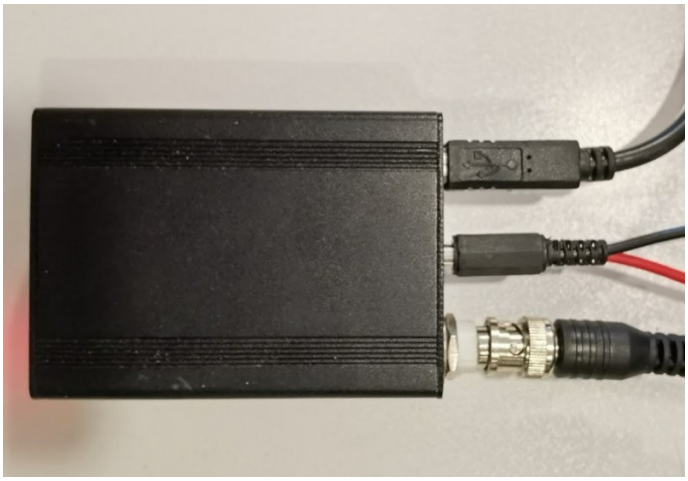
QRP Labs

QDX

QDX: QRP Labs Digital Xcvr (transceiver)

Assembly, design and operating manual



1. Introduction

The QDX is a high performance, four-band 5W Digital modes transceiver with CAT control and built-in USB sound card. QRP Labs presents QDX, a digital transceiver with a ratio of performance to price not available until now.

- Four band 80, 40, 30 and 20m; 5 W from 9 – 10V supply
- Clean single signal output (zero residual carrier, zero unwanted sideband)
- Solid state PIN-diode switched Low Pass Filters and solid state Band Pass Filters
- Solid state transmit/receive switching
- High performance embedded-SDR SSB receiver using 110dB 24-bit stereo ADC chip
- Built-in USB sound card: 48ksps 24-bit stereo
- Built in USB Virtual COM port serial for CAT control
- Si5351A Synthesized local oscillator with better than 0.001Hz resolution and high precision 25MHz TCXO reference as standard
- Built-in signal generator
- Built-in suite of configuration and analysis tools
- Lifetime free firmware upgrades with QRP Labs Firmware Update (QFU) bootloader for easy firmware update on any OS with no extra software, or drivers, or programming hardware
- All SMD components pre-installed by factory, only through-hole component soldering by the kit constructor
- Receive current: 100mA; Transmit current 1.0 – 1.1A (9V supply, 5W output)
- Only three connectors: USB (audio and serial for CAT), Power and RF
- Optional smart aluminium extruded enclosure measuring just 89 x 63 x 25mm

No test equipment is required to build, align and operate this digi modes transceiver. There are no alignment tasks.

We hope you enjoy building and operating this kit! Please read this manual carefully, and follow the instructions step by step in the recommended order. Later in the manual the circuit design is described in detail and we recommend reading and understanding this section too, to get the maximum enjoyment and education from your new radio.

Typical performance measurements are shown in the measurements section. The operation section will get you started with QDX and your WSJT-X or other digi modes software in minutes.

PLEASE READ THE BASIC ASSEMBLY AND USE INSTRUCTIONS IN THIS MANUAL VERY CAREFULLY BEFORE APPLYING POWER TO THE BOARD!

IMPORTANT!

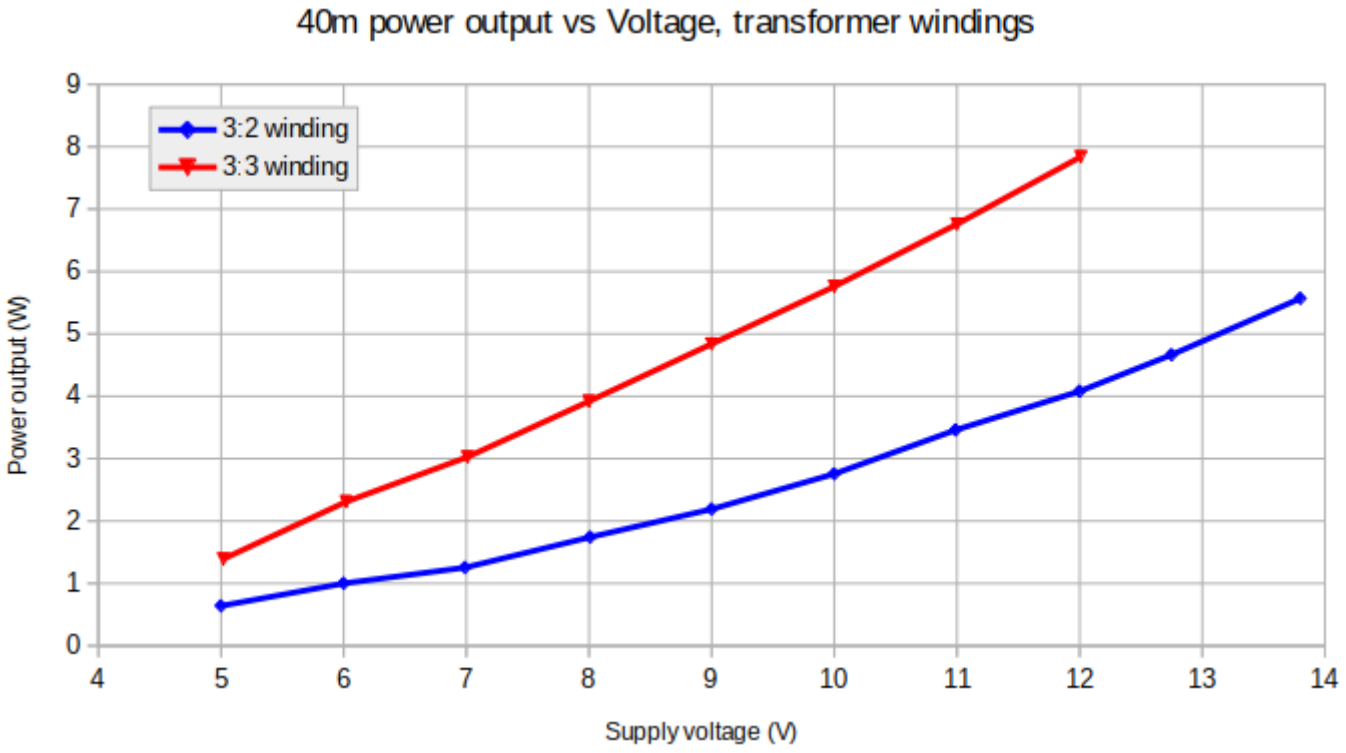
QDX can be built for 9V or 12V operation! You need to decide NOW!

A 9V QDX produces 5 W power output from a supply voltage of 9V or a little over. At 12 V QDX could be producing 8 W power output which is likely to cause over-heating and perhaps failure of the BS170 final transistors.

Operation of QDX at more than 6W power output is NOT RECOMMENDED.

If you wish to operate using a 12 V supply, you may use a two turn secondary winding on the output transformer T1, so a 3:2 ratio instead of the 3:3 turns ratio documented in this manual. Remember this when you come to the assembly step for preparing and installing the output transformer T1. The “primary” is still 3 turns, with a tap half way at 1.5 turns. The secondary (no tap) will now be only two turns.

The chart below shows the measured power output vs supply voltage for the standard 3:3 winding (Red line); at 12 V supply the output power of around 8 W is too high and likely to cause over-heating or failure of the power amplifier transistors. If you wish to use a supply of 12 - 13 V the 3:2 winding style is more suitable and will produce 4 – 5 W output for 12 – 13 V supply. The graph shows 40m but other bands are very similar.



2. Assembly

2.1 General guidelines

Assembly of this kit is quite straightforward, most components are SMD and have already been pre-assembled by the PCB factory. The usual kit-building recommendations apply: work in a well-lit area, with peace and quiet to concentrate. S**ome of the other semiconductors in the kit are sensitive to static discharge. Therefore, observe Electrostatic discharge (ESD) precautions**.

And I say it again: **FOLLOW THE INSTRUCTIONS!!** Don't try to be a hero and do it without instructions!

A jeweler's loupe is really useful for inspecting small components and soldered joints. You'll need a fine-tipped soldering iron too. It is good to get into the habit of inspecting every joint with the magnifying glass or jeweler's loupe (like this one I use), right after soldering. This way you can easily identify any dry joints or solder bridges, before they become a problem later on when you are trying to test the project.

You could also take photos with a mobile phone, and use the phone's zoom features to view the board in detail.

Triple check every component value and location BEFORE soldering the component!

It is easy to put component leads into the wrong holes, so check, check and check again! It is difficult to de-solder and replace components, so it is much better to get them correctly installed the first time. In the event of a mistake, it is always best to detect and correct any errors as early as possible (immediately after soldering the incorrect component). Again, a reminder: removing a component and re-installing it later is often very difficult!

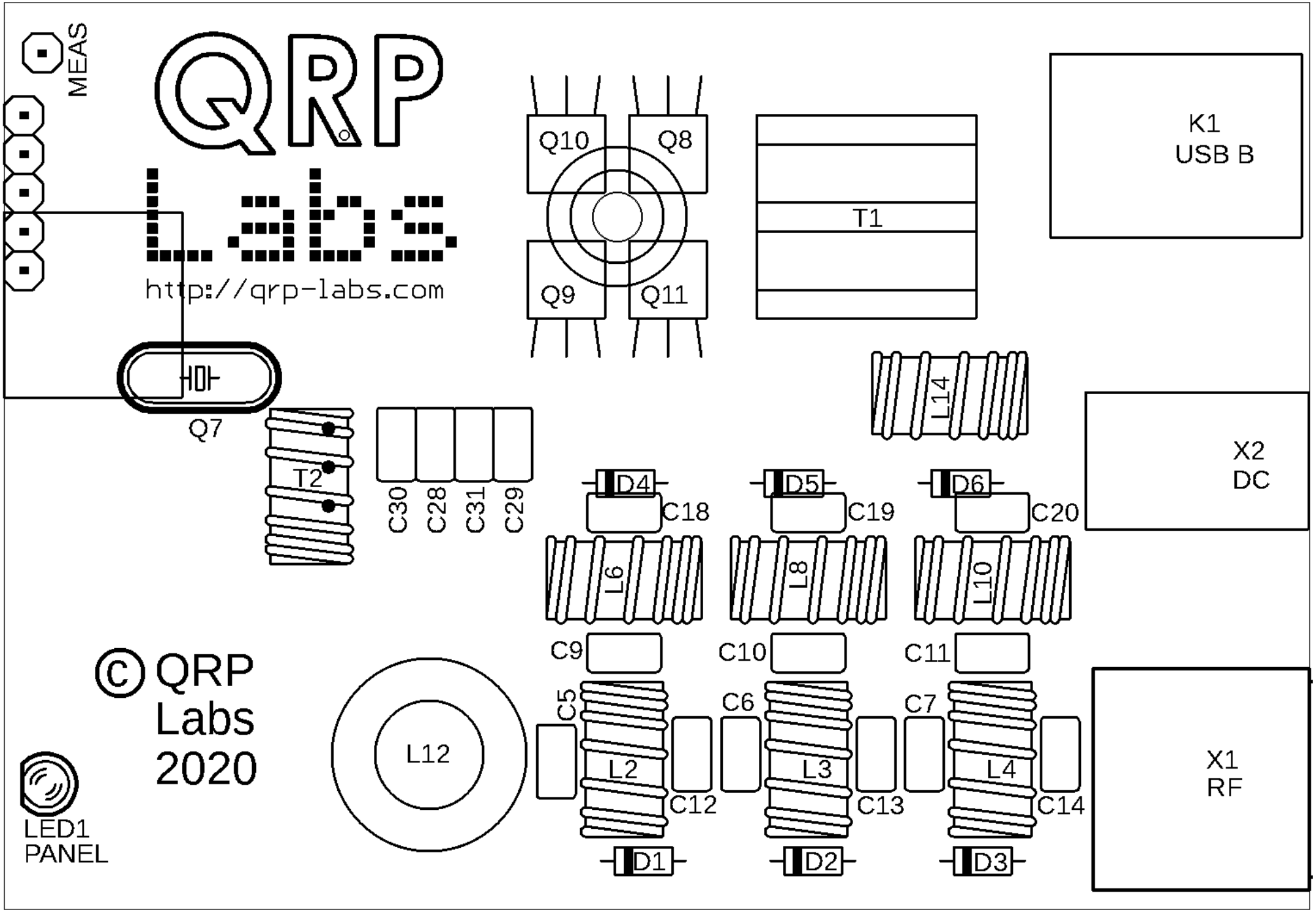
Please refer to the layout diagram and PCB tracks diagrams below, and follow the steps carefully.

Use of a good quality soldering iron and solder is highly recommended for best results!

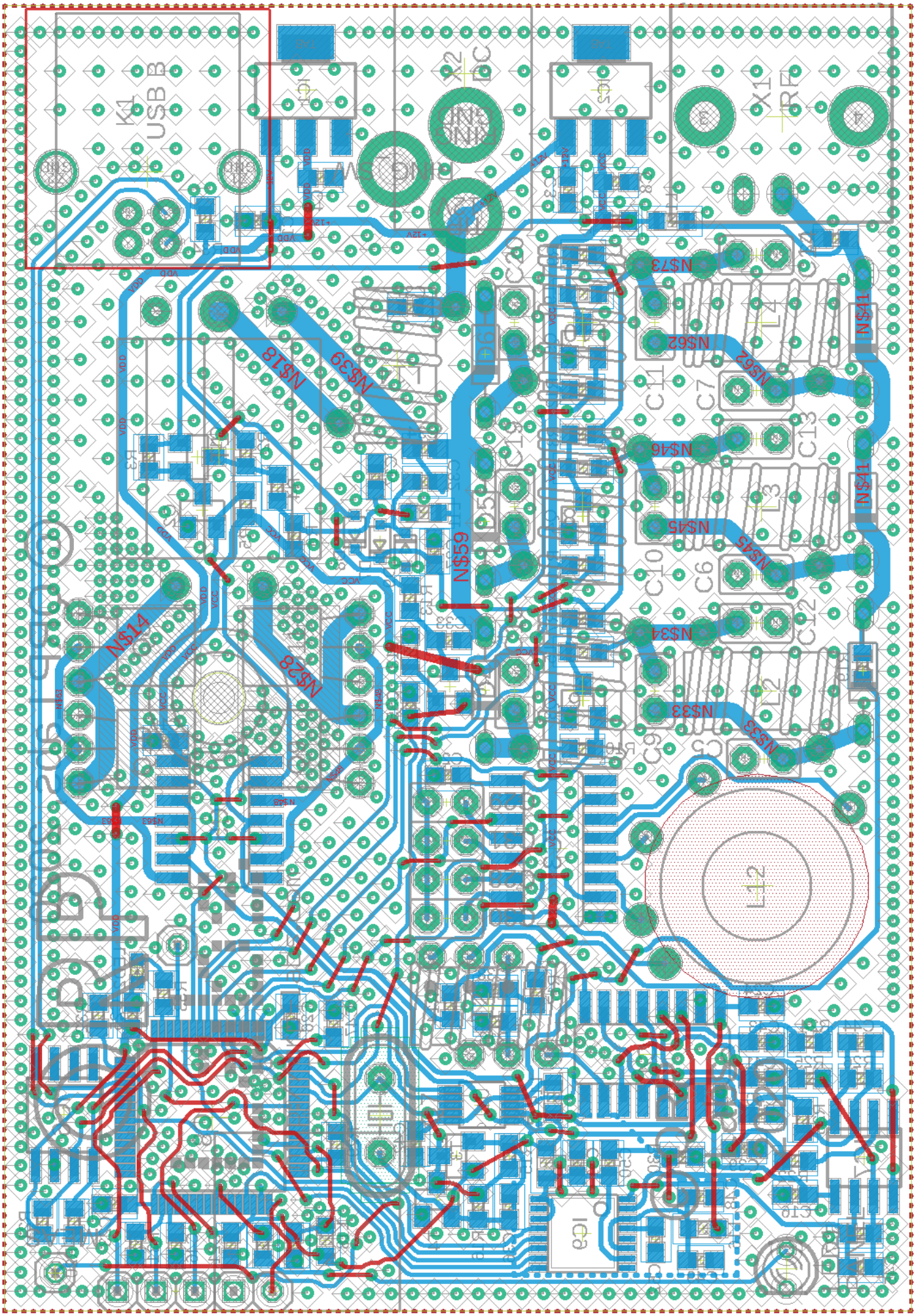
The following diagrams show the PCB layout and track diagrams of the QDX.

