HF-Dreiermatrix (Kreuzschienenverteiler)

Vorwort

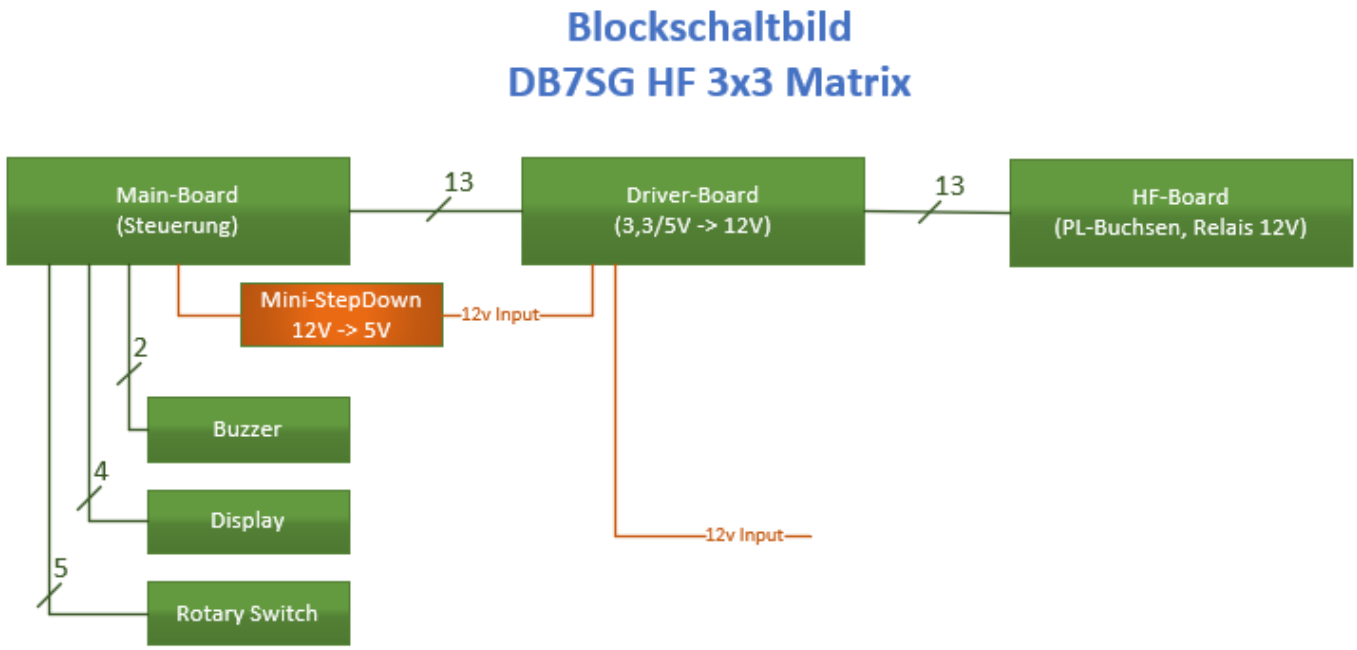
Alle Anleitungen ohne Gewähr! Verwendung auf eigenen Gefahr!

Ich wollte eine Lösung haben, recht einfach mehrere TXen (bis 30MHz, vielleicht auch 50) an mehrere Antennen zu verteilen, und das am besten in allen erdenklichen Kombinationen. Die am Markt erhältlichen Geräte sind eher selten, so versuchte ich einfach mal einen Selbstbau.

Eckdaten

- Drei Eingänge (RX, TX, ...).
- Drei Ausgänge (Antennen, Dummyload, ...).
- Alle Eingänge sollten mit allen Ausgängen verschaltet werden können (also sechs Möglichkeiten).
- Nettes Display: OLEDs sind mir zu klein, Touchscreen (Nextion, ...) ist etwas oversized, also ein LCD mit 4 Zeilen (Überschrift und drei Schaltwege).
- Möglichst einfach aufgebaut, ich nehme drei Platinen:
  - Abgesetzte HF-Platine mit den PL-Buchsen und den Relais, ansonsten so wenig wie möglich *(Entkopplung)*.
  - Treiberplatine zum ansteuern der 12V-Relais mit 3,3V oder 5V *(wird noch für andere Projekte verwendet)*.
  - Prozessorboard zur Ansteuerung der Treiberplatine, entsprechend des Schaltwunsches *(könnte mal umgestellt werden, zentrale Steuerung, vielleicht durch Raspberry Pi o.ä.)*.

