

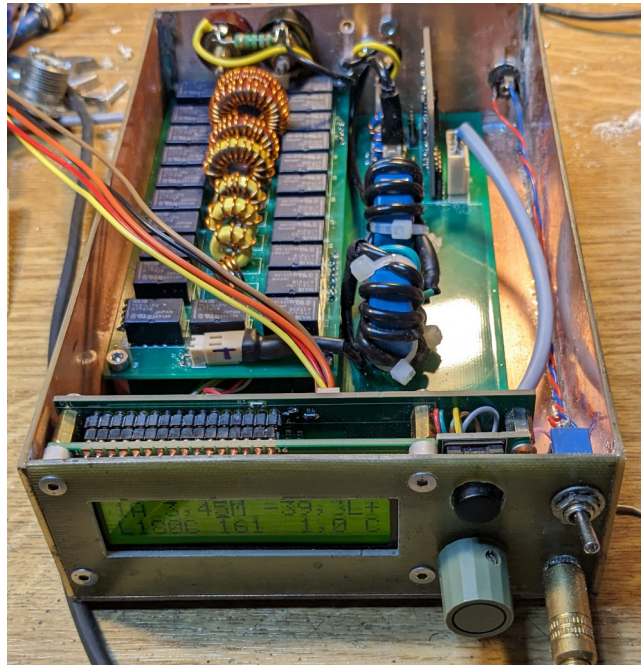
PicATUB QRP-Antennentuner mit bistabilen Relais

Andreas Lindenau DL4JAL

24. Juli 2024

Zusammenfassung

Der „PicATU20B“ ist ein Bastelprojekt von mir. Die FW im PIC habe ich neu geschrieben. Durch den Einsatz von bistabilen Relais, ist der Stromverbrauch sehr gering. Als Programmiersprache verwende ich Assembler. Assembler ist mir sehr geläufig. Ich habe eine große Sammlung an Sourcecode, den ich immer wieder verwenden kann.



Hier ist der PicATU20B zu sehen. Nach der Anpassung an die Antenne kann der PicATU20B aus geschaltet werden. Durch den Einsatz von bistabilen Relais bleibt die gefunden Einstellung auch im ausgeschalteten Zustand erhalten.

Inhaltsverzeichnis

1	Software	2
1.1	Display Inhalt	2
1.2	Power ON	3
1.3	Menü-Funktionen	4
1.4	SETUP-Funktionen	10
1.5	Sonder-Funktionen	20
1.5.1	Test des „Watchdog“	20
2	Schlusswort	21

Kapitel 1

Software

Die gesamte Software wurde in Assembler geschrieben. Der Haupt Quelltext hat etwa 8600 Zeilen. Hinzu kommen noch die Quelltexte für die Pakete Gleitkomma-Berechnung, HF-Formeln und komplexe Berechnungen. Die Software ist über mehrere Jahre gewachsen und wird bei jedem ATU etwas abgewandelt. Die Grundfunktionen zur Suche der richtigen Antennenanpassung ist aber bei all meinen entwickelten Tuner gleich.

1.1 Display Inhalt



- Display Zeile 1
 - **1A** Antenne 1 ist ausgewählt (Speicherbereich 1 im ext.Eeprom, 1,5MHz bis 30MHz alle 10kHz ein Speicherplatz)
 - **7,20c** Frequenz 7,20 MHz ist eingestellt. „c“ bedeutet das die Frequenz von der Fernbedienung gesendet wurde.
 - **-42,4** Sendepiegel -42,4 dBm, Im Stromspar-Modus (Hintergrundbeleuchtung aus) wird nichts angezeigt, da auch die Richtkoppler-Messplatine stromlos ist.
- Display Zeile 2
 - **L 0 L** hat den binären Wert von 0.
 - **C 96 C** hat den binären Wert 96
 - **0,2** Return Loss beträgt 0,2 dB, Im Stromspar-Modus (Hintergrundbeleuchtung aus) wird nichts angezeigt, da auch die Richtkoppler-Messplatine stromlos ist.
- Display Zeile 1 und 2 ganz rechts
 - **LC-Variante** In beiden Zeilen sehen wir die eingestellte LC-Variante.

1.2 Power ON

Nach *Power ON* werden alle Baugruppen initialisiert. Die bistabilen Relais werden in die richtige Stellung gebracht. Alle Relais werden der Reihe nach eingeschaltet und anschließend wieder ausgeschaltet, da der geschaltete Zustand der Relais unbekannt ist. Dabei ist zu beachten, dass die 8 Relais des L-Gliedes in Ruhe alle eingeschaltet sind. Es müssen ja alle Induktivitäten kurz geschlossen sein, wenn L den Wert=0 hat.

Anschließend wird die letzte gespeicherte Frequenz geladen und wenn vorhanden eine LC-Einstellung und die LC-Variante dazu. Der PicATU20B fängt *grundsätzlich* nicht automatisch an zu tunen. Die Tune-Funktionen können nur über das *Menü* gestartet werden.

Wurde im SETUP in der Funktion *Timer I-Redukt.* der Timer aktiviert, reduziert sich die Stromaufnahme nach der angegebenen Zeit auf etwa 10mA. Die Hintergrundbeleuchtung des Displays wird abgeschaltet und die Versorgungsspannung der Messplatine. Ansonsten verbraucht der Tuner etwa 70mA.

Ist der Stromreduzierung noch nicht aktiv, wird bei Senden des TRX im Display das SWR und die Sendeleistung angezeigt (die Messplatine hat ja noch ihre Versorgungsspannung und kann messen).



```
1A 7,20c -42,4L+  S 1,04 16,0mW L+
L 00 96 0,2 C    L 00 96 33,4 C
```

Links: normaler Betriebszustand. Rechts: Ich habe den TX mit nur 15mW eingeschaltet. Das SWR und die Sendeleistung ist im Display zu sehen.