Tracks shown in BLUE are on the bottom layer. Tracks shown in RED are on the top layer. There are only two layers (nothing is hidden in the middle). Not shown in this diagram are the extensive ground-planes, on both sides of the board. Practically everything on both layers that isn't a RED or BLUE track, is ground-plane! The two ground-planes are connected at frequent intervals (not more than 0.1-inches) by vias.

NOTE: the capacitor lead spacing on the PCB is 0.1-inches (2.54 mm) and most of the capacitors are sized appropriately for this. From time to time, due to availability constraints, we may have to use capacitors with 0.2-inch lead spacing (5.08 mm); this is not a mistake, it is just due to component availability. In this case simply use a pair of long-nosed pliers (etc) to straighten out the wires and make them spaced for the 0.1-inch pads.


MEAS
QRP
Labs
http://qrp-labs.com
Q10
Q8
Q9
Q11
T1
K1
USB B
Q7
T2
L14
X2
DC
C30
C28
C31
C29
D4
D5
D6
C18
C19
C20
L6
L8
L10
C9
C10
C11
C5
C6
C7
L2
L3
L4
C12
C13
C14
D1
D2
D3
L12
X1
RF
© QRP
Labs
2020
LED1
PANEL

© QRP Labs, 2020
IC1
IC2
IC3
IC4
IC5
IC6
IC7
IC8
IC9
U1
T3
T4
Q1
Q2
Q3
Q4
Q5
Q6
D7
D8
D9
LED2
L1
L5
L7
L9
L11
L13
L15
L16
L17
L18
C1
C2
C3
C4
C8
C15
C16
C17
C21
C22
C23
C24
C25
C26
C27
C32
C33
C34
C35
C36
C37
C38
C39
C40
C41
C42
C43
C44
C45
C46
C47
C48
C49
C50
C51
C52
C53
R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
R11
R12
R13
R14
R15
R16
R17
R18
R19
R20
R21
R22
R23
R24
R25
R26
R27
R28
R29
R30
R31
R32
R33
R34


K1
USB B
X2
DC
X1
RF
N$73
N$62
N$46
N$45
N$34
N$33
N$41
N$59
N$14
N$28
N$18
N$39
L4
L3
L2
L1
C1
C7
C13
C10
C6
C2
C9
C3

2.2 Parts list

Many components are SMD, pre-soldered to the PCB in the factory. Only through-hole components need to be installed by the constructor. SMD components in the parts list are identified in the Description column and by the text colour being purple.

Resistors

Qty	Value	Description	Component numbers
3	0-ohms	SMD	R30, 31, 32
4	100-ohms	SMD	R22, 23, 24, 25
2	330-ohms	SMD	R2, 4
2	470-ohms	SMD	R28, 33
4	1K	SMD	R1, 6, 20, 21
1	1.5K	SMD	R34 (NOT INSTALLED)
12	10K	SMD	R3, 5, 10, 11, 12, 13, 14, 17, 18, 26, 27, 29
3	100K	SMD	R15, 16, 19
3	470K	SMD	R7, 8, 9

Capacitors (50V, Multi-layer Ceramic capacitors)

Qty	Value	Description	Component numbers
1	22pF	Label “220”	C29
1	30pF	Label “300”	C31
2	47pF	SMD	C27, 32
2	56pF	Label “56J”	C10, 28
1	82pF	Label “820”	C11
1	100pF	Label “101”	C9
1	180pF	Label “181”	C20
1	220pF	Label “221”	C30
1	270pF	Label “271”	C7
2	390pF	Label “391”	C6, 19
2	470pF	Label “471”	C5, 14
2	820pF	Label “821”	C13, 18
1	1200pF	Label “122”	C12
3	10nF	SMD	C36, 44, 45
21	0.1uF	SMD	C15, 16, 17, 21, 22, 23, 24, 25, 26, 33, 35, 37, 39, 40, 41, 43, 46, 47, 49, 51, 53
11	2.2uF	SMD	C1, 2, 3, 4, 8, 34, 38, 42, 48, 50, 52
1	220uF	16V electrolytic	Extra DC supply capacitor (see text)

Semiconductors

Qty	Description	Component numbers
5	SMD: 1N4148	D7, 8, 9
6	1N4007 diode	D1, 2, 3, 4, 5, 6
1	SMD: AMS1117-3.3	IC1
1	SMD: AMS1117-5.0	IC2
2	SMD: FST3253	IC3, 4
1	SMD: 74ACT08	IC5
1	SMD: 24C64	IC6
1	SMD: LM4562	IC7
1	STM32F401RBT6	IC8
1	AK5386	IC9
1	SMD: Si5351A	U1
1	SMD: BSS84 MOSFET	Q1
5	SMD: BSS123 MOSFET	Q2, 3, 4, 5, 6
4	BS170: TO92 MOSFET	Q8, 9, 10, 11
1	SMD: BC857 PNP	T3
1	SMD: BC817 NPN	T4
1	SMD: Red LED	DIAG
1	3mm Red LED	PANEL

Inductors

Qty	Description	Component numbers
6	SMD: 47uH	L11, 13, 15, 16, 17, 18
4	47uH radial inductor	L1, 5, 7, 9 (replace SMD inductors, see text)
1	T37-2 toroid (red), 2.40 uH (24t)	L2
1	T37-2 toroid (red), 2.88 uH (26t)	L6
1	T37-6 toroid (yel), 1.06 uH (18t)	L3
1	T37-6 toroid (yel), 393 nH (11t)	L4
1	T37-6 toroid (yel), 1.20 uH (20t)	L8
1	T37-6 toroid (yel), 525 nH (13t)	L10
1	T50-2 toroid – tapped, see text	L12
1	FT37-43 10t	L14
1	BN43-202 binocular, 3:3	T1
1	FT37-43 10t trifilar	T2

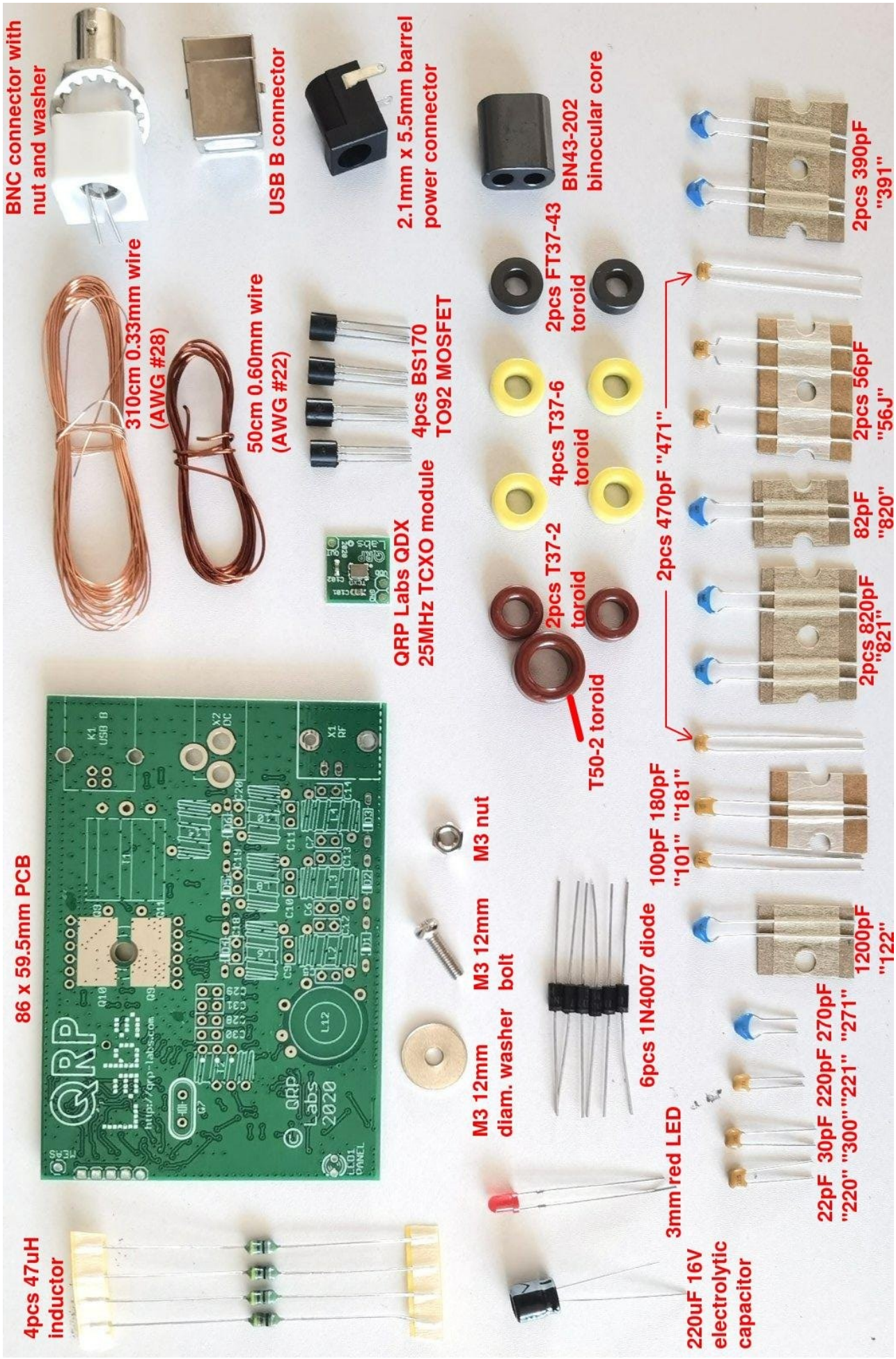
Miscellaneous

Qty	Value	Description
1	2.1 mm DC	2.1 x 5.5 barrel DC power connector
1	USB B	USB type B connector
1	BNC	RF connector
1	25 MHz TCXO module	Replaces 25 MHz crystal – see text
1	25 MHz	HC49/4H quartz crystal – **NOT USED**
1	PCB	Main PCB, 86 x 59.5 mm
1	310 cm	0.33 mm diameter wire (AWG #28)
1	50 cm	0.60 mm diameter wire (AWG #22)
1	M3 12 mm	Steel 12mm long M3 screw
1	M3	Steel M3 nut
1	M3 12 mm	Steel 12mm diameter M3 washer

Enclosure (OPTIONAL)

Qty	Value	Description
2	Top, Bottom	Extruded aluminium top and bottom cover
1	Front panel	Laser etched, drilled for 3mm LED
1	Rear panel	Laser etched, drilled for three connectors
8	M2.5 machine screw	Screws to secure end panels
4	Rubber foot	Self-adhesive rubber foot

2.3 Inventory parts

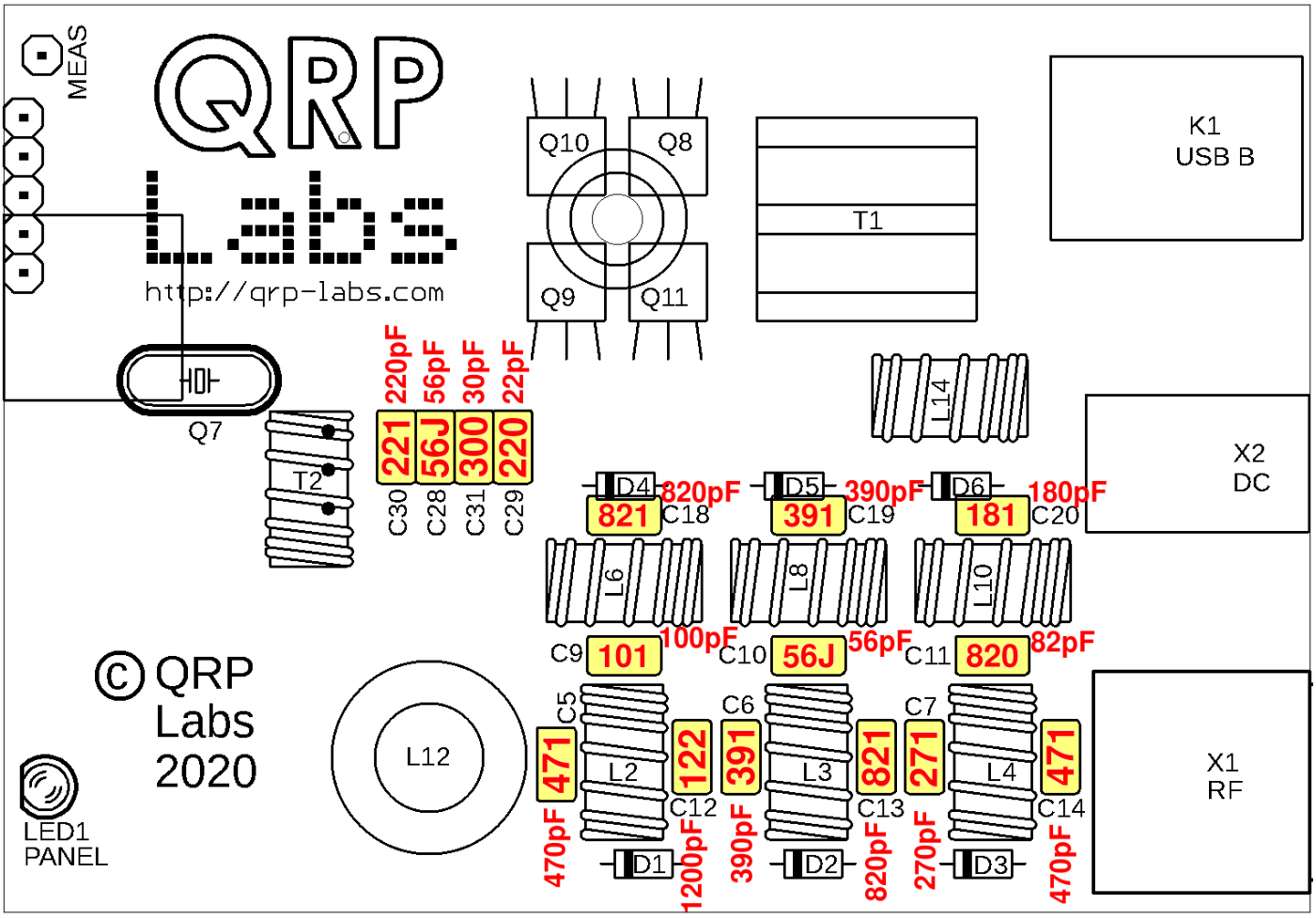


2.4 Install all the ceramic capacitors

Install all 16 through-hole capacitors in accordance with the following diagram. The procedure is so easy that I have included all capacitors in a single assembly step rather than doing one diagram for each capacitor value. Nevertheless be very careful to insert the correct value capacitors in the correct places. Mistakes are hard to correct later.

In the diagram, the component label (capacitor body inscription) is written inside the capacitor body which is coloured yellow. The actual value e.g. 220pF is written in red text next to the capacitor. Note that the leads of the 56pF capacitor ("56J") need to be bent to fit the 2.5mm holes.