Blockschaltbild



HF-Platine

Diese Platine ist im geplanten Gehäuse stehend hinten, die sechs PL-Buchsen werden direkt mit der (leitenden) Rückwand verschraubt. Die drei Antenneneingänge bekommen einen kleinen Überspannungsschutz, optional kann man auch die 3 TX-Eingänge damit bestücken. Als einziger Anschluss dient ein 14-poliger Pfostenstecker „Input“, der folgende Signale erwartet:

- PIN1 bis PIN12 -> -12V zum Ansteuern der 12 Relais
- PIN 13 -> +12V
- *PIN 14 -> -12V (unbenutzt)*

Folgende Tabelle zeigt, welche Relais geschaltet sein müssen um die sechs möglichen Zustände zu erreichen:

Schaltzustände	Rel. 1	Rel. 2	Rel. 3	Rel. 4	Rel. 5	Rel. 6	Rel. 7	Rel. 8	Rel. 9	Rel. 10	Rel. 11	Rel. 12
1-2-3	X	X	X							X	X	
1-3-2	X	X	X					X				X
2-1-3			X	X	X		X		X	X		
2-3-1		X		X	X				X			X
3-1-2			X	X		X	X	X				
3-2-1		X		X		X					X	

Im stromlosen Zustand, also wenn alle Relais abgefallen sind, sind die drei TX-Eingänge freigeschaltet. Ferner gibt es keine Möglichkeit, durch Fehlschaltung die drei TX-Eingänge irgendwie kurzzuschließen.

Der kürzeste Weg (also die wenigsten beteiligten Relais) ist TX1 -> Ant1, gefolgt von TX1 -> Ant2 und TX1 -> Ant3, es ist daher ratsam, das meist genutzte Gerät (oder auch die höheren Bänder) an TX1 anzuschließen (TX2 und TX3 gehen über ein weiteres Relais)!

Treiberplatine

Die Treiberplatine hat nur die Funktion, die Schaltzustände des Prozessors mit 3,3V, bzw. 5V mit 12V an die HF-Relaisplatine weiterzureichen.

- Der 14-polige Pfostenstecker „Output“ wird pinkompatibel mit der Relaisplatine verbunden, am 14-poligen Stecker „Input“ werden folgende Signale erwartet:
  - PIN1 bis PIN12 -> +3,3V oder +5V zum Ansteuern der 12 Relais
  - PIN13 -> -3,3V oder -5V (GND)
  - *PIN14 -> +3,3V oder +5V (unbenutzt)*
- 2-poliger Stecker „Input“: Versorgungsspannung 12V
- Die beiden 2-poligen Stecker „Output“: Abgriff Versorgungsspannung für weitere Zwecke, z.B. Speisung Prozessorplatine

Prozessorplatine

Aufgabe dieser Platine ist es, die Bedienelemente (Display, Schalter, Beeper, ...) bereitzustellen und je nach Wunsch 12 digitale Ausgänge zur Treiberplatine High oder Low zu setzen. Da ich noch einige Arduino Nano hier hatte, hab ich einfach eine Platine gebastelt, die die entsprechenden PINs herausführt. Welche PINs ich für welchen Zweck verwendet habe, ist weiter unten im dafür geschriebenen Arduino-Sketch ersichtlich.

Zur Spannungsversorgung verwende ich eine kleine StepDown-Platine (12V -> 5V). Um andere Wege der Versorgung offen zu halten (z.B. anderes Design einer StepDown-Platine), hab ich zur Spannungsversorgung einen 7-poligen Pfostenstecker vorgesehen, welcher auf den rechten und linken 6 PINs GND und am mittleren +5V erwartet.

Ich verwende für Schalteingänge ungern die internen Pullup-Widerstände des Arduino. Der hier verwendete Rotary-Switch KY-040 hat bereits auf seiner kleinen Platine Widerstände verbaut. Falls man einen anderen Schalter verwendet, kann man auf der Platine die drei optionalen Widerstände verbauen, bzw. zum Abschalten die Brücken neben den Widerständen öffnen.

