (STK500 komp., Wiring, USBASP)
2. verwendeter COM-Port (bei STK500 oder Wiring)
3. löschen der Konfiguration im EEPROM

Danach werden die aktuell gesetzten Fuses angezeigt und auf Tastendruck neu gesetzt. Auf Wunsch kann anschließend eine Batchdatei `flash_firmware.bat` erstellt werden, die im 2. Schritt – dem Programmieren der Firmware – verwendet werden kann und die bereits eingegeben Parameter enthält.

Ein Video ist auf YouTube abrufbar:

https://www.youtube.com/watch?v=KFFmAwlJ9ns (Deutsch)

https://www.youtube.com/watch?v=ijMVng-y_vA (Englisch)

Hinweis zur Fehlermeldung „stk500v2_command(): command failed“:

AVRDUDE kann je nach Programmer einige Meldungen ausgeben, die irritierend sein können.

Die Meldung

```
avrdude: stk500v2_command(): command failed
avrdude: stk500v2_getparm(): failed to get parameter 0x9a
```

ist kein Fehler, sondern nur ein Hinweis, dass ein Kommando vom Programmer nicht unterstützt wird.

Der Atmel STK-500 unterstützt Erweiterungen (Top Cards), die mit dem Kommando 0x9a von AVRDUDE abgefragt werden. Bis auf das Original werden diese Erweiterungen von keinem Programmer unterstützt, daher die Meldung.

Wichtiger Hinweis (nur für Experten, alle anderen können das Überspringen ;)):

Mit den obigen Einstellungen wird der ATmega bzw. der Quarz im Modus „Full Swing Crystal Oscillator“ betrieben, der etwas mehr Strom benötigt. Das macht Sinn, da damit auch Quarze mit geringem Spannungshub (= schlechte Qualität) verwendet werden können. Wer möchte, der kann gerne den „Low Power Crystal Oscillator“ ausprobieren. Sollte es später zu sporadischen Resets oder Probleme beim Flashen der Firmware kommen, kann jederzeit auf den „Full Swing Crystal Oscillator“ zurückgewechselt werden.

Soll der „Low Power Crystal Oscillator“ verwendet werden, muss beim Setzen der Fuses der Wert der `lfuse` von "`0xf7`" auf "`0xff`" geändert werden.

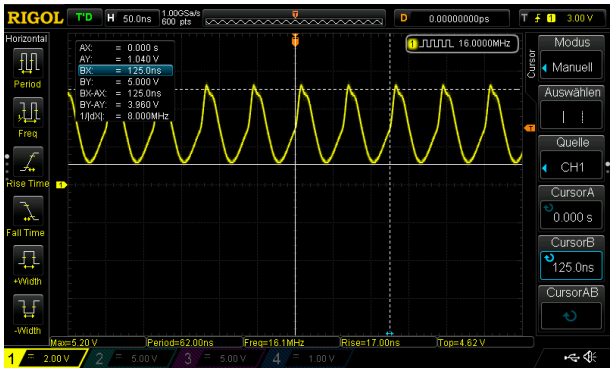
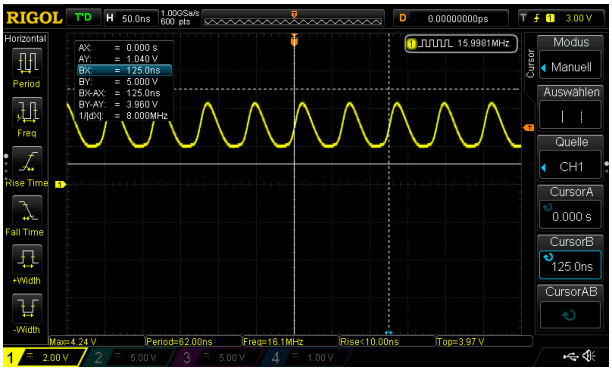


Abbildung 4.2: „guter“ Quarz



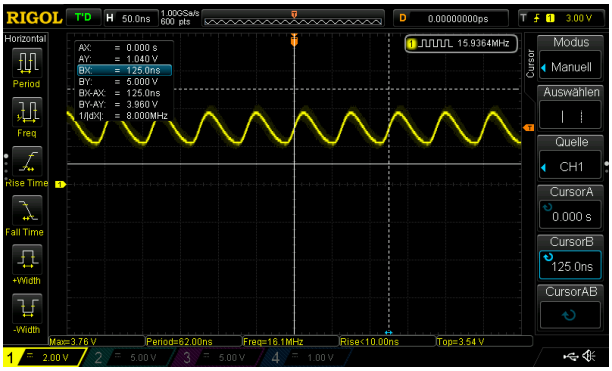
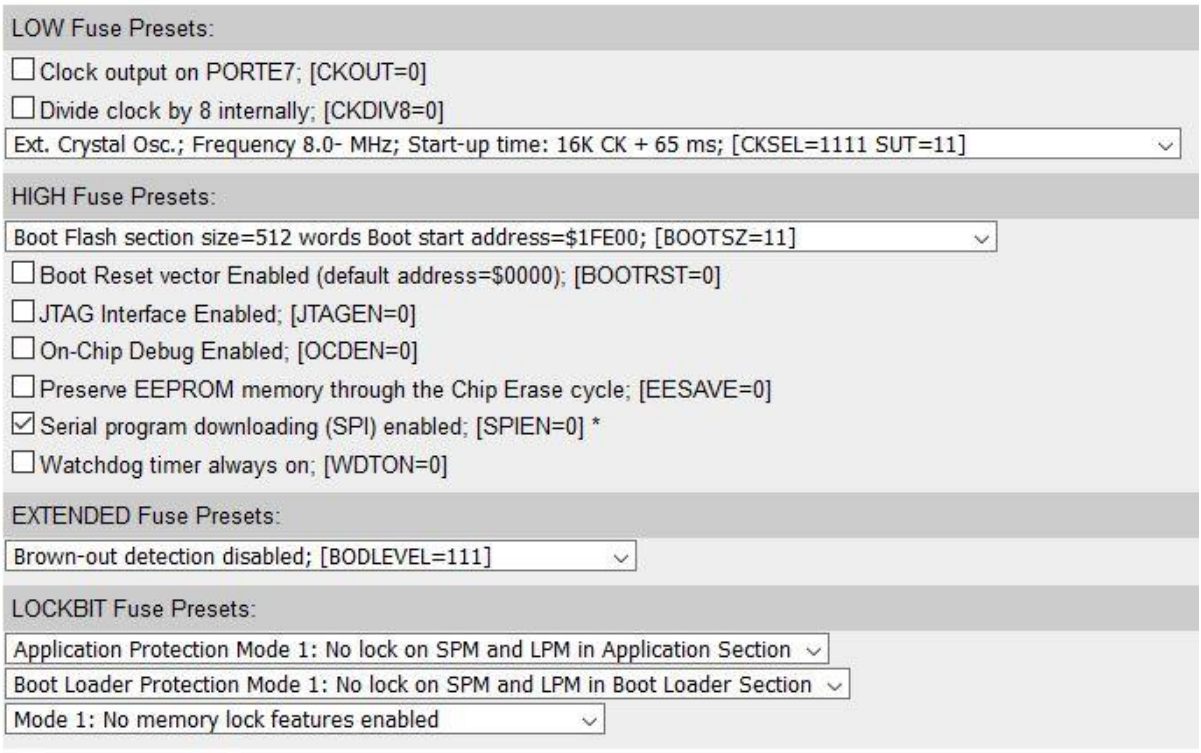


Abbildung 4.3: zwei „weniger gute“ Quarze

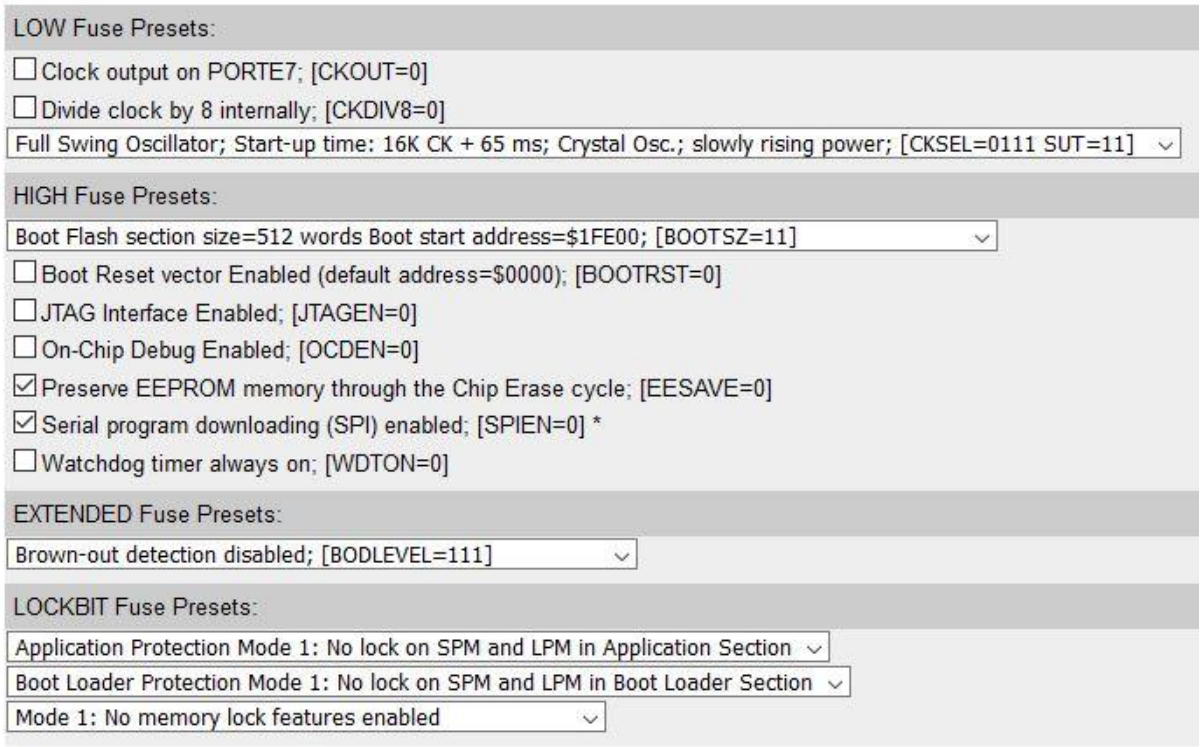
In den obigen Bildern wurden drei verschiedene Quarze in dieselbe Oszillatorschaltung eingesetzt.

Wer gerne die Fuses mit einem anderen Programm setzen möchte, findet die empfohlenen Einstellungen in den folgenden Bildern.

LOW = FF (“Low Power Crystal Oscillator“), HIGH = DF (“Don’t save EEPROM”), EXTENDED = FF



LOW = F7 (“Full Swing Crystal Oscillator”), HIGH = D7 (“Preserve EEPROM”), EXTENDED = FF



4.2 Programmieren der Firmware

Möglichkeit 1: Per Kommandozeile

Die Firmware wird wie folgt programmiert:

```
avrdude.exe -C"avrdude.conf" -v -patmega2560 -cstk500 -PCOM5
-Uflash:w:Chip-TesterPro-FW-v0.XX.hex:i
```

Der Dateiname muss jeweils an die zu brennende Version der Firmware angepasst werden.

```
avrdude.exe: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.01s

avrdude.exe: Device signature = 0x1e9801 (probably m2560)
avrdude.exe: safemode: lfuse reads as F7
avrdude.exe: safemode: hfuse reads as D7
avrdude.exe: safemode: efuse reads as FF
avrdude.exe: NOTE: "flash" memory has been specified, an erase cycle will be performed
             To disable this feature, specify the -D option.
avrdude.exe: erasing chip
avrdude.exe: reading input file "Chip-TesterPro-FW-v0.19.hex"
avrdude.exe: writing flash (256214 bytes):

Writing | ################################################## | 100% 8.63s

avrdude.exe: 256214 bytes of flash written
avrdude.exe: verifying flash memory against Chip-TesterPro-FW-v0.19.hex:
avrdude.exe: load data flash data from input file Chip-TesterPro-FW-v0.19.hex:
avrdude.exe: input file Chip-TesterPro-FW-v0.19.hex contains 256214 bytes
avrdude.exe: reading on-chip flash data:

Reading | ################################################## | 100% 6.28s

avrdude.exe: verifying ...
avrdude.exe: 256214 bytes of flash verified

avrdude.exe: safemode: lfuse reads as F7
avrdude.exe: safemode: hfuse reads as D7
avrdude.exe: safemode: efuse reads as FF
avrdude.exe: safemode: Fuses OK (E:FF, H:D7, L:F7)

avrdude.exe done.  Thank you.
```

Abbildung 4.4: fertig programmierter ATmega2560 (hier FW v.19)

Wenn die Meldung

```
avrdude.exe: verifying ...
avrdude.exe: verification error, first mismatch at byte 0x0000
             0xXX != 0xYY
avrdude.exe: verification error; content mismatch
```

erscheint, dann ist mit hoher Wahrscheinlichkeit der Programmer (USBASP benutzt?) nicht geeignet einen ATmega2560 zu programmieren (siehe Kapitel 8.1).

Möglichkeit 2: Per Batchdatei (selbst erstellt)

Der Batchdatei `flash_firmware_(please_edit).bat` muss an den verwendeten Programmer und den COM-Port angepasst werden. Die Datei ist so geschrieben, dass jeweils die vorgefundene Firmware gebrannt wird.

Möglichkeit 3: Per Batchdatei (automatisch erstellt)

Wurde beim Setzen der Fuses die Batchdatei `flash_firmware.bat` erstellt, kann diese dazu verwendet werden die Firmware zu brennen. Hierzu diese in den Firmware-Ordner kopieren und starten.

Möglichkeit 4: Per interaktiver Batchdatei

Es wird die Batchdatei `winflash`.bat mitgeliefert. Wird die Firmware-Datei auf dieser abgelegt, fragt diese anschließend interaktiv den verwendeten Programmer und ggf. den COM-Port ab.

TIPP:

Sind die Fuses korrekt gesetzt, kann die Firmware geflasht werden. Mit dem Parameter „`-B1`“ kann der Vorgang erheblich beschleunigt werden. Dieses sollte aber bei der Erstprogrammierung ggf. noch nicht ausprobiert werden.

4.3 Update der Firmware

Ein Firmware-Update beschränkt sich auf das Programmieren der Firmware wie in Kapitel 4.2 beschrieben. Es ist nicht nötig die Fuses noch einmal zu setzen.

5 Erste Inbetriebnahme

Der Chip Tester sollte nach dem Aufbau und Programmierung direkt einsatzbereit sein. Sobald er mit Spannung versorgt wird, sollte für ca. zwei Sekunden eine Einschaltmeldung erscheinen und danach das Menü.

Sollte das nicht der Fall sein: Keine Panik und die folgende kurze Checkliste durchgehen:

Es erscheint kein Text auf dem Display und der Chip Tester wird ohne DC/DC-Modul betrieben?

- Sind die Jumper auf dem Board gesetzt?

Es erscheint kein Text auf dem Display und der Chip Tester wird mit DC/DC-Modul betrieben?

- Leuchten alle drei (grünen) LEDs auf dem Modul für die Versorgungsspannungen? Sollte eine LED nicht leuchten, deutet dieses ggf. auf einen Kurzschluss hin. Bitte den Chip Tester von der Spannung trennen und in Abs. 8 nachlesen.

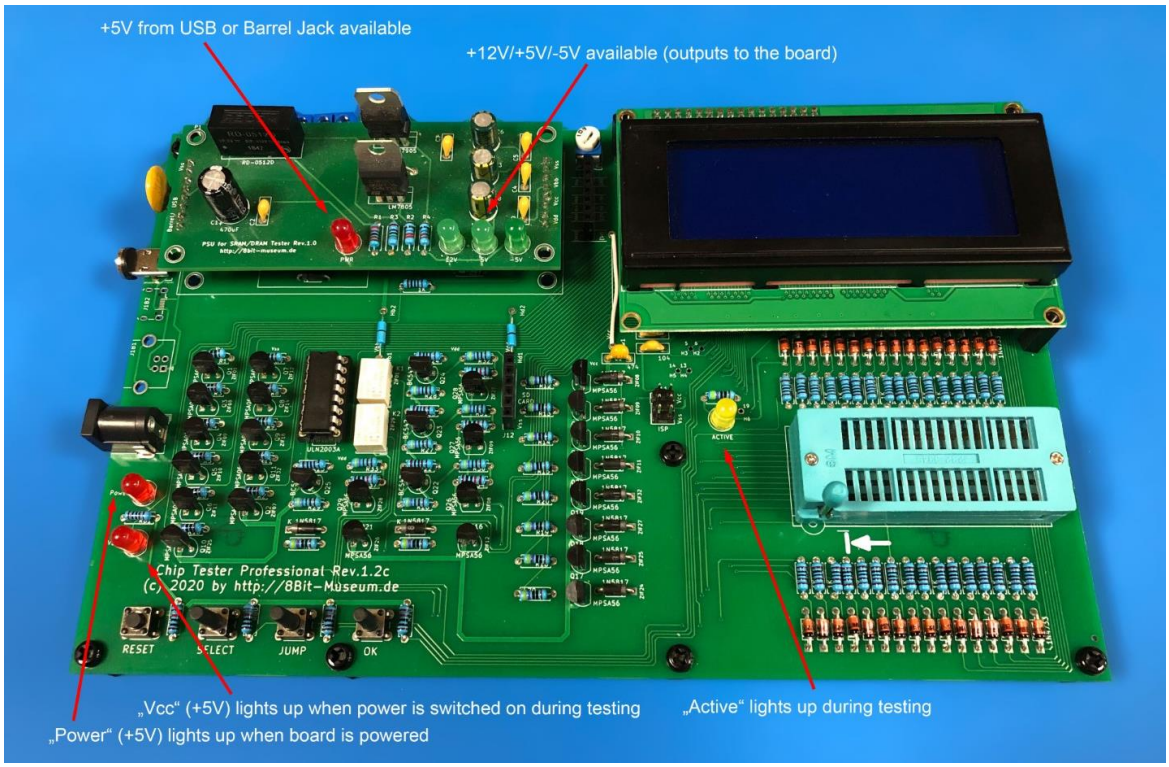
Leuchtet das Display, aber es erscheint kein Text?

- Wenn ja, dann ist vermutlich der Kontrast nicht korrekt eingestellt. Mit dem Potentiometer links neben dem Display kann dieser verändert werden.

Erscheint das Einschaltmenü, aber die Bedienung ist extrem langsam?

- Wenn der Tester sehr langsam reagiert, dann sind die Fuses des ATmega2560 nicht korrekt gesetzt worden, siehe Abs. 4.1. Wurde dieses vergessen, läuft der Mikroprozessor nur mit 1 MHz und nicht mit 16 MHz.

Dieses sind die häufigsten Probleme bei der ersten Inbetriebnahme. Bei weiteren Problemen bitte in Abs. 8 nachlesen.



„Vcc“ leuchtet, wenn während der Tests die Versorgungsspannung eingeschaltet ist. „Power“ leuchtet sobald der Tester mit Spannung versorgt wird. „Active“ zeigt an, dass gerade getestet wird.

6 Anschluss des SD-Kartenadapters

Um Speicherbausteine auslesen zu können, muss ein SD-Kartenadapter an den Speichertester angeschlossen werden. Hierzu bieten sich mehrere Möglichkeiten an.

6.1 Anschluss an der 6-pol. Pfostenleiste

Platinen ab der Rev.1.2c besitzen eine 6-pol. Pfostenleiste. In diese Leiste kann der SD-Kartenadapter direkt eingestellt werden. Der Anschluss „Vss“ ist gekennzeichnet.

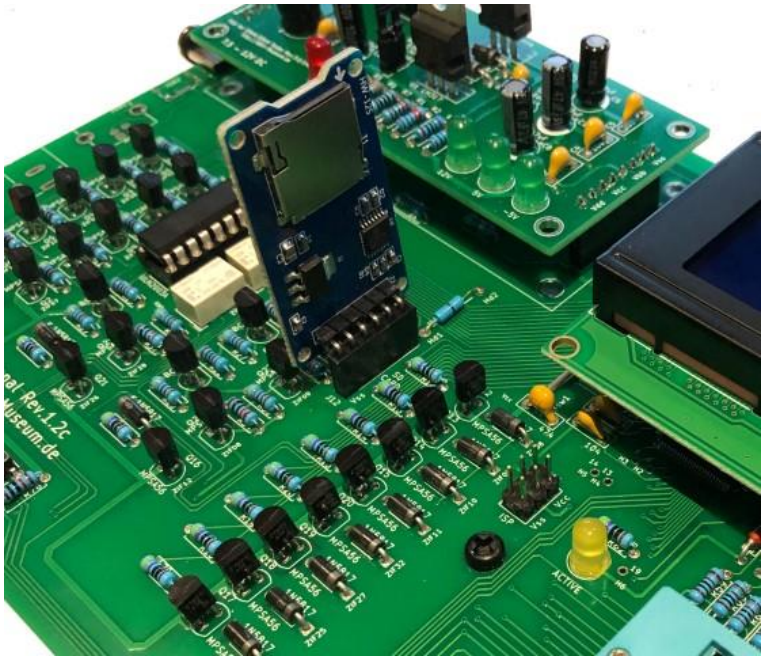


Abbildung 6.1: SD-Kartenadapter in der bestückten Pfostenleiste

Wer möchte, kann die vorhandene Stiftleiste am SD-Kartenadapter auch entfernen und auf der Rückseite eine neue Stiftleiste einsetzen. Dadurch kann der SD-Kartenadapter liegend eingebaut werden (auf Kurzschlüsse achten).

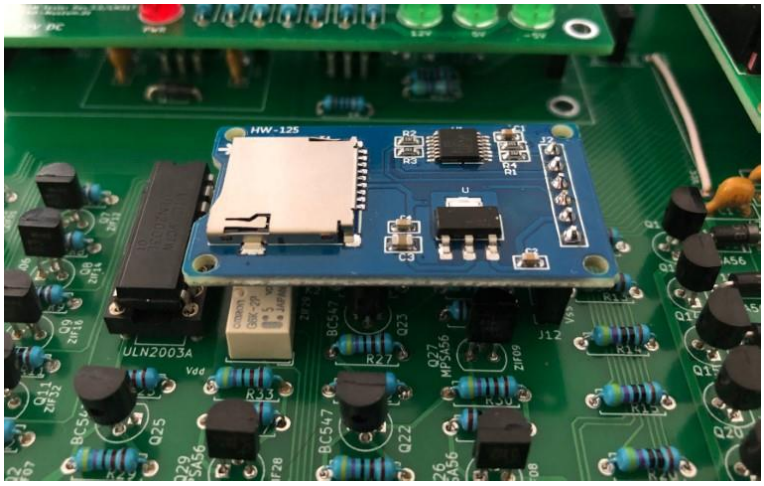


Abbildung 6.2: SD-Kartenadapter horizontal eingebaut

6.2 Vorbereitung der SD-Karte

Damit die SD-Karte verwendet werden kann müssen in der Konfiguration die Unterstützung für SD-Karten eingeschaltet werden (“**SD Card: 1**”) und die SD-Karte muss FAT32 formatiert sein. Die unterstützte maximale Größe beträgt 32 GByte.

Falls die SD-Karte nicht erkannt wird, sollte die Karte probeweise mit dem „SD Memory Card Formatter“ von der SD Association formatiert werden.

https://www.sdcard.org/downloads/formatter/

Dieses Tool formatiert eine beliebige SD-Karte standardkonform. Alle Dateien werden im Hauptverzeichnis der SD-Karte abgelegt.

6.3 Anschluss an den SPI-Anschluss über den 6-pol. ISP-Stecker

Der Anschluss an den SPI-Port ist wie folgt möglich (Hinweis: CS ist mit GND verbunden, da der ISP kein CS bereitstellt).