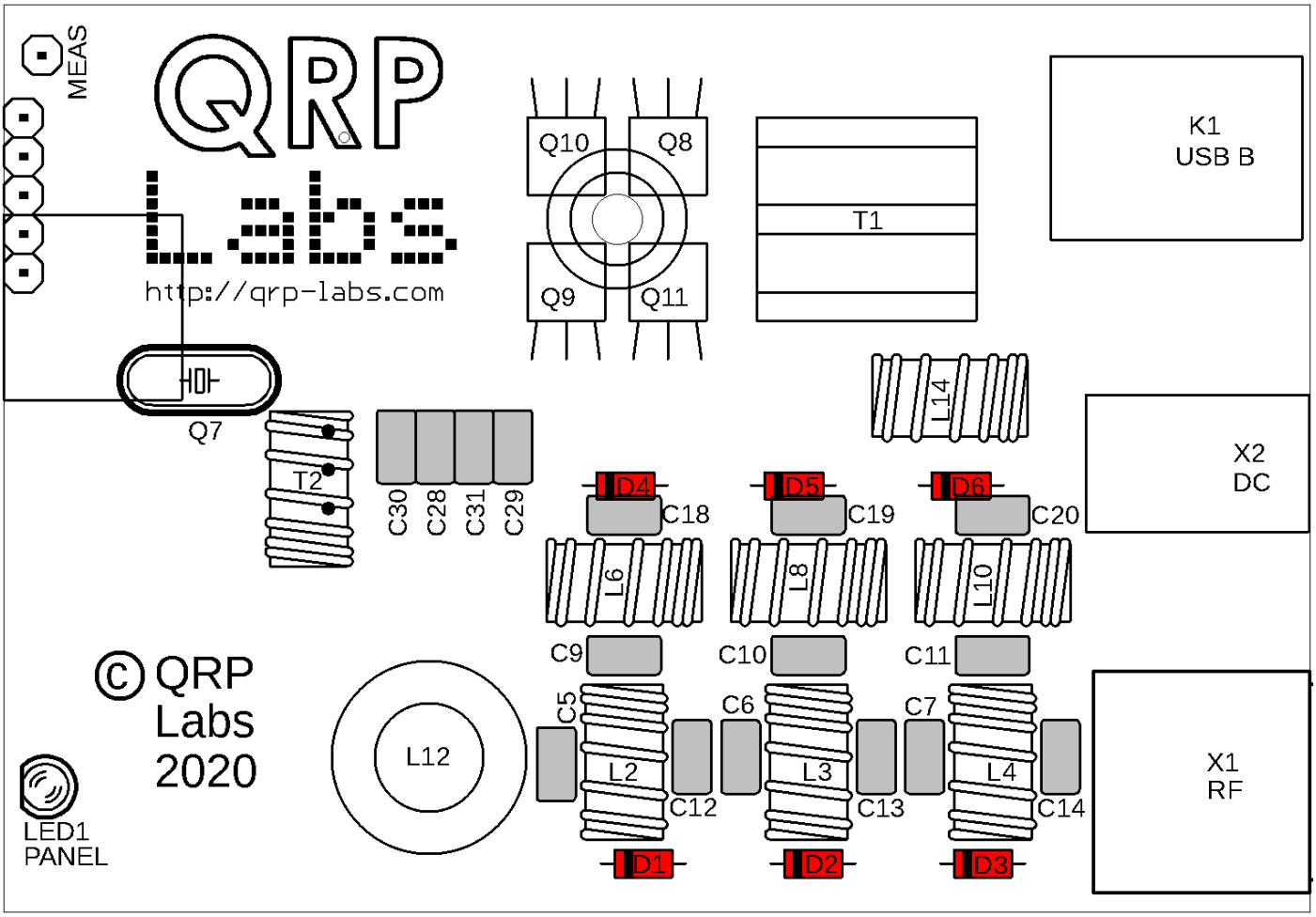
Qty	Value	Description	Component numbers
2	470pF	Through-hole, label "471"	C5, 14
2	390pF	Through-hole, label "391"	C6, 19
1	270pF	Through-hole, label "271"	C7
1	100pF	Through-hole, label "101"	C9
2	56pF	Through-hole, label "56J"	C10, 28
1	82pF	Through-hole, label "820"	C11
1	1200pF	Through-hole, label "122"	C12
2	820pF	Through-hole, label "821"	C13, 18
1	180pF	Through-hole, label "181"	C20
1	22pF	Through-hole, label "220"	C29
1	220pF	Through-hole, label "221"	C30
1	30pF	Through-hole, label "300"	C31



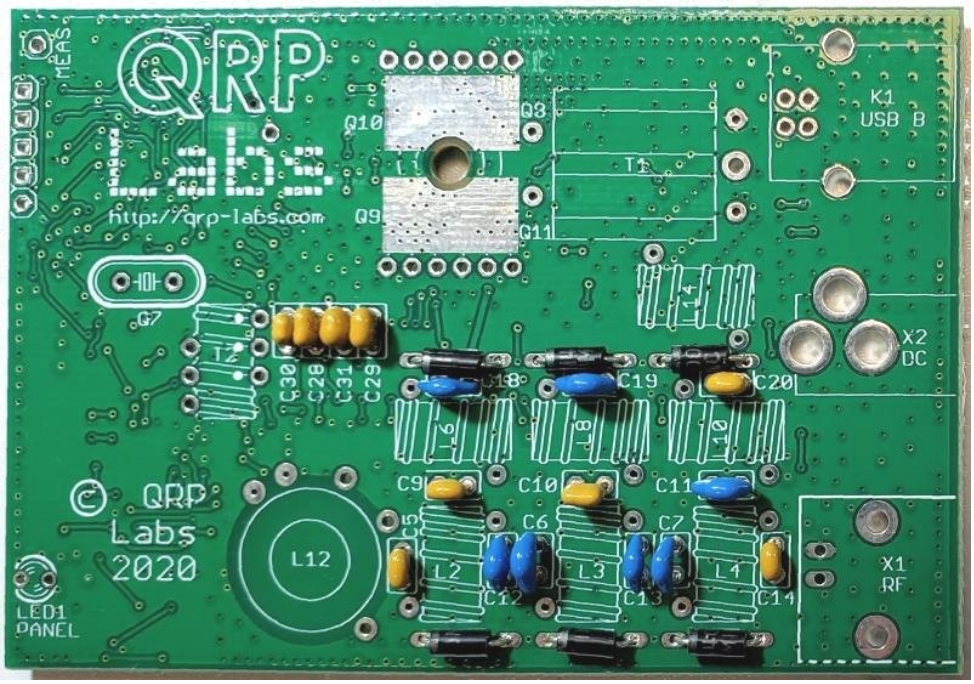
2.5 Install 1N4007 diodes

Install the six 1N4007 diodes D1-D6.

Pay attention to the orientation of the diodes which is critically important. The diodes have a black body and a white stripe. The white stripe end must be aligned with the white stripe on the image on the PCB silkscreen. In the diagram below, all the white stripe ends of the diodes are on the left of each diode position.



So far, the assembly should look like this →

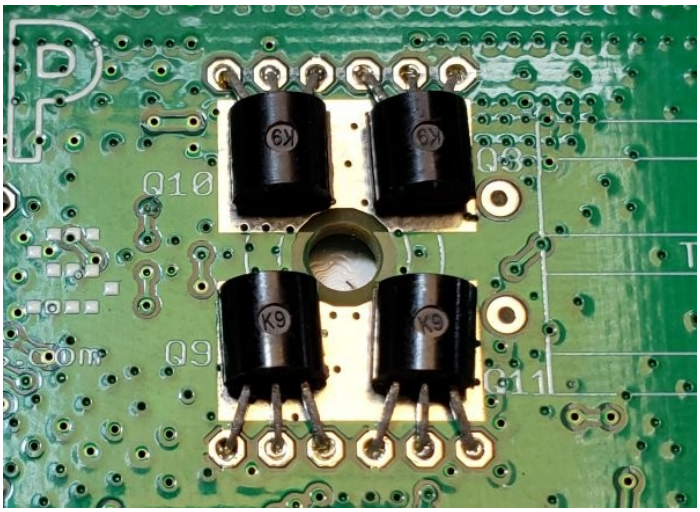


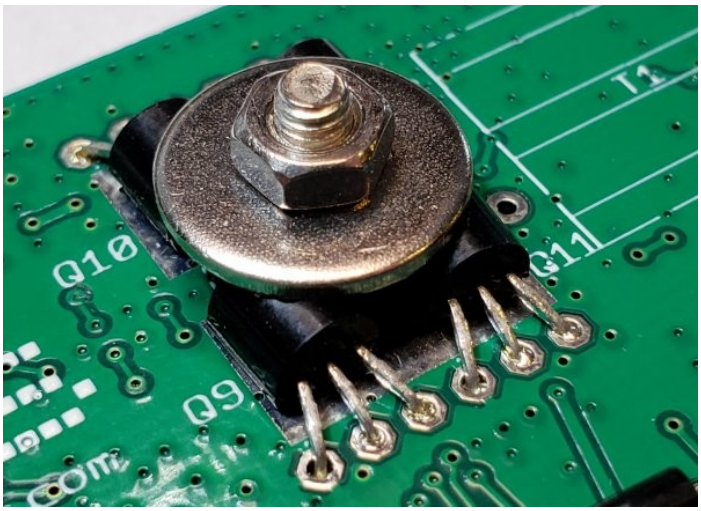
2.6 Install PA transistors

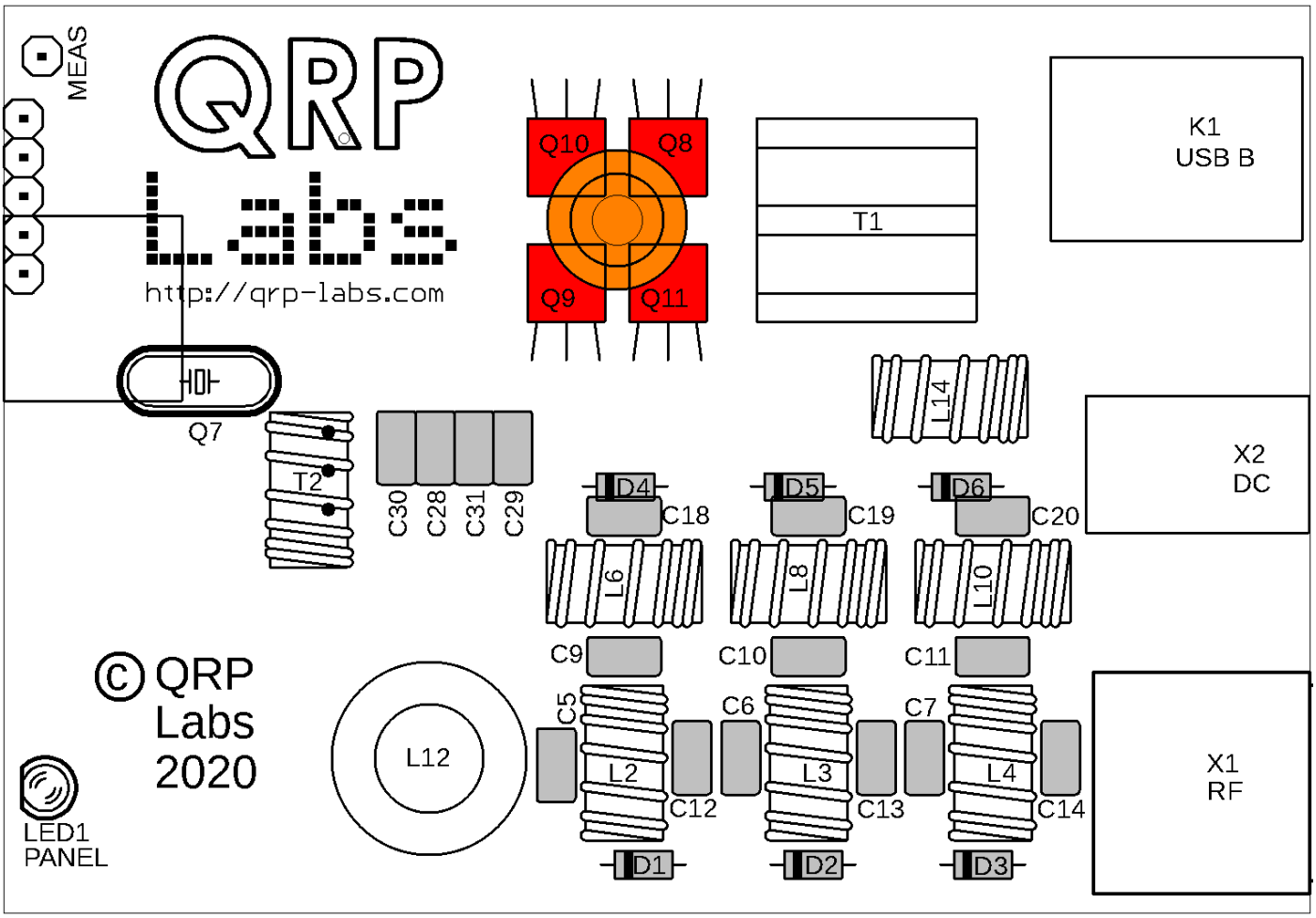
Install the transistors in the positions shown, with their flat faces flush against the PCB.

Use the 12mm M3 bolt, washer and nut to firmly press the transistor faces against the PCB, as shown.

Refer to the diagram and photographs below.





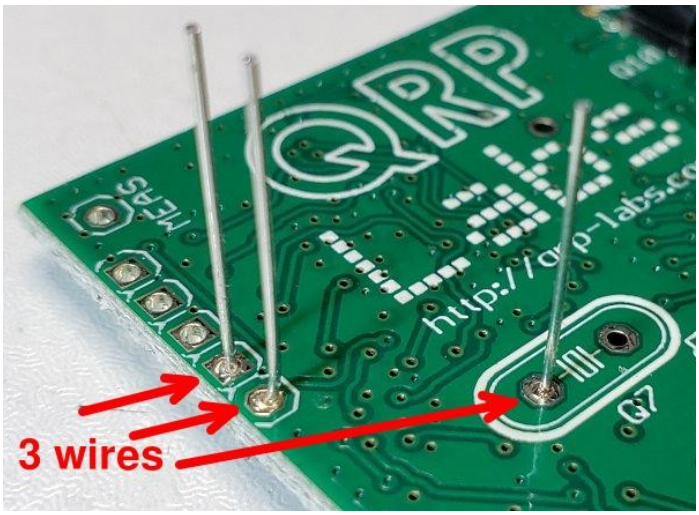


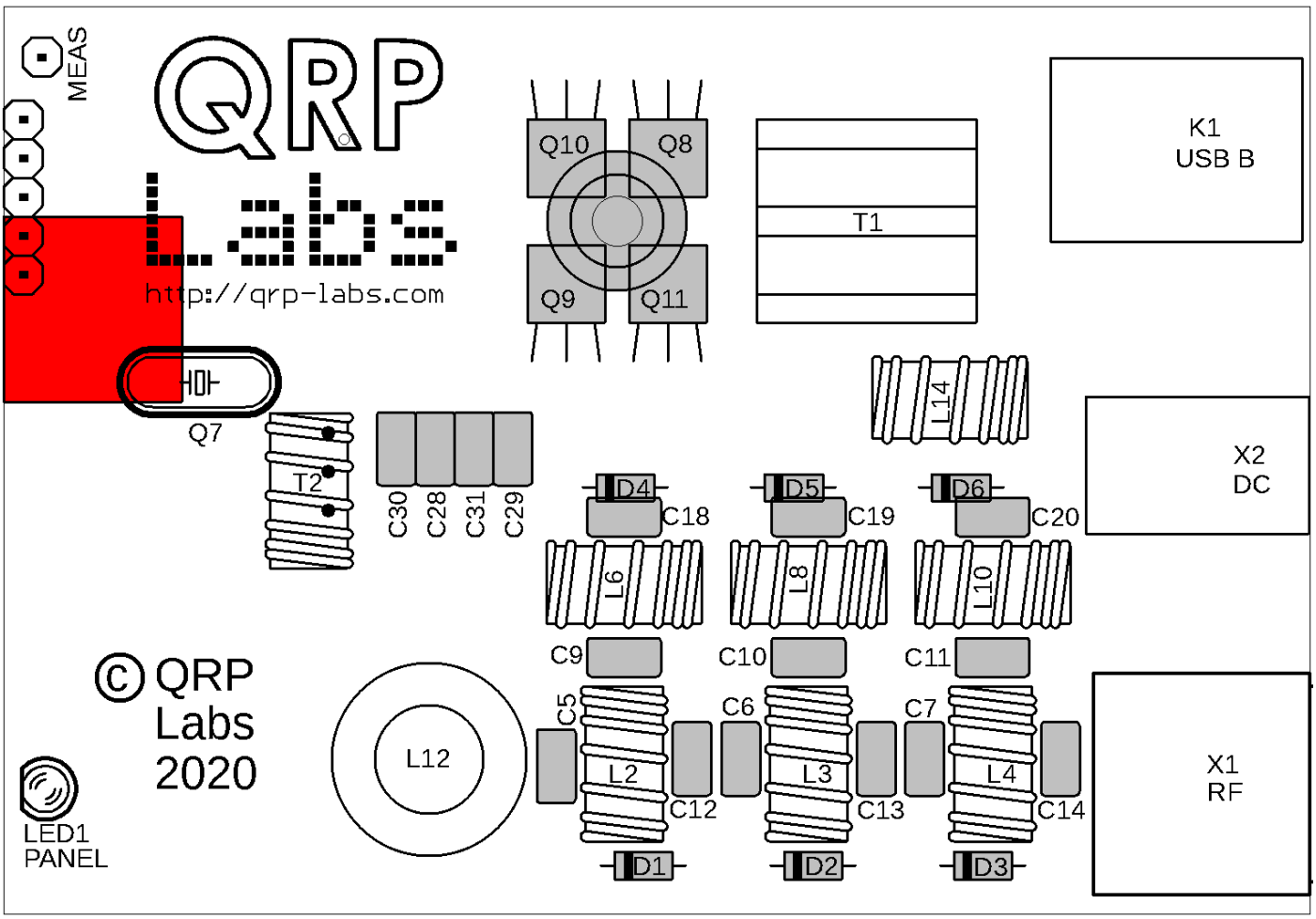
2.7 Install QRP Labs QDX TCXO module

The QDX TCXO module is a small PCB containing the actual TCXO and two capacitors. It is to be installed on the QDX PCB instead of the crystal marked “Q7”.