Anschlüsse der Platine:

- Spannungsversorgung 5V (7-polig)
- Optionaler Beeper (2-polig)
- LCD-Display 2004 mit I2C (4-polig)
- Rotary-Switch (5-polig)
- *Zwei zusätzliche PINs (unbenutzt, für freie Verwendung oder anderen Displaytyp)*

Firmware

Ich hab einen passenden Sketch geschrieben, mal zum ersten Test, das Einzige was man dazu noch einbinden muss (Bibliotheken verwalten in der Arduino IDE) ist die „LiquidCrystal_I2C.h“. Ein paar Grundeinstellungen (Namen von TXen und Antennen, Displaylicht-Timeout, ...) kann man vor dem Aufspielen anpassen, ein ausgefeiltes Setup-Menü ist was für irgendwann, wenn mal Zeit bleibt. Man kann per Rotary Encoder jedem TX eine Antenne zuweisen, diese wird dann getauscht mit dem TX, der die gewünschte Antenne aktuell hat.

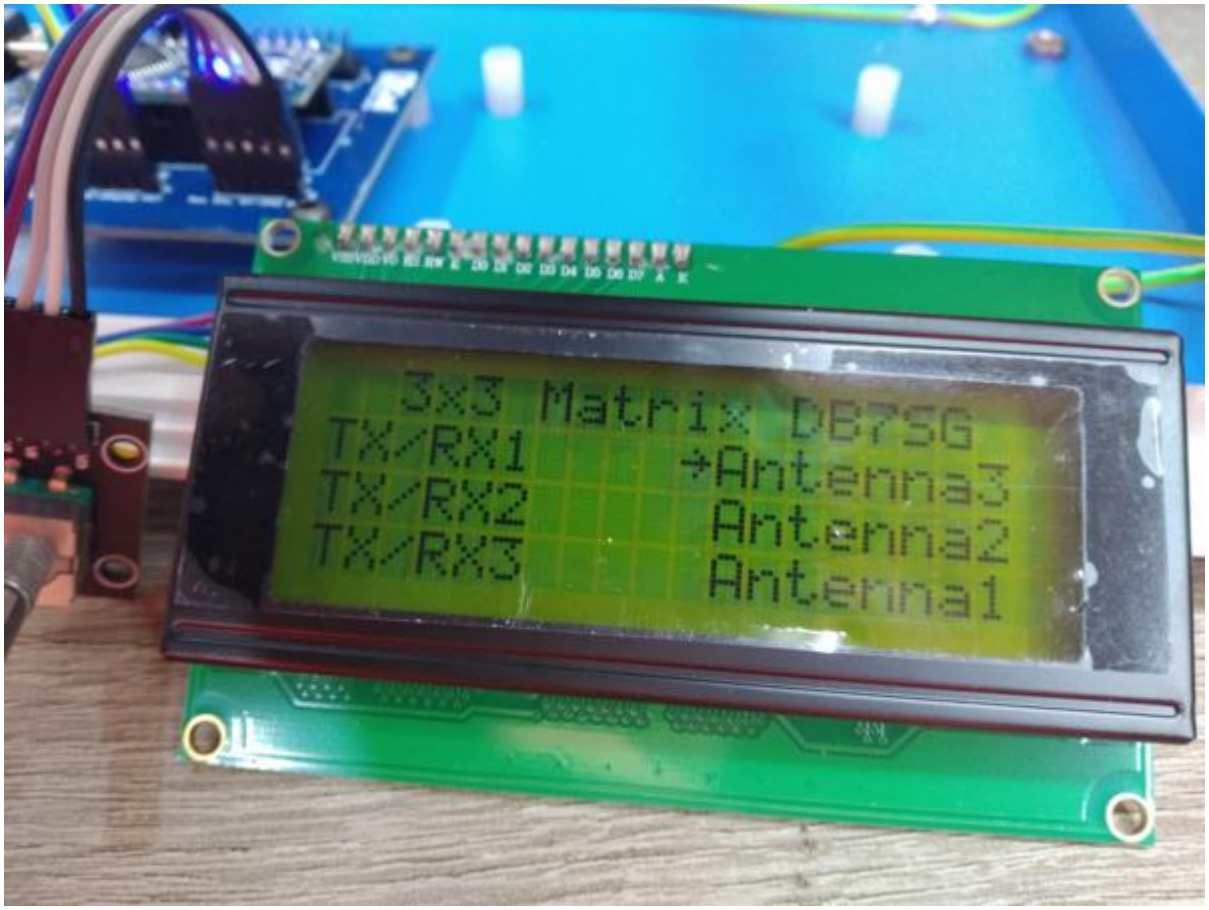
Ich habe den Sketch mal von Grund auf selbst geschrieben, weil es mich eben interessiert hat, ein kleines Menü zu basteln, ohne Bibliotheken dafür zu verwenden. Vielleicht kann jemand das ja auch als Beispiel nehmen und mir auch Rückmeldung geben, was ganz furchtbar ist und was nicht. Der Sketch ist dadurch recht simpel und übersichtlich gehalten und in englisch im Quellcode dokumentiert. Im oberen Bereich gibt es eine kleine Sektion, in der man alle nötigen Anpassungen vornehmen kann.