Power OFF

Wurde eine Anpassung an die Antenne erfolgreich beendet, können wir den PicATU20B auch ausschalten. **Alle Relais bleiben im geschalteten Zustand!** Das ist der große Vorteil der bistabilen Relais.

Frequenz-Erkennung

Der PicATU20B hat keine Frequenzmessung. Die momentan benutzte Frequenz wird im 10kHz-Raster mit dem Drehgeber eingestellt. Ist passend zu dieser Frequenz eine Abstimmeneinstellung im externen Eeprom vorhanden, wird diese geladen und die Relais geschaltet. Mit der Taste im Drehgeber wird die Position des Cursors geändert, so dass auch 1 MHz Schritte beim Verstellen der Frequenz möglich sind. So kann ganz schnell der gesamte Kurzwellenfrequenzbereich eingestellt werden.

Externer Speicher für LC-Werte

Ich habe auf der Steuerplatine einen externen Eeprom-Speicher vorgesehen, der sehr groß ist (64kByte). Dadurch ist es möglich, im gesamten Kurzwellenfrequenzbereich alle 10kHz eine Abstimmeneinstellung zu speichern. Da bei einem Frequenzbereich von 1,5 MHz bis 30 MHz immer noch viel Speicherplatz vorhanden ist, habe ich noch 4 weitere Frequenz-Bereiche 1,5 MHz bis 30 MHz vorgesehen. Es gibt also 5 Speicherbereiche von 1,5 MHz bis 30 MHz, in denen alle 10 kHz eine Tuner-Einstellung gespeichert werden kann. Diese 5 Speicherbereiche

bezeichne ich als Antenne 1 bis 5. Pro Datensatz benötige ich 4 Byte. Für den Frequenzbereich 1,5 MHz bis 30 MHz benötige ich bei einem 10kHz-Raster 2850 Speicherplätze mit je 4 Byte. Das ergibt 11400 Byte. Folgende Speicherbereiche ergeben sich:

- **Antenne 1** Speicherplatz 0 bis 11399
- **Antenne 2** Speicherplatz 11400 bis 22799
- **Antenne 3** Speicherplatz 22800 bis 34199
- **Antenne 4** Speicherplatz 34200 bis 45599
- **Antenne 5** Speicherplatz 45600 bis 56999

Der Speicherbereich 57000 bis 65535 bleibt frei. Da könnte man noch eine Sicherung des Pic-Eeproms einrichten. Vielleicht mache ich das später noch einmal.

1.3 Menü-Funktionen

Mit der *Einzel Taste* kommen wir in die *Menü-Funktionen*. Es folgt der Erleuchtung der Funktionen.

Match, Match deep



In dieser Beschreibung habe ich beide Funktionen (*Match*, *Match deep*) zusammen gefasst. Die „Match-Funktion“ wurde von mir noch einmal grundlegend überarbeitet. Ich habe auf eine intelligente Flächensuche umgestellt. Die Fläche ergibt sich aus den Seitenlängen X(C:Kapazität) und Y(L:Induktivität). In unserem Fall ist X=C-Werte 0 bis 2047 und Y=L-Werte 0 bis 255. Das ist eine Fläche von 2047 x 255 Feldern, das ergibt multipliziert 512985 Felder. Das ist eine enorme Menge. Deshalb bin ich auf die Idee gekommen die erste Flächensuche mit exponentieller Schrittweite zu beginnen.

Grundmatch

Nachdem das Sendesignal erkannt ist beginnt die erste Such-Funktion *Grundmatch*. Das ist die Wichtigste. Die Schrittweite steigt quadratisch und es wird immer nur ein Relais der binären L-Kette und C-Kette eingeschaltet.

Das ist die erste Suche des SWR-Minimums. Die Suche beginnt mit C Wert=0 und L Wert=0. Es folgt der nächste Schritt mit C-Wert=8 und L-Wert=1. Bei jedem weiteren Schritt verdoppelt sich der Wert. Es ergeben sich 9x9 also 81 L/C Kombinationen. Die L- und C-Werte kann man als eine quadratische Fläche betrachten mit 81 Feldern.

L-Werte 0, 1, 2, 4, 8, 16, 32, 64, 128

C-Werte 0, 8, 16, 32, 64, 128, 256, 512, 1024

		C									
		0,0pF	12,5pF	25pF	50pF	100pF	200pF	400pF	800pF	1600pF	
		C-Glied	0	8	16	32	64	128	256	512	1024
L	L-Glied										
0,00uH	0										
0,125uH	1										
0,25uH	2										
0,5uH	4										
1uH	8										
2uH	16										
4uH	32										
8uH	64										
16uH	128										

Die Wertigkeit von L geht von 0 (0,00uH) bis 128 (16uH) und die Wertigkeit von C von 0 (0,0pF) bis 1024 (800pF). Die Suche auf der Fläche (gelb) beinhaltet 81 Felder. Bei jeder Kombination von L und C wird das Return-Loss gemessen. Das Feld mit dem höchsten Return-Loss merkt sich die Funktion und eine neue Suchfläche (hier grün) wird festgelegt. Die Schrittweite in der grünen Fläche wird neu gewählt, so das wieder etwa 64 Suchfelder entstehen.

Submatch

Submatch setzt die Suche fort mit reduzierter Suchweite in der quadratischen Fläche von L und C. Die Suche geht in unserem Beispiel C-Werte von 64 bis 256 und L-Werte von 8 bis 32 (oben grünes Feld). Die Schrittweite wird passend eingestellt. In unserem Beispiel C-Step=32 und L-Step=4. Mit der neue Schrittweite wird noch einmal die Suchfläche aktualisiert.

C-Wert	64	96	128	160	192	224	256
L-Wert							
8							
12							
16							
20							
24							
28							
32							

Jetzt ist die Schrittweite konstant linear. C Step 32 und L Step 4. Im Beispiel wurde neues RL maximum ist gefunden (Rot, L=16, C=96).

Die Funktion *Submatch* wird immer wieder wiederholt, mit reduzierter LC-Fläche und reduzierten Step.