First use three pieces of wire, off-cuts from the capacitor leads soldered in step 2.4. Solder them in the positions shown, soldering on the bottom side of the board (Photo, below left).

Next, slide the TCXO module onto the wires and lower it to sit flat on the main QDX PCB. Solder the wires carefully and quickly – the TCXO PCB is only a single sided board so the pads will not withstand a lot of heat or abuse – then carefully cut off the excess wire. The result is as shown (photo, below right).

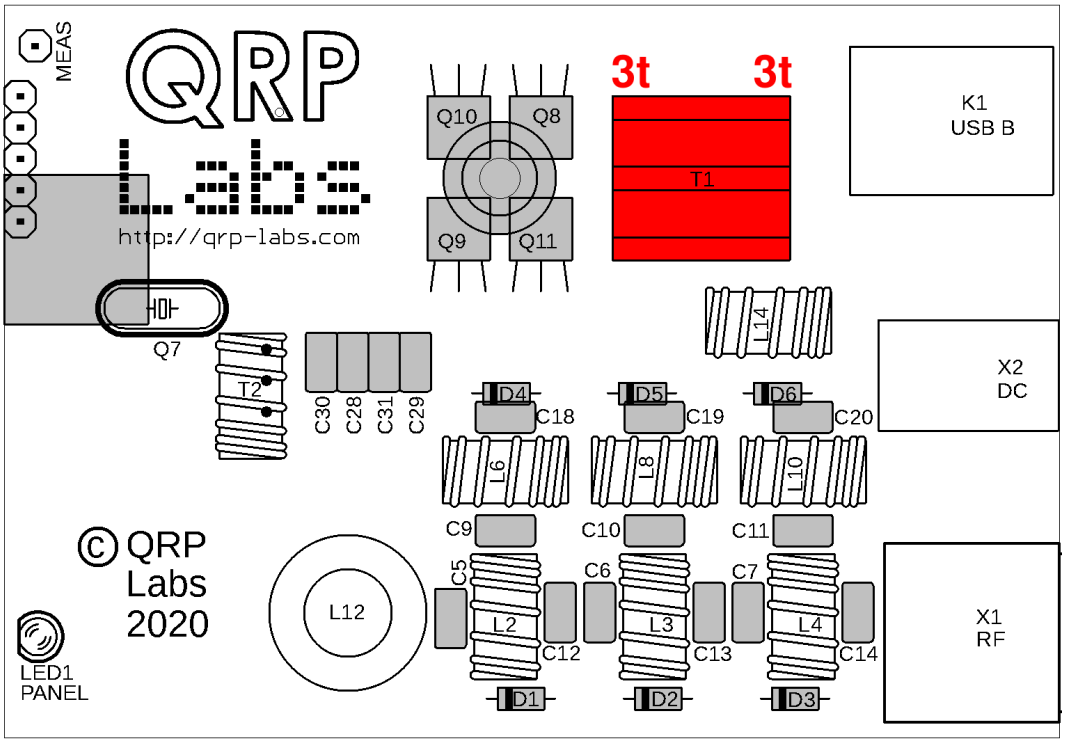




2.8 Assemble and install transformer T1

Transformer 1 is wound on the BN43-202 binocular ferrite former, using the thick 0.6mm (AWG #22) wire. This wire is also used for L14 so do not use all of it on the transformer.

The transformer has two windings, both 3 turns (3:2 if building for 12V supply). The three-turn primary winding has a center-tap. In the nomenclature of binocular cored transformers, “1 turn” means the wire passes through both sets of holes, ending up back at the end where it started.

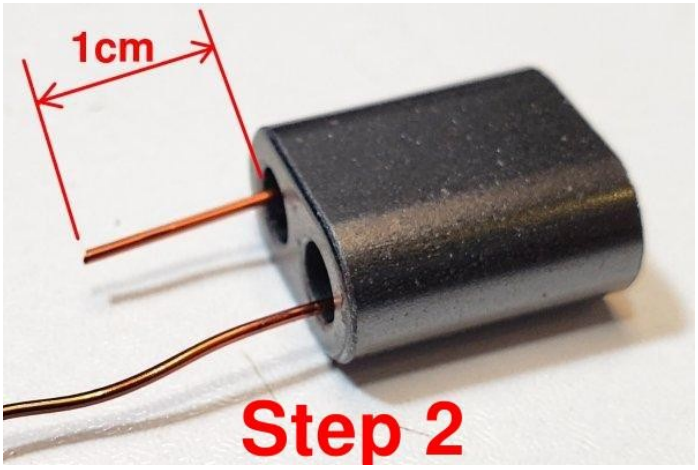


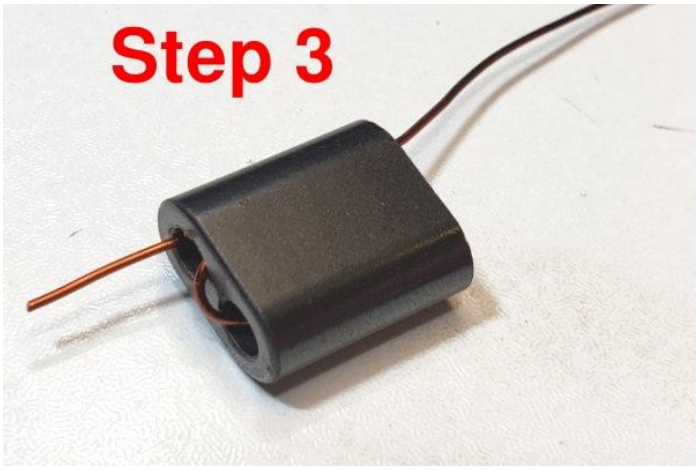
Assembly of this transformer is best done in steps as follows. Please read all the steps before commencing the assembly:

Step 1: Carefully unwrap the thick 0.6mm (AWG #22) wire, and straighten it ensuring there are no kinks.

Step 2: Pass the wire through both holes, starting at the top left, as shown. This is the first turn of the 3-turn primary winding.

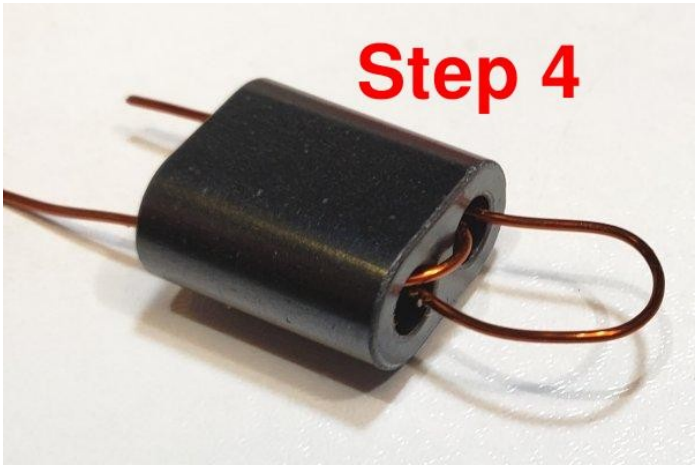
Step 3: Now pass the wire through the top hole of the binocular core, back from left to right. This is the next half-turn of the primary winding, bringing the number of turns so far to 1.5; now we must make the center tap.

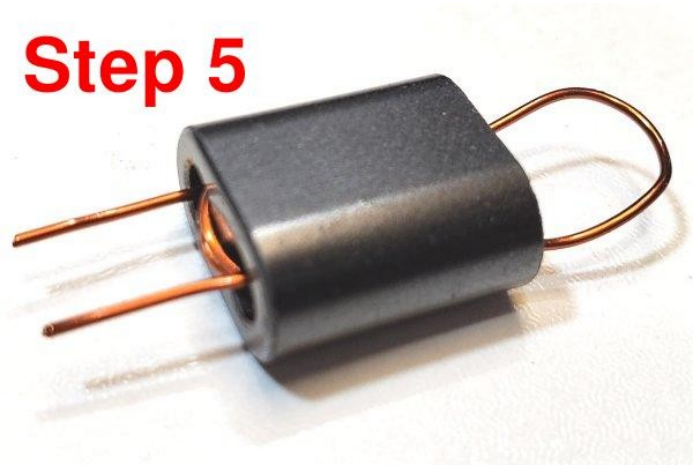




Step 4: Now pass the wire back through the bottom hole of the binocular core, from right to left; but do not pull it tight. Leave a small loop as shown in the photograph. This will be soldered into the center-tap pad on the PCB.

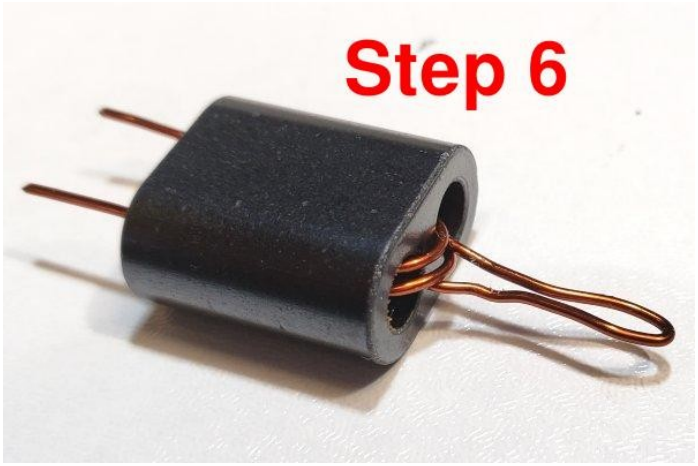
Step 5: Wind the wire through both the top hole of the binocular core and the bottom hole, pulling it a little tight as normal; this forms the final turn of the 3-turn primary winding. You end up with the wire coming out of the bottom left hole as shown; cut it to about 1cm protrusion.

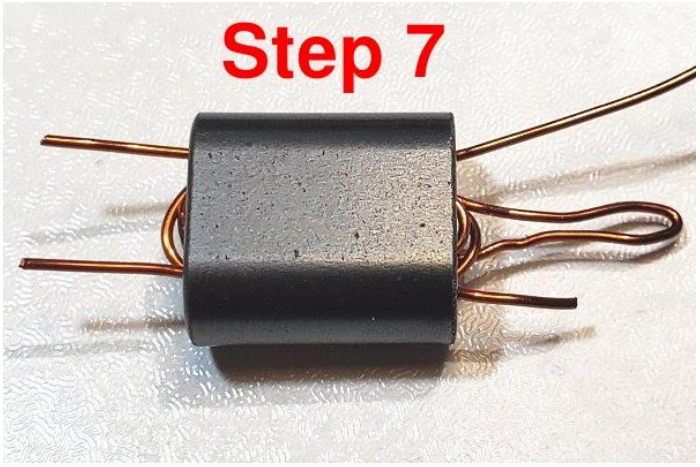




Step 6: Squeeze together the center-tap to try to avoid subsequent confusion.

Step 7: Start the 3-turn secondary winding by pushing the wire from right to left through the bottom hole of the binocular core, then from left to right through the top hole. This is the first turn.

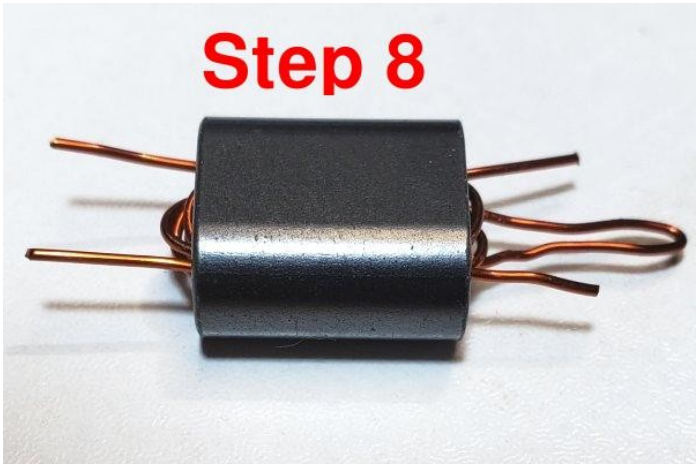




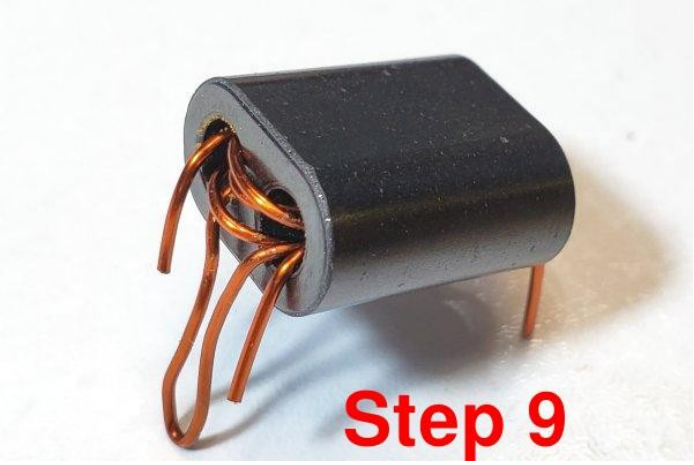
Step 8: Push the wires through both holes two more times to create the second turn: right to left through the bottom hole, then left to right through the top hole. Now we have three turns on the secondary. Cut the wire leaving about 1cm spare.

Remember to wind only 2 turns secondary, if winding the transformer as 3:2 for 12V operation.

Please refer to page 4 for details of 9V or 12V operation choices.

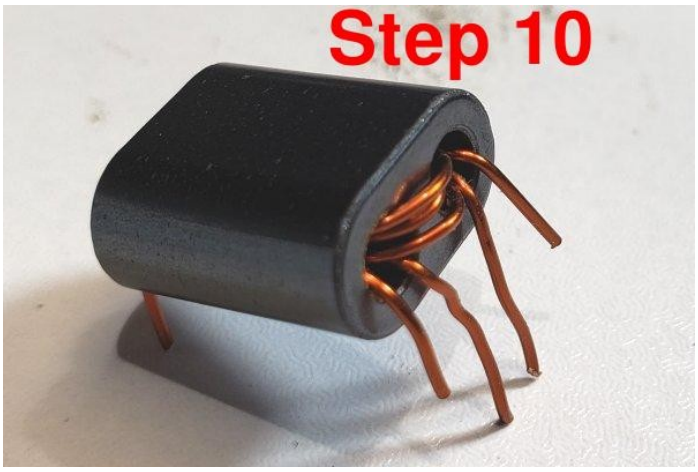


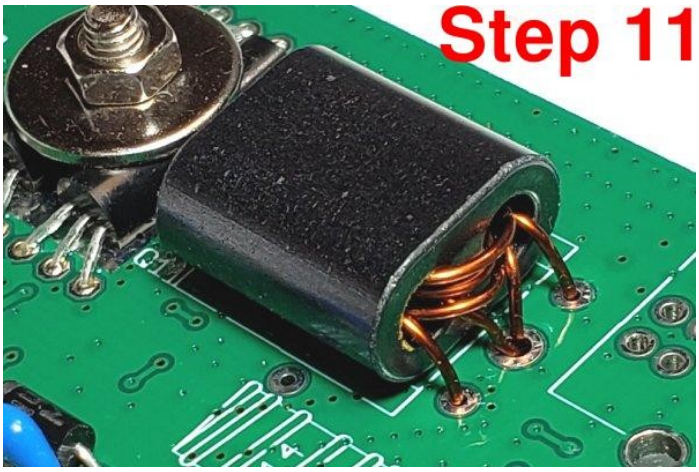
Step 9: Bend all the wires to point downward in the position they will need to be, to fit through the holes in the PCB.