Die Funktionsweise ist in folgendem Video etwas dargestellt.

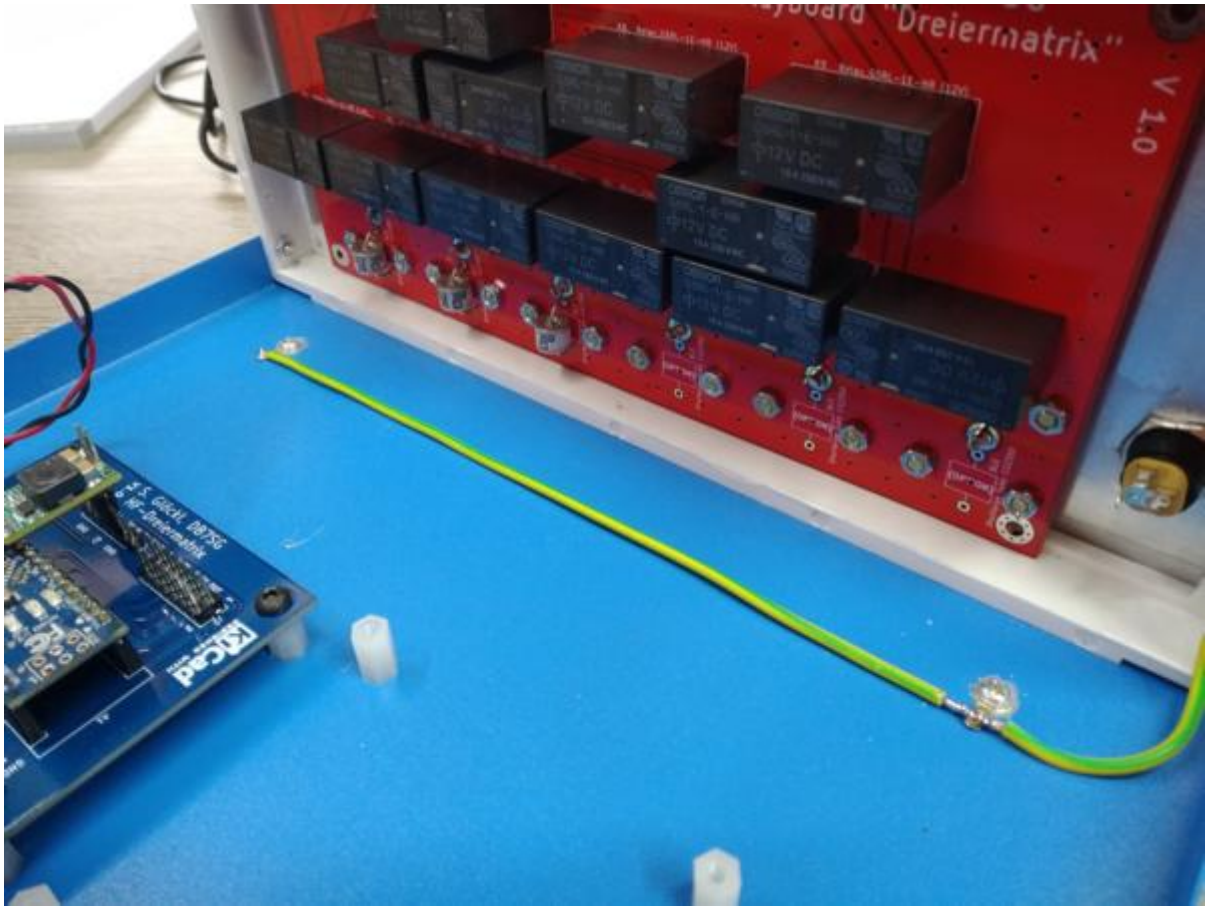
dreiermatrix_display_.mp4

Zusammenbau (stehende Version)