C-Wert	64	80	96	112	128	144	160
L-Wert							
8							
10							
12							
14							
16							
18							
20							
22							
24							

Jetzt ist die Schrittweite halbiert. C Step 16 und L Step 2. Ein neues RL maximum ist gefunden (Rot, L=16, C=112).

Wiederholung der Funktion *Submatch* mit reduzierter LC-Fläche und reduzierten Step.

C-Wert	88	96	104	112	120	128	136
L-Wert							
12							
13							
14							
15							
16							
17							
18							
19							
20							

Die Schrittweite wird wieder halbiert. Bei C Step 8 und bei L Step 1. Ein neues RL maximum ist gefunden (Rot, L=15, C=104).

Die Schrittweite wird immer wieder halbiert und mit der Funktion *Submatch* neu gesucht. **Die Funktion *Submatch* wird beendet, wenn:**

- Ist das SWR kleiner 1,2 wird *Match* beendet. Der Match-Vorgang war erfolgreich. Die L/C Kombination wird gemerkt und verwendet und im externen Eeprom im 10kHz-Raster gespeichert.
- Ist das SWR 1,2 noch nicht erreicht aber beide Schrittweiten für L und C sind 1, wird *Match* ebenfalls beendet. Die beste L/C Kombination wird gemerkt und verwendet. Ist das SWR kleiner 1,5 wird diese Einstellung auch im externen Eeprom im 10kHz-Raster gespeichert.

So finde ich ganz schnell eine Impedanzanpassung. Es dauert im Idealfall etwa 10 bis 15 Sekunden.

Ist am Beginn der *Match-Funktion* im *Grundmatch* in der ersten L/C Variante das SWR < 2,5, wird in dieser L/C Variante weiter gesucht bis das SWR Minimum erreicht wurde. Ist im *Grundmatch* das SWR > 2,5, wird sofort in die nächste L/C Variante gewechselt, wieder beginnend mit *Grundmatch* usw...

Zuerst wird angenommen, dass die Antennenimpedanz $> 50 \text{ Ohm}$ ist, wie das meistens der Fall ist. **Folgende Suchreihenfolge habe ich programmiert:**

- **Variante1:** L/C Glied Impedanz größer 50 Ohm , Grundmatch, Submatch...
- **Variante3:** C/L Glied Impedanz größer 50 Ohm , Grundmatch, Submatch...
- **Variante2:** L/C Glied Impedanz kleiner 50 Ohm , Grundmatch, Submatch...
- **Variante4:** C/L Glied Impedanz kleiner 50 Ohm , Grundmatch, Submatch...
- Ist bei einer Varianten das beste $\text{SWR} > 2,5$, wird sofort in der nächsten Variante gesucht. Bei *Match deep* wird nicht bei $\text{SWR} > 2,5$ abgebrochen, sondern bis zum Ende weiter gesucht. Bis das $\text{SWR} < 1,2$ ist oder beide Schrittweiten 1 sind.
- Wird mit *Match* keine Anpassung gefunden sollte mit *Match deep* erneut eine Anpassung gesucht werden.
- Ein erneutes „Match deep“ bricht nicht mehr bei „ $\text{SWR} > 2,5$ “ ab, sondern sucht immer bis zum Ende (bis beide Schrittweiten 1 sind). So ist die Suche nach dem besten SWR intensiver. Es wird fast immer ein vertretbares SWR gefunden.
- Ist das $\text{SWR} < 1,5$, werden die Werte von L, C und die L/C Variante im Speicher des „10kHz-Segmentes“ abgelegt.

Zusätzlich kann man noch die gefunden Einstellung mit dem Befehl (*B*)and für das ganze KW-Band abspeichern (in alle 10kHz-Segmente des Bandes).

ReMatch

```
--- Menu --- OK
ReMatch      X
```

Grundsätzlich sollte man beim Wechsel in einem anderen 10kHz-Bereich noch einmal *ReMatch* aufrufen und Nachstimmen. *ReMatch* sucht so lange bis die Schrittweite bei L und C gleich 1 ist.

Auch hier gilt wieder, ist das $\text{SWR} < 1,2$ wird die gefundene Einstellung im 10 KHz.Segment des externen Eeprom abgespeichert. Oft werden aber noch kleiner SWR-Werte erreicht.

ReMatch deep

```
--- Menu --- OK
ReMatch deep X
```

Die Funktion *ReMatch deep* habe ich für schwierige Fälle noch programmiert. Vorher wird manuell die LC-Variante gesetzt und anschließend *ReMatch deep* gestartet. *ReMatch deep* beginnt mit der Suche von ganz vorn *Grundmatch*, *Submatch* usw..

Auch bei den LC-Varianten „nur L“ und „nur C“ ist *ReMatch deep* zu verwenden.

Band Seg.save

```
--- Menu --- OK  
Band Seg.save X
```

Wird das erste mal eine Anpassung im KW-Band gesucht ist es ratsam die gefundene Anpassung für alle 10kHz-Segment des KW-Bandes zu speichern. An Hand der eingestellten Frequenz weiß die SW welches KW-Band gemeint ist.

Steht die Frequenz im Tuner als Beispiel auf 3,76 MHz, weiß die SW das ist das 80m-Band und speichert die gefundene Einstellung in allen 10kHz-Segmenten von 3,4MHz bis 3,9MHz ab. Also noch 100kHz zusätzlich unterhalb und oberhalb des Bandes.

Das hat den Vorteil, wir haben im gesamten 80m-Band eine einigermaßen gültige Anpassung die mit *ReMatch*, *ReMatch4x4* oder *ReMatch8x8* optimiert werden kann.