Step 10: Now cut the center-tap loop; hopefully, as mine are in this photograph, these two wires formed by cutting the loop are LONGER than the two ends of the 3-turn secondary, so that you cannot mix anything up.

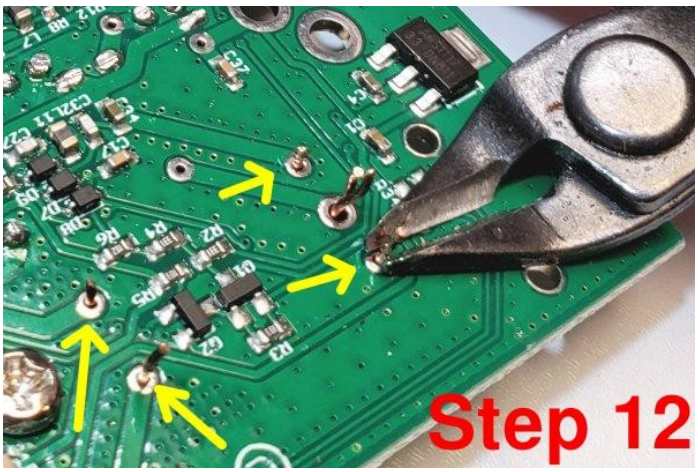
Step 11: Insert the wires into the holes on the PCB. The two wires (of the cut loop) that belong to the center-tap of the 3-turn primary, which are the longer wires, are inserted into the middle hole on the right-hand side; this hole is larger than the others and is big enough for both wires to fit in.

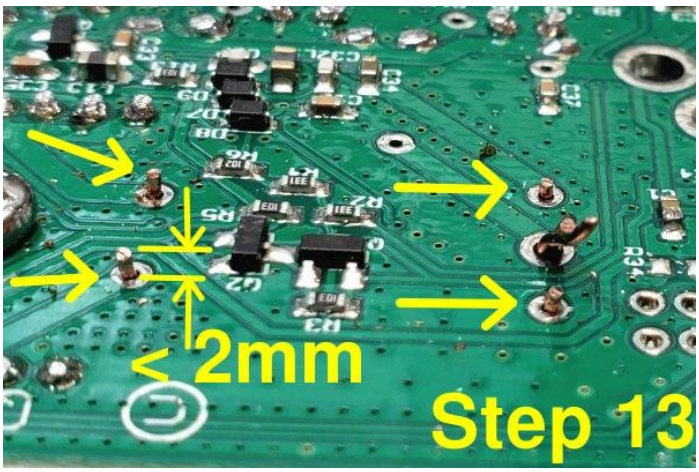




Step 12: Now we need to get the enamel off the wire. Usually on thinner wire, I hold the soldering iron to the wire until the enamel burns off. But that doesn't work so well on thicker wire such as this. So my technique here is to scrape the enamel off, at least partially, using wire cutters. The correct pressure needs to be applied to the wire cutters, so as NOT to actually cut the wire. I hold the wire cutter as close to the PCB as possible, then gently but firmly pull the cutter away from the PCB, scratching off the enamel. Turn the cutter to a different angle and scrape again, 2 or 3 times. It is not necessary to remove ALL the enamel, if you get a few good scrapes on, enough enamel will be removed that the soldering iron heat will burn off the rest of the enamel and a good joint will be achieved. Do this for the 4 winding ends first, leave the two center-tap for later, to make it easier.

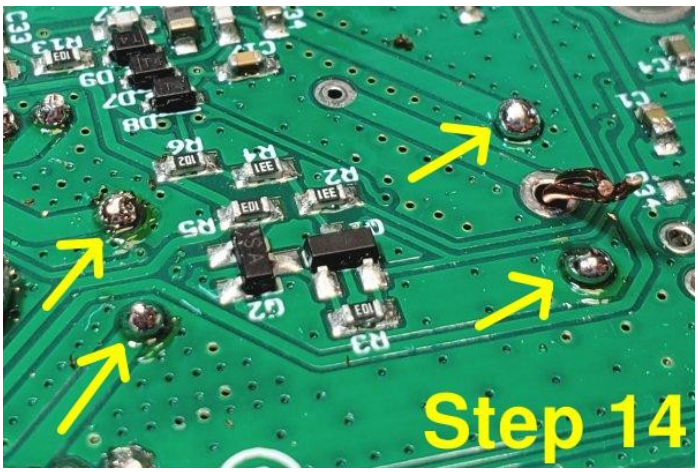
Step 13: Cut each of the four wire-endings, leaving about 2mm or less, sticking up from the PCB.

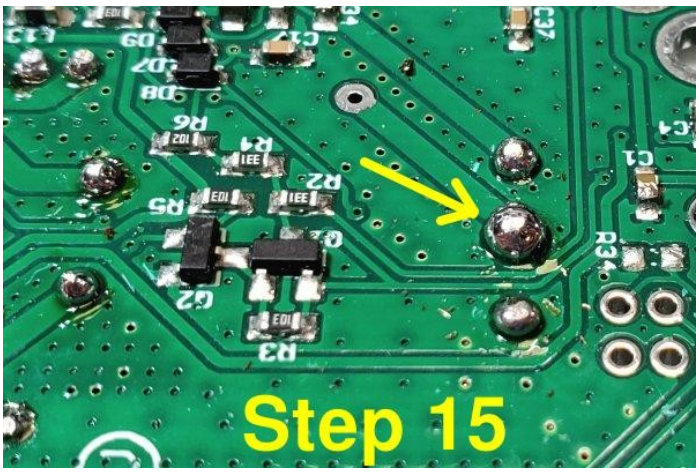




Step 14: Solder the four wire endings. At each pad, hold the soldering iron firmly, apply plenty of solder, and wait for about 10 seconds. This will ensure a good connection, and any remaining enamel will be burned away.

Step 15: Finally repeat the procedure with the two center-tap wires that came through the large center hole. Scrape them, cut them to 2mm, and solder them. Apply plenty of solder and hold the soldering iron tip to the joint for maybe 15 seconds, to really be sure of a good joint and burning away any remaining enamel.





Step 16: Check that none of the wires protrude more than 2mm from the surface of the PCB, since if you are using the optional aluminium enclosure, there are only a few mm clearance.

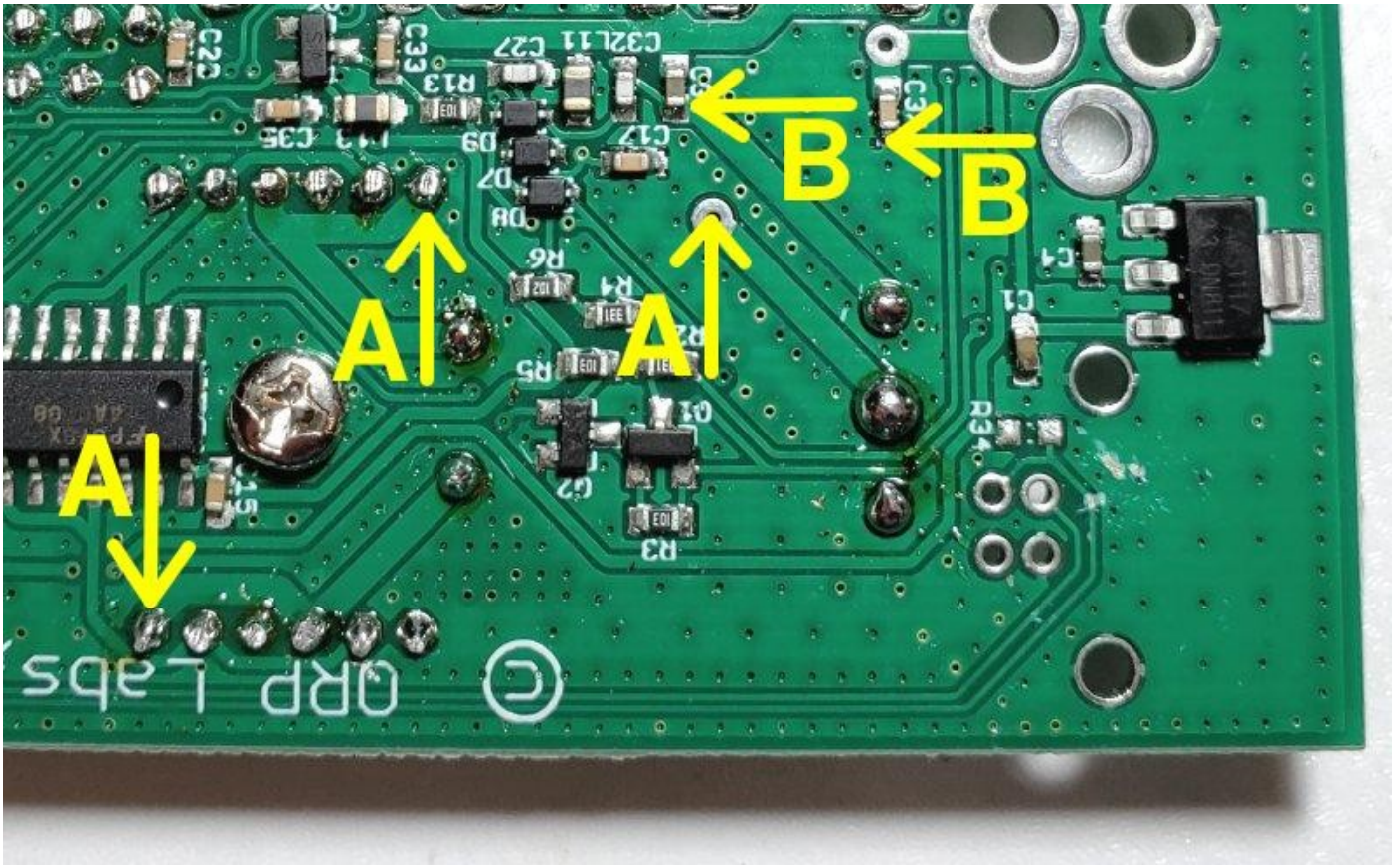
Step 17: Verify good connections for all five soldered joints of T1 using a DMM set to resistance or continuity mode. My cheap DMM hasn't got a continuity mode and I'm using the 2000-ohm resistance mode; in this mode when there is continuity, the reading on mine shows 001 (not zero-ohms; this is just a DMM thing, of no significance).

You should see continuity between all the points labeled A in the photo below. Touch the probes to pairs of these points, to verify continuity.

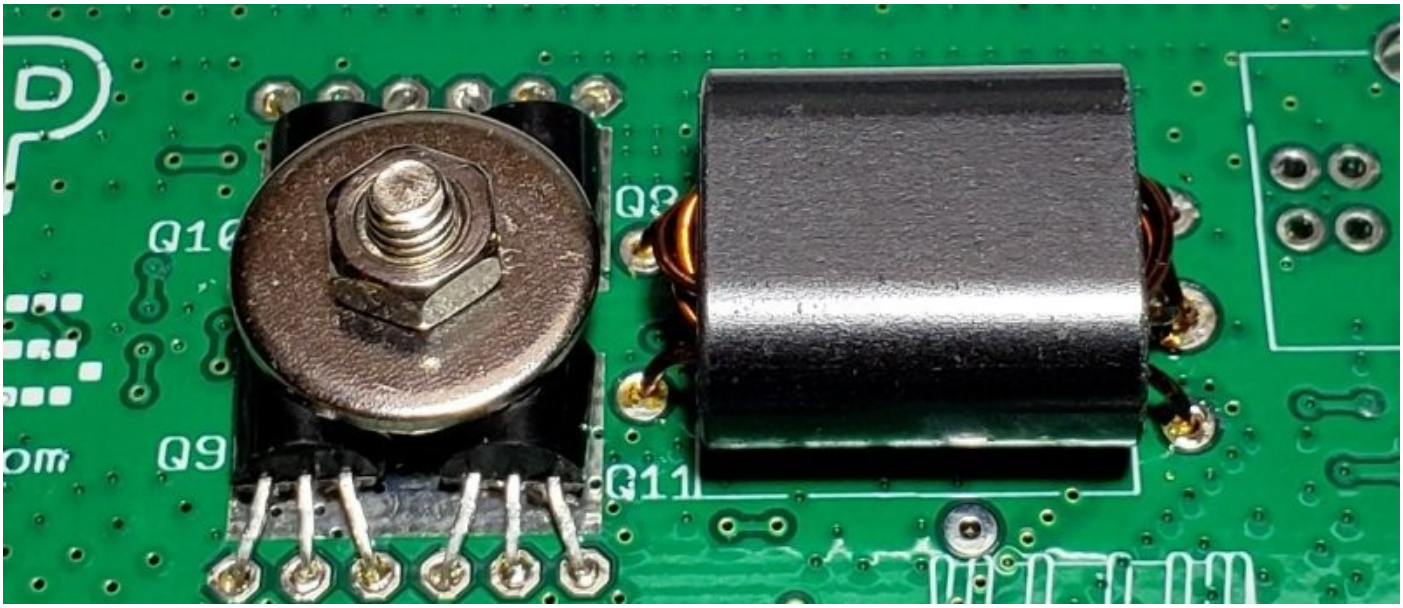
You should see continuity between the points labeled B also. Touch the two probes to the two pads labeled B and check for continuity.

Finally, there should be NO connection between A and B. Hold one probe on any A point and the other probe on a B point. That should read infinite resistance (no continuity).

If any of these tests fail, then you have a soldering problem somewhere, or the wrong wire in the wrong hole, or some short-circuit somewhere etc.