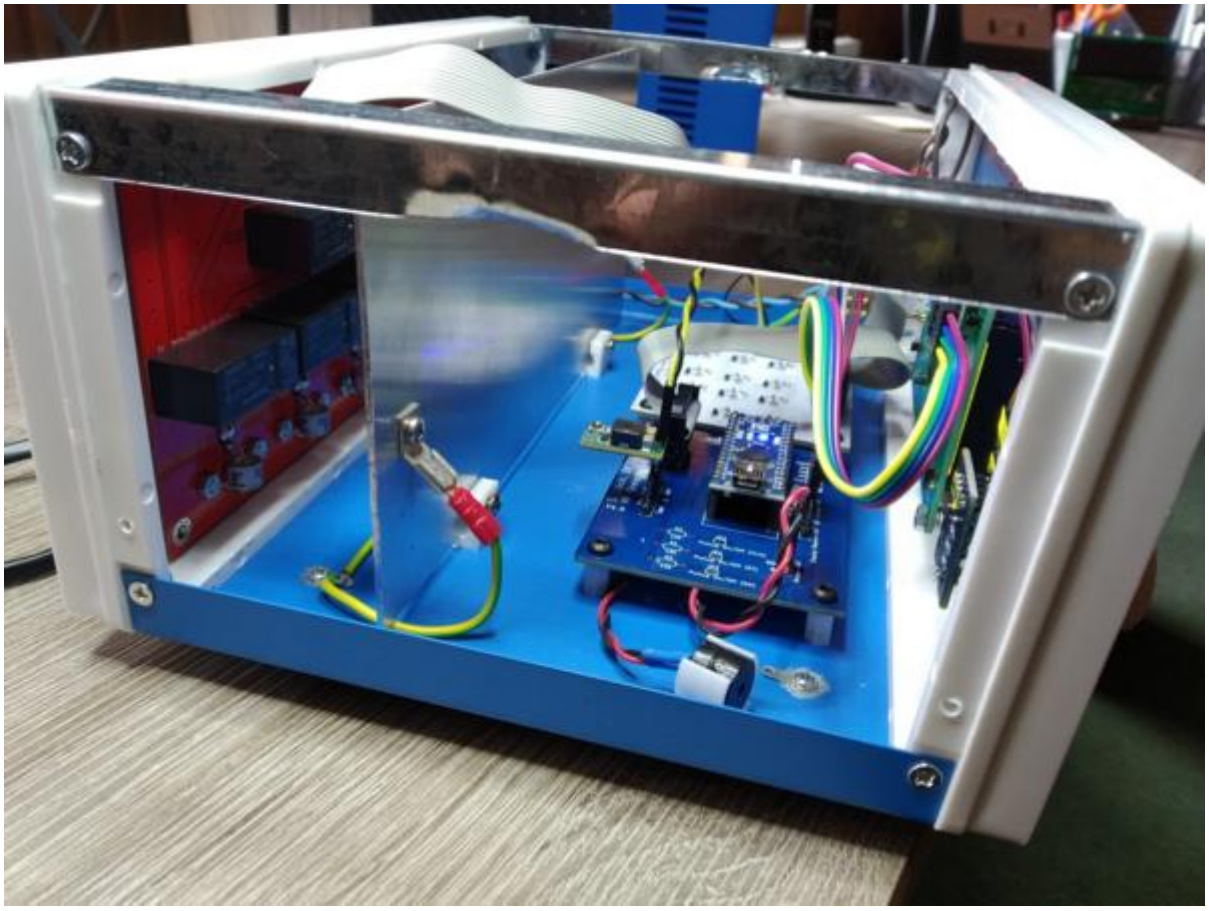
Ich habe zwei Displays probiert, LCD2004 I2C, ein Blaues und ein Grünes. Das Blaue sieht zwar schick aus, aber ich hab in der Firmware einen Parameter, um nach ein paar Minuten die Beleuchtung abzuschalten, solange bis man den Encoder benutzt. Ohne Licht lässt sich das Grüne einfach besser ablesen. Aber Geschmackssache, man kann in der Firmware auch einstellen, das Licht immer an zu lassen.