10kHz Seg.save

```
--- Menu --- OK  
10kHz Seg.save X
```

Diese Funktion speichert die LCV-Einstellung im 10kHz-Segment ab, auch wenn das SWR sehr schlecht sein sollte. Es gibt Situationen da ist das nötig.

```
Seg.10kHz var OK  
7,03M- 7,03M X
```

```
Seg.10kHz var OK  
7,00M- 7,06M X
```

Neu ist jetzt das Speichern von 10kHz-Segmenten in der Nachbarschaft. Das ist bei höheren Frequenzen sinnvoll. Mit dem Drehgeber wird in die Nachbar-segmente rechts und links erweitert. Der Frequenzbereich wird im Display angezeigt.

In den beiden Bilder ist links der normale Speicherbereich, nur ein Segment 7,03MHz. Rechts habe ich mit dem Drehgeber den Speicherbereich erweitert auf 7,00 MHz bis 7,06 MHz.

Die Einzeltaste Speichert die Einstellung ab. Der Taster im Drehgeber ist *Abbruch*.

10kHz Seg.ab

Diese Funktion ist ab FW 1.09 vorhanden und ist ähnlich der vorherigen Funktion. Besser ist das der Speicherbereich getrennt nach unten (a) und oben (b) erweitert werden kann. So kann man gezielt einen definierten Frequenzbereich mit der gefundenen Einstellung belegen.

```
--- Menu --- OK  
10kHz Seg.ab X
```

```
Seg.10kHz 62 OK  
3,50M- 3,58M X
```

Links der neue Menüpunkt. Rechts die Funktion. Der Speicherbereich wurde erweitert, siehe Zeile 2.

Einzeltaste kurz gedrückt Einstellung wird genommen und im eingestellten Frequenzbereich abgespeichert.

Drehgeber-Taste kurz gedrückt Der Cursor wechselt zwischen Anzahl Segmente nach unten (6) oder oben (2), Zeile 1 im Display (62).

Tasten lange gedrückt Abbruch der Funktion.

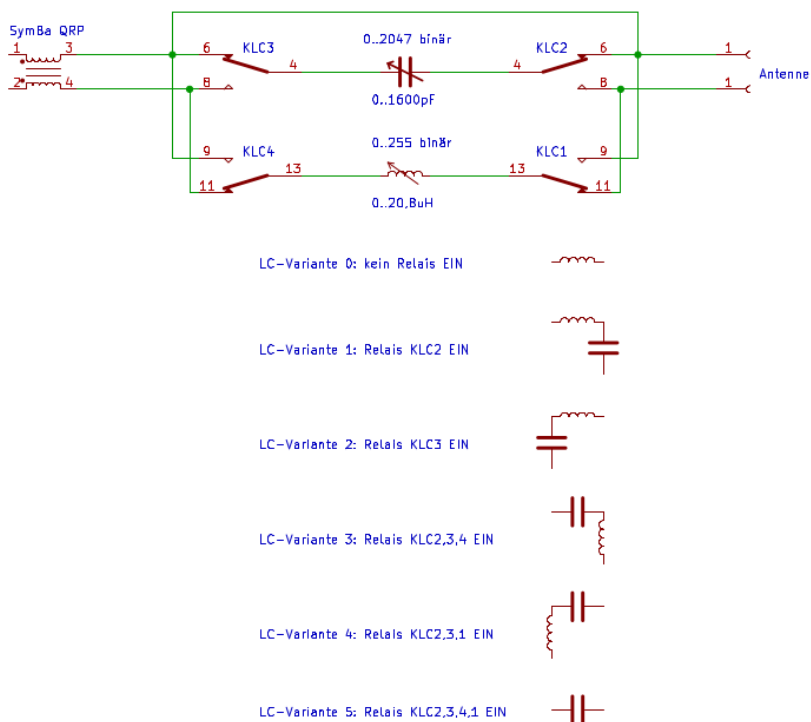
LC-Variante



Hier habe ich L und C untereinander vertauscht. Das ergibt manchmal eine bessere Anpassung.

In dieser Funktion kann die LC-Variante manuell geändert werden.

Das Prinzip der LC-Varianten



Ant.Impedanz

```
--- Menu --- OK |Z|= 20,8Ω OK  
Ant.Impedanz X 0,3- 20,8j
```

Das ist nur ein Beispiel.

Wurde eine gute Anpassung gefunden, berechnet diese Funktion mit der eingestellten Frequenz, der LC-Variante und den Werten von L und C die Antennenimpedanz. Der Zahlenwert der Impedanz stimmt natürlich nicht ganz genau, aber eine gute Orientierung ist das auf alle Fälle.

ReMatch 4x4

```
--- Menu --- OK  
ReMatch 4x4 X
```

Ist die Abweichung der Anpassung sehr gering kann man ganz schnell mit dieser *ReMatch-Funktion* das SWR verbessern.

ReMatch 8x8

```
--- Menu --- OK  
ReMatch 8x8 X
```

Auch diese *ReMatch-Funktion* verbessert eventuell das SWR. Es wird ein etwas größerer Bereich abgesucht.

1.4 SETUP-Funktionen

Ein *langer Tastendruck* der Einzeltaste startet das SETUP-Menü.

Abbruch!

Das bedarf keiner Erklärung. Abbruch!

TRX Cat. select

Diese Funktion wählt den Typ des TRX aus.

Auf der mc-Platine ist eine RS232-Schnittstelle für die TRX-CAT-Verbindung vorhanden. Auf der Platine sind 3 gleichberechtigte 5-polige Steckerbuchsen für verschieden HW-Varianten. Die Hardware der RS232-Schnittstellen müssen ja zum HW-Protokoll des TRX passen.

Folgende HW-Schnittstellen habe ich vorgesehen:

5V RX-TTL Diese Schnittstelle hat in beiden Richtungen TTL-Pegel 5V entspricht dem HIGH-Pegel und 0V entsprechen dem LOW-Pegel. Mein TRX PicAStar und vom Icom die CI-V Schnittstelle arbeitet ebenfalls mit TTL-Pegel.