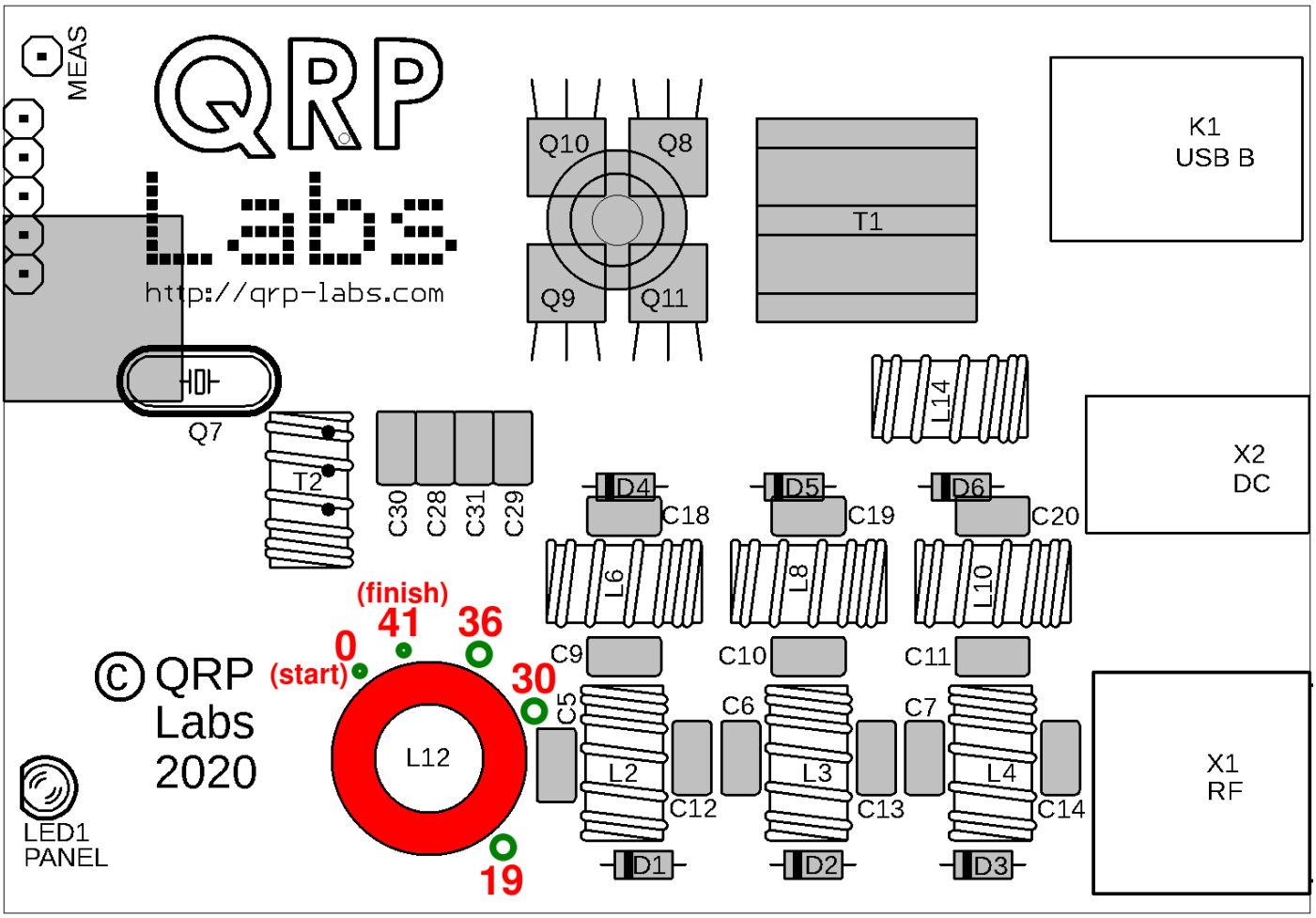
This is the final appearance of T1 when the installation is complete:



2.9 Prepare and install tapped inductor L12

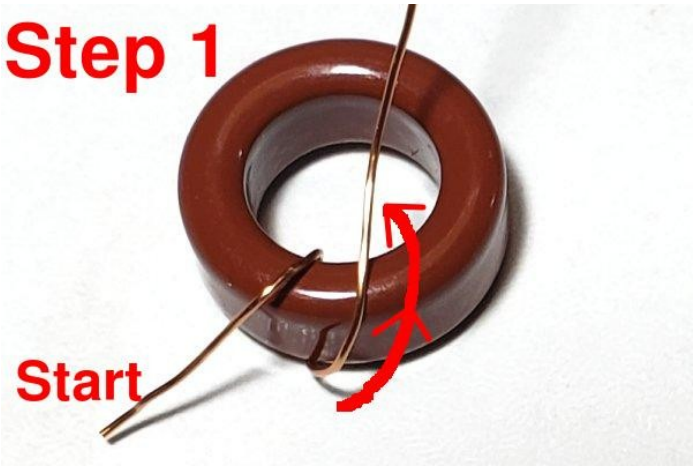
L12 is an inductor wound on a T50-2 toroid (red colour, large size), having several taps which are switched in according to the operating frequency (band). There are 41 turns in total, with taps at 19, 30 and 36 turns. The diagram shows the location of the taps at 19, 30 and 36 turns, which have larger holes, such that two wires can be inserted.

Carefully unwind and cut off 75cm of the 0.33mm (AWG #28) enameled wire, and straighten it, ensuring no kinks.



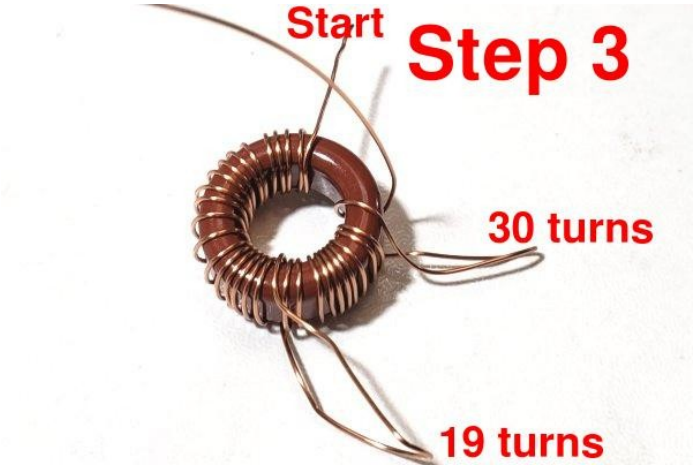
Step 1: Begin winding L12 as shown. Remember that in the nomenclature of toroidal inductors, each time the wire passes through the center of the toroid counts as 1 turn.

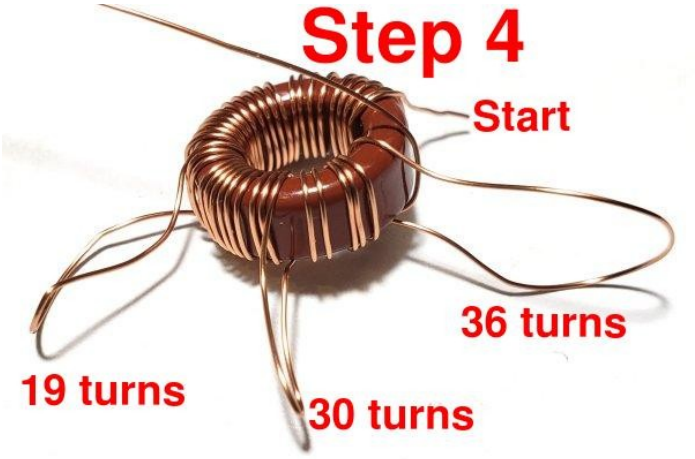
Step 2: Wind 19 turns then make a loop between the 19th and 20th turns. This is for the 19 turn tap.



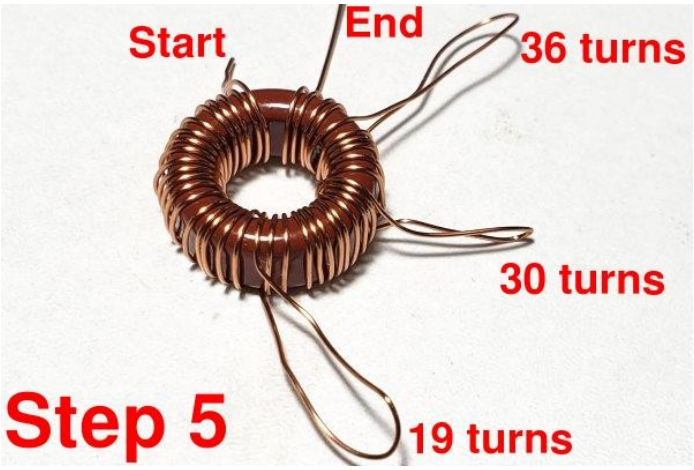
Step 3: Wind up to the 30th turn then make another loop between the 30th and 31st turns. This is for the 30 turn tap.

Step 4: Wind up to the 36th turn then make another loop between the 36th and 37th turns. This is for the 36-turn tap.





Step 5: Wind the remaining turns to complete the total of 41 turns.

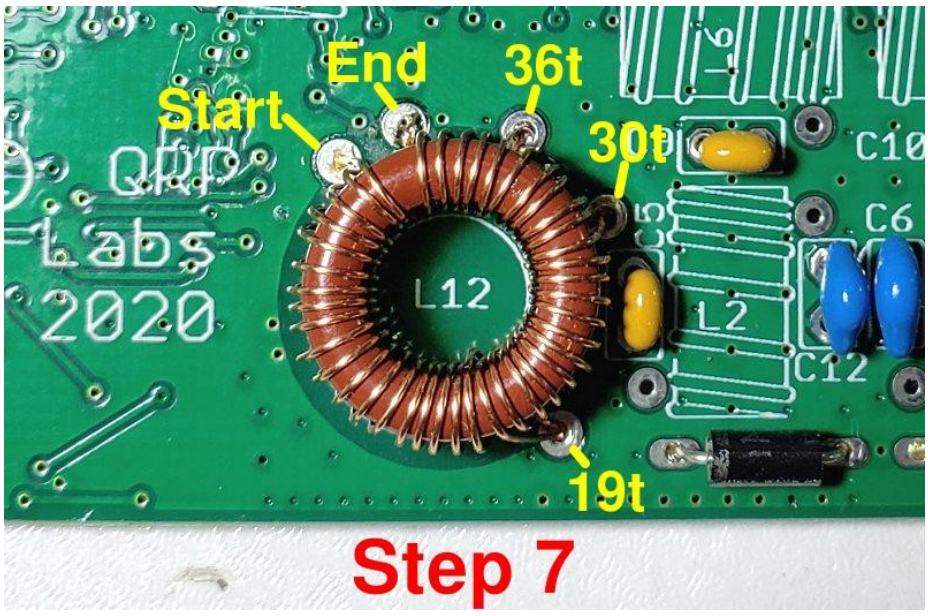


Step 6: Now with a pair of pliers, you can squeeze each of the loops into a sharp point, such that it is easy to fit through the large holes at the 19, 30 and 36-turn taps (see photo, right).

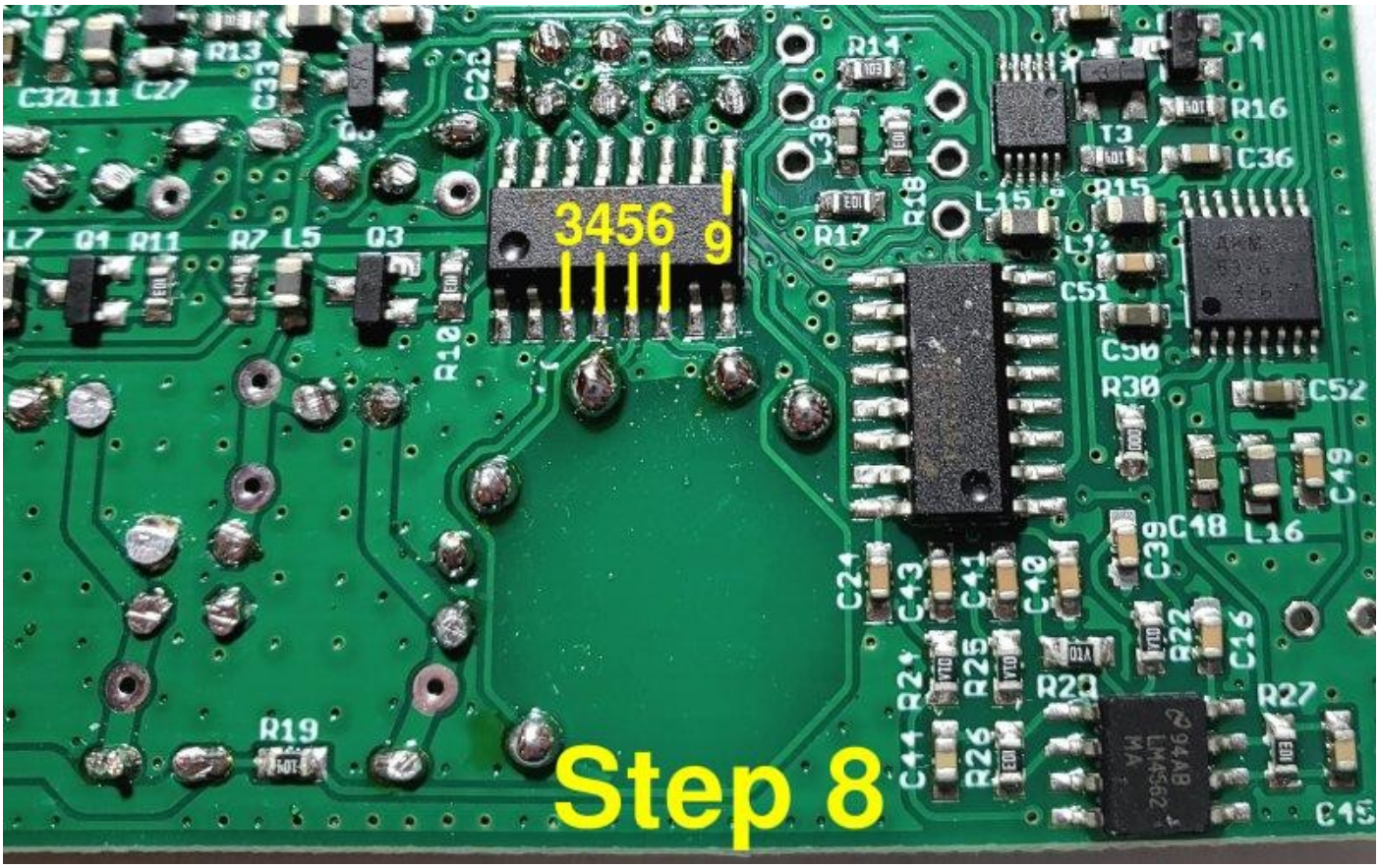
Step 7: Insert all the wires in their correct holes.

You should pull them firmly from the underside of the board to ensure that there are no loose wires on the top side.

Now cut each wire to about 2mm length (at most) and solder in place. It is essential to remove the enamel from the wire. My favourite method of doing this is simply to hold the soldering iron to the joint, with plenty of solder, for at least 10 seconds. The enamel burns away in this time.



Step 8: Verify the joints are properly soldered, by using a DMM in continuity testing mode (if it has this mode) or check for zero ohms in resistance mode. On the reverse of the PCB, check for continuity between pins 3, 4, 5, 6 and 9 of IC3 as shown. You should measure 0 ohms (continuity) between any pairs of these. If you do not, then there is a mistake somewhere, most probably a failure to burn away the enamel at one or more of the L12 connections to make a good joint.



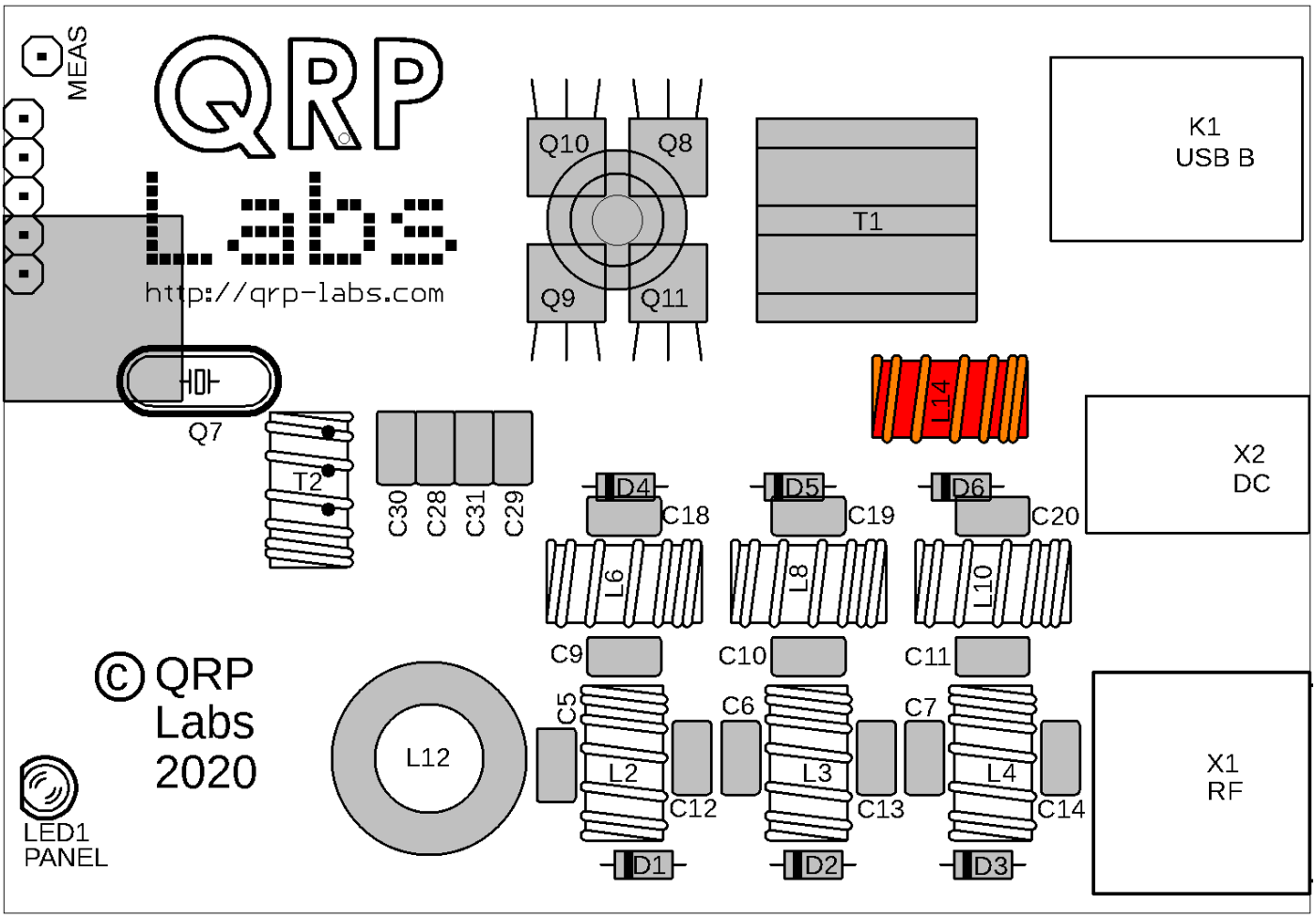
Here's the story so far, including the nicely installed L12 tapped inductor.