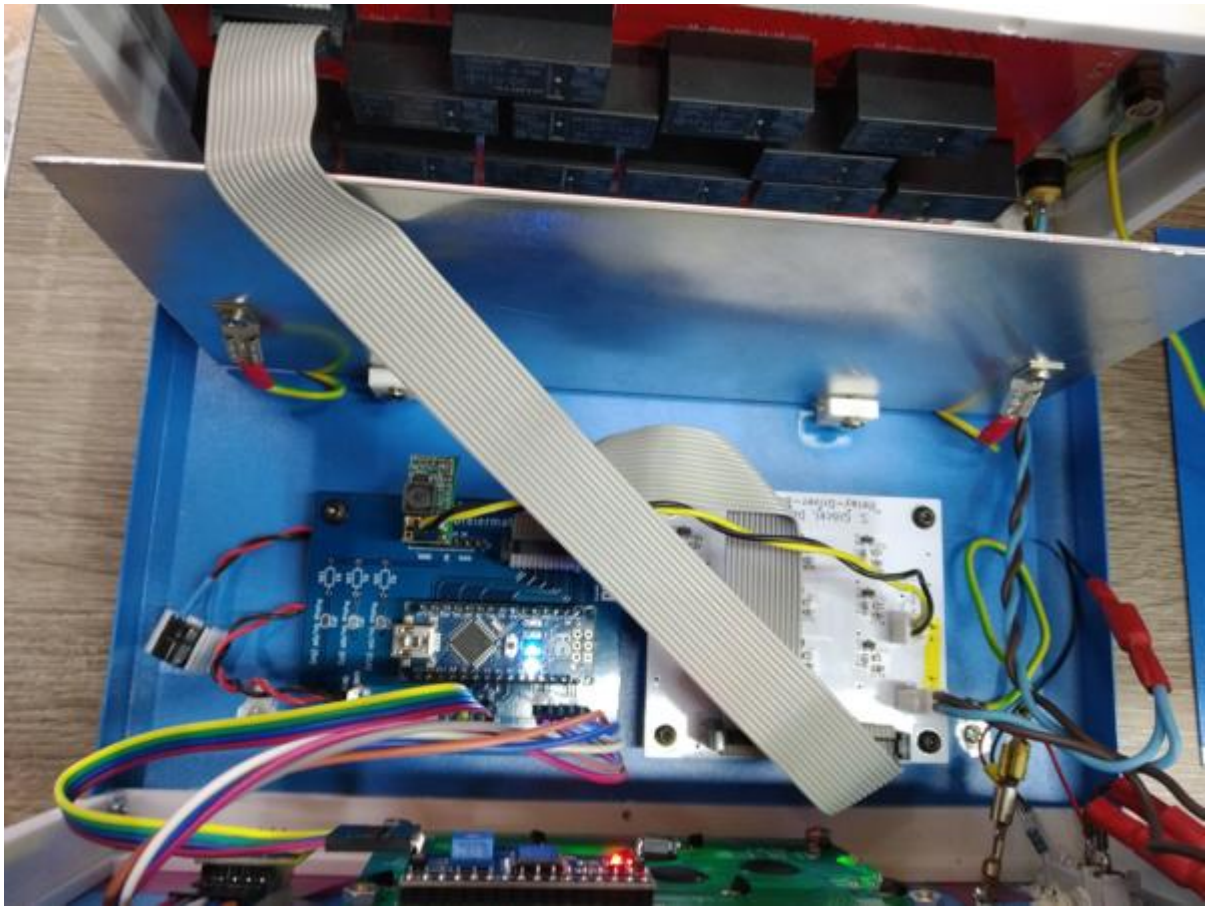


Das Relaisboard wird an der Rückplatte mit Messing-M3-Spacern (StandOffs) verschraubt.



Die beiden anderen Platinen hab ich möglichst weit weg zur Frontplatte montiert und noch ein Schirmblech zwischen HF-Platine und Steuerung gebaut.