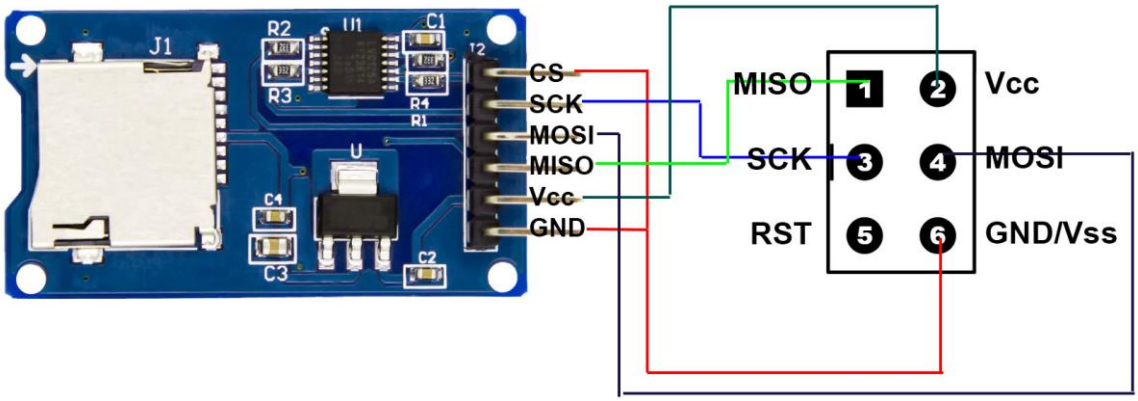


Abbildung 6.3: Beschaltung SD-Kartenadapter

Mit einer Crimpzange für Dupont-Stecker lässt sich schnell ein geeignetes Kabel konfektionieren:

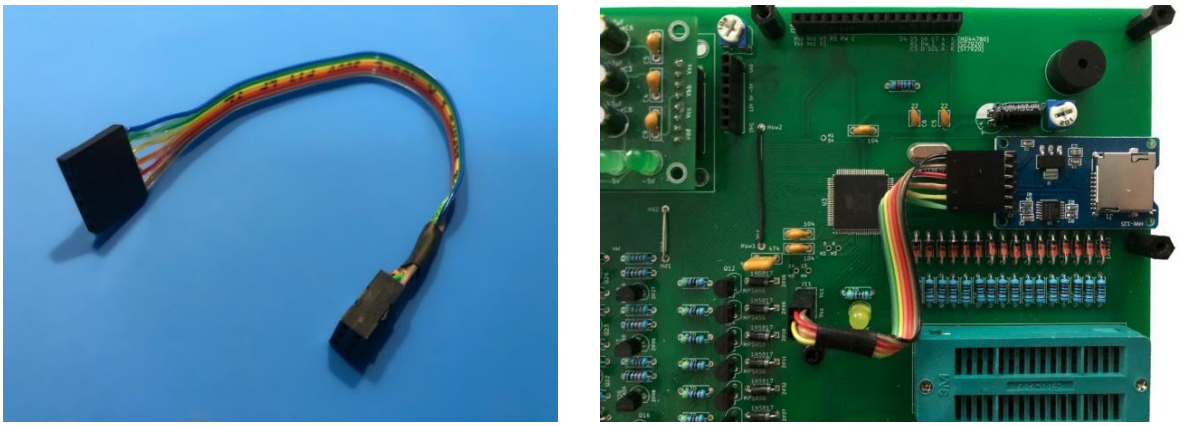


Abbildung 6.4: SD-Kartenadapter mit Verlängerung

Der SD-Kartenadapter passt rechts auf die Platine direkt unter das Display. Hierbei unbedingt darauf achten, dass kein Kurzschluss entsteht. Danke an Greg64 für die Idee.

Ab der Rev.1.2c sollte der Anschluss über die dafür vorgesehene Pfostenleiste (siehe Kap. 6.1) erfolgen.

7 Eigene Experimente

Auch wenn die eigentliche Soft- und Hardware „closed-source“ sind, dürfen gerne eigene Experimente durchgeführt werden. In diesem Kapitel wird erklärt, wie der Sockel belegt ist und wie die Versorgungsspannungen geschaltet werden können.

7.1 Belegung des Sockels

Der Belegung des Sockels (AVR-Pinout) ist wie folgt:

SRAM / DRAM Sockel

Vbb	Vdd	Vcc	Vss	Port		ZIF		Port	Vss	Vcc	Vdd	Vbb
79			55	52	1		32	21	63	72		
				51	2		31	22				
				50	3		30	23				
				49	4		29	24				80
				48	5		28	25			78	
				47	6		27	26		71	77	
			64	46	7		26	27		74		
	75	65	56	45	8		25	28	85	70		
	76	66	57	44	9		24	29		69		
		67	58	43	10		23	30				
		68	59	42	11		22	31				
		73	60	41	12		21	32				
				40	13		20	33				
			61	39	14		19	34				
				38	15		18	35				
			62	37	16		17	36				

Abbildung 7.1: Ansteuerung des Sockels

SELECT, OK, JUMP sind Eingänge (Taster), ACTIVE (LED), Buzzer und Vcc Power (schaltet den MOSFET) sind Ausgänge.

SELECT	10
JUMP	11
OK	12
Vcc switch	20
Active LED	7
Buzzer	53

Das Display wird über RS=9, EN=8, D4-D7: 81, 82, 83, 84 angesteuert.

Alle Bezeichnungen entsprechen noch die des MegaCore Framework, allerdings verzichtet die Software inzwischen darauf und spricht aus Geschwindigkeitsgründen die Ports direkt an.

Ein Bootloader wird nicht benötigt, da über den ISP-Anschluss programmiert wird (schneller und spart Speicher und Kosten).

7.2 Programmierung mit Hilfe des Arduino Frameworks

Aufgrund seiner Vergangenheit kann der Chip Tester immer noch mit Hilfe des Arduino Frameworks programmiert werden. Da das Standard Framework des Arduino Mega 2560 nicht alle Digital-I/O unterstützt und die Nummerierung der Pins auch nicht wirklich intuitiv ist, wurde die Entwicklung an das MegaCore Framework angelegt. Der Vorteil: Wer möchte, kann dieses Framework benutzen, um eine eigene Firmware zu erstellen.

Für einen leichteren Start sind hier einige Beispiele für die Programmierung des Chip Testers aufgeführt. Das Wiring-Framework ist nicht besonders schnell, ggf. sollten alternative Wege zum Setzen der Zustände verwendet werden.

7.2.1 Ansprechen des HD44780 kompatiblen Displays

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(9, 8, 81, 82, 83, 84);
lcd.begin(20, 4);
lcd.clear();
lcd.setCursor(0,0);
lcd.print("Hello Chip Tester");
```

7.2.2 Ein-/Ausschalten von Vcc

```
#define PIN_POWER 20
pinMode(PIN_POWER, OUTPUT);
digitalWrite(PIN_POWER, HIGH);        // Vcc ein
digitalWrite(PIN_POWER, LOW);         // Vcc aus
```

7.2.3 Taster und LED

```
#define PIN_SELECT 10
pinMode(PIN_SELECT, INPUT);
buttonState = digitalRead(PIN_SELECT); // Status von SELECT


#define PIN_ACTIVE 7
pinMode(PIN_ACTIVE, OUTPUT);
digitalWrite(PIN_ACTIVE, LOW);        // LED aus
digitalWrite(PIN_ACTIVE, HIGH);       // LED ein
```

7.2.4 Signale setzen und abfragen

```
pinMode(52, OUTPUT);                  // ZIF01 auf Ausgabe
digitalWrite(52, LOW);                // ZIF01 auf LOW

pinMode(52, INPUT);                   // ZIF01 auf Eingabe
ZIF01State = digitalRead(52);         // Status von ZIF01
```

7.2.5 Schalten von Vss, Vcc, Vbb und Vdd

```
pinMode(79, OUTPUT);                    // ZIF01 Vbb
digitalWrite(79, HIGH);                 // ZIF01 Vbb einschalten

pinMode(55, OUTPUT );                   // ZIF01 Vss
digitalWrite(55, HIGH);                 // ZIF01 Vss einschalten

pinMode(75, OUTPUT);                    // ZIF08 Vdd
digitalWrite(75, HIGH);                 // ZIF08 Vdd einschalten

pinMode(65, OUTPUT);                    // ZIF08 Vcc
digitalWrite(65, LOW);                  // ZIF08 Vcc einschalten
```

7.3 Belegung des Spannungssteckers

An dem 7-Pin-Connector können die Spannungen 12V, 5V, -5V abgegriffen werden (z.B. falls man einen Adaptersockel für exotische Chips basteln will und dafür Vdd (12V) oder Vbb (-5V) benötigt).

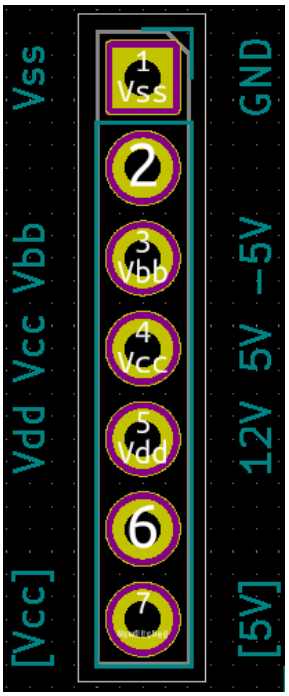


Abbildung 7.2: Spannungsstecker

Die [Vcc] bzw. [5V] liegen nur während eines Tests an. Die anderen Spannungen stehen dauerhaft zur Verfügung.

7.4 Belegung der Stecker der DC/DC-Konverterplatine

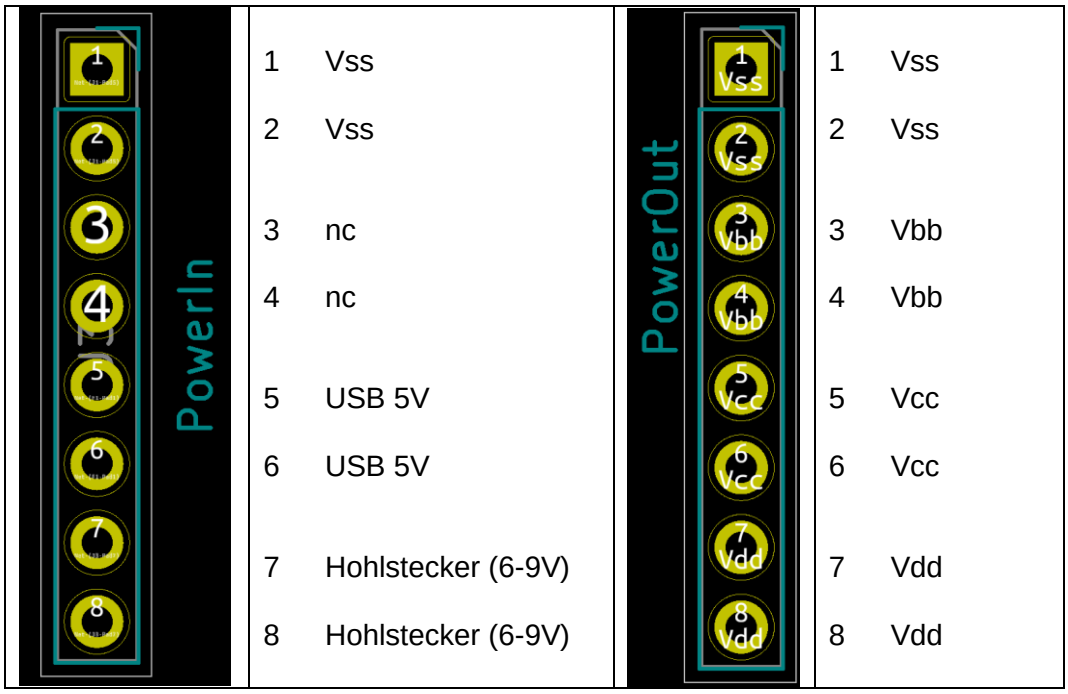


Abbildung 7.3: Belegung der Konverterplatine

7.5 Belegung des ISP-Headers

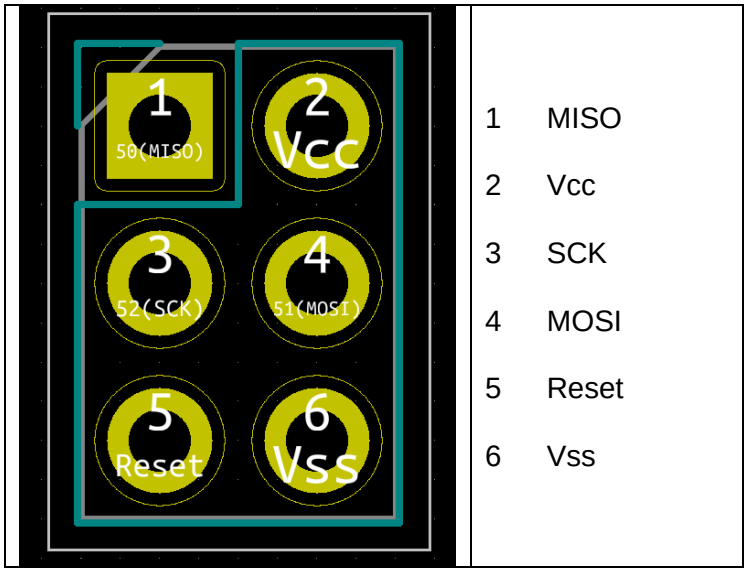


Abbildung 7.4: Belegung des ISP-Headers

7.6 Belegung des SD-Card-Headers

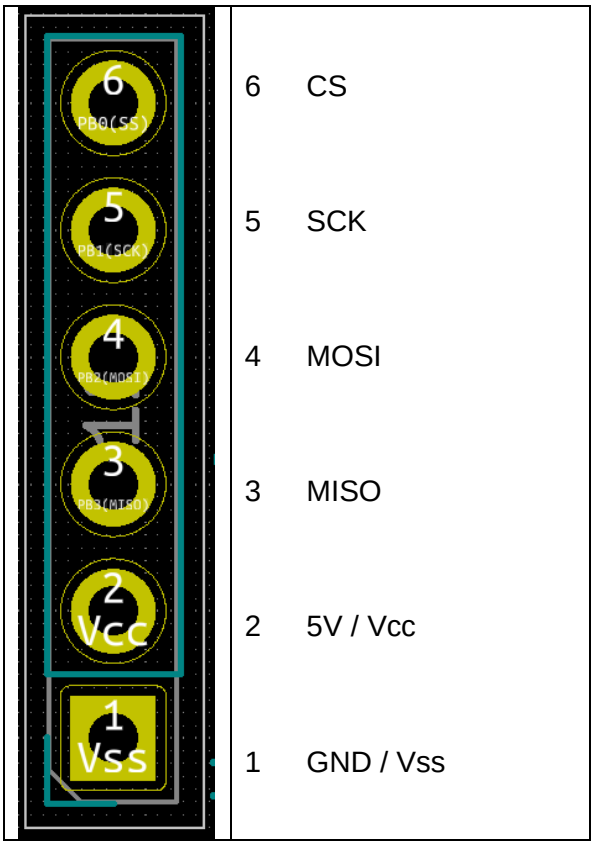


Abbildung 7.5: Belegung des SD-Card-Headers

8 Fehlersuche

8.1 Übersicht und Hinweise zu geeigneten ISP-Programmierern

8.1.1 Bisher erprobte Programmierer

Die folgenden Programmierer sind erprobt und können einen ATmega2560 fehlerfrei programmieren:

- Diamex USB ISP-Programmer für AVR, STM32, LPC-Cortex (Prog-S2) (Artikel Nr. 102106), ca. 20 Euro
- Diamex USB ISP-Programmer für AVR, STM32, LPC, ESP32/8266 (Prog-S2E) (Artikel Nr. 102106), ca. 20 Euro
- Diamex USB ISP-Programmer für Atmel AVR (Artikel Nr. 102104), ca. 18 Euro
- Tremex USB ISP-Programmer, ca. 20 Euro
- Pololu USB AVR Programmer, ca. 20 Euro
- Pololu USB AVR Programmer v2.1, ca. 12 Euro
- mySmartUSB MK2, ca. 30 Euro
- mySmartUSB MK3, ca. 40 Euro
- mySmartUSB light, ca. 16 EUR
  Die Programmierung scheint mit AVRDUDE nicht zu funktionieren, verwenden Sie hierzu das beiliegende „myAVR ProgTool“.
- Atmel ICE, ca. 120 Euro
- Atmel STK500
- Diamex USB ISP-Programmer Stick für AVR (Artikel Nr. 102108, 2019+(?)), ca. 18 Euro
- diverse China Nachbauten des AVR ISP MKii, ca. 16 Euro, Treiberinstallation notwendig (funktionieren, wird aber nicht empfohlen)
- Microchip PICkit4, ca. 100 EUR

Wer ein einfach zu bedienendes Programmiergerät sucht, der sollte sich den Programmierer von Diamex/Tremex ansehen. Wer ein umfangreich konfigurierbares Gerät sucht, sollte sich den Pololu ansehen. Beide Programmiergeräte benötigen keine Treiber und werden unter Windows durch einen virtuellen COM-Port angesprochen.

Rückmeldungen, dass es Probleme gab oder die Programmierung nicht funktionierte:

- USBASP (diverse Modelle), ca. 10 Euro
- Bus Pirate, ca. 25 Euro
- Diamex USB ISP-Programmer Stick für AVR (Artikel Nr. 102108, vor 2019), ca. 18 Euro

Nicht erprobt:

- Pololu PGM03A (gem. Hersteller liefert der Programmierer keine Spannungsversorgung, d.h. das Board müsste durch USB selbst versorgt werden).

8.1.2 Hinweise zu einigen speziellen ISP-Programmierern

Der Programmer muss geeignet sein einen ATmega2560 zu programmieren, was nicht jeder ist, der verspricht AVR-Chips programmieren zu können.

USBtinyISP

In der Anleitung zum z.B. USBtinyISP steht sehr deutlich

> *Works with any AVR ISP chip with 64K of flash (or less) - does not work with Atmega1281/1280/2561/2560*

Dabei ist dieser mit 22 US$ noch nicht einmal günstig.

USBASP

Die Erfahrungen mit dem USBASP sind sehr schlecht. Google liefert zig Ergebnisse mit Problemen im Zusammenhang mit dem ATmega2560. Wer es aber trotzdem mit diesem probieren will findet hier erste Informationen:

https://forum.arduino.cc/index.php?topic=363772.0

Es gibt zumindest einen Hinweis, dass die Firmware

> https://www.fischl.de/usbasp/usbasp.2011-05-28.tar.gz
>
> avrdude.exe -p atmega8 -c usbasp -U flash:w:usbasp.atmega8.2011-05-28.hex:i -F -P usb

funktionieren soll, wenn man den USBASP im „slow-sck“ Modus betreibt (dazu muss J3 gesetzt sein). Die Programmierung dauert dann aber extrem(!) lang (über 40 Minuten).

Zur Programmierung kann „Khazama AVR Programmer“ verwendet werden. Bitte beachten, dass ein anderes gängiges Programm namens „eXtreme Buner AVR“ den ATmega2560 nicht korrekt programmiert und nicht zum Programmieren verwendet werden sollte.

Eine gute Anleitung, wie die Firmware des USBASP mit Hilfe eines Arduino aktualisiert werden kann, ist hier zu finden: https://www.electronics-lab.com/project/usbasp-firmware-update-guide/

mySmartUSB light

mySmartUSB light scheint Probleme mit AVRDUDE zu haben:

> https://www.mikrocontroller.net/topic/272299

Mit der herstellereigenen Software scheint der Programmiervorgang zu klappen.

Andere Programmer:

In der Anleitung von AVRDUDE gibt es noch diesen Hinweis:

```
m2560 ATmega2560 (**)
m2561 ATmega2561 (**)

(**) Flash addressing above 128 KB is not supported by all programming
hardware. Known to work are jtag2, stk500v2, and bit-bang programmers.
```

siehe https://www.nongnu.org/avrdude/user-manual/avrdude_4.html#Option-Descriptions

Der an anderer Stelle empfohlene Programmer kostet unter 20 EUR und ist stk500v2 kompatibel.

8.2 Probleme beim Programmieren des ATmega2560

8.2.1 Versorgungsspannung prüfen!

Sollte der Chip nicht erkannt werden, zunächst prüfen, ob dieser überhaupt mit Spannung versorgt wird, denn nicht alle Programmierer liefern eine Versorgungsspannung für den ATmega2560. Manchmal muss der Programmierer erst entsprechend eingestellt werden.

Sollte der Programmierer selbst keine Versorgungsspannung bereitstellen, dann muss der Tester zusätzlich per USB mit Spannung versorgt werden. In diesem Fall muss aber sichergestellt werden, dass der Programmierer wirklich keine solche liefert, sonst kann es zu Problemen kommen.

8.2.2 Wurde der korrekte Programmierer bei AVRDUDE angegeben?

Ist die Versorgungsspannung i. O. und wird der Chip nicht erkannt, bitte prüfen, ob der Programmierer korrekt bei AVRDUDE angegeben wurde.

Bei vielen Programmierern, die eine serielle Schnittstelle simulieren, ist **`-cwiring`** oder **`-cstk500`** korrekt. Die serielle Schnittstelle wird mit dem Parameter „-P“ bestimmt, also z.B. **`-PCOM5`** für Port 5. Die Geschwindigkeit kann mit **`-b115200`** auf 115200 bit/s festgelegt werden.

Bei USBASP ist als Parameter **`-cusbasp`** anzugeben. „-P“ und „-b“ entfallen hier.

Mit folgendem Befehl kann die Kommunikation gefahrlos getestet werden (es werden nur die aktuellen Fuses ausgelesen ohne eine Änderung durchzuführen):

```
avrdude.exe -C"avrdude.conf" -v -patmega2560 -cstk500 -PCOM5 -b115200 -U lfuse:r:-:i -U hfuse:r:-:i -U efuse:r:-:i
```

bzw.