2.10 Wind and install L14

L14 consists of 10 turns of 0.6mm (AWG #22) wire wound on an FT37-43 toroid (black colour). Firstly, you have to understand that there's a right way and a wrong way to wind a toroidal inductor, too. The terms "right" and "wrong" are probably not as appropriate here, as "handedness" or "winding direction". There are two directions you can wind the toroid. If you choose the right one, then all the wires will automatically end up near the holes they are to be soldered into. If you do it wrongly, it will be a bit messy.

For all the toroidal inductors in the QDX kit, you will get it right, if you start as shown (photo, above right) and pass the wire through the toroid from the top side down through the hole, out and then around and over again; accumulating turns in the anticlockwise direction.

With this in mind, wind 10 turns and install the toroid. As with the output transformer T1, I recommend scratching the enamel with a knife or wire-cutter, and then cutting it to about 2mm length on the underside of the board. Then solder, applying the soldering iron for at least 10 seconds and plenty of solder, to ensure a good connection. Do NOT tin the wires before inserting into the holes: the holes are not large enough to accommodate tinned wires.



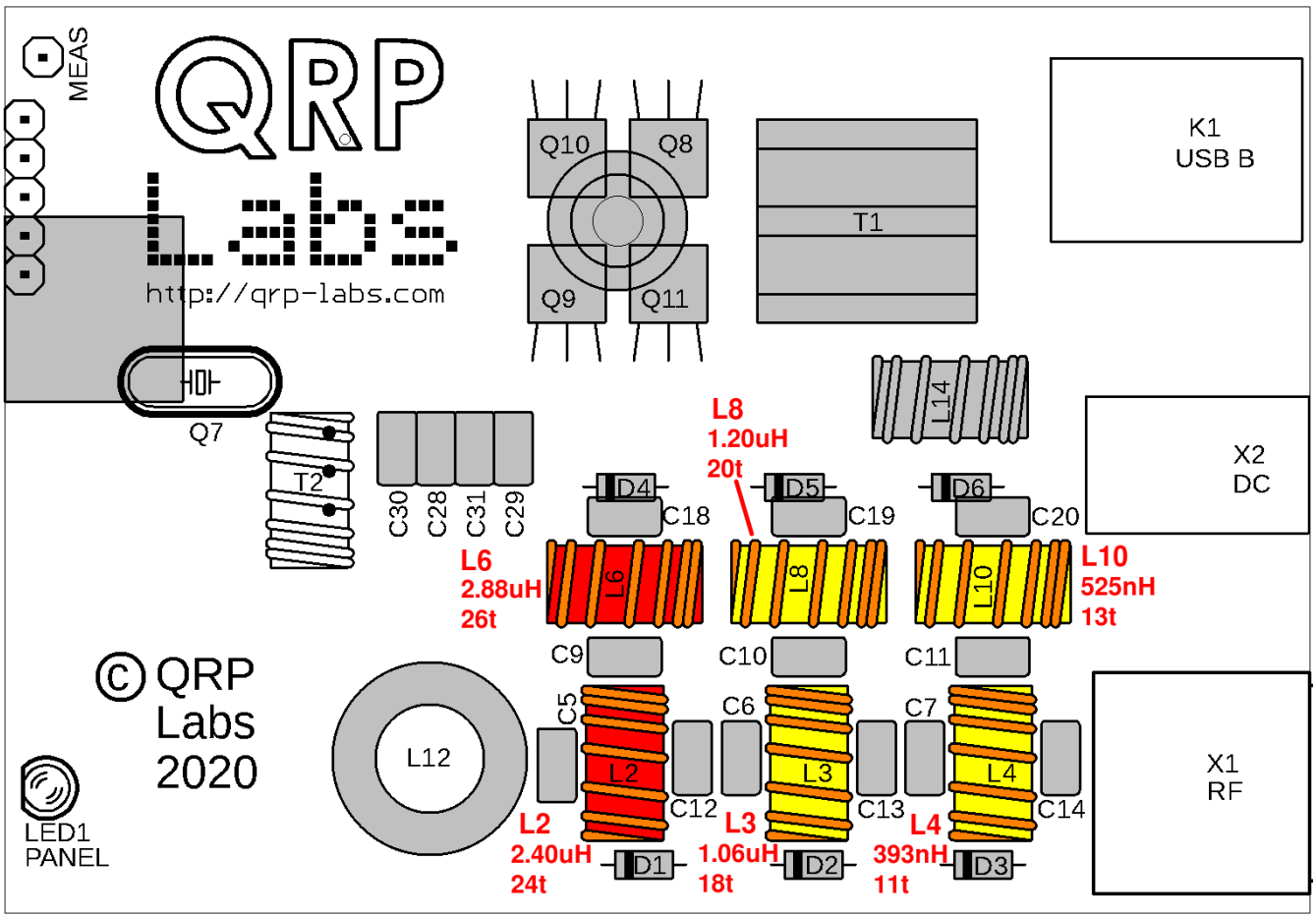
2.11 Install Low Pass Filter toroids

Now we'll wind and install the six Low Pass Filter toroids using 0.33mm (AWG #28) enameled wire. Remember, as per the previous direction, using the right winding direction will make the toroids much easier to fit to the PCB. Refer to the table below. Make sure to count carefully and install the toroids in the correct places! Remember that each time the wire passes through the center of the toroid counts as one turn. Spread the turns evenly with a small gap.

Winding the toroids as tightly as possible will slightly improve the performance of the Low Pass Filters. But don't pull so tightly that you break the wire!

The following table lists the required number of turns, cut wire length, and inductance.

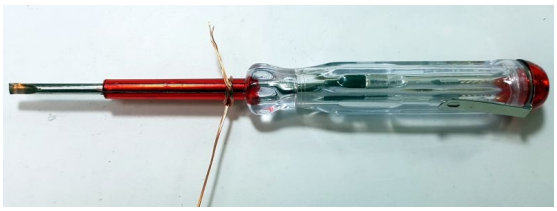
Band	Part number	Core	Inductance	Number of turns	Wire length
80m	L2	T37-2 (RED)	2.40uH	24	36cm
80m	L6	T37-2 (RED)	2.88uH	26	39cm
40m	L3	T37-6 (YELLOW)	1.06uH	18	29cm
40m	L8	T37-6 (YELLOW)	1.20uH	20	30cm
30m/20m	L4	T37-6 (YELLOW)	393nH	11	20cm
30m/20m	L10	T37-6 (YELLOW)	525nH	13	22cm

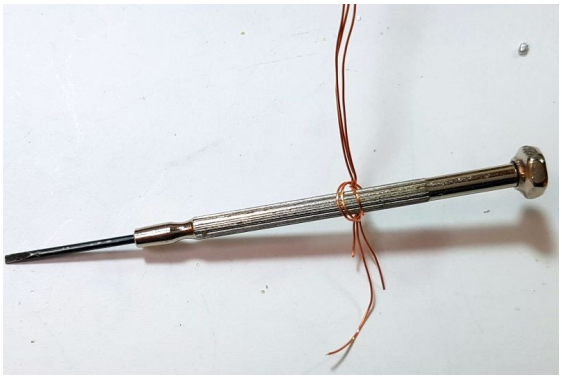


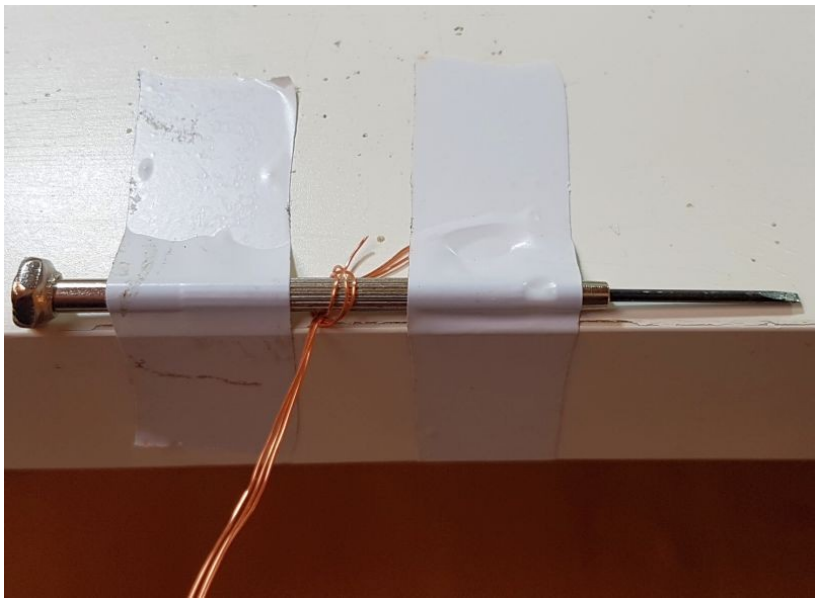
2.12 Wind and install trifilar toroid T2

This toroid needs some care so please follow these instructions very carefully.

Cut three lengths of 0.33mm (AWG #28) enameled wire, of length 20cm each. These three pieces now need to be tightly twisted together to make the trifilar wire. My method for this is to tie one end in a knot around a small screwdriver shaft. Similarly tie the other end around another small screwdriver. Now clamp one end somehow to something solid. You could use a vise, if you have one. If you don't, then you have to get creative and think of something. Here I taped it to the edge of the desk. Now you can twist the screwdriver at the free end, repeatedly until you twist the three wires together thoroughly. You need to keep the wire under a little tension to keep the twists evenly spaced.





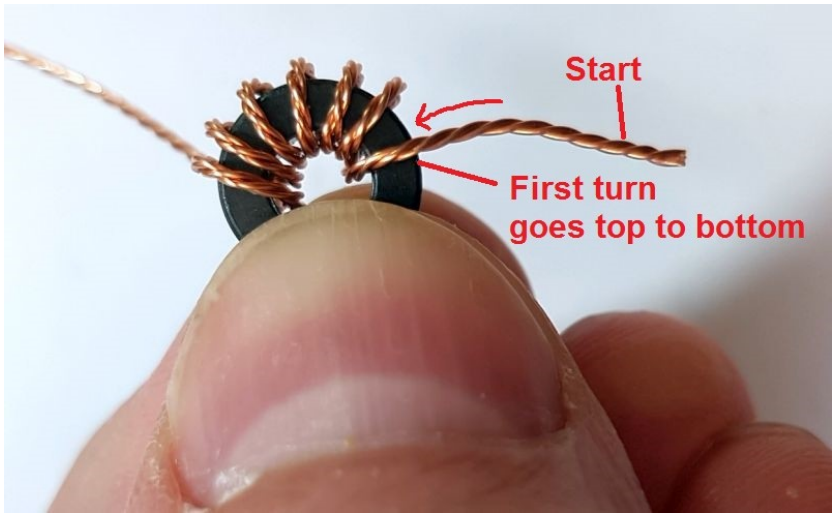


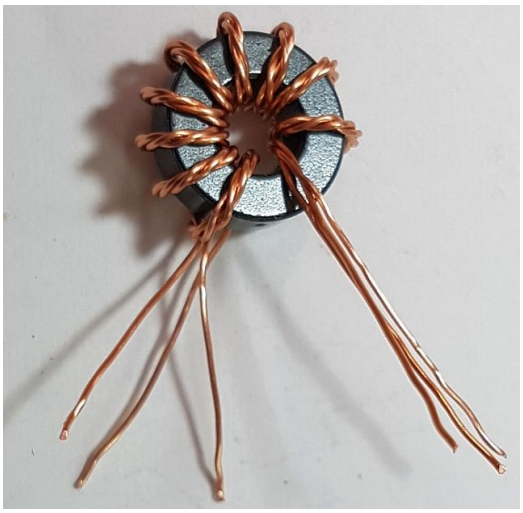
A 20cm length of wire should have about 60 twists. It is not critical.

The end result is something like the photo (right). The measurement scale is in cm.

Now cut off the untidy ends, and this is the piece of wire that will be used to wind the FT37-43 toroidal core as a trifilar transformer.

Hold the core between thumb and finger. Pass the wire first from above, to below. Then take the wire from below, and bring it around to pass through the toroid again to form the second turn. After each turn, ensure the wire is fitting snugly around the toroidal core. Wind 10 turns on the core. Each time through the toroid's central hole counts as one turn. Cut off the excess wire, leaving about 2.5cm remaining.

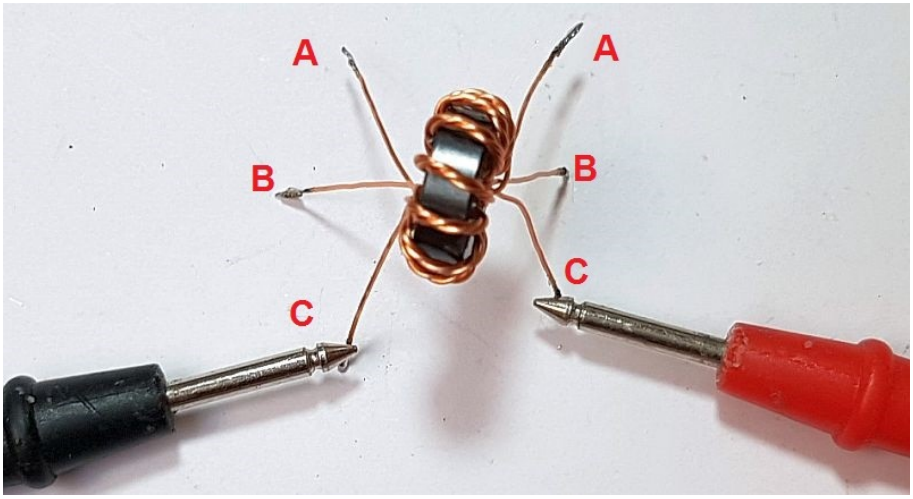




Now it's necessary to identify which wire belongs to which winding. You have three windings twisted together, they all use the same wire. The only way to do this is with a DMM as continuity tester. First, untwist and straighten the wire ends that are not wound around the toroidal core.

Now tin the last few mm at the ends of each wire. You can do this by scraping off the enamel then tinning with the soldering iron; or, if your soldering iron is powerful enough, hold the wire end in a blob of molten solder for a few (maybe 10) seconds, until the enamel burns off.