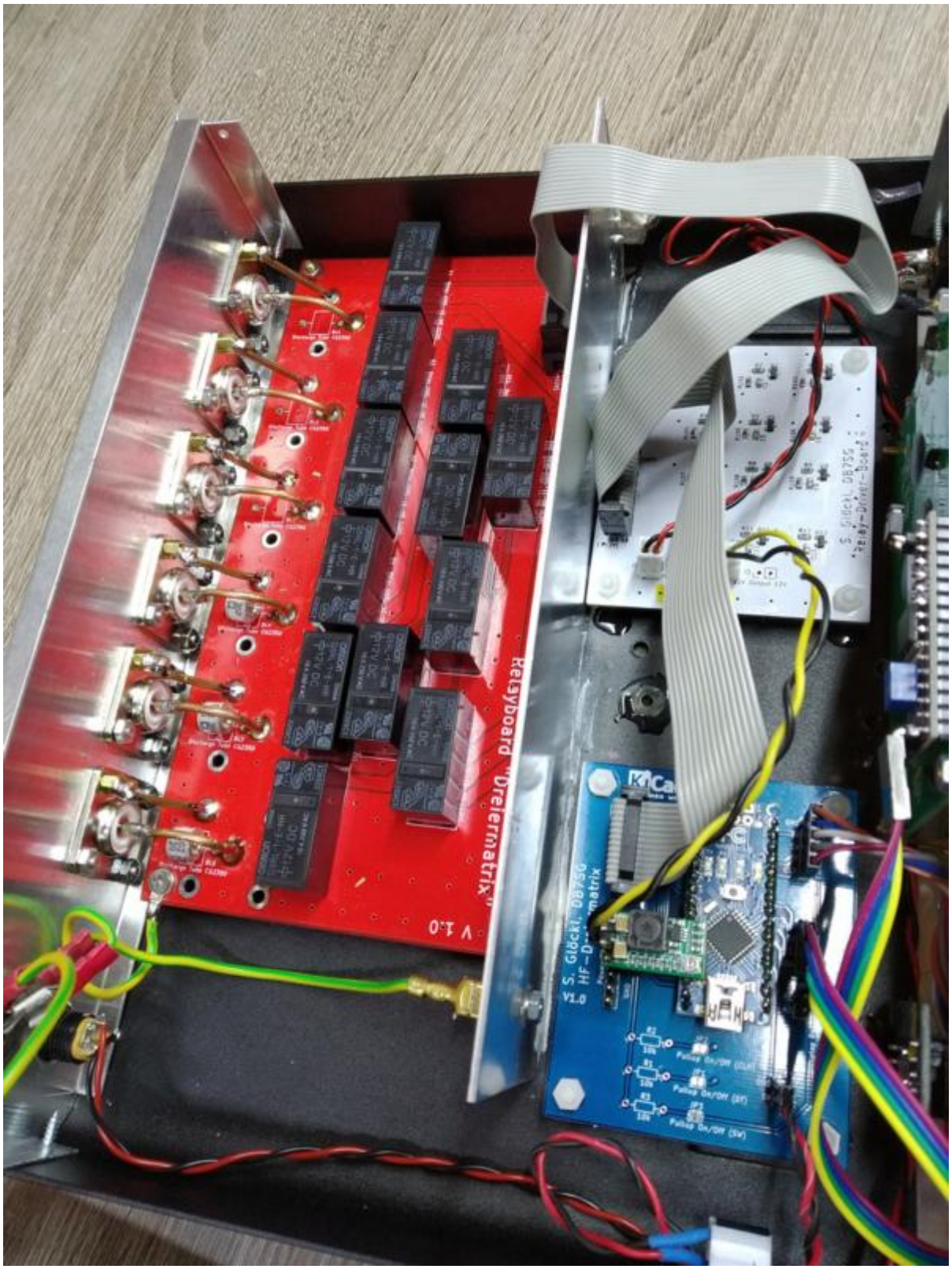




Fertig aufgebaut sieht das Ganze dann so aus.

Zusammenbau (liegende Version)

Ich habe, so als Anregung, noch eine flachere Version gebaut, mit liegendem HF-Board.