```
avrdude.exe -C"avrdude.conf" -v -patmega2560 -cusbasp -U lfuse:r:-:i -U hfuse:r:-:i -U efuse:r:-:i
```

Sollte beim Lesen der Fuses die Chip-Signatur nicht korrekt sein (oder sogar jedes Mal eine unterschiedliche Signatur angezeigt werden), liegt es wahrscheinlich an einer zu hohen Geschwindigkeit mit welcher der ISP-Programmer mit dem Chip kommuniziert. In diesem Fall die Geschwindigkeit heruntersetzen. Beim USBASP den Parameter „-B“ erhöhen, z.B. auf `-B4` oder `-B8`.

Von der Verwendung des USBASP wird abgeraten!

8.2.3 Beim Programmieren erscheint ein „verify error“

Wenn am Ende des Programmiervorgangs ein „verify error“ erscheint, ist der Programmierer sehr wahrscheinlich nicht zum Programmieren des ATmega2560 geeignet. Der ATmega2560 ist einer der wenigen ATmega, die über 256kb verfügen und viele einfache Programmer kennen als Obergrenze nur 128kb (siehe auch 128kb Bug beim USBASP).

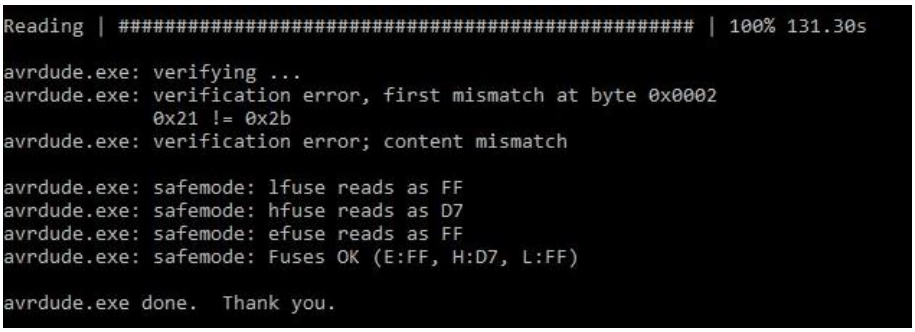


Abbildung 8.1: „verify error“ (128kb Bug)

In diesem Fall bitte nachsehen, ob es ein Firmwareupdate für den Programmer gibt oder auf einen geeigneten Programmer wechseln. Alte Programmer geben nur eine pauschale Kompatibilität zum ATmega an, die seit Erscheinen des ATmega2560 nicht unbedingt gegeben ist.

Einige Quarze liefern einen zu geringen Spannungshub, so dass der Prozessor, der einen „Low Power Oscillator“ („`lfuse`“ ist auf `0xff` programmiert) erwartet, nicht korrekt funktioniert. In diesem Fall sollte der „Full Swing Oscillator“ ausprobiert werden. Hierzu wird die „`lfuse`“ auf `0xf7` programmiert:

```
avrdude.exe -C"avrdude.conf" -v -patmega2560 -cstk500 -PCOM5
-U lfuse:w:0xf7:m -U hfuse:w:0xdf:m -U efuse:w:0xff:m
```

Für die „`hfuse`“ gilt wieder: `0xdf` = die Konfiguration wird bei einem Firmwareupdate gelöscht, `0xd7` = die Konfiguration bleibt nach einem Firmwareupdate erhalten.

In sehr seltenen Fällen sind der Quarz oder die 22p Kondensatoren der Fehler. Falls der „verify error“ mit einem der empfohlenen Programmer auftritt, sollten diese drei Bauelemente getauscht werden. Es sollte vorher aber noch probiert werden den ATmega nicht im „Low Power Crystal Oscillator“, sondern im "Full Swing Crystal Oscillator" zu betreiben. Näheres hierzu im Abs. 4.1.

8.2.4 Verbindung zum ATmega prüfen!

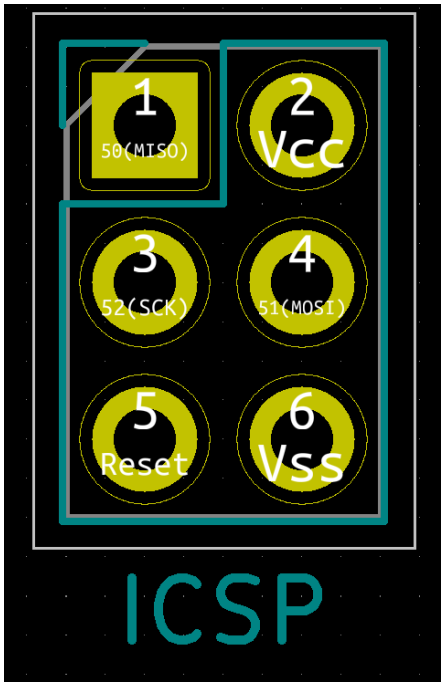
Sollte der ATmega2560 schon vormontiert gewesen sein, ist dieses Prüfung nicht notwendig!

Falls immer noch keine Kommunikation möglich ist, könnte es sein, dass der Chip nicht korrekt aufgelötet wurde (kalte Lötstelle) oder einen Kurzschluss hat.

Mit einem Durchgangsprüfer kann die Verbindung wir folgt geprüft werden:

- Pin 1 des Headers geht an Pin 22 des ATmega2560 (MISO)
- Pin 2 des Headers geht an Pin 10 des ATmega2560 (Vcc)
- Pin 3 des Headers geht an Pin 20 des ATmega2560 (SCK)
- Pin 4 des Headers geht an Pin 21 des ATmega2560 (MOSI)
- Pin 5 des Headers geht an Pin 30 des ATmega2560 (Reset)
- Pin 6 des Headers geht an Pin 11 des ATmega2560 (Vss)

Die Belegung des ISP-Headers ist wie folgt:



Die Belegung des ATmega2560 ist wie folgt (die Zählung beginnt an der Markierung entgegen des Uhrzeigersinns):

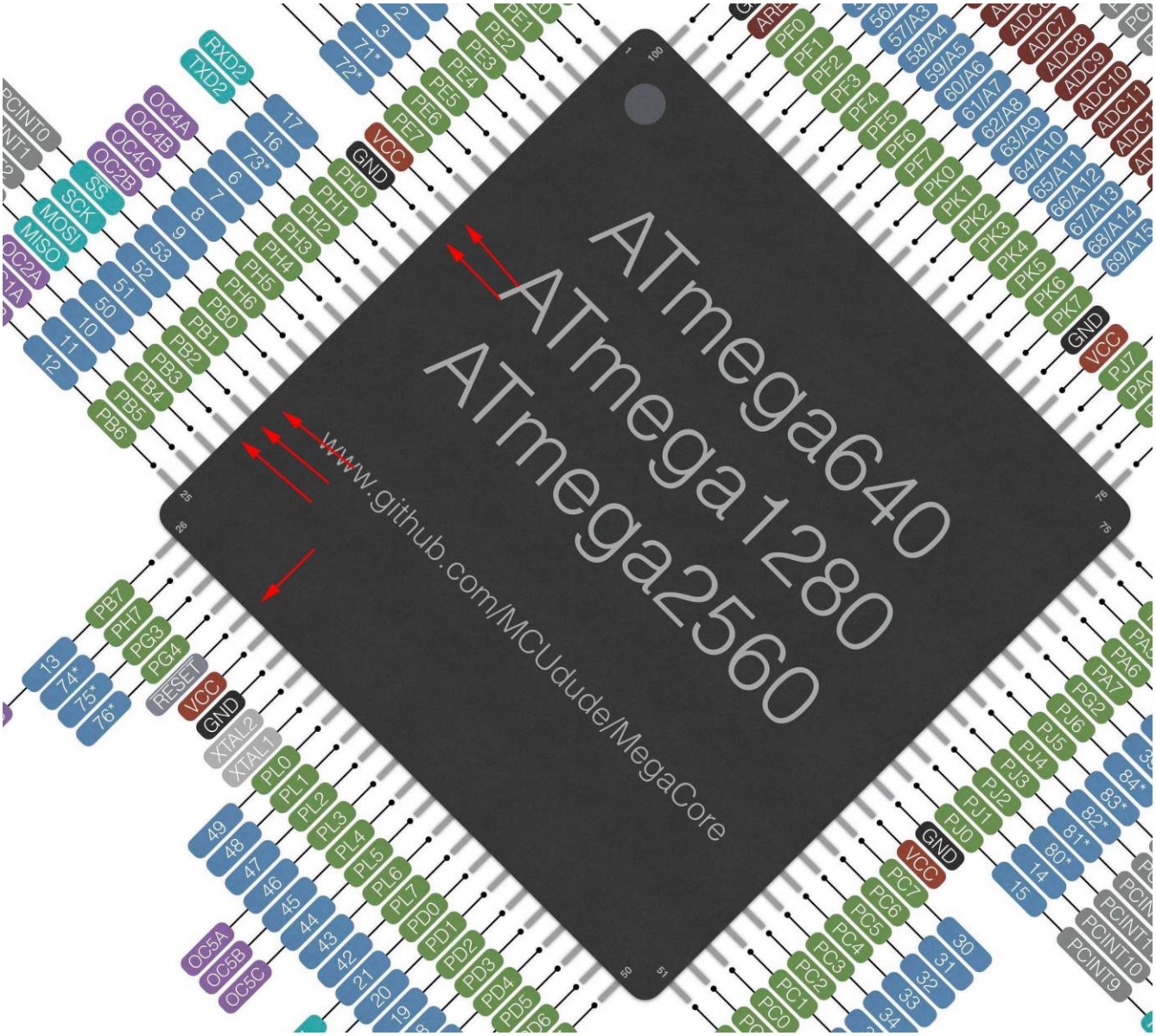


Abbildung 8.2: Pinbelegung ATmega2560

Bitte auch testen, ob ggf. eine Verbindung zu einem benachbarten Pin besteht. Ist alles i.O. und ist immer noch keine Kommunikation möglich, könnte der Chip defekt sein. Wurde der Chip beim Einlöten ggf. zu stark erhitzt? Stammte der Chip aus einer verlässlichen Quelle (bei Händlern in China ist etwas Vorsicht angebracht)?

8.2.5 „Hilfe! Ich kann den ATmega2560 nicht mehr programmieren!“

Konnte der ATmega2560 beim Programmieren der Fuses oder beim Programmieren der Firmware angesprochen werden, dabei trat aber ein Fehler auf und jetzt lässt sich der Chip nicht mehr ansprechen?

Grundsätzlich sind zwei Arten von Fehler möglich:

1. Durch das Setzen der Fuses wird vom internen 1MHz Takt auf den externen Quarz umgeschaltet. Kann direkt danach der ATmega2560 nicht mehr angesprochen werden, kann der Quarz oder einer der 22pF Kondensatoren defekt sein.
2. Wurde nach dem Setzen der Fuses damit begonnen die Firmware zu flashen und trat dabei ein Fehler auf und ist jetzt keine Kommunikation mit dem ATmega2560 mehr möglich, können die Fuses in Mitleidenschaft gezogen worden sein.

Wenn folgende Fehlermeldung erscheint: Erst einmal keine Panik! Es ist fast unmöglich sich aus seinem ATmega2560 softwaremäßig auszusperren.

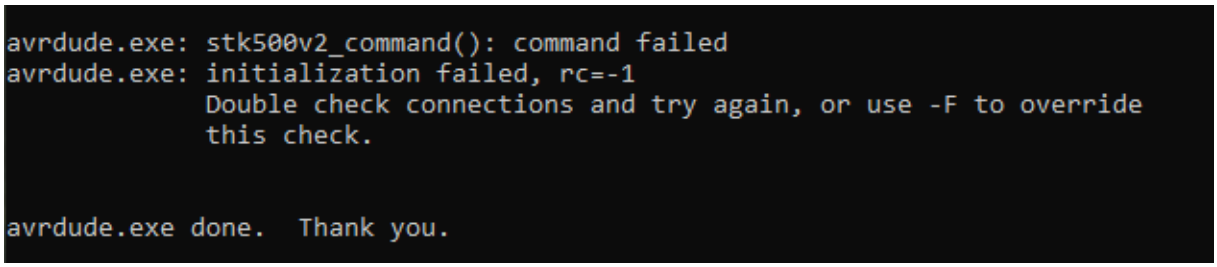


Abbildung 8.3: AVRDUDE – Keine Kommunikation mit dem ATmega2560

Vermutlich sind die Fuses in einem Zustand, dass der externe Quarz nicht verwendet wird und auch die internen 1 MHz nicht zur Verfügung stehen (z. B. wenn alle Fuses gelöscht wurden und ein externer Takt benötigt wird), d.h. dem Chip fehlt die Taktquelle.

Man kann die Fuses recht einfach neu setzen: Es muss nur eine Taktquelle an Pin 34 (XTAL1) des ATmega2560 angelegt werden. Das hört sich komplizierter an, als es ist.

Zunächst benötigt man eine Taktquelle (wer einen Diamex-Programmierer oder einen anderen Programmierer besitzt, der einen Takt unterstützt, springt bitte ans Ende dieses Kapitels). Hier kann man einen Arduino Uno/Nano oder Mega verwenden, den man mit folgendem Programm programmiert:

```
#include <avr/io.h>

void setup() { }

void loop() {
    DDRB = 0xFF;
    while(1) {
        PORTB ^= 0xFF;
    }
}
```

Dieses Programm erzeugt einen Takt (1.59 MHz) an Pins 8 bis 13 beim Arduino UNO/NANO und Pins 10-13, 50-52 beim Arduino MEGA. Wird dieses Programm auf den Arduino geladen, leuchtet die an Pin 13 angeschlossene LED nur noch ganz schwach durch die ständigen Ein-/Ausschaltvorgänge.

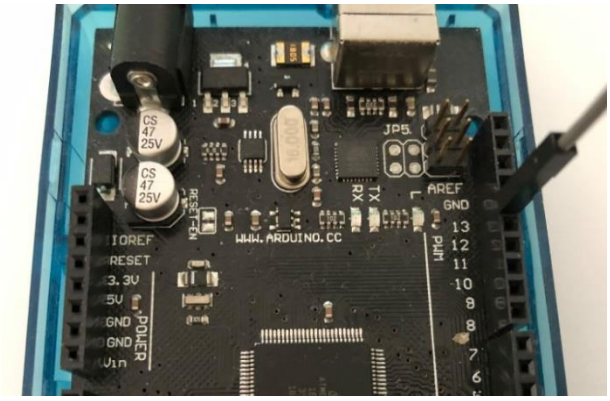


Abbildung 8.4: Pin 13 beim Arduino Mega

Jetzt schließt man Pin 13 an XTAL1 an. Hier bietet sich das rechte Beinchen des 1 MOhm Widerstands unterhalb des Quarzes an. Der Anschluss darf keine Wackler aufweisen und nicht mit dem Quarz in Kontakt kommen. Nicht vergessen GND (Masse) mit GND des Testers zu verbinden (i.d.R, klappt es auch so, wenn beide Geräte am selben Computer betrieben werden).

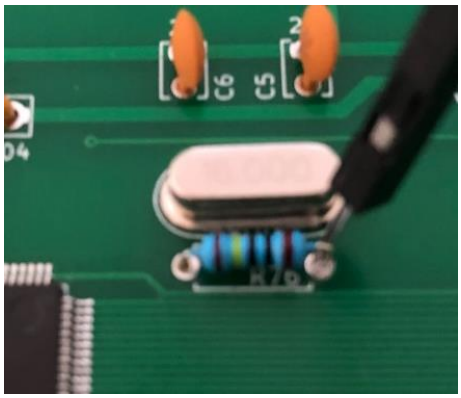


Abbildung 8.5: XTAL1

I.d.R. muss der Quarz nicht ausgelötet werden, da der 5V Takt des Arduino sehr viel stärker ist als das schwache Signal des Quarzes. Es sollte so aber nicht die Firmware vollständig programmiert werden, sondern nur „schnell“ die Fuses korrekt programmiert werden, damit der Programmiervorgang wieder ohne den externen Takt (mit dem vorhandenen Quarz) stattfinden kann.

Ist diese Verbindung hergestellt und der ISP-Programmierer angeschlossen, kann jetzt noch einmal probiert werden die Fuses zu setzen (der Programmierer und Port muss entsprechend angepasst werden).

```
avrdude.exe -C"avrdude.conf" -v -patmega2560 -cstk500 -PCOM5
-U lfuse:w:0xf7:m -U hfuse:w:0xd7:m -U efuse:w:0xff:m
```

Dieser Vorgang klappt nicht unbedingt beim ersten Aufruf, ggf. mehrmals wiederholen. Ggf. muss der Quarz auch entfernt werden.

Konnten die Fuses jetzt korrekt gesetzt werden, kann die Firmware programmiert werden.

Programmierer mit Takt-Unterstützung (z.B. Diamex)

Wenn der verwendete ISP-Programmierer einen Takt-Ausgang anbietet, so kann dieser sehr einfach benutzt werden.

Beim Diamex einfach den Parameter `-xfosc=1MHz` beim Setzen der Fuses zusätzlich mit angeben und den Takt von Pin 3 (OSC) des 10-poligen Steckers mit XTAL1 (siehe oben) verbinden.

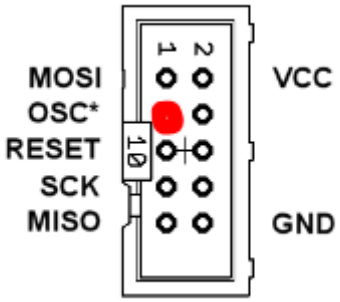


Abbildung 8.6: Takt am 10-poligen ISP-Connector

Die Kommendozeile lautet somit:

```
avrdude.exe -C"avrdude.conf" -xfosc=1MHz -B0.5 -patmega2560 -cstk500
-PCOM5 -U lfuse:w:0xf7:m -U hfuse:w:0xd7:m -U efuse:w:0xff:m
```

8.2.6 Verwendung von COM10 und höher (Windows)

Wenn AVRDUDE an COM10 und höher betrieben werden soll, muss die Schnittstelle ggf. wie folgt angegeben werden:

Zum Beispiel für COM12:

```
-P \\.\com12
```

8.3 Probleme mit dem Display

8.3.1 Es erscheint kein Text auf dem Display

Ist der Kontrast korrekt eingestellt? Mit dem Potentiometer kann dieser verändert werden.

8.3.2 Das Display ist ziemlich dunkel

Auf der Platine befindet sich neben dem Display ein 220 Ohm Widerstand. Der sorgt dafür, dass auch Displays, die über keinen Vorwiderstand für das Backlight verfügen, sicher betrieben werden kann (i.d.R. sollten auch diese Displays mit 100 Ohm auskommen). Bei allen anderen Displays kann der Widerstand komplett entfallen. Auf der Rückseite ist leicht zu erkennen, ob ein entsprechender Vorwiderstand vorhanden ist.

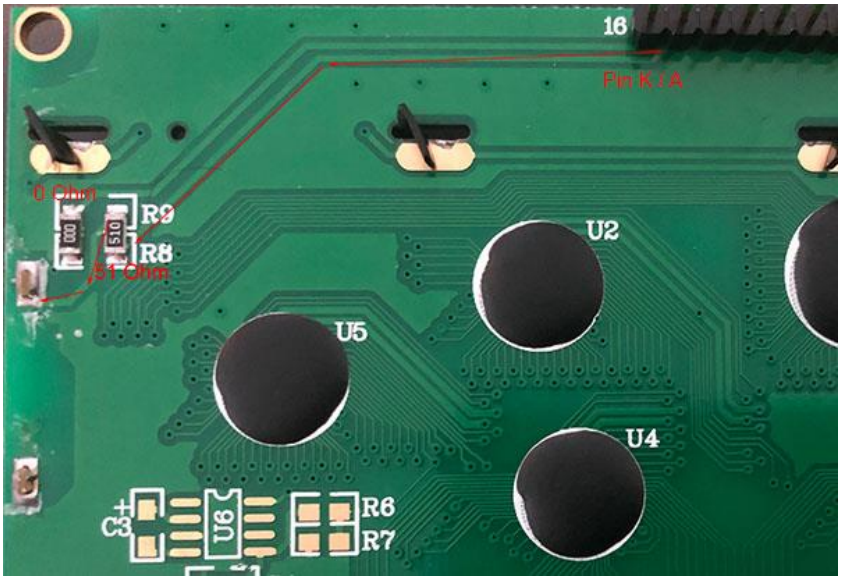


Abbildung 8.7: 51 Ohm Vorwiderstand zum Backlight

Sollte ein Vorwiderstand vorhanden sein, kann der 220 Ohm Widerstand entfallen und durch eine Drahtbrücke oder 0 Ohm Widerstand ersetzt werden.

8.3.3 Die Anzeige eines OLED ist ziemlich dunkel

Wurde ein OLED verbaut und ist die Anzeige ungewöhnlich dunkel, dann kann es daran liegen, dass die Pins 15/16 nicht - wie im Datenblatt beschrieben - nicht angeschlossen (also „nc“) sind. Normalerweise sollten die Pins 15/16 bei einem OLED nicht angeschlossen sein, da diese normalerweise für die Hintergrundbeleuchtung verwendet werden:

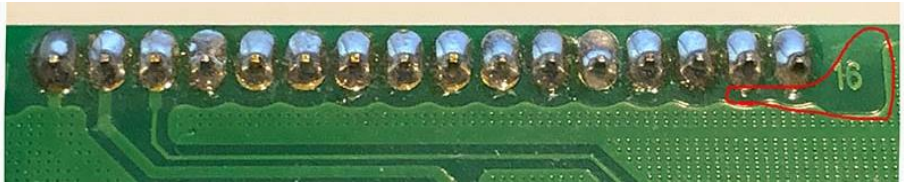


Abbildung 8.8: Pins 15/16 nicht angeschlossen

Bei einem Winstar OLED werden diese Pins aber verwendet, obwohl diese im Datenblatt als „nc“ gekennzeichnet sind:

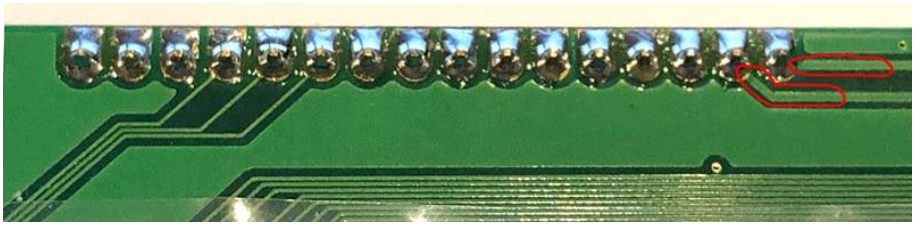


Abbildung 8.9: Pins 15/16 sind angeschlossen

In diesem Fall die beiden Pins einfach kürzen, so dass diese keinen Kontakt mehr zum Sockel haben. Das Display sollte dann den Text mit maximalem Kontrast darstellen.

8.3.4 Die Anzeige eines OLED ist nicht korrekt (1)

Leider sind nicht alle OLEDs voll kompatibel zum HD44780 Controller. Die meisten(?) Displays scheinen problemlos zu funktionieren, es gibt aber auch OLEDs, die eine fehlerhafte Anzeige produzieren.

Normalerweise sollte die Einschaltmeldung wie folgt aussehen:

Abbildung 8.10: Anzeige in Ordnung

Nach mehreren Resets können plötzlich vertauschte Zeilen auftreten:

Abbildung 8.11: vertauschte Zeilen

Selten kann das Display nach einem Reset auch eine fehlerhafte Darstellung liefern:

Abbildung 8.12: fehlerhafte Darstellung

Der Fehler tritt u.a. bei OLEDs von Winstar und Klonen (z.B. Newhaven) auf (allerdings nicht bei allen, vermutlich von der Platinenrevision abhängig).

Um das Display fehlerfrei betreiben zu können ist ein kleiner Eingriff notwendig. Auf der Rückseite des OLEDs wird ein dünnes Drähtchen mit der Reset-Leitung des Controllers verbunden. Der linke Anschluss des 10k Widerstands eignet sich dazu den Draht anzulöten.

Abbildung 8.13: Rückseite des OLEDs

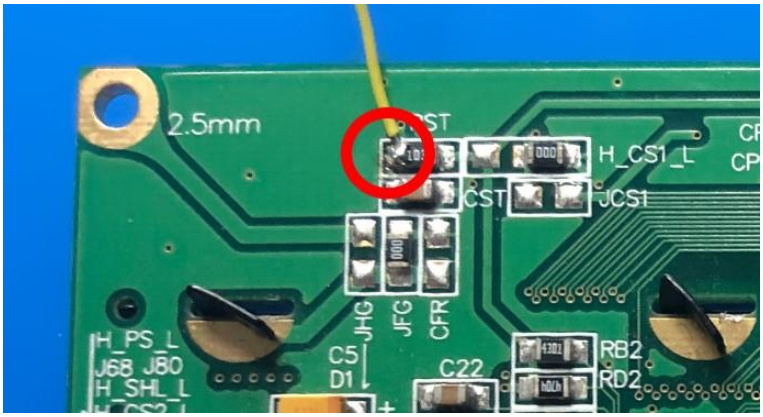


Abbildung 8.14: linke Seite des 10k Widerstands

Der Draht wird dann mit der Reset-Leitung des Testers verbunden. Hierzu eignet sich der ISP-Anschluss (von vorne gesehen, der linke untere Anschluss; von der Rückseite, der rechte untere Anschluss).

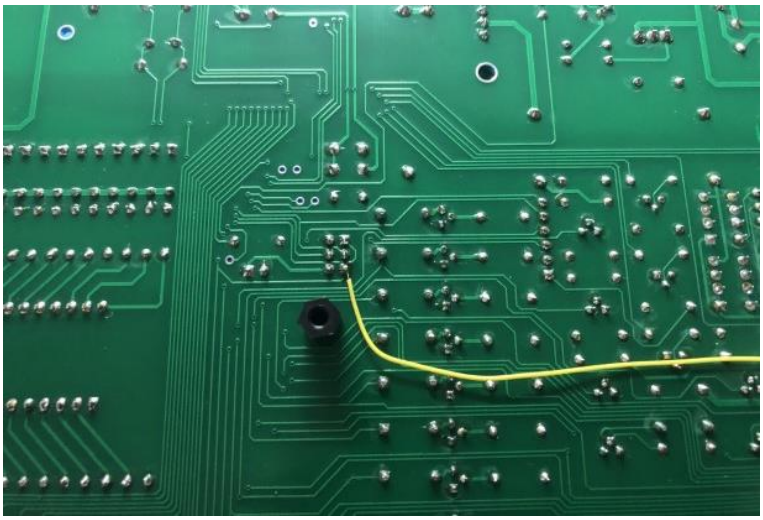


Abbildung 8.15: Reset am ISP-Anschluss

Nach dieser Modifikation sollten nach einem Reset keine vertauschten Zeilen oder ein fehlerhafter Bildschirm mehr auftreten.

8.3.5 Die Anzeige eines OLED ist nicht korrekt (2)

Damit ein Winstar 4x20 OLED-Display am RCT funktioniert, muss die Platine durch Brücken, wie auf dem Bild gezeigt, konfiguriert sein. Bei der Lieferung kann es in seltenen Fällen vorkommen, dass diese anders konfiguriert sind. In diesem Fall bleibt das Display dunkel bzw. es erscheint nur Zeichensalat.

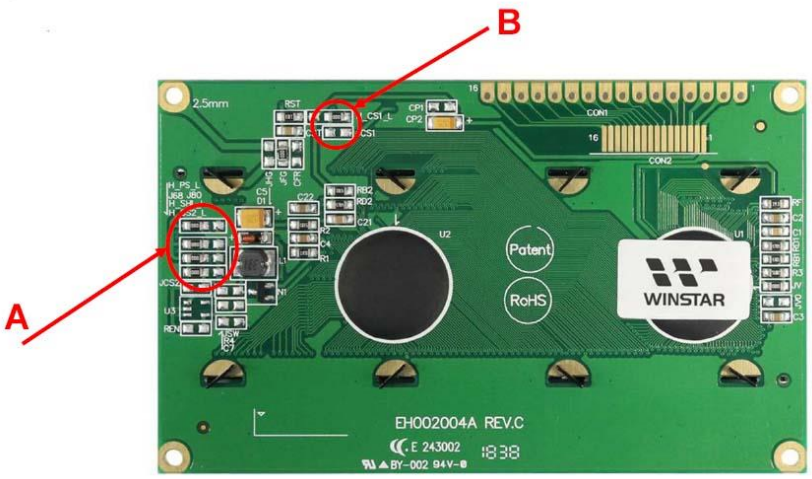


Abbildung 8.16: korrekt gesetzte Jumper

Die Brücken müssen bei diesem Display-Typ wie folgt gesetzt sein:

A: Alle vier Brücken auf der linken Seite.

B: Brücke auf H_CS1_L; CS1 keine Brücke

8.4 Probleme mit dem DC/DC Modul

8.4.1 Die grünen LEDs des DC/DC Moduls leuchten nicht, ggf. ist das Display dunkel

Fehler auf dem DC/DC-Modul

Es könnte sein, dass das DC/DC-Modul die Versorgungsspannung nicht liefert. Bitte das DC/DC-Modul vom Tester entfernen und die Pins „Vss“ und „Barrel Jack“ auf der linken Seite mit 5V bis 9V verbinden (Vss = Minus, Barrel Jack = Plus). An der rechten Seite sollten die Spannungen 5V (Vss/Vcc), -5V (Vss/Vbb) und 12V (Vss/Vdd) zu messen sein. Auch sollten alle drei LEDs leuchten. Leuchten die LEDs nicht bzw. fehlt eine Spannung, dann ist ein Bauteil auf dem DC/DC-Modul defekt. Leuchten die LEDs bzw. sind die Spannungen vorhanden, dann könnte ein Kurzschluss vorliegen.

Fehler auf der Hauptplatine

Es könnte ein Kurzschluss auf dem Board vorliegen. Um einen möglichen Kurzschluss einzugrenzen, bitte den Widerstand jeweils zwischen Vss/Vcc, Vss/Vbb und Vss/Vdd messen (am besten an der Pfostenleiste J4 unterhalb des Potentiometers). Sollte ein Wert nahe 0 Ohm angezeigt werden, liegt das Problem vermutlich in einem den folgenden Bereichen:

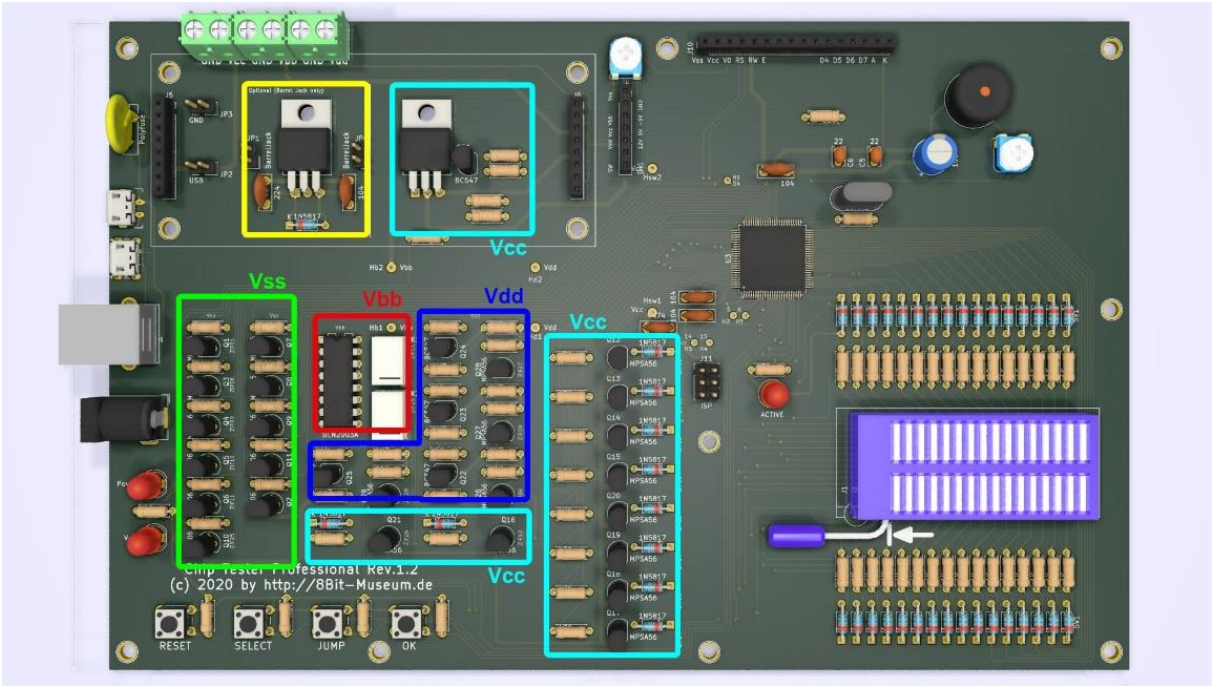


Abbildung 8.17: Aufbau der Platine

Bitte insbesondere die Transistoren auf Löt-Kurzschlüsse untersuchen. Auch andere nahe beieinanderliegende Pads können bei zu viel Lötzinn einen Kurzschluss verursachen.

Es ist auch schon vorgekommen, dass einer der Kondensatoren fehlerhaft war und einen internen Kurzschluss hatte (hier bitte den Widerstand der Kondensatoren messen, ein fehlerhafter Kondensator hat sehr Wahrscheinlich einen Widerstand nahe 0 Ohm).

8.5 Weitere Fehler

8.5.1 Die -5V (Vbb) liegen nicht an, am Netzteil sind die -5V messbar

Klacken die Relais, wenn z.B. der Test für 4116er ICs ausgewählt wird?

Sind die Relais korrekt herum eingelötet? Sitzt der ULN2003 korrekt herum? Ggf. diesen einmal mit dem Tester prüfen (Misc Logic Chips) bzw. direkt tauschen.

8.5.2 Tests schlagen sehr oft fehl

Wenn der Tester per USB mit Spannung versorgt wird, diese bitte mit einem Multimeter einmal prüfen. Hierzu kann die Spannung „Vcc“ an der Pfostenleiste zwischen DC/DC-Modul und Display abgegriffen werden.

Die Spannung „Vcc“ sollte im Idealfall 5V betragen. Bis 4,8V sind gerade noch akzeptabel, Spannungen unter 4.8V können zu Problemen führen. In diesem Fall einmal ein anderes USB-Netzteil ausprobieren oder den Tester über den Hohlstecker mit Spannung versorgen (7-9V).

Falls die Versorgungsspannung i.O. scheint (und auch eine Anzeige auf dem Display erscheint), kann es praktisch jedes der über 200 Bauelemente sein, welches einen Fehler produzieren kann.

1. Die Zener-Dioden schützen den ATmega vor zu hohen Spannungen. Eine defekte Zener-Diode kann den Signalpegel schwächen oder einen Kurzschluss nach GND erzeugen.
2. Widerstände sind nur selten defekt, was aber durchaus vorkommt. Unterbrechungen sind schnell mit einem Ohmmeter zu finden.
3. Sollte ein falscher Widerstandswert eingebaut worden sein, kann ggf. eine Versorgungsspannung nicht mehr korrekt geschaltet werden. Z.B. ein 4.7k Ohm Widerstand als Basiswiderstand anstelle der 470 Ohm führt dazu, dass der Transistor nicht mehr voll durchschaltet und der Spannungsabfall Uce zu hoch ist. Gleiches gilt für falsche Werte bei den Pullup/Pulldown Widerständen.
4. Ein falscher Widerstandswert von z.B. 4.7k Ohm anstelle der 4.7 Ohm für R57 (Gate Widerstand am MOSFET) führt dazu, dass Vcc komplett ausfällt oder bei geringer Belastung zusammenbricht.
5. Ein defektes Subminiatur-Relais erkennt man meistens am fehlenden Klicken beim Testen eines 4116 (Relais #1) bzw. 2708 (Relais #2) oder Dauerklicken eines Relais. Letzteres kann auch an einem defekten ULN2003A liegen.
6. Schwieriger zu finden sind kalte Lötstellen. Matte Lötstellen ggf. kurz Nachlöten, um einen hochohmigen Übergang auszuschließen.
7. Wurde im Bereich der Transistoren mit zu viel Lötzinn gearbeitet, können zwischen den Pins leicht Kurzschlüsse entstehen. Diese sind leicht mit der Lupe zu kontrollieren.
8. Der Quarz ist auch ein häufiger Fehler. Mehr dazu in Abschnitt 8.5.3.
9. Ein defekter Kondensator führt häufig zu einem Kurzschluss zwischen Vcc und GND.
10. Am schwierigsten sind defekte Transistoren zu identifizieren. Sollten z.B. SRAMs und viele DRAMs ohne Probleme testbar sein, ein spezieller DRAM Typ aber nicht, kann das darauf hindeuten, dass ein Transistor, der verantwortlich für Vcc, Vss oder Vdd ist, nicht korrekt schaltet. Hier sollte geprüft werden, in welchen Pins sich der problematische Speichertyp unterscheidet.

Sollte aber nur ein Baustein eines speziellen Typs immer fehlschlagen (z.B. ein uPD416 kann nicht getestet werden, ein TMS4116 wird aber problemlos getestet), sollte davon ausgegangen werden, dass dieser Baustein (oder die gesamte Charge) ggf. aufgrund von Alterungseffekten nicht mehr getestet werden kann.

8.5.3 Bei einem Test tritt ein Reset auf oder die Firmware verhält sich merkwürdig

Sollte beim Testen ein plötzlicher Reset auftreten, IC-Tests wiederholt fehlschlagen oder die Firmware „merkwürdig“ verhalten, kommen mehrere Fehler in Frage:

1. Probleme mit dem Takt

Symptome: Plötzlich auftretende Resets, Firmware „springt“ an andere Programmstellen (Tests brechen nicht ab, Anzeige zeigt ungewöhnliche Ausgaben)

Fehlerursache: Vermutlich liefert der Quarz einen zu geringen Spannungshub.

Fehlerbehebung: Anstelle des "Low Power Crystal Oscillator", sollte der "Full Swing Crystal Oscillator" des ATmega probeweise eingestellt werden (siehe Abs. 4.1). Sollte das nicht funktionieren, sollte der Quarz und die zwei 22p Kondensatoren gewechselt werden.

2. Zu geringe Spannungsversorgung

Symptome: Ein bestimmter IC lässt sich nicht testen (möglich bei Speichertests und Logik-Tests auf).

Fehlerursache: Der Fehler kann an einer zu niedrigen / fehlerhaften Versorgungsspannung liegen, wenn der Fehler immer bei einem bestimmten IC-Typ auftritt, z.B. einem 14pol. Logik-Chip oder einem 32pol. Speicherchip.

Fehlerbehebung: Zunächst sollte der Selbsttest ausgeführt werden. Treten hier Fehler auf, möglicherwiese immer an demselben Pin, ist sehr wahrscheinlich ein Bauteil defekt oder es liegt eine kalte Lötstelle vor. Der Fehler kann sowohl auf der Hauptplatine vorliegen als auch auf dem DC/DC Modul. Bei einem Kurzschluss sinkt die Spannungsversorgung des Prozessors so stark ab, dass dieser einen Reset durchführt oder nicht mehr korrekt arbeitet.

Tritt der Fehler an verschiedenen Pins auf, ist vermutlich die Versorgungsspannung nicht verlässlich. Hier könnte der Fehler dann auf dem DC/DC-Modul liegen oder, sollte hauptsächlich Vcc betroffen sein, dann am MOSFET oder die umliegenden Bauelemente (Q30, Q31, R54-R57).

3. Andere Fehler

Symptome: Ein spezieller Speicher oder Logik-IC kann nicht getestet werden.

Fehlerbehebung: Der Speicher oder Logik-IC kann natürlich defekt sein. Tritt der Fehler aber bei mehreren ICs desselben Typs auf, bitte prüfen, ob der IC besondere technische Anforderungen hat. Dazu gehören: Spannungsversorgung (z.B. bei NVRAMs), benötigter Strom an Eingängen bei bipolaren ICs (z.B. Standard-TTLs).

8.6 Selbsttest

Über das Menü kann ein einfacher Selbsttest aufgerufen werden, der die Versorgungsspannungen Vcc, Vdd und Vss prüft. Vbb ist aus technischen Gründen nicht testbar.

Die Anzeige zeigt den Zustand der einzelnen ZIF-Pins, Ein „X“ bedeutet, dass die Versorgungsspannung korrekt an diesem Pin geschaltet werden konnte. Eine „1“ bedeutet, dass die Spannung (Vcc, Vdd, Vss) dauerhaft anliegt, eine „0“, dass keine Spannung vorhanden ist.

Abbildung 8.18: Test von Vcc, Interpretation der Anzeige

Wenn der Tester ohne DC/DC-Modul verwendet wird, schlägt der Selbsttest immer fehl (Vcc und Vss sollten "X" anzeigen, Vdd sollte "0" anzeigen). Also keine Panik.

Es darf kein IC eingesetzt sein. Dieser würde das Testergebnis verfälschen und ggf. Schaden nehmen.

Es darf kein Entkopplungskondensators aktiviert sein (es darf rechts vom ZIF-Sockel kein Jumper gesetzt sein).

8.6.1 Es wird bei allen Spannungen (Vcc, Vdd) eine „0“ angezeigt

Sind die drei Drahtbrücken unterhalb des DC/DC-Moduls eingelötet?

8.6.2 Es werden bei 5V (Vcc) nur 0-en angezeigt

Bitte prüfen, ob für R57 auch 4.7 Ohm eingelötet wurde. Ggf. wurde hier 4.7k Ohm eingesetzt. Es kann auch der MOSFET (IRF5305) defekt sein.

8.6.3 Es wird bei Vcc/Vdd/Vss eine einzelne „1“ oder „0“ angezeigt

Der Selbsttest läuft wie folgt ab:

Es wird Vcc/Vdd eingeschaltet
Es sollte jetzt ein logisch H gelesen werden können → Status A

Es wird Vcc/Vdd ausgeschaltet
Es sollte jetzt ein logisch L gelesen werden können → Status B

Wenn

- A=HIGH und B=LOW → Test OK, Ausgabe “X”
- A=HIGH und B=HIGH → Test Fail, Ausgabe “1” (Signal bleibt High)
- A=LOW und B=LOW → Test Fail, Ausgabe “0” (Signal bleibt Low)
- A=LOW und B=HIGH → Test Fail, Ausgabe “-”

Dasselbe gilt auch für Vss bis zur Firmware v.15 wurde eine „1“ ausgegeben, wenn Vss geschaltet werden konnte (also ein L gelesen wurde), dieses aber nicht „ausgeschaltet“ werden konnte (also weiterhin ein L gelesen wurde). Ab der Firmware v.16 ist die Ausgabe an die Signalpegel angepasst worden, d.h. es wird eine „1“ ausgegeben, wenn trotz Aktivierung von Vss ein logisch H gelesen wurde.