RS232 Eine kleinen Platine mit einem RS232-IC (z.B.:MAX232) bildet die echte RS232-Schnittstelle mit einem SUB-D9 Stecker oder Buchse. Je nach dem wie es gewünscht wird. Ich habe einen weiblichen SUB-D9. Bei dieser HW ist etwa -8V der HIGH-Pegel und +8V der LOW-Pegel.

USB extern Die USB-HW ist ebenfalls auf einer kleinen Platine mit dem IC FT232RNL.

Die 3 Stecker J8 bis J10 arbeiten parallel. Es darf jeweils nur eine Möglichkeit benutzt werden, sonst gibt es „Datensalat“. Alle Steckplätze sind gleichberechtigt Die MC-Platine wurde auch mit Kicad neu entworfen. **Hier noch mal die zwei verschieden Module:**

echte RS232 mit DB9 Buchse Das ist eine echte RS232-Schnittstelle, die einige TRX benötigen. Mit einem 9-poligen Stecker und einem IC *MAX 232 CWE* im SMD SOL16 Gehäuse. Also etwas breiter als normal.

echte USB mit FT232RNL Der FT232RNL ist eine virtuelle USB-RS232 in Richtung PC. Der IC FT232RL befindet sich jetzt nicht mehr auf der Hauptplatine sondern wurde in dieses USB-Modul ausgelagert.

In der Software habe ich folgende TRX berücksichtigt:

PicAStar(DL4JAL), TTL mit 3,5mm Stereo Klinkenbuchse Der CAT-Anschluss ist ein Ausgang mit 5V TTL-Pegel. Bei jedem VFO-Frequenzwechsel wird sofort die neue Frequenz ausgegeben.

Yaesu FT847, RS232 mit SUBD9 Buchse 9600-8N-2 Anschluss für den FT847

Icom CI-V Remote, TTL mit 3,5mm Stereo Klinkenbuchse für den Icom CI-V Anschluss

K2/K3 Elecraft, RS232 mit SUBD9 Buchse Funktioniert erst ab FW 1.13 richtig. Baudrate ist fest eingestellt 4800 Baud.

PC PowerSDR, USB-Buchse oder RS232 Für Windows-Software „PowerSDR“.

Yaesu FTDX101, RS232 mit SUBD9 Buchse Anschluss für FTDX101. Einstellung 9600Baud.

Yaesu FT1000MP Field, RS232 mit SUBD9 Buchse Anschluss FT1000MP-Field, 4800Baud.

Yaesu FT1000MP, RS232 mit SUBD9 Buchse Anschluss FT1000MP, 4800Baud.

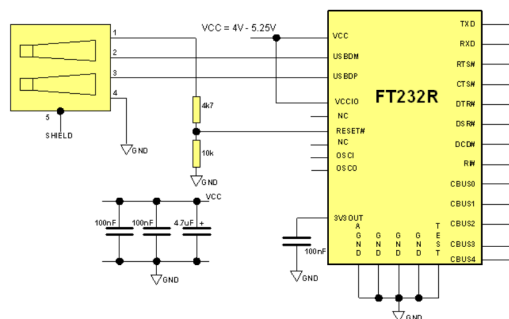
Die Auswahl mit dem Drehgeber. Mit der Einzeltaste bestätigt. Ein *langer Tasterdruck* führt zum Abbruch der Funktion.

Beschreibung der „PowerSDR“ Schnittstelleneinrichtung (PC-Software)

Zuerst stecken wir die Verbindung mit USB-Kabel „Remote-BG“ und „PC“. Im PC wird eine neue serielle Schnittstelle sichtbar. Jetzt machen wir im „PowerSDR“ folgende Einstellungen. Im „SETUP, CAT Control“ wählen wir die neue Schnittstelle aus (9600,none,8,1). Es folgt noch **ID as** „PowerSDR“ und bei „Allow Kenwood AI Command“ setzen wir den Haken. Jetzt noch „Enable CAT“ Haken setzen und mit „OK“ bestätigen. PowerSDR sendet jetzt bei jeder Frequenzänderung ein Kommando zu unserer Remote-BG.

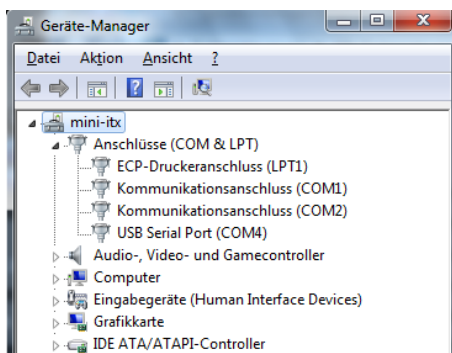
Die Inbetriebnahme der USB-Schnittstelle gestaltet sich manchmal etwas schwierig. Hier einige Tips dazu. Ich habe eine zusätzliche Beschaltung des IC FT232RL vorgenommen.

6.2 Self Powered Configuration

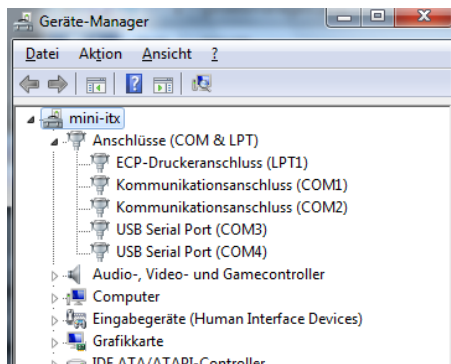


Die Beschaltung des FT232RL, wenn die Versorgungsspannung Baugruppe und 5 Volt USB getrennt sind. Der FT232RL wird beim anstecken des USB-Kabels neu gestartet (Reset).