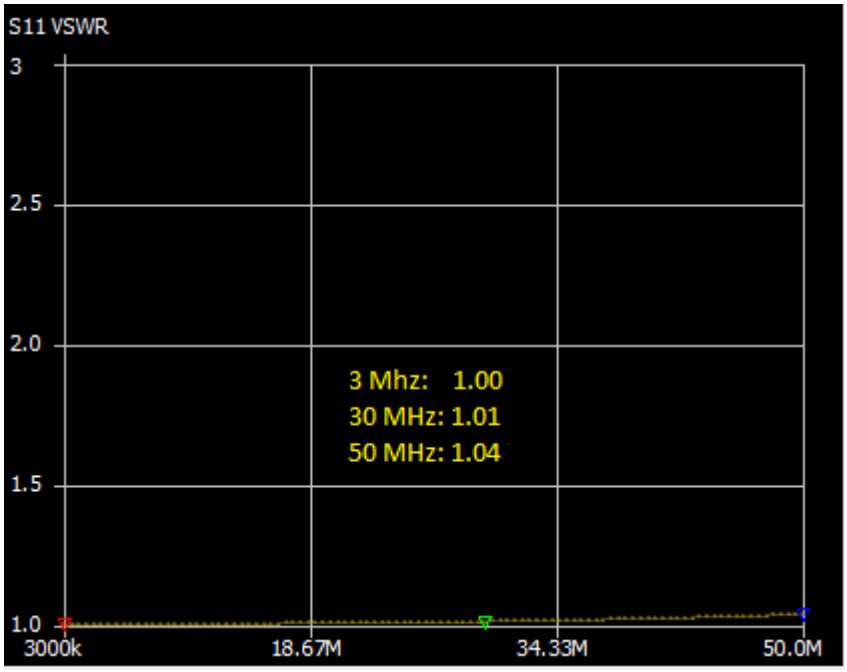


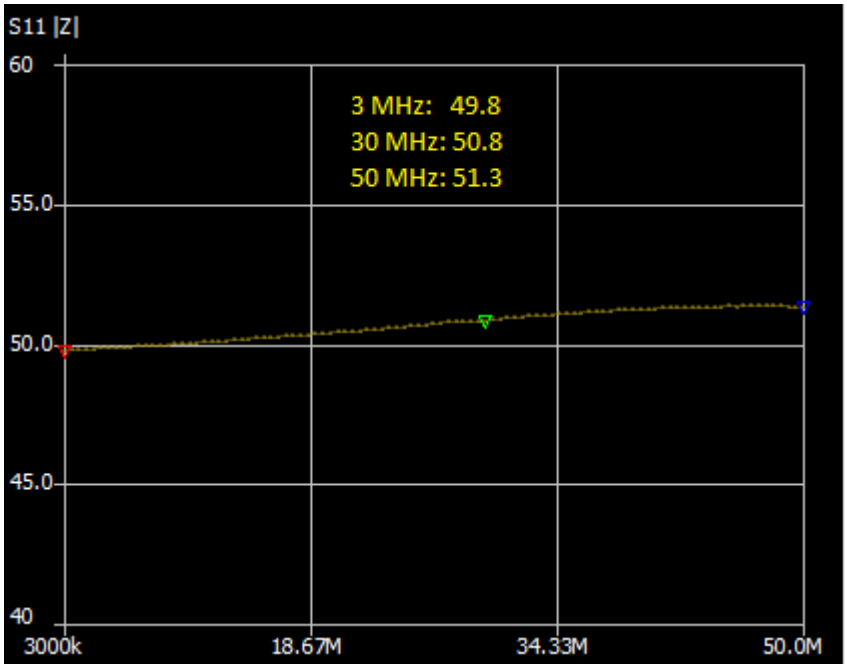
Die Black-Edition, in Carbon-Optik... 😉

Messungen

Also erstes hab ich mal geschaut, ob die Relais überhaupt so arbeiten, wie sie sollen, einfach mal die sechs möglichen Kombinationen schalten und Durchgang messen.

Die Impedanz und das SWR wurden durch alle Möglichen Schaltkombinationen über den Bereich 0 - 50MHz gegen einen Dummyload gemessen. Die Impedanz liegt im Messbereich immer um 50 Ohm, Das SWR geht gegen 1 und hebt sich Richtung 50 MHz leicht auf 1,04.





Die Dämpfung ist mit meinen Mitteln kaum messbar und liegt je nach Frequenz und Schaltweg zwischen 0,01dB und 0,05dB. Der kürzeste Weg durch die Matrix geht von TX/RX1 zu Antenne1, dann Antenne2, dann Antenne 3.

Downloads

- Gerberdaten für die Leiterplatten *(Kicad-Daten zum selbst Ändern gern auf Anfrage)*
- Stückliste (BOM) zu den drei Platinen
- Schaltpläne
- Firmware (Arduino-Sketch)
- Einbaurahmen LCD2004 (stl-Datei)