Wenn beim Testen von Vcc/Vdd eine „0“ ausgegeben wird, kann auch die zugehörige Zener-Diode defekt sein (Kurzschluss).

8.6.4 Es werden mehrere Fehler beim Vcc und Vdd Selbsttest angezeigt

Sollten beim Selbsttest von Vcc und Vdd mehrere Fehler auftreten, dann sind vermutlich die Fuses nicht korrekt gesetzt. Das Fehlerbild äußert sich in folgender Ausgabe (kann auch leicht unterschiedlich sein):

Abbildung 8.19: Fehler beim Test von Vcc

Sind die Fuses nicht korrekt gesetzt, kann u.U. das JTAG-Interface der MPU aktiviert sein. In diesem Fall sind die Pullup-Widerstände für die Pins PF7 (TDI, Pin 26), PF5 (TMS, Pin 9) und PF4 (TCK, Pin 12) aktiviert, die so auch einen Reset überstehen.

Einfach die Fuses korrekt setzen und der Fehler sollte behoben sein.

8.6.5 Funktionsweise

Vss wird durch NPN-Transistoren geschaltet. Zeigt der Selbsttest an, dass Vss nicht geschaltet werden kann, kann das Problem an einem defekten Transistor (MPSA06) liegen. Es ist aber ebenso eine kalte Lötstelle oder ein defekter Widerstand möglich. Der zum ZIF-Pin zugehörige Transistor ist auf der Platine vermerkt.

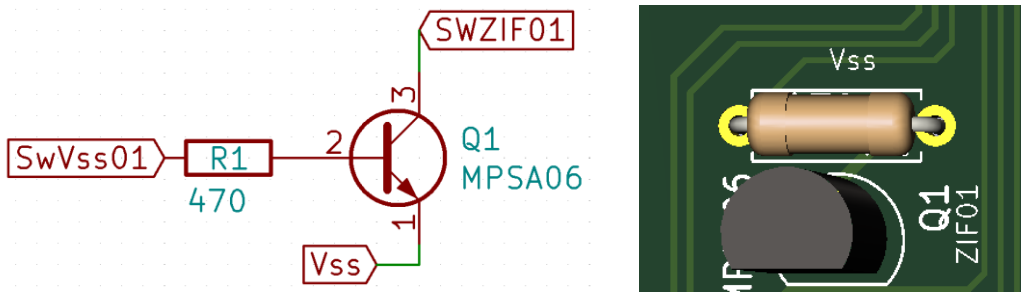


Abbildung 8.20: Schalten von Vss, Identifikation auf der Platine

Vcc wird durch die PNP-Transistoren (MPSA56) in der Mitte der Platine geschaltet. Kann Vcc nicht geschaltet werden, könnte der Transistor defekt sein. Liegt Vcc überhaupt nicht an, kann die Diode defekt sein oder es kann eine kalte Lötstelle vorliegen.

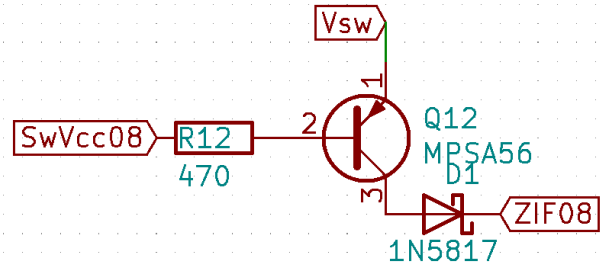


Abbildung 8.21: Schalten von Vcc

Vdd wird durch die PNP-Transistoren (MPSA56) im Zusammenspiel mit den NPN-Transistoren (BC547) im ersten Drittel der Platine geschaltet. Kann Vdd nicht geschaltet werden, könnte ein Transistor defekt sein. Das Problem kann aber auch an einem falschen oder defekten Widerstand liegen oder es kann eine kalte Lötstelle vorliegen.

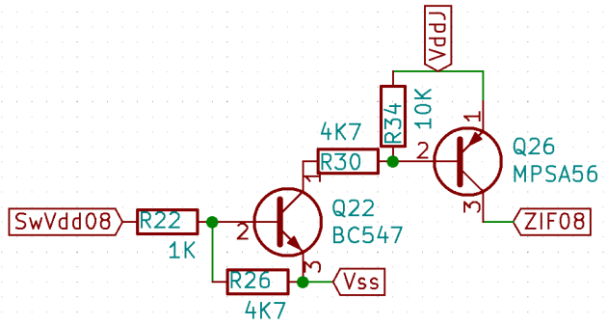
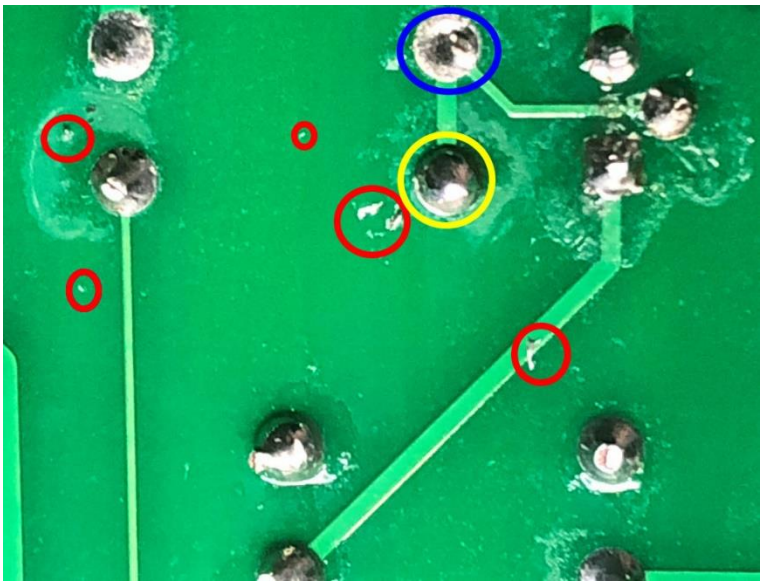


Abbildung 8.22: Schalten von Vdd

8.6.6 Schlechte Lötung

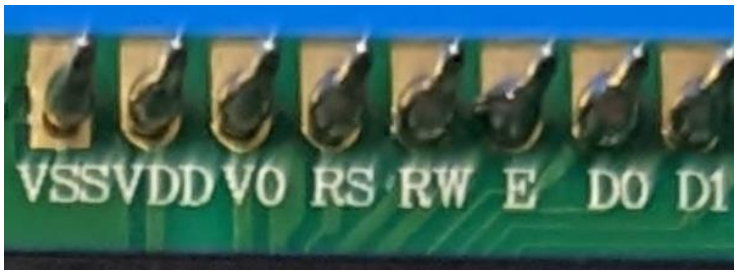
Bitte unbedingt darauf achten, dass nicht zu viel, aber auch nicht zu wenig Lötzinn verwendet wird. Zudem können bei schlechtem Lot und/oder zu heißer Lötspitze Spritzer von Lötzinn entstehen, die Kurzschlüsse erzeugen können.

Die folgenden Bilder zeigen ein paar Beispiele:

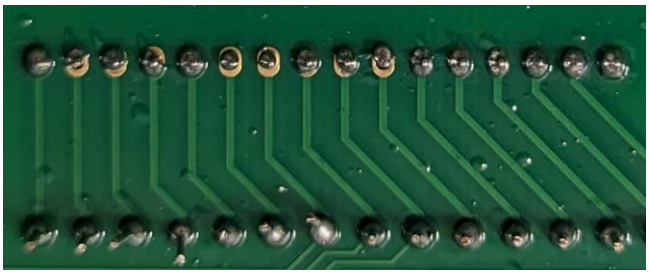


rot: Spritzer von Lot, die Kurzschlüsse erzeugen können

gelb: zu viel Lötzinn (Kugel aus Lötzinn); blau: zu wenig Lötzinn

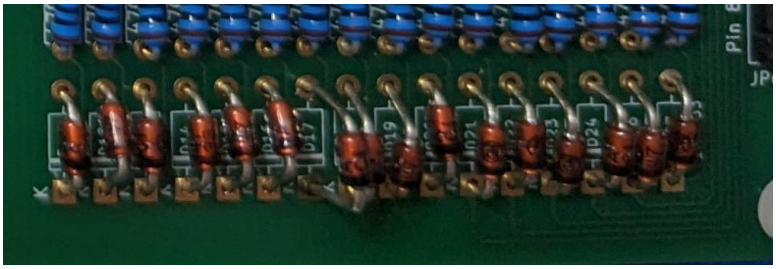


zu viel Lötzinn (Kugel aus Lötzinn), kein Kontakt

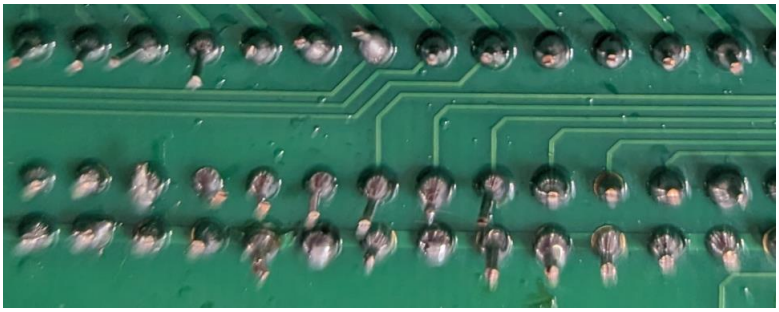


zu wenig Lötzinn, kein Kontakt

Sollte die Lötverbindung „matt“ aussehen, könnte ebenfalls eine kalte Lötstelle (also ohne Kontakt) vorliegen (im Bild oben links).



Nicht weit genug durchgesteckt, Kurzschluss möglich



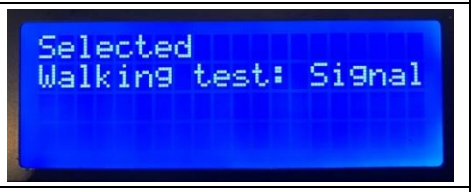
Pins nicht gut gekürzt, Kurzschluss möglich

Platine nicht gesäubert, Kurzschlüsse möglich

8.7 Diagnose-Software

In dem Archiv `upload-tester-pro-test.zip` befindet sich eine (einfache) Diagnose-Software. Die Firmware wird, wie in Abschnitt 4 beschrieben, programmiert.

Mit SELECT und OK kann ein Test in dem Menü ausgewählt werden. Mit JUMP wird ein Test i.d.R. beendet.

Menüpunkt	Beschreibung	Bild
Selftest Vcc,Vdd,Vss	Der Tester testet selbst die Versorgungsspannungen Vcc, Vdd und Vss durch. Vbb ist aus technischen Gründen nicht testbar.	
Selftest LOW/HIGH	Es wird getestet, ob ohne Pullups ein LOW gelesen wird und mit Pullups ein HIGH.	
Walking test: Signal	Dieser Test legt nacheinander ein Signal an jedem einzelnen Pin des ZIF32-Sockels an. Mit SELECT wird ein Pin weitergeschaltet.	
Walking test: Vcc	Dieser Test legt nacheinander die Versorgungsspannung Vcc an den möglichen Pins des ZIF32-Sockels an. Mit SELECT wird ein Pin weitergeschaltet.	
Walking test: Vss	wie oben, nur für Vss	
Walking test: Vdd	wie oben, nur für Vdd	
Walking test: Vbb	wie oben, nur für Vbb	
All ON: Vcc	Dieser Test schaltet alle möglichen Pins des ZIF32-Sockels gleichzeitig auf Vcc.	
All ON: Vss	wie oben, nur für Vss	
All ON: Vdd	wie oben, nur für Vdd	
All ON: Vbb	wie oben, nur für Vbb	
Input Test: Low	Wird dieser Test gestartet, kann ein beliebiger Pin mit Vss verbunden werden. Der Pin wird im Display angezeigt.	
Input Test: High	Wird dieser Test gestartet, kann ein beliebiger Pin mit Vcc verbunden werden. Der Pin wird im Display angezeigt.	

Menüpunkt	Beschreibung	Bild
Benchmark	Dieser Test führt einen einfachen Benchmark durch. Hiermit kann erkannt werden, ob der Takt des ATmega2560 korrekt gesetzt ist.	

Um die Signale schnell zu testen, kann ein Logik-Tester verwendet werden (Vorsicht bei Vdd: 12V). Wer keinen Logik-Tester besitzt, kann sich schnell einen einfachen Tester selbst bauen:

Benötigt wird nur eine LED und ein 1k Ohm Widerstand. Der Widerstand wird direkt an die LED gelötet (Kathode, kurzer Pin bzw. abgeflachte LED-Seite).

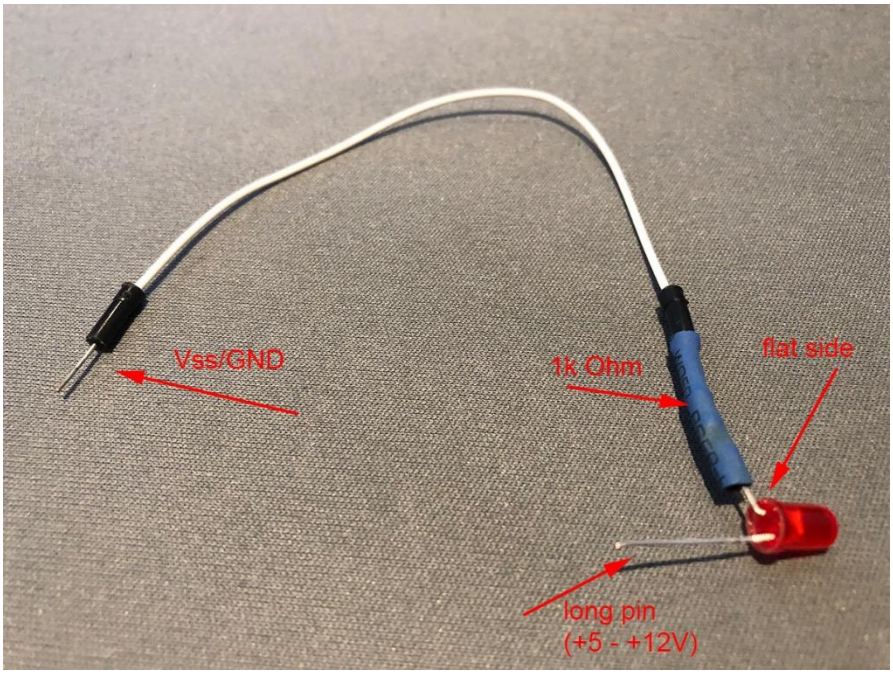
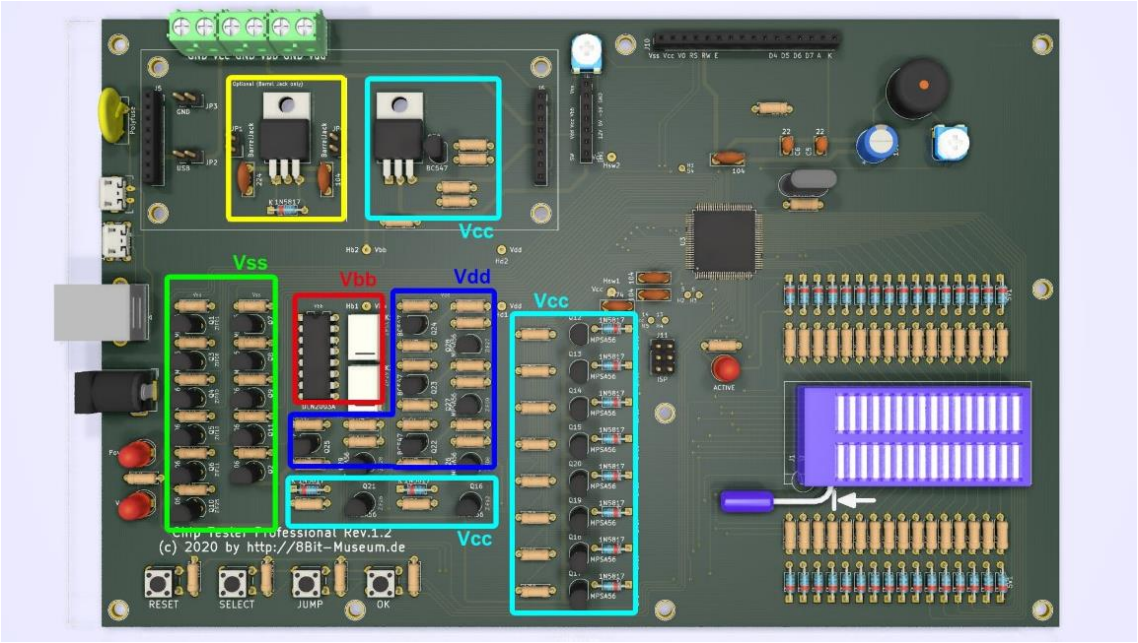


Abbildung 8.23: Einfacher Logik-Tester

Zum Testen eines positiven Signals den Stecker in die Pfostenleiste „Vss“ stecken und den langen Anschluss der LED (die Anode) an einen Pin des ZIF32-Sockels halten. Mit diesem „Tester“ kann auch das Vorhandensein von Vdd geprüft werden.

Zum Testen von Vbb (-5V) den langen Anschluss der LED (Anode) in die Pfostenleiste „Vss“ stecken und mit dem Pin die Spannung testen.

Die zuständigen Transistoren sind mit der Pin-Nummer des ZIF-Sockels beschriftet. Dabei sind folgende Bereiche für Vss, Vbb, Vcc und Vdd zuständig:



8.8 Still-Alive Modus

Die Firmware (ab v25) beinhaltet eine kleine Testroutine mit der geprüft werden kann, ob der Prozessor funktioniert. Das ist sinnvoll, wenn nach dem Zusammenbau keine Reaktion erfolgt und die beschriebenen Maßnahmen zur Fehlersuche nicht funktionierten.

Für den Still-Alive Modus wird nur vorausgesetzt, dass der Takt funktioniert, also zumindest der 16 MHz Quarz und die zwei 22pF Kondensatoren müssen in Ordnung sein.

Sollten die Fuses nicht korrekt gesetzt ein oder der Quarz/Kondensatoren nicht bestückt sein, kann auch ein externer Takt eingespeist werden (z.B. über den Oszillator-Pin des Programmierers).

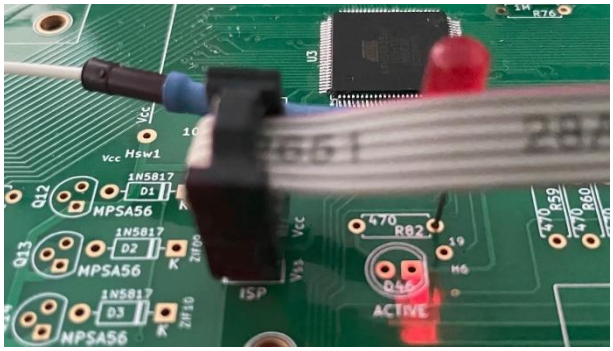
Ist ein Takt vorhanden und das Board bereits bestückt, ist nichts weiter zu beachten. Der Modus wird aufgerufen, wenn nach dem Anlegen der Versorgungsspannung der Taster „Select“ für mindestens fünf Sekunden gedrückt gehalten wird.

Der Still-Alive Modus:

- füllt das Display (falls angeschlossen) vollständig mit „*“,
- gibt jede Sekunde einen kurzen Signalton aus und
- lässt die Active-LED viermal in der Sekunde blinken.

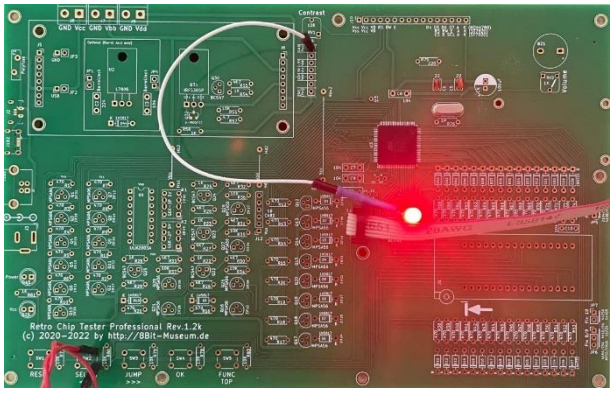
Ist das Board noch unbestückt, dann kann wie folgt eine Active-LED angeschlossen werden:

- diese zusammen mit R82 bestücken oder
- eine LED mit einem Vorwiderstand (siehe vorheriges Kapitel) wie im folgenden Bild gezeigt anstecken (Vss und rechter Kontakt von R82).



Anstelle des Select-Tasters kann eine Drahtbrücke verwendet werden. Die Versorgungsspannung kann über den ISP-Programmierer erfolgen (i.d.R. ist diese dort dauerhaft vorhanden).

Wenn die Platine noch unbestückt ist, muss nach dem Anlegen der Versorgungsspannung ggf. ein Reset ausgelöst werden, indem kurz eine Drahtbrücke an die Kontakte des Reset-Tasters halten wird.



Kurz nach einem Reset sollte die LED dann anfangen zu blinken.

9 Anhang: Beispiele zur Programmierung des ATmega

In diesem Kapitel werden Methoden zur Programmierung des ATmega2560 vorgestellt, die mir von Anwendern zur Verfügung gestellt wurden. Ich kann hierzu leider keinen Support geben.

Vielen Dank an

- Tom64 für die Bereitstellung der Anleitung zu MacOS 10,
- Joao für die Anleitung zum Olimex AVR-ISP-MK2 Programmer.

9.1 DIAMEX PROG-S2 ISP-Programmer für AVR / STM32 / NXP / LPC

Abbildung 9.1: Diamex PROG-S2

Der Programmer wird per USB mit dem PC verbunden. Über den Gerätemanager findet man schnell heraus, welcher COM-Port für die Kommunikation verwendet wird.

Abbildung 9.2: Diamex im Gerätemanager (hier: COM5)

Die DIP-Schalter 1 und 2 müssen auf ON stehen (5V und Spannungsversorgung ein), DIP-Schalter 3 und 4 auf AUS. Der Chip Tester wird in diesem Fall durch den Programmer mir Spannung versorgt und darf nicht mehr per USB oder Hohlstecker mit Spannung versorgt werden.

Mit dieser Konfiguration kann die Programmierung, wir in Abs. 4 beschrieben, erfolgen.

Der Anschluss erfolgt mit dem 6pol. Anschlusskabel (markierte Leitung oben):

Abbildung 9.3: Anschluss des Diamex

Als nächstes können die Fuses gesetzt werden. Hierzu gibt es zwei Möglichkeiten:

1) Manuell, über die Kommandozeile (Batch: “flash_fuses_manual_(please_edit).bat”):

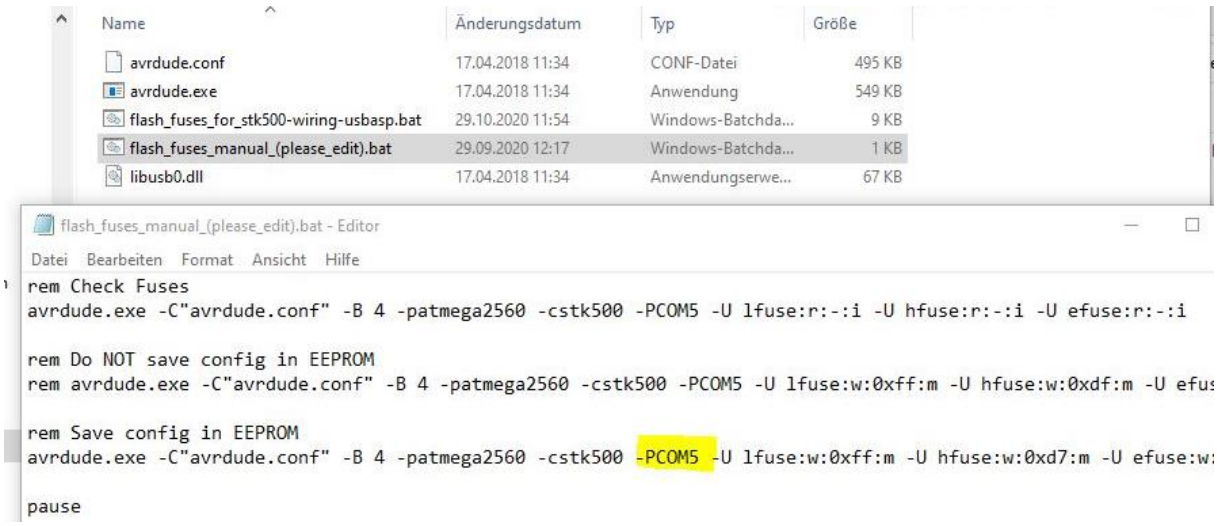


Abbildung 9.4: Vorbereitete Batch-Datei

In diesem Fall muss der COM-Port (gelb markiert) angepasst werden. Danach kann die Datei gestartet werden.

2) Interaktiv, durch mehrere Abfragen (Batch: „flash_fuses_for_stk500-wiring-usbasp.bat“):

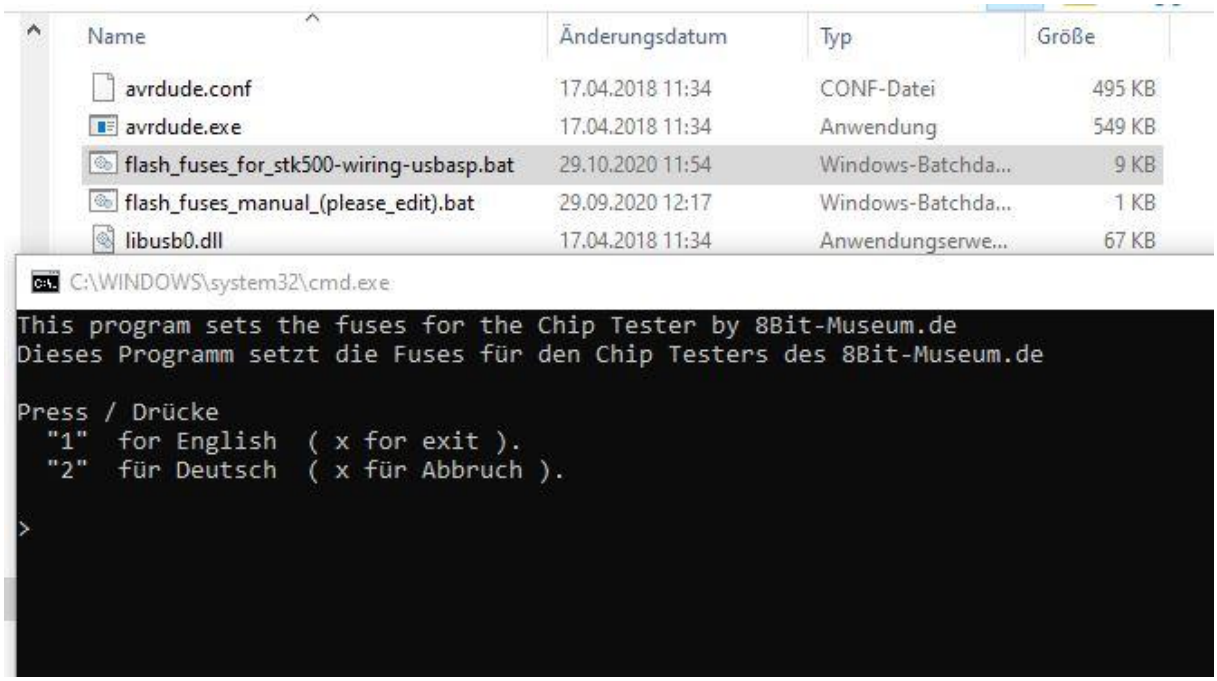


Abbildung 9.5: Interaktive Batch-Datei

In diesem Fall werden die erforderlichen Daten nacheinander abgefragt. Am Ende wird auf Wunsch eine vorbereitete Batchdatei angelegt, die das Flashen der Firmware erleichtert.

Nachdem die Fuses gesetzt wurden, kann die Firmware geflasht werden. Hierzu einfach die vorbereitete Batchdatei „flash_firmware.bat“ in den Firmware-Ordner kopieren und starten, oder die dort vorhandene Batchdatei „flash_firmware_(please_edit).bat“ entsprechend anpassen und starten.

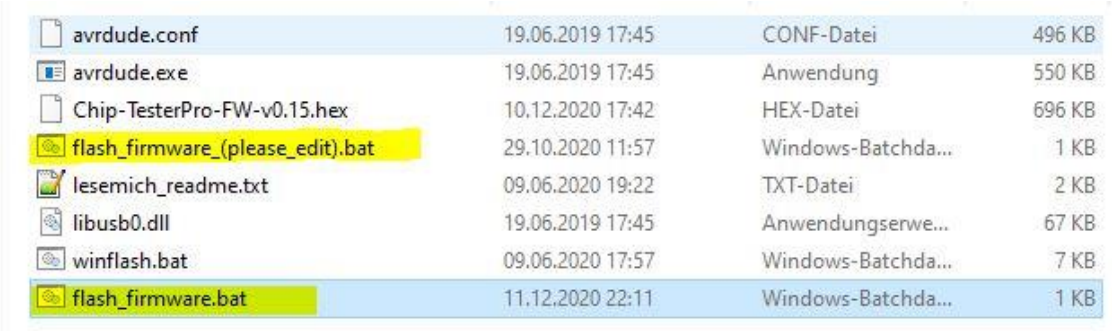


Abbildung 9.6: Flashen der Firmware

Nach ca. 2 Minuten sollte der Programmiervorgang abgeschlossen sein.

9.2 Pololu USB AVR Programmer v2.1 / MacOS 10

Das Display darf auf den Speichertester nicht aufgesteckt sein. Verwendet wird das Programm „AVRFuses 1.5.2 (avrdude v6.3)“[1] zum Programmieren des ATmega2560.

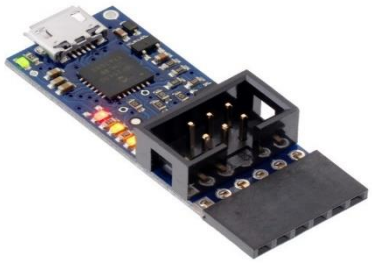


Abbildung 9.7: Pololu USB AVR Programmer v2.1

Der Pololu Programmer ist nicht dazu geeignet das Board mit Spannung zu versorgen (es kann funktionieren, muss aber nicht), deshalb sollte das Board per USB mit +5V versorgt werden und die Versorgung über den Programmer ausgeschaltet werden.

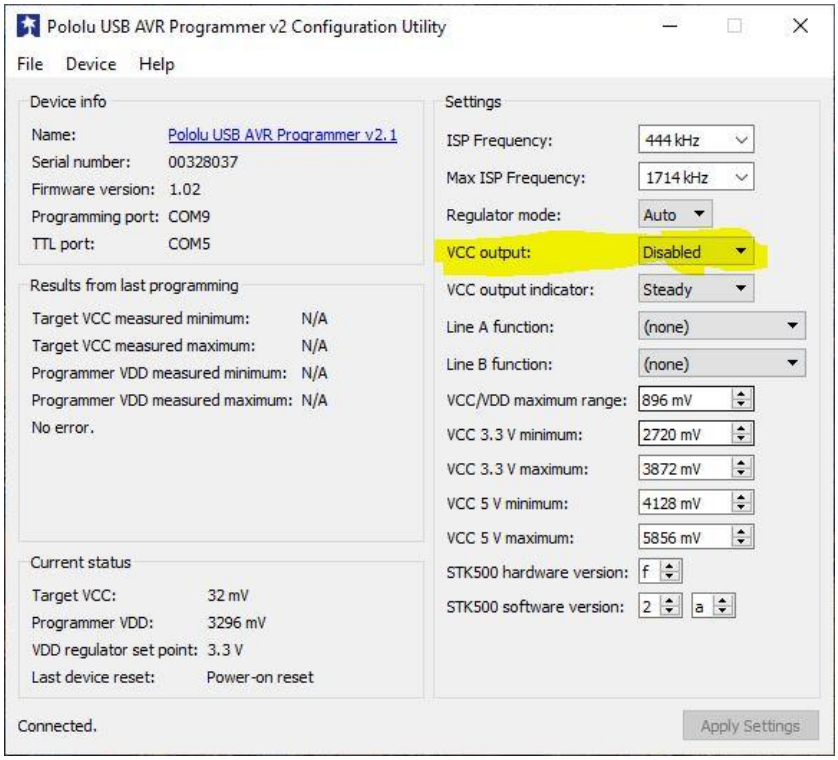


Abbildung 9.8: Vcc output

[1] https://vonnieda.org/software/avrfuses

9.2.1 Voreinstellungen AVRFuses

Menü: → AVRFuses → Preferences

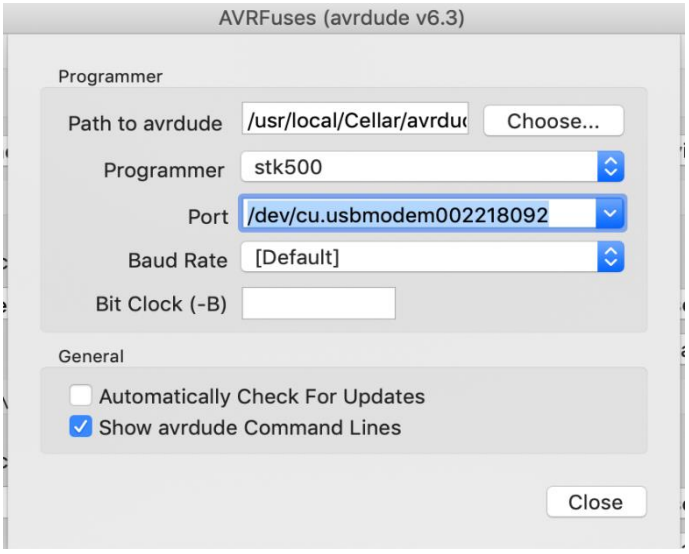


Abbildung 9.9: AVRFuses

- Programmer: STK500 oder STK500v2
- Port: dev/cu.usbmodemXXXXXXX {das erste „usbmodem“ auswählen}
- Baud Rate: default
- Bit Clock: {leer, kein Eintrag}

9.2.2 Fuses setzen

Die Fuses werden wie folgt gesetzt Low: 0xf7, High: 0xd7, Extended: 0xff (die Einstellungen im Bild bitte ignorieren).

Bei der Eingabe der Werte darauf achten, dass die Werte auch übernommen werden. Die Häcken ändern sich entsprechend.

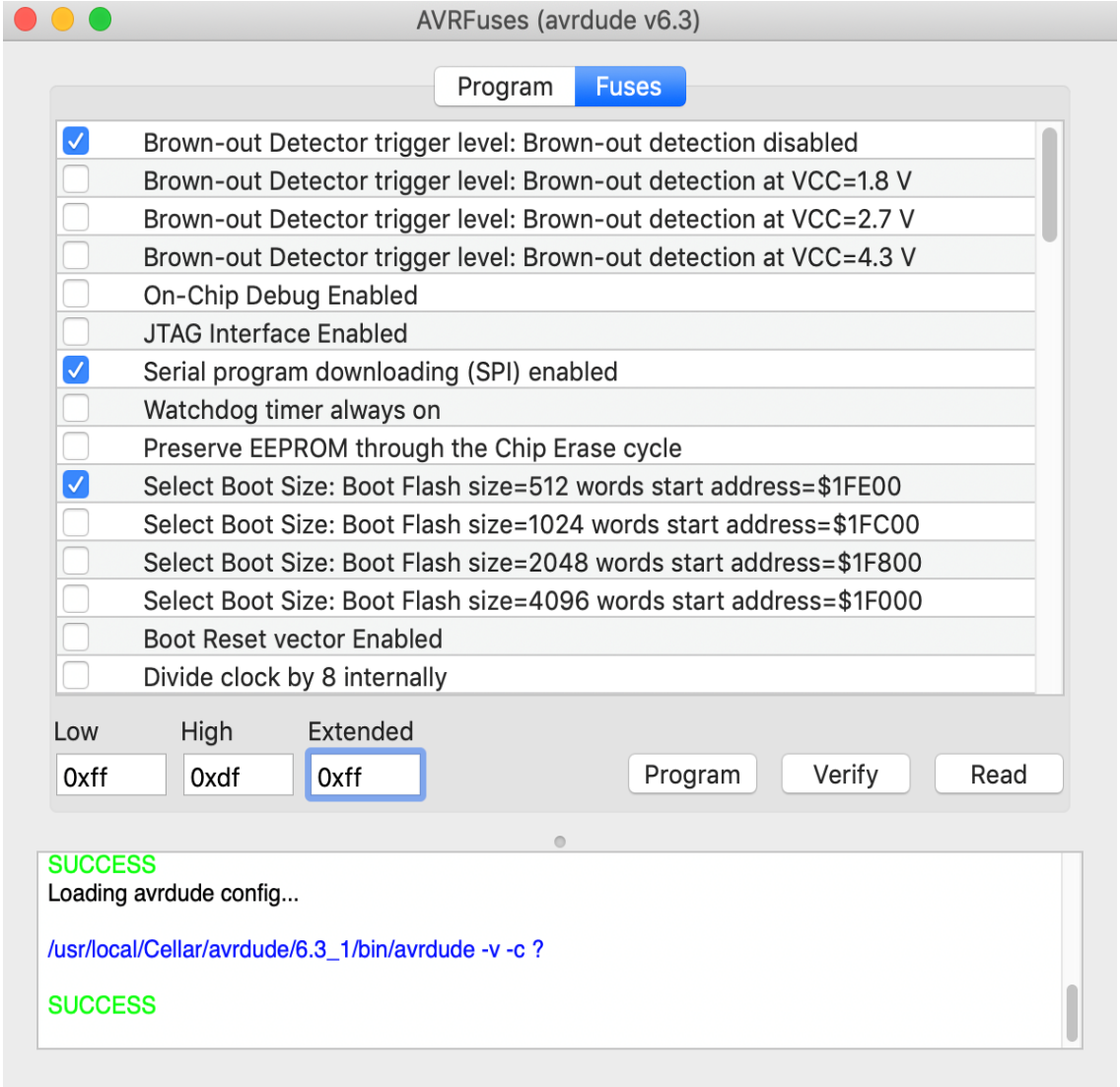


Abbildung 9.10: Setzen der Fuses mit AVRFuses

Mit „Program“ die Werte in den ATmega2560 schreiben.

9.2.3 Firmware flashen

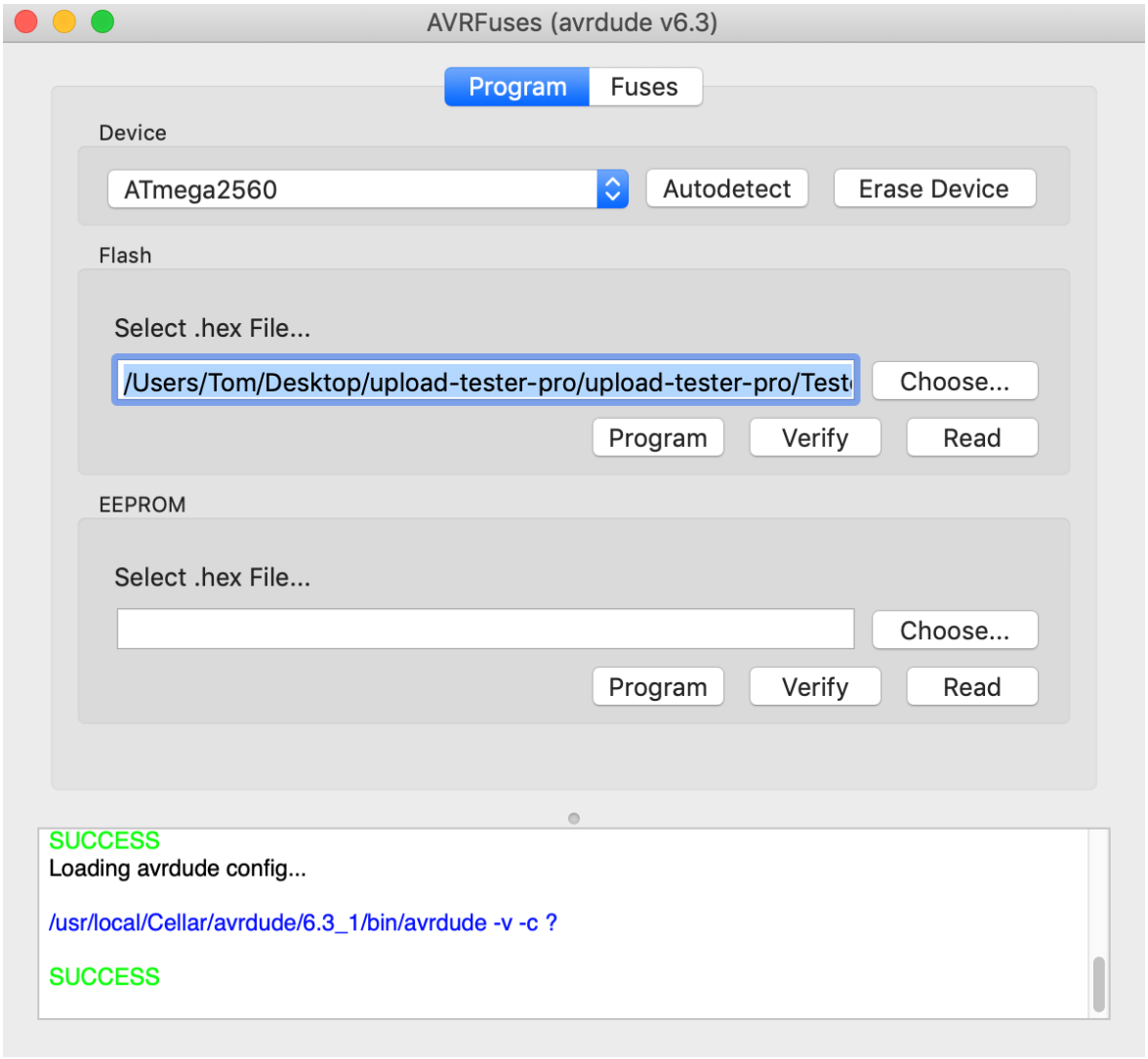


Abbildung 9.11: Programmieren mit AVRFuses

- Device: „ATmega2560“ auswählen
- Flash, Select .hex File: Firmware auswählen
- EEPROM: {leer, kein Eintrag}
- Mit „Program“ die Firmware in den ATmega2560 schreiben.

9.3 Pololu USB AVR Programmer v2.1 / Microsoft Windows

Das Display darf auf den Speichertester nicht aufgesteckt sein. Verwendet wird das Programm „AVRDUDESS 2.4“[2] zum Programmieren des ATmega2560.

Der Pololu Programmer ist nicht dazu geeignet das Board mit Spannung zu versorgen (es kann funktionieren, muss aber nicht), deshalb sollte das Board per USB mit +5V versorgt werden und die Versorgung über den Programmer ausgeschaltet werden.

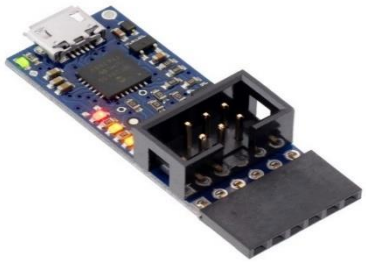


Abbildung 9.12: Pololu USB AVR Programmer v2.1

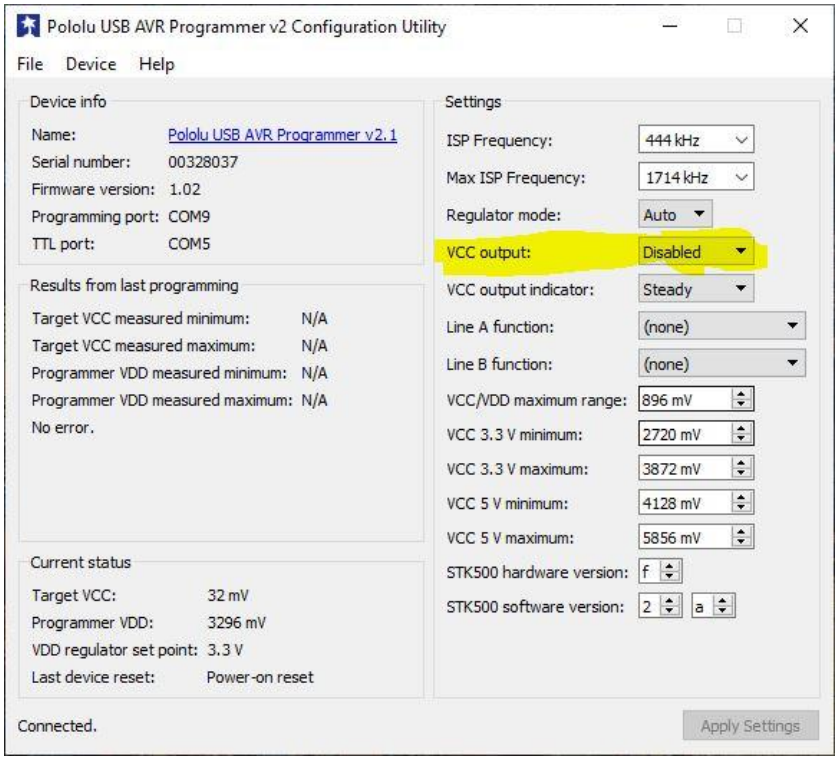


Abbildung 9.13: Vcc output

Weiter mit Kapitel 9.3.1 oder Kapitel 9.3.2…

[2] https://github.com/zkemble/AVRDUDESS

9.3.1 Programmierung mit AVRDUDESS

Die Bedienung von AVRDUDESS ist relativ einfach:

1. Auswählen des Programmers (Atmel STK500 kompatibel)
2. Auswahl des verwendeten Ports (Pololu legt zwei virtuelle Ports an, i.d.R. sollte der niedrigere Port korrekt sein)
3. Auswahl der MCU (ATmega2560)
4. Auswahl der zu programmierende Firmware unter Flash (HEX-File) mit dem Modus „Write“ und als Format „Auto“
5. Festlegen der Fuses „L“, „H“, „E“, „LB“
6. Danach sollte „Set fuses“ angekreuzt werden und mit „Write“ (über „Set Fuses“) diese gesetzt werden.
7. Sind die Fuses gesetzt, kann mit „Go“ (unter „Flash“) die Firmware in den ATmega geschrieben werden.

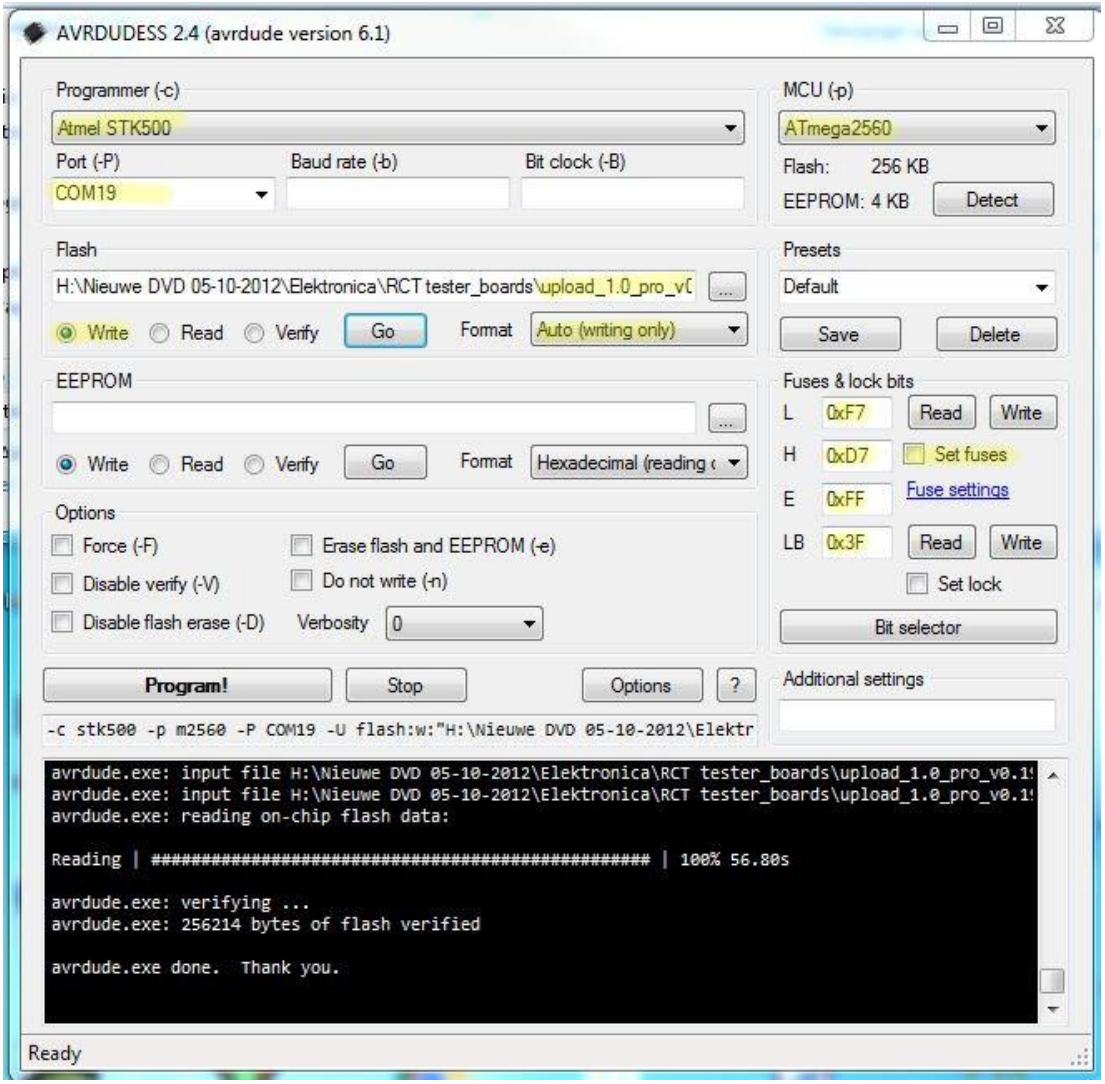


Abbildung 9.14: AVRDUDESS

9.3.2 Programmierung über die Kommandozeile

Der Vorgang entspricht dem in Kapitel 4.2 (ggf. vorher die Fuses setzen) beschriebenen Prozess.