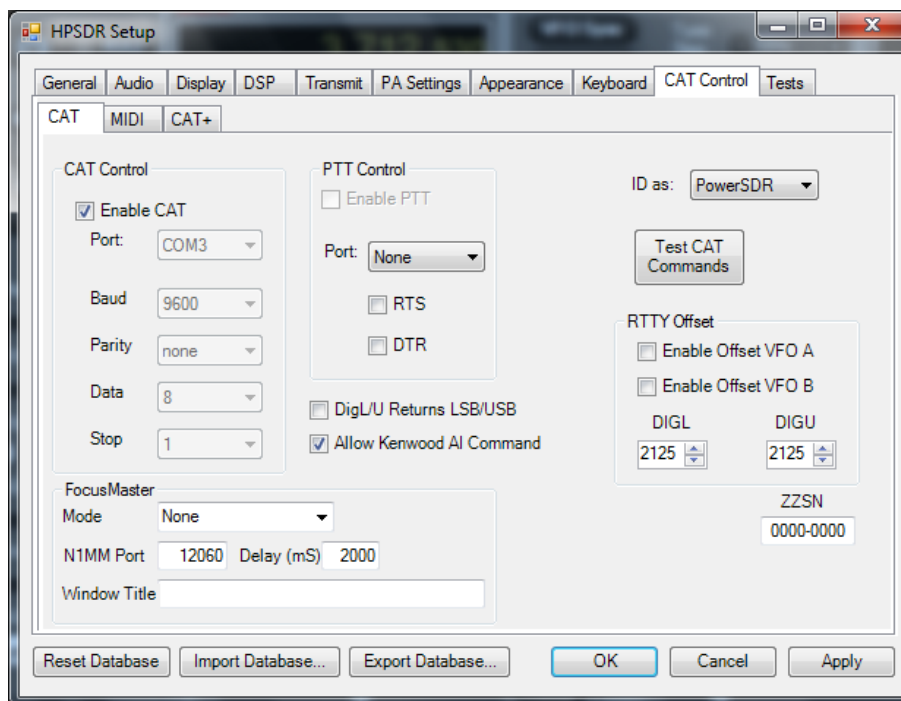
Als erstes öffnen wir am PC die „Systemsteuerung Gerätemanager“. Wird das USB-Kabel am PC angesteckt entsteht eine neue virtuelle RS232. Im „Gerätemanager“ ist das Online sichtbar. Als Test können wir das USB-Kabel mehrmals anstecken und wieder abziehen.



Das USB-Kabel ist noch nicht angesteckt. COM1, COM2 und COM4 sind zu sehen.



Das USB-Kabel wurde angesteckt. Wir sehen die neu entstandene COM. In diesem Fall die COM3.



Jetzt können wir im „PowerSDR“ die neu entstandene Schnittstelle im Dialogfenster einstellen und benutzen. Wir sehen alle „CAT Einstellung“ im „PowerSDR“.

Auf der MC-Platine wird mit den Dioden D3, D4, D5 und D6 alle ankommenden TX-Daten entkoppelt und können ohne Umschaltung parallel genutzt werden. Es darf aber nur immer eine Schnittstelle aktiv sein sonst gibt es „Datensalat“.

Wichtig ist, es darf nur eine TRX-Schnittstelle aktiv sein.

MK.Return Loss

In der PDF der Hardware und Aufbau des PicATU20B wurde diese Funktion schon behandelt. Ist der Richtkoppler aufgebaut und die Messplatine gesteckt, wissen wir nicht ob der Der Richtkoppler mit der Messplatine korrekt funktioniert. Im Kapitel Inbetriebnahme könnt ihr das nachlesen.

Kalib. Vorlauf

Der PicATU20B funktioniert auch ohne ein spezielle Kalibrierung des Vorlaufs und Rücklauf. Möchte man aber genauere Angaben der Sendeleistung kommt man um eine Kalibrierung der beiden AD8307 nicht herum.

Für die Kalibrierung benötigen wir eine möglichst genauen 0,0dBm-Generator und ein Dämpfungsglied 30dB.



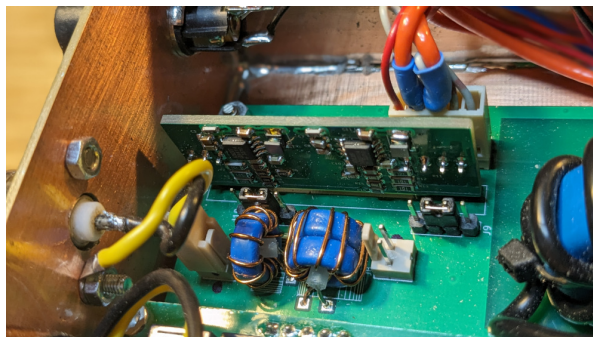
Mein 0,0dBm Generator mit aufgeschraubten Dämpfungsglied 30dB von *Mini-Circuits*.

Ich habe mehrere Möglichkeiten einen genauen Pegel zu erzeugen. Mein Homemadedoppel-DDS-VFO geht auch sehr genau im Pegel und Frequenz.



Doppel-VFO -140dBm bis etwa +3dBm in 0,1dB Schritten, IM3 -90dBc.
Frequenz 0,1Hz bis 150MHz.
Leistungsausgang Doppel-Pegel bis 15 Watt PEP IM3 -60dBc.
Frequenz 1,6 MHz bis 55 MHz.

- Zuerst wird die Brücke zwischen Richtkoppler und Messplatine geöffnet und das Kabel zum Generator angesteckt.



Wir sehen die beiden Brückenstecker. Links ist für den Rücklauf und rechts für den Vorlauf.



Das Kabel zum Generator wird angesteckt.

- Wir starten die Kalibrierfunktion und sehen im Display den Messwert des A/D Wandlers bei $0,0\text{dBm}$ und anschließend bei -30dBm .