```
avrdude.exe -C"avrdude.conf" -v -patmega2560 -cstk500 -PCOMx
-Uflash:w:Chip-TesterPro-FW-v0.XX.hex:i
```

Bei **-PCOMx** den entsprechenden Port angeben, z.B. **-PCOM9** und die Versionsnummer der Firmware anpassen.

Sollten Probleme auftreten, dann zusätzlich die Geschwindigkeit setzen:

```
avrdude.exe -C"avrdude.conf" -v -patmega2560 -cstk500 -PCOMx -b115200
-Uflash:w:Chip-TesterPro-FW-v0.XX.hex:i
```

Durch Angabe von **-B1** (Faktor 4) oder **-B0.5** (Faktor 8) kann der Programmiervorgang erheblich beschleunigt werden.

9.4 Arduino Uno als ISP Programmer

Es lässt sich der Arduino als Universal-Programmieradapter nutzen, um einzelne Mikrocontroller wie AVRs und PICs zu flashen. Auch wenn das Programmieren i.d.R. problemlos funktioniert, ist diese Lösung etwas „wackelig“.

Zuerst muss der Code zum Simulieren eines ISP-Programmers auf den Arduino Uno geladen werden:

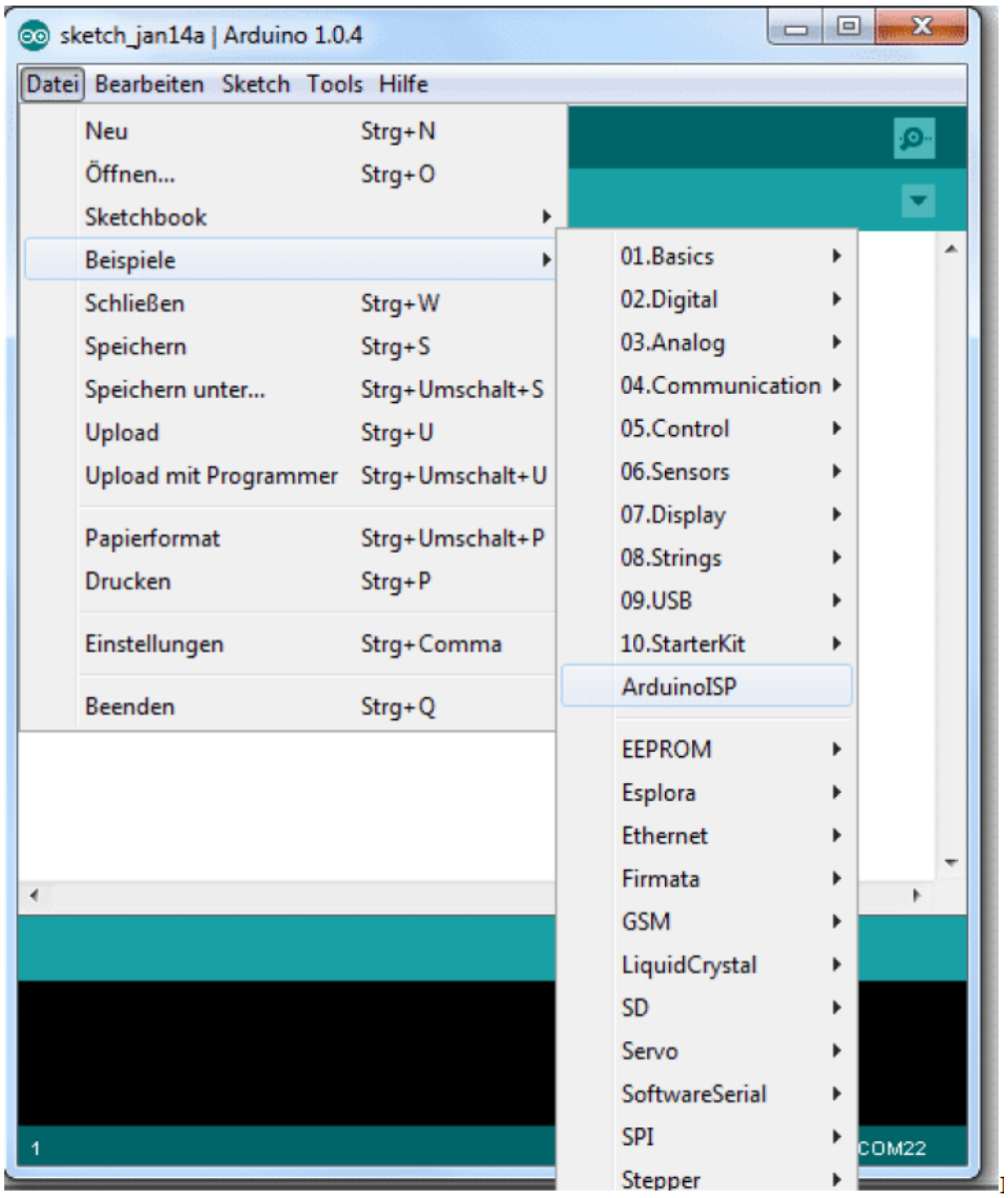
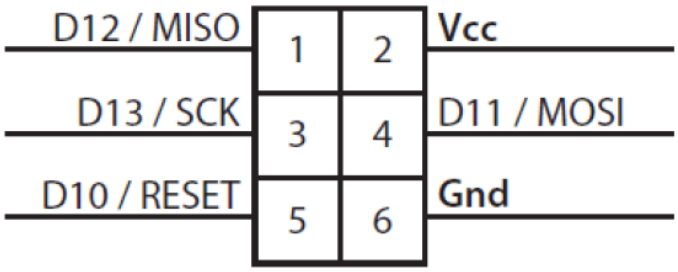


Abbildung 9.15: Arduino IDE

Für die Steuerung von MISO, MOSI und SCK dienen beim Arduino Uno die Pins 11, 12 und 13, den RST-Eingang des Mikrocontrollers steuert Pin 10. Die klassische Belegung eines sechspoligen ISP-Wannensteckers zum Programmieren eines AVR-Controllers in einer fertigen Schaltung ist im Bild zu sehen.



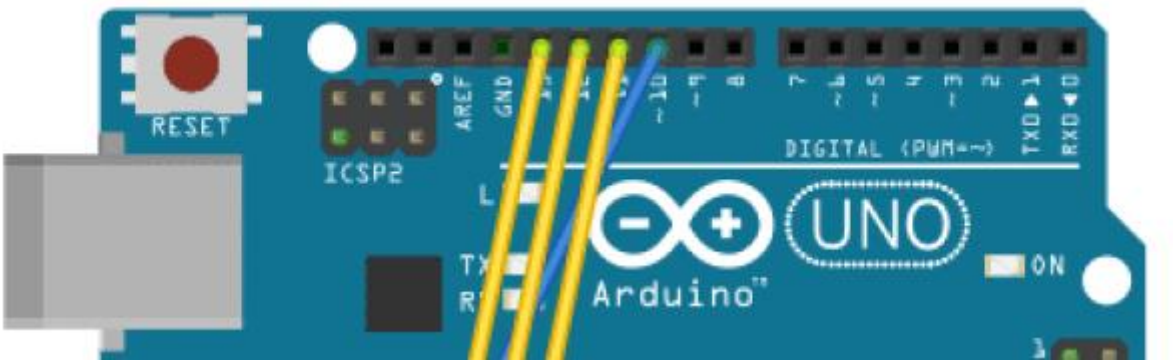
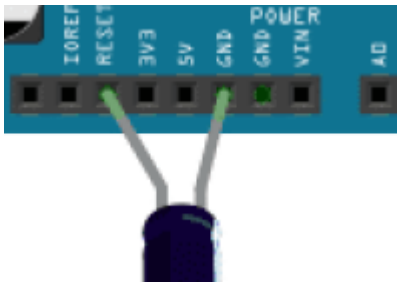


Abbildung 9.16: Arduino Uno als Programmierer

Zum Programmieren muss sichergestellt sein, dass „Programmer > Arduino ISP" ausgewählt ist.



Beim Arduino Uno muss man für die weiteren Schritte einen Kondensator (Elko mit 10µF) zwischen GND und Reset legen. Er fängt den Reset-Impuls ab, den der USB-Seriell-Wandler auf dem Board des Arduino Uno beim Starten der seriellen Kommunikation sendet. Der Reset-Impuls führt nämlich dazu, dass der Mikrocontroller auf dem Arduino-Board seinen Bootloader zur Neuprogrammierung startet, statt den Arduino-Sketch zu starten.

Es gibt wohl zwei unterschiedliche Sketche, so dass der Arduino-Programmierer entweder als `avrisp` oder als `stk500v1` angesprochen wird. In beiden Fällen erfolgt die Programmierung mit AVRDUDE:

1) Als Programmer wird der `avrisp` angegeben:

```
avrdude -C"avrdude.conf" -v -p atmega2560 -c avrisp -P COM3 -b 19200 ...
```

2) Als Programmer wird der `stk500v1` angegeben:

```
avrdude -C"avrdude.conf" -v -p atmega2560 -c stk500v1 -P COM3 -b 19200 ...
```

Ansonsten wird der ATmega2560 wie in Abs. 4 beschrieben.

9.5 AVRISP-MKII kompatibler Programmer

Wer einen zum AVRISP-MKII kompatiblen Programmer einsetzen möchte, muss zuerst dafür sorgen, dass AVRdude mit diesem kommunizieren kann. Werden die Standard-Atmel Treiber installiert, klappt die Kommunikation zuerst nur mir Atmel Studio, nicht aber mit AVRdude.

Abbildung 9.17: AVRISP-MKII kompatibler Programmierer

Zunächst muss der verwendete USB-Treiber gewechselt werden. Dieses erledigt das kostenlose Tool ZADIG zuverlässig, das hier heruntergeladen werden kann: https://zadig.akeo.ie/

Nach der Installation die Option wählen, dass alle Geräte aufgelistet werden. Danach „AVRISP mkII“ in der Liste auswählen und den Treiber auf „libusb-win32“ ändern. Abschließend „Replace Driver“ anwählen.

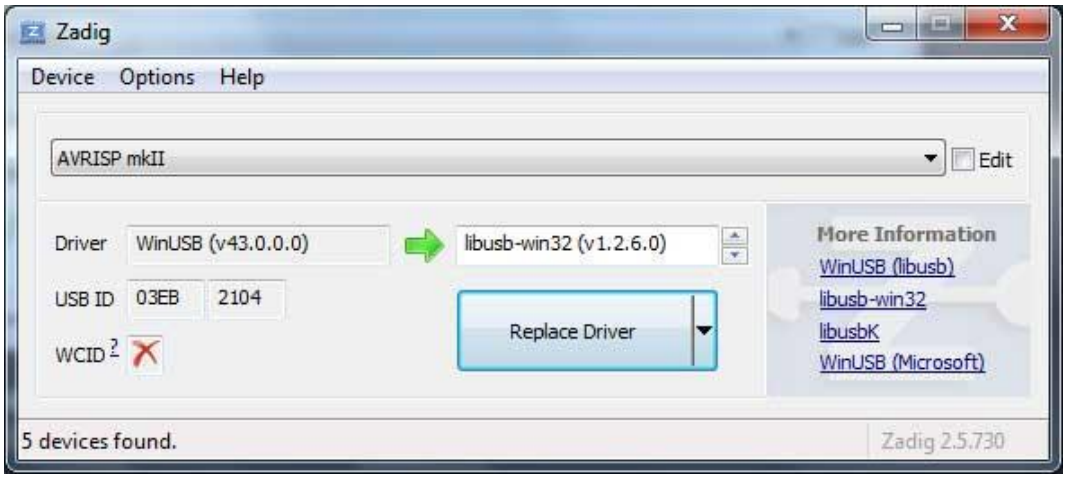


Abbildung 9.18: ZADIG

Wird der Programmer neu angeschlossen, wird der neue Treiber aktiv. Er wird jetzt mit AVRdude über den Parameter „`-cavrispmkii -Pusb`“ angesprochen.

9.6 Microchip PICkit4

Die Firmware kann mit einem PICkit4 von Microchip aktualisiert werden, hierzu die MPLAB X IDE-Software verwenden. Bitte beachten, dass pickit4 den Mikroprozessor nicht mit Spannung versorgt, also muss hierzu USB benutzt werden, um den RCT mit Strom zu versorgen. Das Hex-File einfach in MPLAB importieren und den Kommunikationsmodus des pickit4 von JTAG auf ISP umschalten. Danach funktioniert die Programmierung.

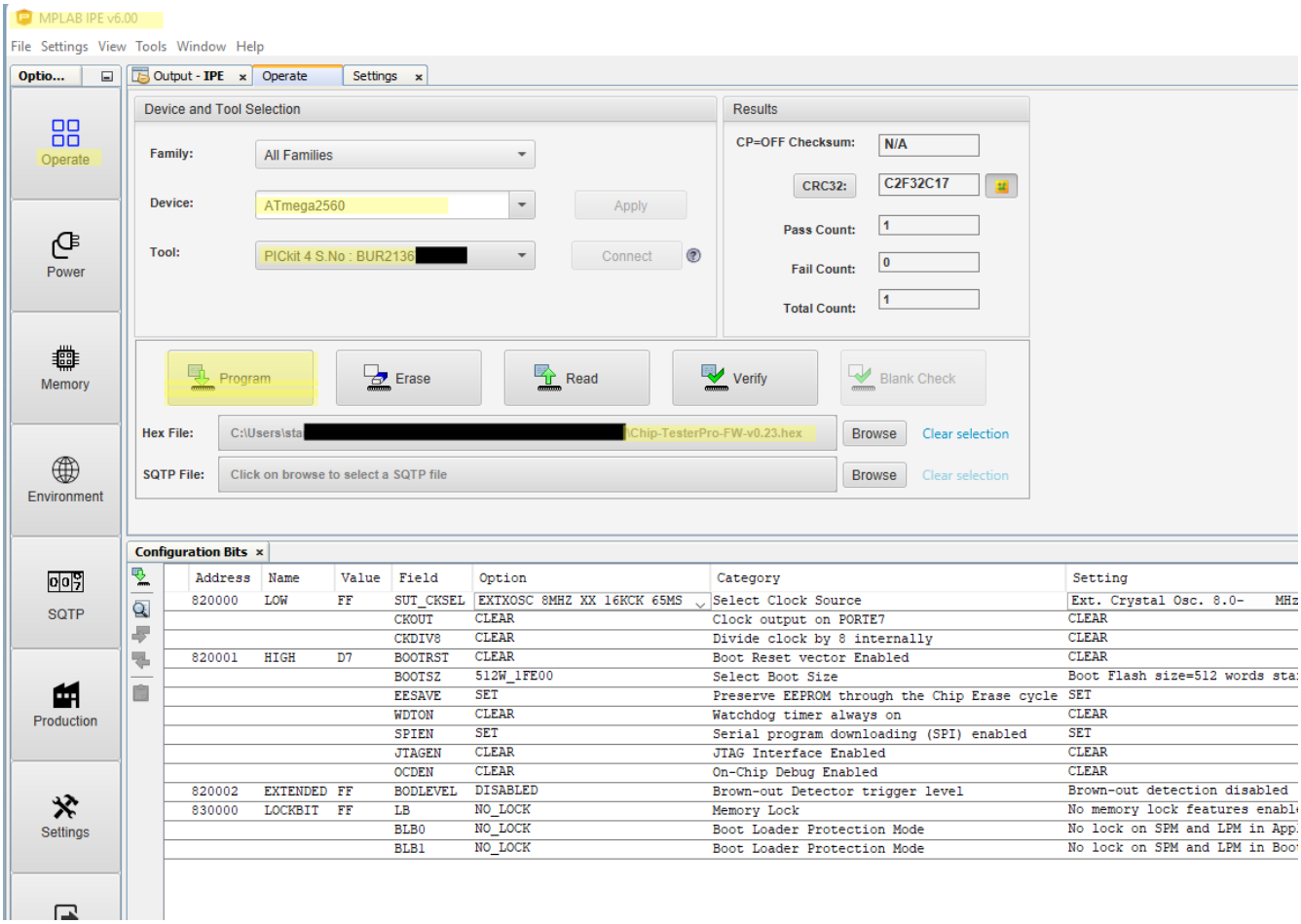
9.7 Microchip MPLABX with IPE Application / PICkit4

Sie können die IPE-Anwendung MPLABX (unter Windows 11) verwenden, um vorkompilierte HEX-Dateien auf den RCT zu laden. Die IPE-Anwendung ist einfacher zu verwenden, wenn nur der Download einer HEX-Datei erforderlich ist.

Beachten Sie, dass der pickit4 den Mikrocontroller nicht mit Spannung versorgt. Versorgen sie daher den RCT per USB mit Strom. Wenn der ATmega2650 im Menü ausgewählt wird, wird die Option für die interne Stromversorgung im „Power“-Tab automatisch ausgegraut.

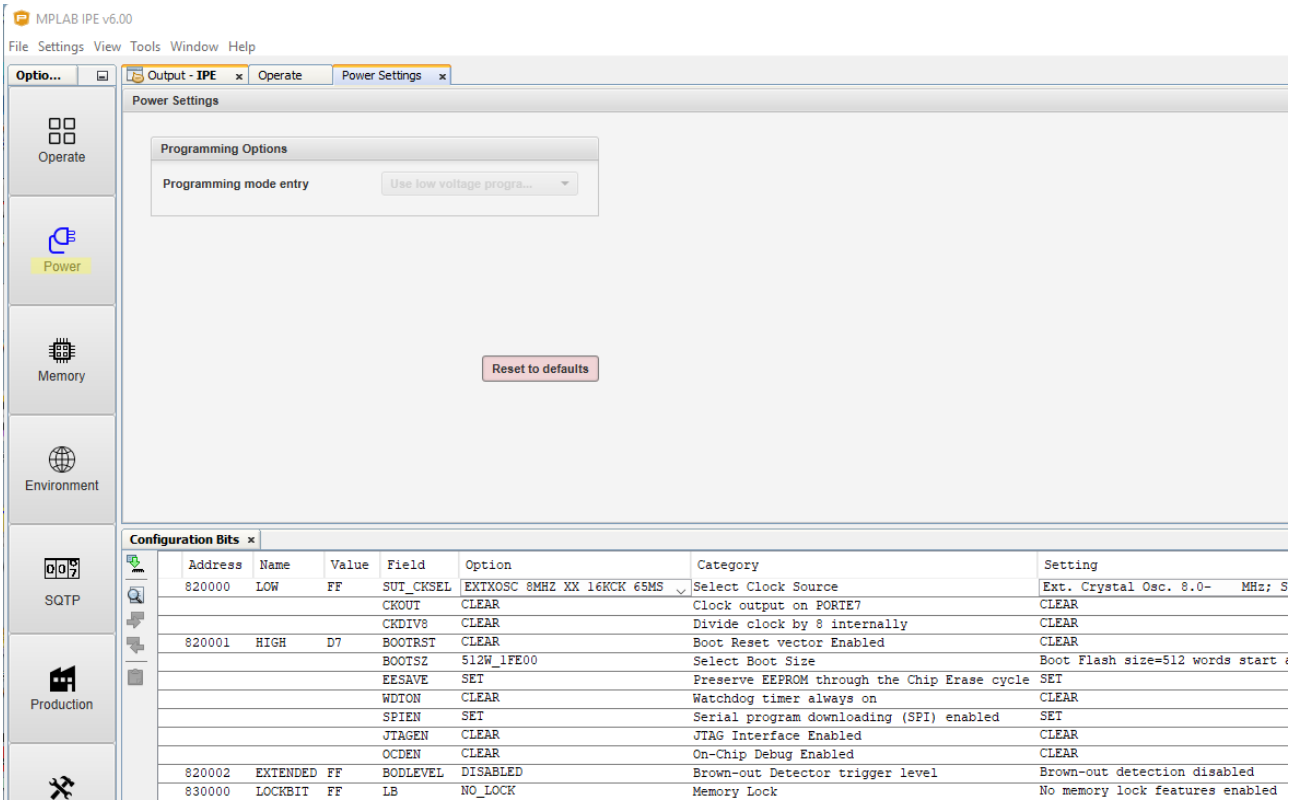
Operate Tab:

Wähle den ATmega2560, den verwendeten Programmer, die HEX Datei, die Konfigurationsbits sollten lauten FF, D7, FF, FF (low, high, extended, lockbit).



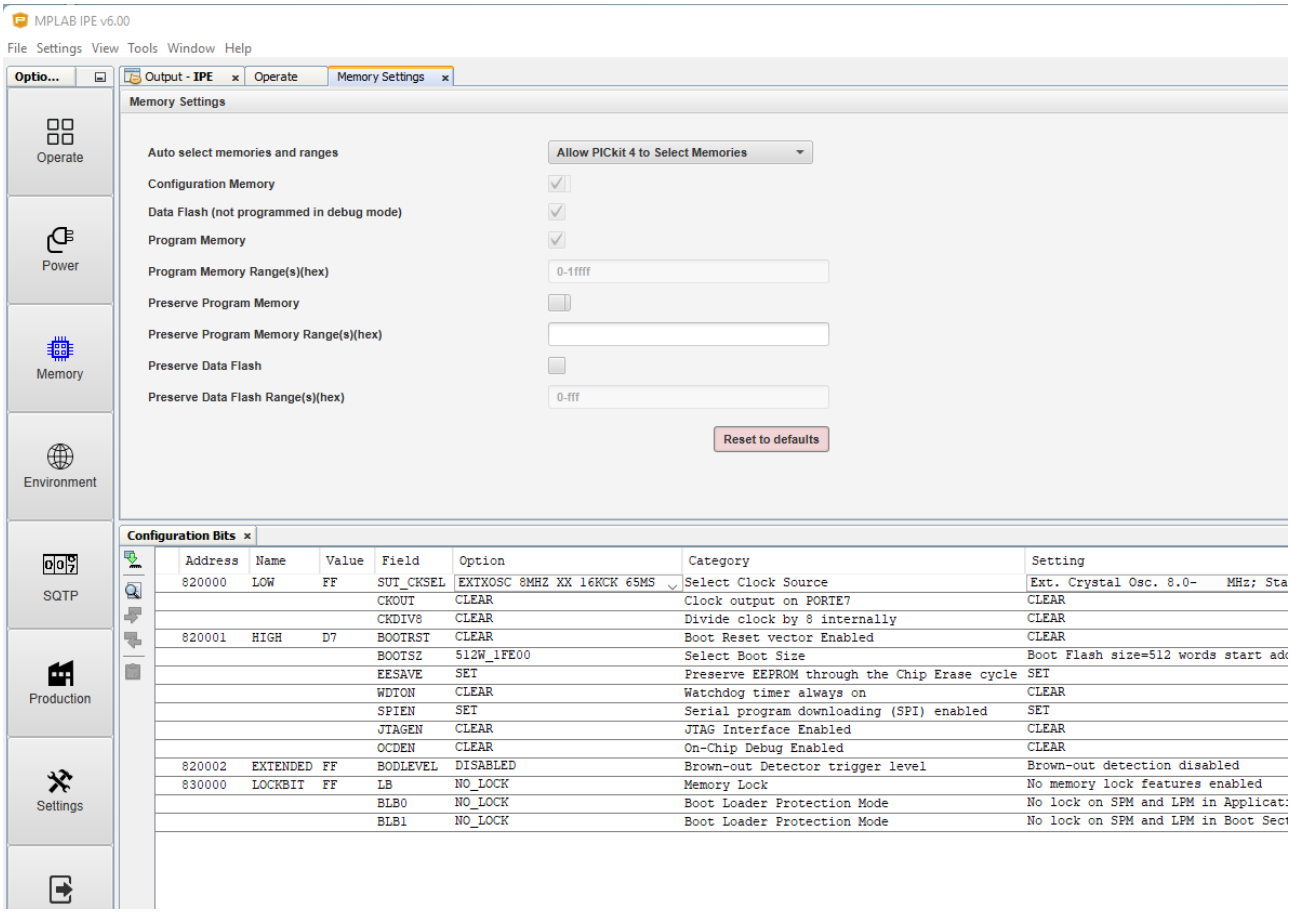
Power Tab:

Power ist ausgegraut.



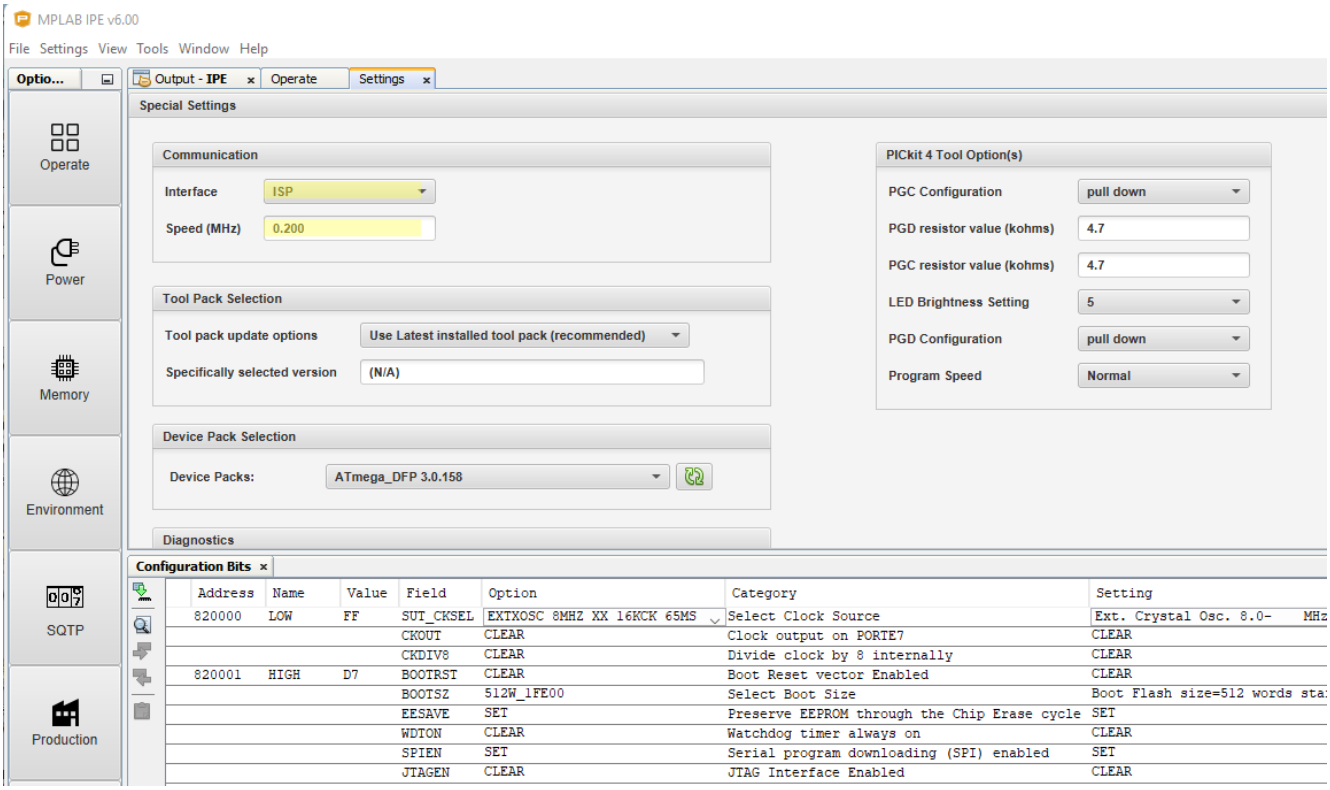
Memory Tab:

Wähle “Allow PICkit 4 to Select Memories”

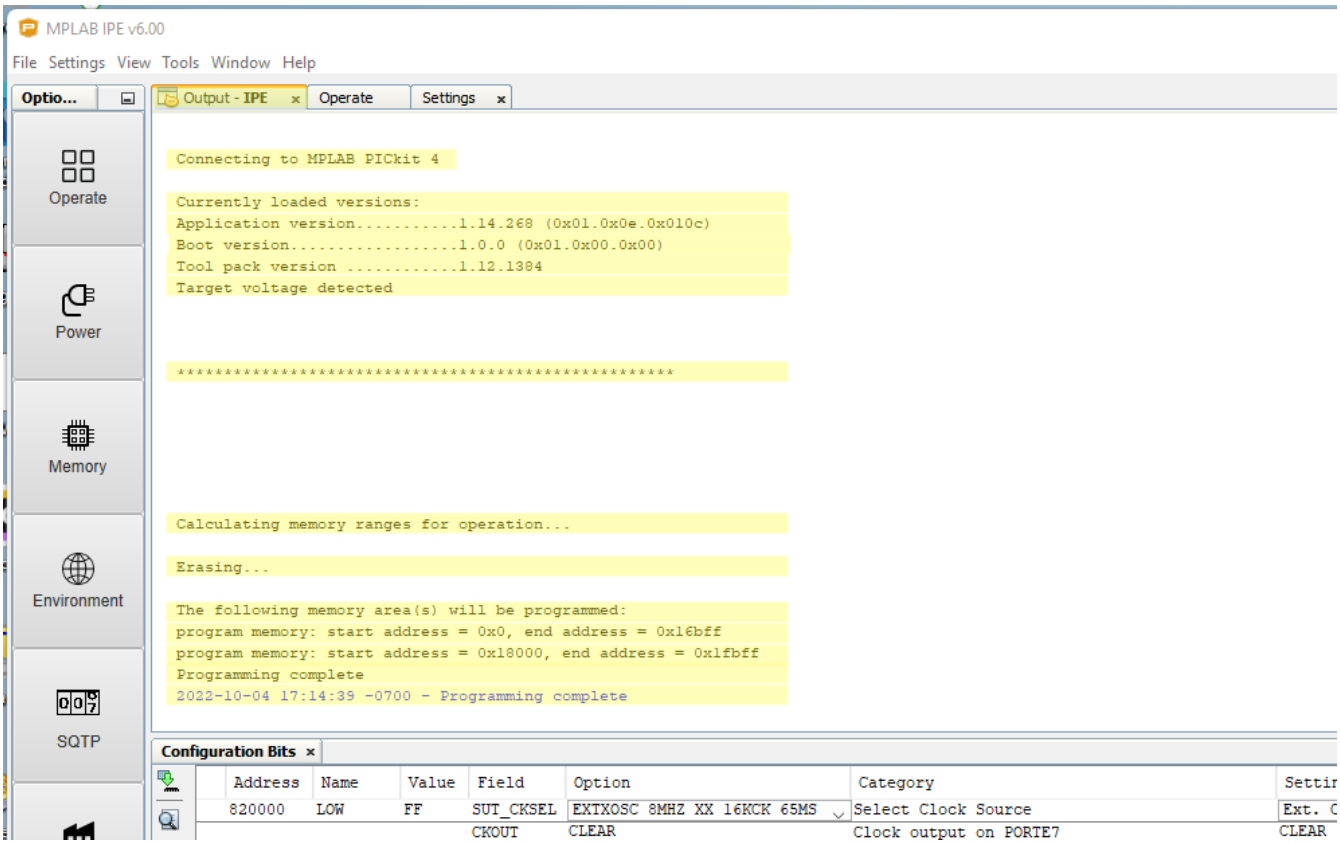


Settings Tab:

Wähle das ISP Interface und als Geschwindigkeit 0.200 MHz.



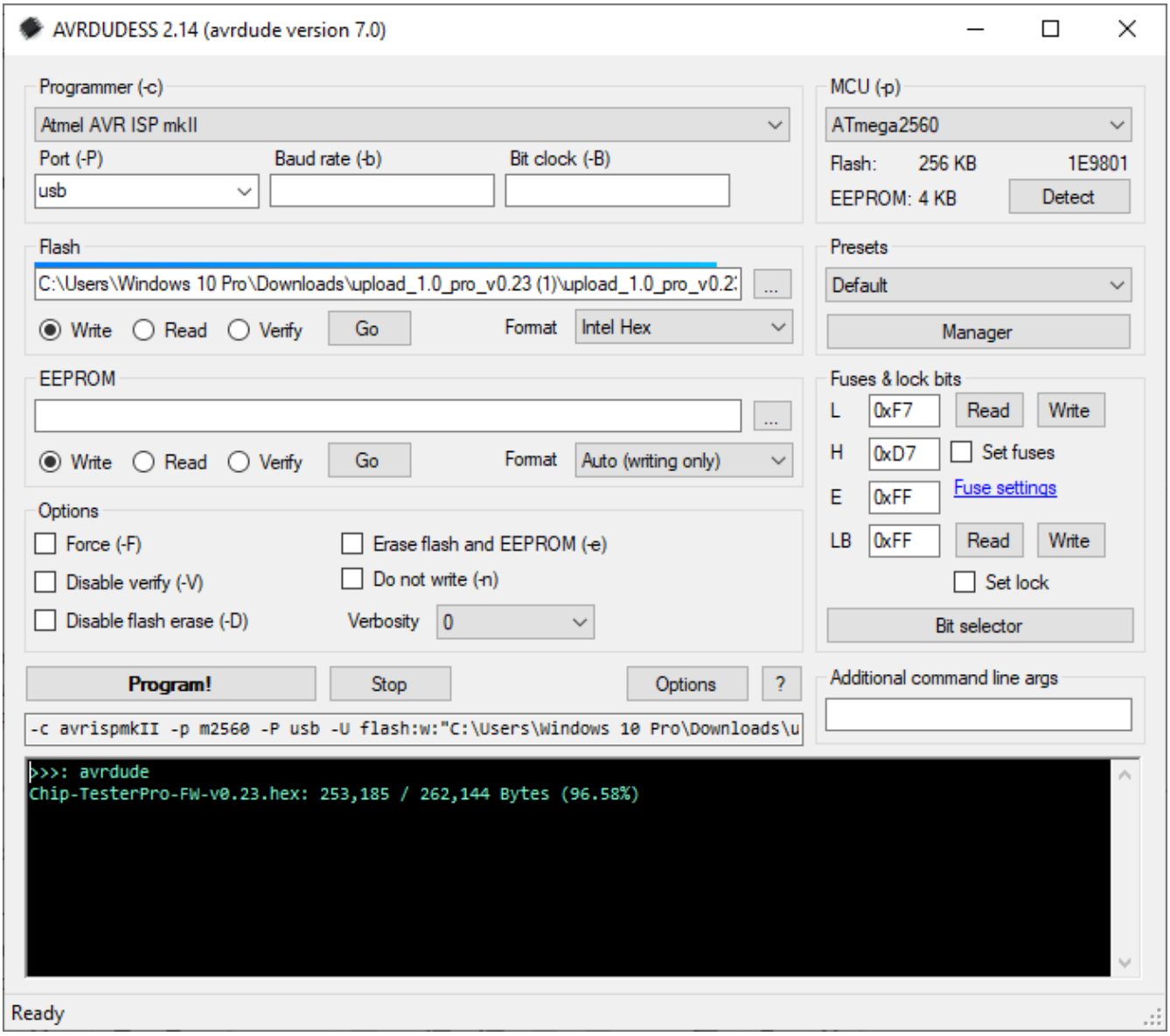
Output IPE Tab:



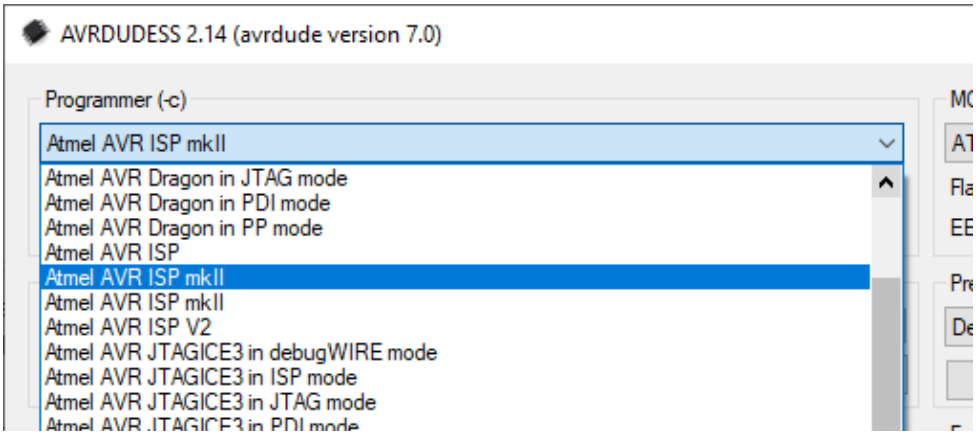
9.8 OLIMEX AVR-ISP-MK2

Der Programmierer OLIMEX AVR-ISP-MK2 kann das RCT-Board mit Strom versorgen (er verfügt über zwei Jumper, einer zum Ein- oder Ausschalten der Stromversorgung und einer zum Auswählen von 5V oder 3,3V).

Es kann AVRDUDESS 2.14 mit AVRDUDE v7.0 zum Programmieren verwendet werden (dieses ermöglicht den Port „-P“ als USB anstelle eines COM-Ports zu konfigurieren).



Bei der Option „-c“ sollte der erste „ATMEL AVR ISP mkII“ Eintrag aus der angezeigten Liste ausgewählt werden, da der installierte Treiber den Gerätenamen „avrispmkII“ entspricht.



Um zu überprüfen, ob der Programmierer den ATMega2560-Chip erkennt, muss das Feld „-p“ (ATmega2560) konfiguriert und dann die Schaltfläche „Detect“ gedrückt werden.

Wenn der Chip erkannt wird, kann die neue Firmware-HEX-Datei ausgewählt werden, die auf den Chip hochgeladen werden soll, zudem „Write“ und das Format „Intel HEX“ auswählen.

Trennen Sie vor dem Aktualisieren der neuen Firmware den Programmierer von der Platine und testen Sie die Eingaben durch Drücken der Schaltfläche "GO" im Flash-Bereich, nur um zu sehen, ob der Programmierbefehl korrekt ist:

```
avrdude -c avrispmkII -p m2560 -P usb
-U flash:w:"C:\<...>\Chip-TesterPro-FW-v0.23.hex":i
```

Wenn alles korrekt ist, die Platine wieder anschließen und "Go" wählen, um die neue Firmware zu schreiben.

9.9 Übersicht über graphische Oberflächen für AVRDUDE

Wer möchte, kann auch eine grafische Oberfläche zum Programmieren des ATmega verwenden. Diese Empfehlungen werden ohne weiteren Support gegeben:

AVR8 Burn-O-Mat: a GUI for avrdude (Windows, Mac, Linux)

www.brischalle.de/avr8_burn-o-mat_avrdude_gui/

AVRDUDESS – A GUI for AVRDUDE (Windows, Mac, Linux)

https://blog.zakkemble.net/avrdudess-a-gui-for-avrdude/

AVRDUDESHELL (Russisch, Englisch)

http://matrex-notes.blogspot.com/search/label/AVRDUDESHELL

10 Anhang: Tweaks

Dieses Kapitel enthält einige gesammelte Ideen.

10.1 Verlängerung des Display-Anschlusses

Soll die Platine in ein Gehäuse eingebaut werden, kann es notwendig werden den Anschluss des Displays über ein Flachbandkabel zu verlängern.

Die triviale Methode verwendet ein 16-pol. Flachbandkabel zum Verbinden des Displays mit der Platine (bzw. 12-pol., wenn man die unbelegten Anschlüsse nicht verbindet).

Alternativ sind folgende Varianten denkbar.

10.1.1 Verlängerung über IDE- oder Floppy-Kabel

Anstelle des 16-pol. Pin-Sockets wird ein 16-pol. Pin-Header bestückt (oder ein „Pin-Header auf Pin-Header“-Adapter in den bestückten Pin-Socket gesteckt). Danach kann mit einem IDE- oder Floppy-Flachbandkabel das Display angesteckt werden. Da die Buchsen zweireihig (und ein bzw. vier Pins breiter) sind, darauf achten, wir das Kabel angesteckt wird.

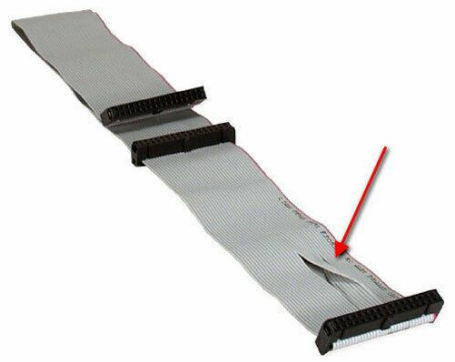


Abbildung 10.1: Floppy-Kabel

Beim einem Floppy-Kabel muss die gedrehte Seite abgeschnitten werden (siehe Pfeil).

10.1.2 Verlängerung über Jumper-Kabel

Das Display kann leicht mit einem Jumper-Kabel (Male-Female) verlängert werden.

Abbildung 10.2: Jumper-Kabel (Male-Female)

10.1.3 Einsparen von Verbindungen

Es können bei einer Verlängerung des Displays bis zu vier Kabel eingespart werden, so dass nur acht Kabel benötigt werden. Hierzu wird die Helligkeits- und Kontrastregelung direkt an das Display verlegt.

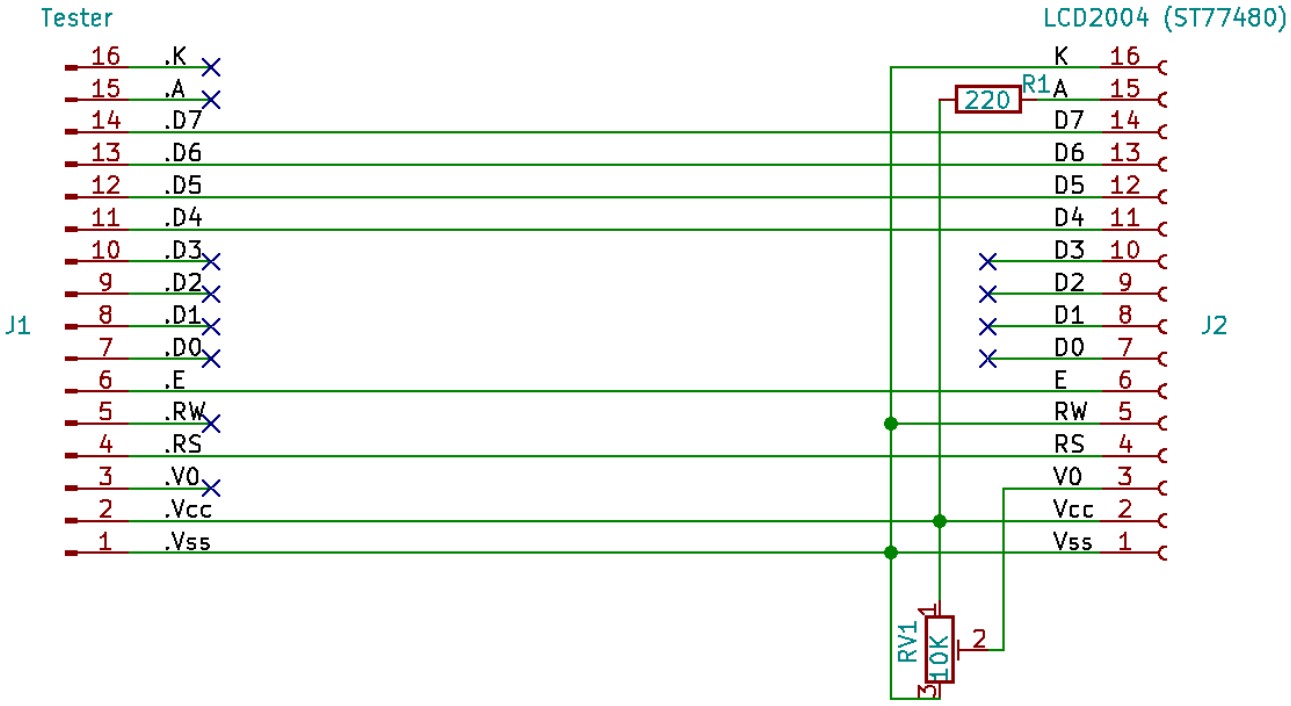


Abbildung 10.3: Beschaltung des Displays

Der 220 Ohm Widerstand ist der Vorwiderstand für die Displaybeleuchtung. I.d.R. kann dieser Widerstand entfallen. Mit dem Poti/Trimmer wird der Kontrast eingestellt.

10.2 Leicht zugänglicher Lautstärkeregler

Ist es notwendig die Lautstärke häufig zu regulieren, kann der Regler auf die Rückseite der Platine gelötet werden. Somit ist er zugänglich ohne das Display demontieren zu müssen.

10.3 Übertragung der Daten per WiFi

Sollen mehrere ROMs ausgelesen werden, dann kann das Umstecken der SD-Karte etwas umständlich sein. Praktischer wäre der direkte Zugriff auf die SD-Karte per WiFi.

Der chinesische Hersteller von 3D-Druckern ShenZhen BigTree Technology (https://www.bigtree-tech.com/) hat hierfür einen günstigen WiFi-Adapter für SD-Karten entwickelt.

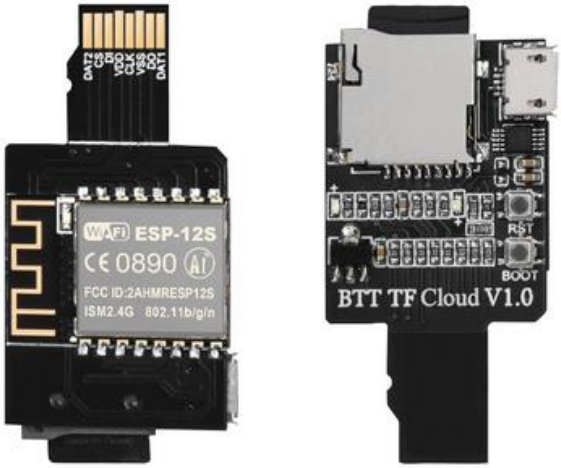


Abbildung 10.4: TF Cloud V1.0 Adapter

Anleitung und Firmware sind auf GitHub erhältlich (https://github.com/bigtreetech/BTT-SD-TF-Cloud-V1.0). Zu bestellen ist der Adapter TF-Cloud V1.0 (Micro SD-Card) z.B. auf AliExpress für unter 10 EUR (https://www.aliexpress.com/item/4001031573171.html).

Der Adapter wird konfiguriert, indem auf der SD-Karte die Datei `SETUP.INI` angelegt wird. Diese Datei enthält zwei Zeilen:

```
SSID=wlan-ssid
PASSWORD=password
```

Als `wlan-ssid` wird der Name des eigenen WLANs eingetragen, als `password` das Kennwort des WLANs. Danach wird die Speicherkarte in den „TF-Cloud“-Adapter eingelegt und dieser in den SD-Micro Kartenadapter gesteckt.

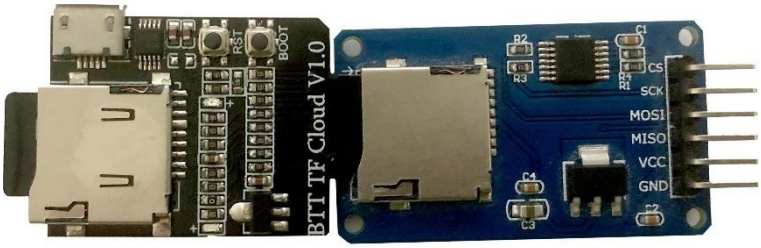


Abbildung 10.5: TF Cloud V1.0 Adapter (links), Micro-SD Kartenadapter (rechts)

Wer möchte kann die TF-Cloud auch mit einem USB-Kabel an einem Computer anschließen. Hierzu muss aber auch ein Treiber für den CH340-Chipsatz installiert werden. Danach ist der Adapter über eine virtuelle serielle Schnittstelle erreichbar. Mit einem seriellen Monitor (115200 bps, z.B. über die Arduino Software oder ein beliebiges Terminal-Programm), kann man sich Debug-Meldungen des Adapters anzeigen lassen, so wird z.B. zu Beginn die zugewiesene IP-Adresse angezeigt. Ggf. vorher einmal RESET auf der Adapterplatine drücken.

Einfacher ist aber im Router nachzusehen, welche IP-Adresse der Adapter zugewiesen bekommen hat. Bei mir meldete sich der Adapter mit „WiFi-SD-Card-3DPrinter“ beim Router an.

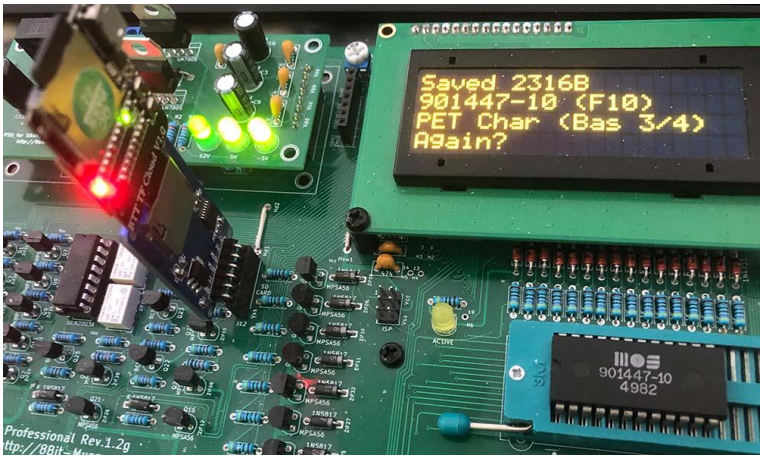


Abbildung 10.6: TF Cloud V1.0 Adapter im Einsatz

Der Zugriff auf die Speicherkarte erfolgt dann über den Windows Explorer über die Eingabe:

```
\\IP-Adresse\DavWWWRoot
```

Falls die Speicherkarte noch in Verwendung ist, erscheint folgende Meldung:

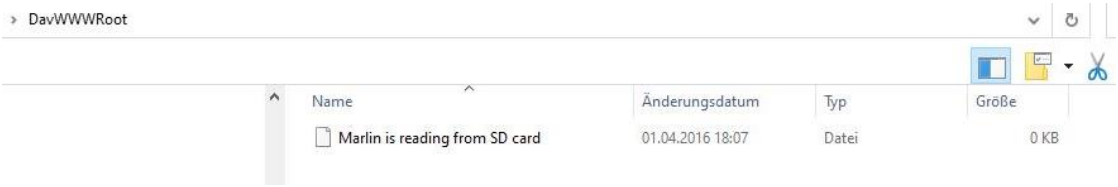


Abbildung 10.7: TF Cloud V1.0 Adapter in Benutzung

ansonsten wird der Inhalt der SD-Karte angezeigt.

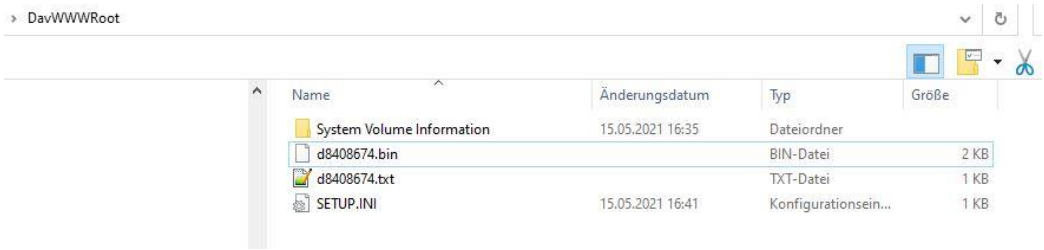


Abbildung 10.8: Inhalt der SD-Karte

Jetzt können die Daten bequem kopiert werden.

10.4 Nachrüsten eines FUNC/TOP Tasters (nur bis Rev.1.2i)

Aufgrund einiger Nachfragen, wurde in die Firmware ab v.19 die Möglichkeit eingebaut, einen weiteren Taster nachzurüsten. Anstelle SELECT und JUMP gleichzeitig zu drücken, kann dann dieser Taster verwendet werden. I.d.R. dient er dazu aus einem Untermenü eine Menüebene zurückzuspringen.

Das Nachrüsten macht dann Sinn, wenn der Tester in einem Gehäuse verbaut ist. Angeschlossen wird der Taster an dem Lötauge 19 / H6 (siehe Bild):

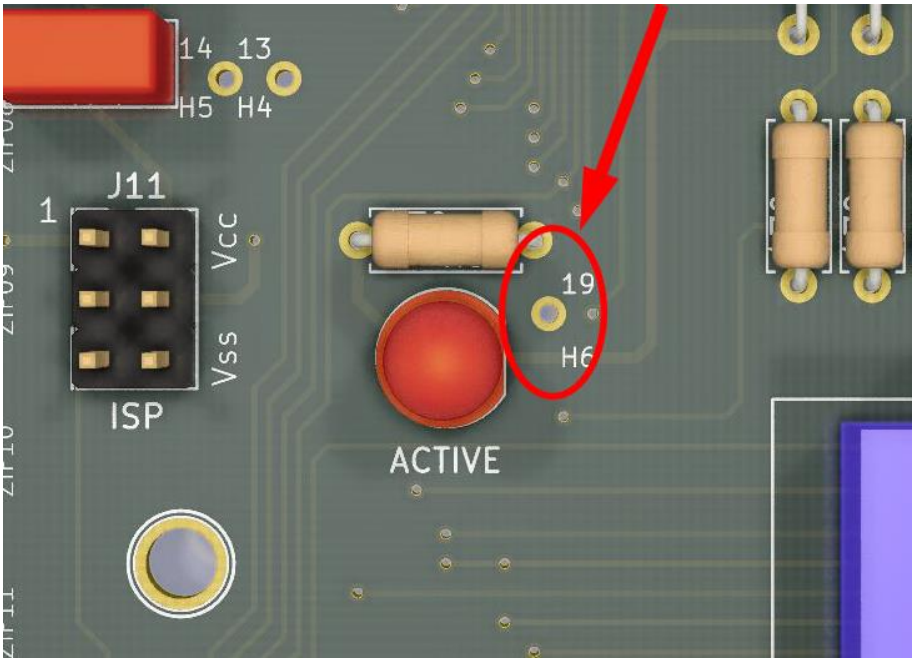


Abbildung 10.9: Anschluss des Tasters (19/H6)

Der Anschluss wird einfach über den Taster mit GND verbunden. Idealerweise wird das Kabel von unten verlötet und auf der Rückseite der Platine verlegt. GND kann z.B. an der Pfostenbuche zwischen Display und DC/DC-Modul abgegriffen werden. Wer die Funktion zuerst ausprobieren möchte, kann dieses mit einem Steckkabel zuerst ausprobieren (ein Ende in GND der Pfostenleiste stecken und das Lötauge mit dem anderen Ende kurz antippen).

Zusätzlich ist dann die gelb markierte Funktion verfügbar:

Im IC-Menü / IC-Untermenü:

SELECT BACK	JUMP FWRD	FUNC TOP	OK	Funktion
X				einen Chip zurück
	X			einen Chip weiter
X	X			Sprung zum Ausgang[3]
		X		Sprung zum Hauptmenü[4] oder ins Untermenü
			X	Auswahl der Funktion
			gedrückt halten	alternativer Mode (in einigen Fällen), z.B. Speichern

[3] nur im Untermenü
[4] nur im Untermenü