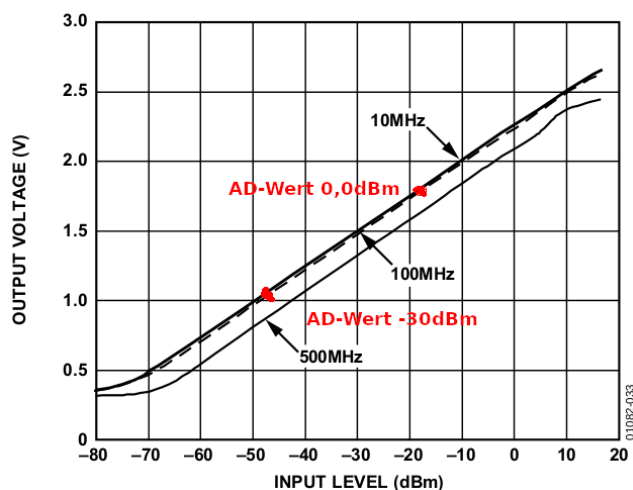
```

Vorlauf      OK      Vorlauf      OK
0dBm:11004  X      -30dBm: 6055  X
  
```

Drücken wir *OK* werden die beiden Kalibrierergebnisse ausgerechnet und abgespeichert.

Die beiden Messergebnisse von „11004“ bei $0,0\text{dBm}$ und 6055 bei $-30,0\text{dBm}$ erscheinen sehr hoch. Der A/D-Wandler im PIC18F26K22 hat nur 10Bit Wandlerbreite. 10Bit ergeben ein maximale Zahl von 1023. Ich habe aber in der Messfunktion der Software eine Messschleife mit 32 Messungen hintereinander. Die Werte werden addiert. 32 mal 1023 ergibt eine maximale Zahl von 32736.

Wer sich für die Berechnungen interessiert kann hier weiter lesen. Zuerst betrachten wir die Messfunktion im Datenblatt des AD8307.



Ich habe das Diagramm aus dem Datenblatt heraus kopiert und die 2 Messpunkte für $0,0\text{dBm}$ und $-30,0\text{dBm}$ eingezeichnet. Bei beiden Messpunkten müssen wir 18dB abziehen. Die Eingangsbeschaltung auf der Messplatine hat 18dB Dämpfung, die wir berücksichtigen müssen.

Im Vorfeld eine Erklärung der verwendeten Bezeichnungen in den Formeln:

Bezeichner	Erklärung
dBmpegel1	Messpegel1 in dBm
dBmpegel2	Messpegel2 in dBm
adcwertpunkt1	Werte des AD Wandlers im PIC beim Pegel 1
adcwertpunkt2	Werte des AD Wandlers im PIC beim Pegel 2
mkx	Wert X der linearen Funktion
mky	Wert Y der linearen Funktion

Formel 1: $mkx = \frac{dbmpegel1 - dbmpegel2}{adcwertpunkt1 - adcwertpunkt2}$

Formel 2: $mky = (adcwertpunkt1 * mkx * -1) + dbmpegel1$

Da der erste Pegel 0dBm beträgt, vereinfacht sich die Formel etwas.

Formel 1: $mkx = \frac{-30dBm}{adcwertpunkt1 - adcwertpunkt2}$

Formel 2: $mky = adcwertpunkt1 * mkx * -1$

Mit diesen beiden Werten aus der Berechnung der Kalibrierung können wir aus der Integer-Zahl des ADC-Wertes (0.32736) den Pegel in dBm berechnen.

Wir setzen die Zahlen der Kalibrierung in die Formeln ein:

$$mkx = \frac{-30dBm}{11004 - 6055} = 0,006062$$

$$mky = 11004 * 0,006062 * -1 = -66,706$$

Mit dem beiden Konstanten mkx und mky können wir nach jeder A/D-Wandlung jetzt mit einem Schritt den dBm-Wert berechnen. Die Formel für die Berechnung des HF-Pegels in dBm ist einfach und lautet:

$$Pegel(dBm) = ADCWert * mkx + mky$$

Zur Kontrolle setzen wir die beiden A/D-Werte für 0,0dBm und -30,0dBm in die Formel ein. Zuerst 11004 für 0,0dBm

$$Pegel(dBm) = 11004 * 0,006062 + -66,706 = 0,000248dBm$$

und 6055 für -30dBm

$$Pegel(dBm) = 6055 * 0,006062 + -66,706 = -30,00059dBm$$

Die Ergebnisse weichen nach der 4 Stelle hinter dem Komma ab, da ich Mx und My etwas gerundet habe.

So einfach ist es mit „Mx“ und „My“ aus dem Wert des ADC-Wandlers den Pegel in dBm zu errechnen.

Ich hoffe ihr habt meine Berechnungen verstanden.

Kalib. Ruecklauf

Die Kalibrierung des Rücklaufes ist wie oben beschrieben der gleiche Ablauf.

Timer I-Redukt.

```
==== SETUP =====  
Timer I-Redukt.
```

Der PicATU20B verbraucht etwa 50mA Strom bei 9Volt. Der Tuner kann also mit einer kleinen 9 Volt Blockbatterie betrieben werden. Damit nicht ständig 50mA fließen, kann nach einer vorgegebenen Zeit die Hintergrundbeleuchtung und die Spannung der Messplatine abgeschaltet werden. Der Stromverbrauch reduziert sich dadurch auf etwa 10mA. Der PIC bleibt noch aktiv, aber die online Anzeige im Display beim Senden geht nicht mehr, da an der Messplatine die Spannung fehlt.

```
--- I-Redukt. --   --- I-Redukt. --  
[0] keine         [1] nach 10 Sek
```

```
--- I-Redukt. --  
[5] nach 30 Min
```

Verschiedene Einstellungen.

View MKxy Vor

```
==== SETUP =====  
View MKxy Vor
```

Anzeige der Kalibrierwerte vom Vorlauf. Es werden der Reihe nach angezeigt:

- mkx und mky als Float-Zahl in exponentieller Darstellung
- Die beiden A/D-Werte von 0,0dBm und -30dBm. Die beiden Zahlen sollte man sich aufschreiben für das *manuelle Kalibrieren*. Da werden nur die beiden Zahlen eingetragen und man ist fertig ohne einen HF-Pegel Einspeisung.
- Als Abschluss die beiden Konstanten mkx und mky als HEX-Zahl. Das ist nur für mich interessant als Programmierer.

```
kx:+6,121199e-3   0dBm:11058  
ky:-6,768822e+1  -30dBm: 6157
```

```
kx:3BC8945861  
ky:C287605F46
```

View MKxy Rueck

Anzeige der Kalibrierwerte vom Rücklauf. Siehe vorherige Beschreibung.

Relais-Test

```
==== SETUP ====  
Relais-Test
```

Der Relais-Test wurde schon bei der Inbetriebnahme beschrieben. Es können alle Relais auf ihre Funktion getestet werden.

```
---- Relais ----  ---- Relais ----  
KL1 L-Gl. 0,08uH  KL8 L-Gl. 10,4uH
```

KL1 bis KL8 Relais.

```
---- Relais ----  ---- Relais ----  
KC1 C-Gl. 0,8pF  KC11 C-Gl. 800pF
```

KC1 bis KC11 Relais.

```
---- Relais ----  ---- Relais ----  
KLC1 ----- SymBa  KLC4 --- Antenne
```

KLC1 bis KLC4 Relais.

Kal.manuell Vor

Das ist die Funktion, die ich schon kurz bei *View MKxy Vor* erwähnt habe. Es werden nur die beiden Integer-Zahlen eingetragen für $0,0dBm$ und $-30dBm$ eingetragen. Daraus berechnet die Software die Konstanten mkx und mky .

Kal.man. Rueck

Diese Funktion ist wie *Kal.manuell Vor*.

View dBm+A/D,V+R

```
==== SETUP ====  
View dBm+A/D,V+R
```

Anzeige der dBm und A/D-Werte für Vorlauf und Rücklauf. Diese Funktion dient der Kontrolle der Messungen.

```
U: -62,4dBm  865  U: -20,5dBm  7712  
R: -58,9dBm  970  R: -19,3dBm  7616
```

```
U: -20,7dBm  7680  
R: -52,3dBm  2069
```

3 verschieden Anzeigen beim Experimentieren.

SET dBm Minimum

```
==== SETUP ====  
Set dBm Minimum
```

Hier wird der minimale dBm-Wert eingestellt. Oberhalb dieses Pegels beginnt der Tuner zu arbeiten *Match* usw...

```
Min:10dBm 10,0mW  
AD-Wert: 9424
```

SET dBm Maximum

```
==== SETUP ====  
Set dBm Maximum
```

Das ist der maximale Pegel beim Tunen. Oberhalb dieses Pegels werden keine Relais mehr geschaltet. Das ist ein Schutz der Relais bei schalten.

```
Max:39dBm 7,94W  
AD-Wert:14162
```

Antenne Clear

Der Speicherbereich im Eeprom für die entsprechende Antennennummer (Speicherbereich) wird gelöscht. Alles LCV-Werte gehen verloren.

Antenne Copy

Kopieren von einer Antennen-Nummer zu einer beliebigen anderen Antennen-Nummer. Alle LCV-Werte werden kopiert.

Antenne Select

Die Auswahl der Antenne (Speicherbereich 1..5) erfolgt hier.

HF-Att. ON/OFF

In der HW wurde ein zusätzliches Relais eingefügt. Das Relais befindet sich am Eingang des Tuners und wird immer bei jedem Abstimmvorgang eingeschaltet. Das Relais aktiviert ein HF-Leistungsdämpfungsglied. Ist die neue HW eingebaut, kann mit dieser Funktion das HF-Leistungsdämpfungsglied auch wieder deaktiviert werden. Die Sendeleistung wird ohne Dämpfung durchgeleitet, wie gehabt. Oder es wird aktiviert. Dann zieht bei jedem Abstimmvorgang als erstes dieses Relais und schaltet das HF-Leistungsdämpfungsglied ein. Ist die Anpassung beendet fällt das Relais wieder ab.

```
1A 7,03M -39,2L+  
L 450 182 1,00C
```

Ab FW 1.08, im Display rechts Zeile 2 ist ein Widerstandssymbol zu sehen. Das HF-Dämpfungsglied ist aktiv.

1.5 Sonder-Funktionen

1.5.1 Test des „Watchdog“

In der Firmware des PIC18F26K22 habe ich den „Watchdog“ aktiviert. Der „Watchdog“ kann nicht ausgeschaltet werden. In der Watchdog-Funktion des PIC ist ein Zähler der einstellbar ist. In der Firmware muss die Zähler mit dem Assembler-Befehl „CLRWTD“ auf NULL gesetzt werden. Geschieht das nicht, zum Beispiel bei einem Software-Fehler (Endlosschleife, usw...) wird ein „RESET“ ausgelöst und die Software beginnt von vorn, wie bei „PowerON“, zu arbeiten. Man kann aber den Unterschied im Register „RCON“ erkennen ob der Watchdog verantwortlich ist für den Neustart.

Damit wir die Funktion des Watchdog testen können habe ich noch zwei Funktionen programmiert. Die eine Funktion ist der Test, ob der Watchdog funktioniert. Die zweite Funktion ist zu schauen wie viele „Watchdog-Resets“ es gab.

Taste bei PowerON

Eine Möglichkeit ist die Einzeltaste bei „PowerON“ gedrückt zu halten. Im Display sehen wir den Funktionsablauf.

Einzeltaste mehr als 3 Sek. drücken

Im Display sehen wir den Funktionsablauf.

Drehgebertaste mehr als 3 Sek. drücken

Im Display wird die Anzahl der „Watchdog-Naustarts“ angezeigt. Normalerweise bleibt der immer auf NULL stehen.

Kapitel 2

Schlusswort

Dieses Projekt darf nicht kommerziell vermarktet oder genutzt werden. Alle Rechte liegen bei DL4JAL (Andreas Lindenau). Ich wünsche viel Spaß beim Basteln.

vy 73 Andreas DL4JAL

✉ DL4JAL@t-online.de

🌐 www.dl4jal.eu