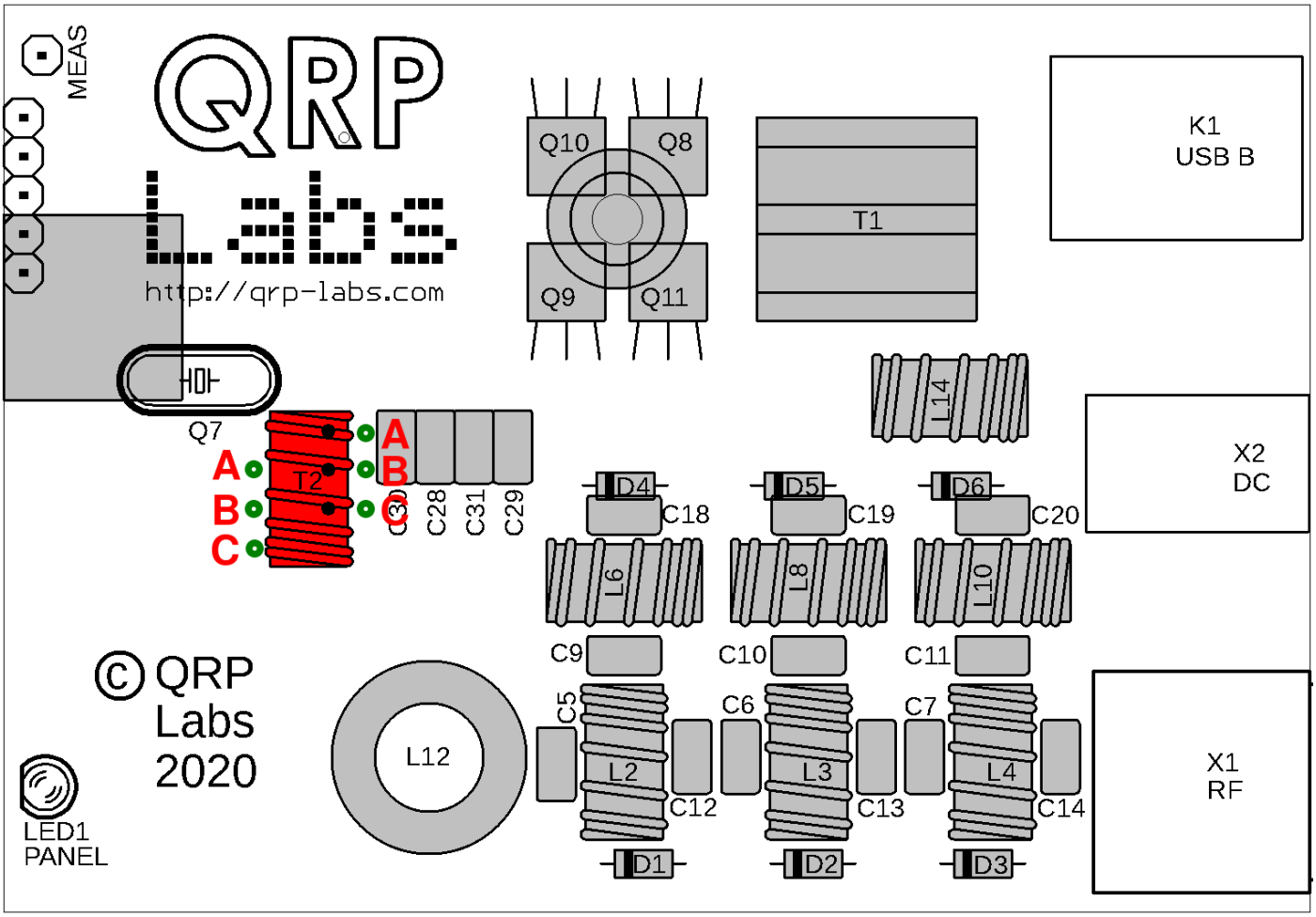
Now use a DMM to test for continuity. Re-arrange the wires so that there is continuity from A-A, B-B, and C-C in this photo.



Carefully keep this orientation of wires and insert the transformer this way into the PCB. BE CAREFUL not to lose the orientation of the wires! The right wires must be in the right holes, so that the windings are connected correctly in the circuit!

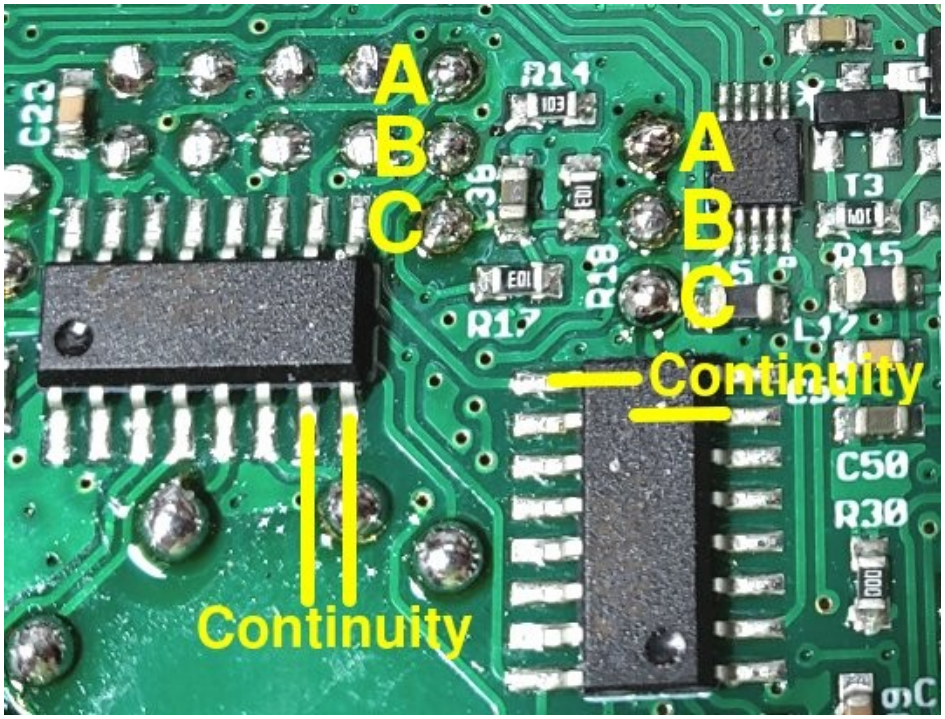
Refer to the diagram on the following page to see the correct orientation.

Once the wires are inserted through the correct holes in the PCB, and pulled tight, check for continuity between the pairs of wires in holes A, B and C once AGAIN. It is much easier to get this right first time, than it is to make repairs later!



When you are satisfied that the wires are all in the correct holes, you can cut them to a length of about 2mm and solder them. It is best to cut-and-solder one wire at a time, since if you cut all the wires to 2mm length then the toroid is more likely to fall out before you've had a chance to solder any of the wires. If that happens, it will be tough to get all the wires back in the correct holes again.

There are numerous SMD components in the vicinity so be VERY careful when soldering the wires of toroid T2, NOT to touch any of the nearby SMD components!



Once again my method involves holding the iron and plenty of solder on the joint for at least 10 seconds to make sure the enamel burns off completely.

The photograph (right) shows the A B C-labeled holes on the lower side of the PCB. You should check for continuity between the pairs of wires, IC3 pins 7 and 8; and IC4 pins 7 and 9. There should be no continuity between IC4 pin 7 and IC3 pin 8 for example.

2.13 Install the 3mm red status LED

Identify the shorter of the two wires of the 3mm red LED and bend it 90-degrees right at the body of the LED.

Bend the other wire 90-degrees but at a distance of approximately 2.5mm (0.1-inches) from the body of the LED (see photograph, right).

Insert the LED leads into the holes in the PCB. As an additional check, the LED will have a flat part on its body, which will sit flat on the surface of the PCB. Solder the LED In place but be careful that it is accurately aligned in position, protruding over the board edge at right-angles. This will ensure that later if you are using the optional enclosure, the LED will fit without difficulty.

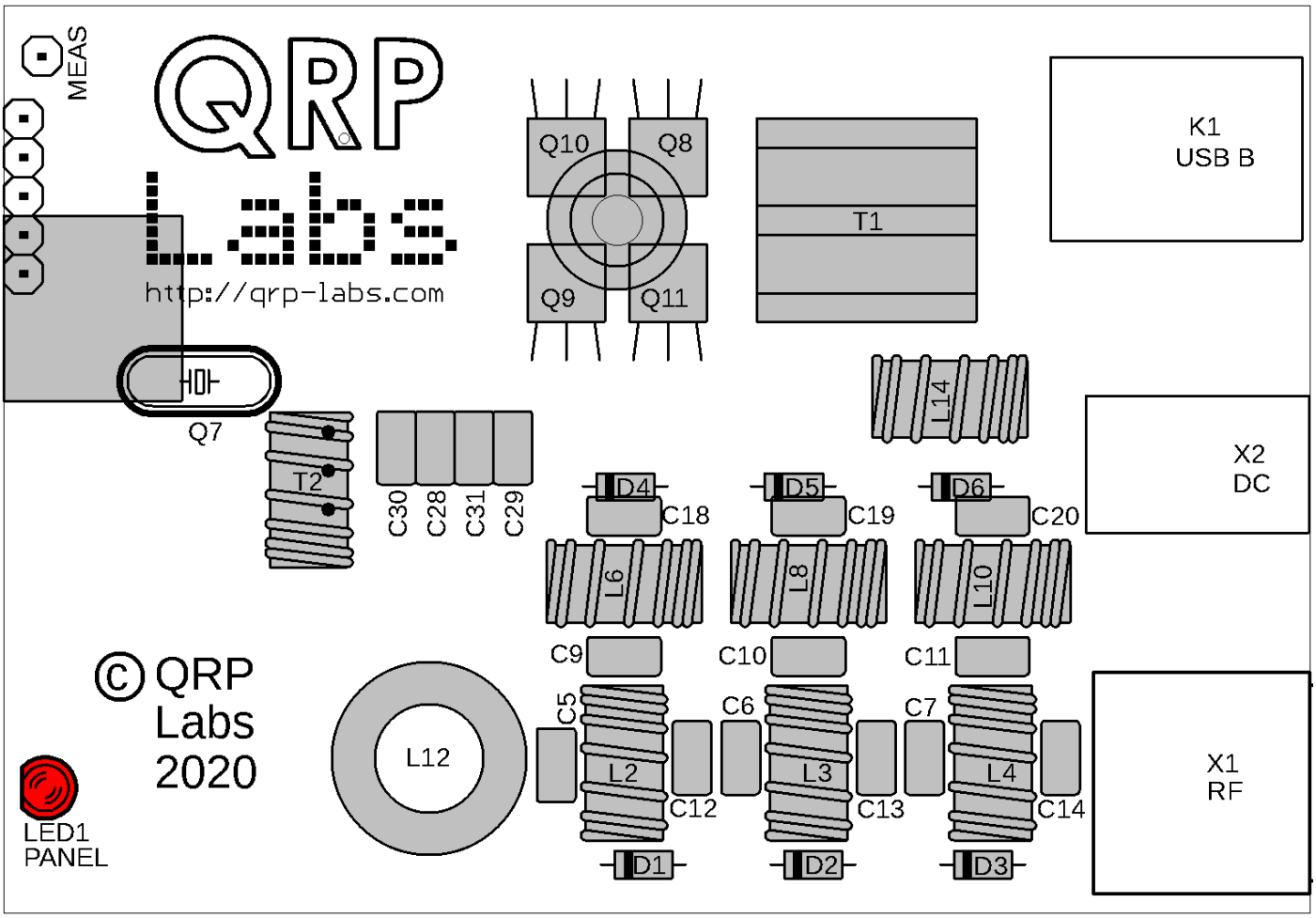


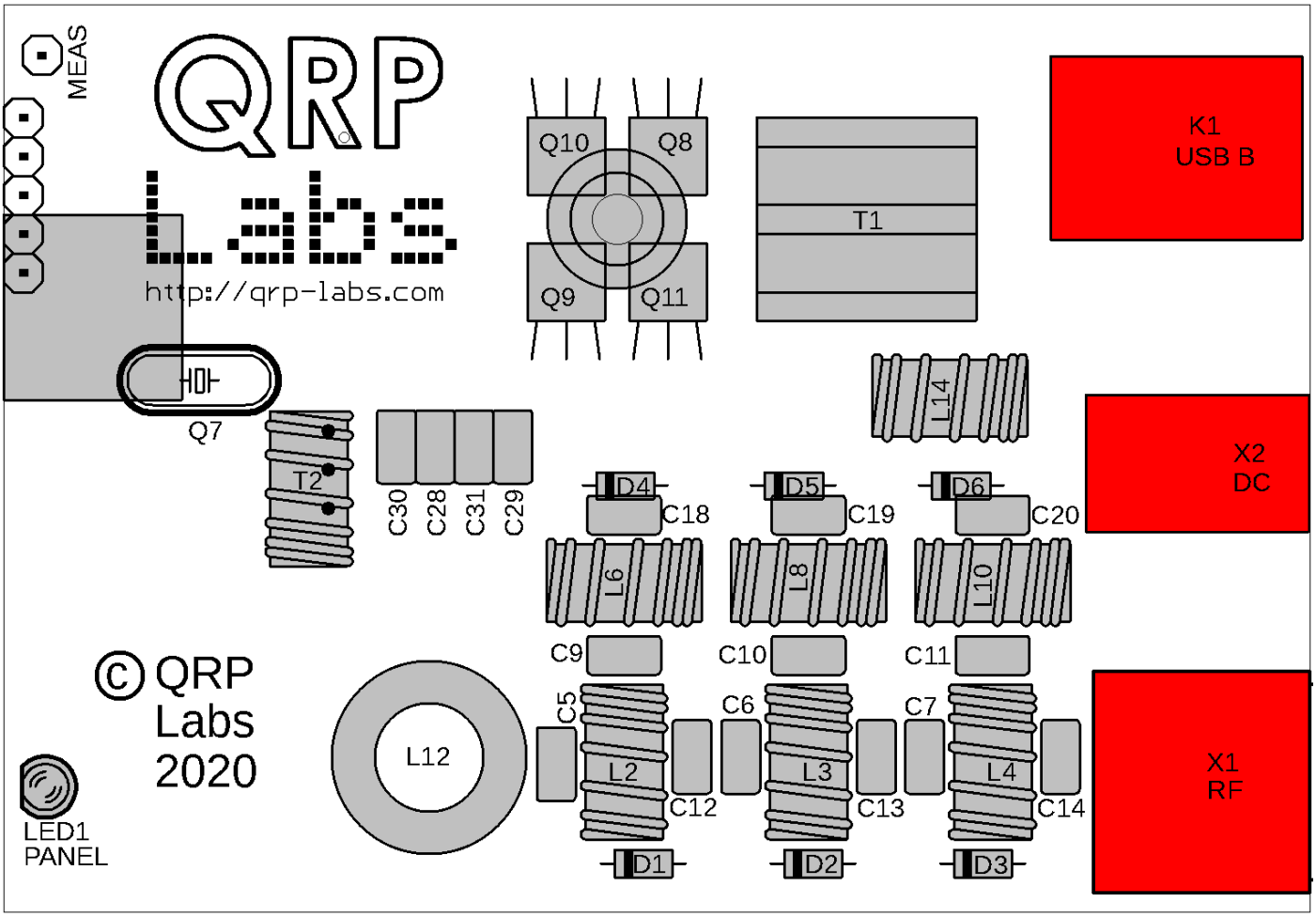
Photo (right) shows the LED installed correctly.

2.14 Install connectors

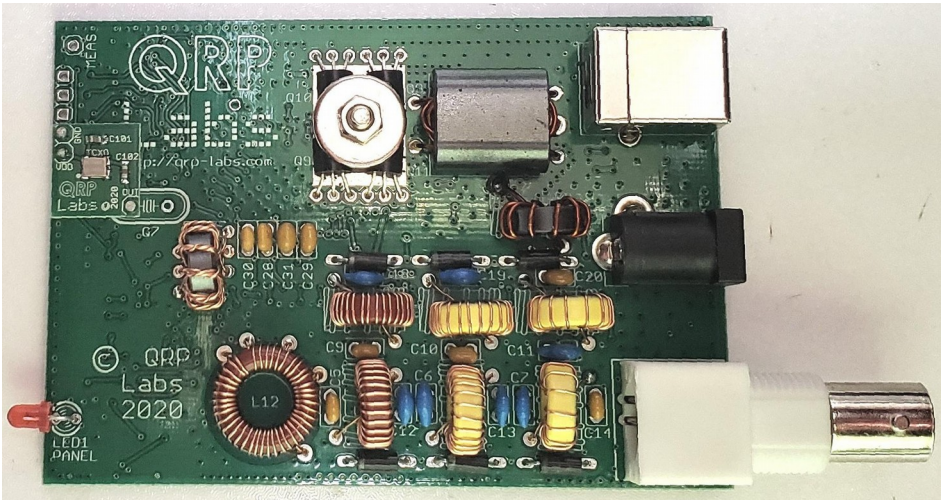
The final components to install are the connectors, from top right to bottom right:

- K1: USB B connector (NOTE: there is no R34, the empty pads for R34 are correct).
- X2: DC Power connector
- X1: BNC RF connectors

It is very important to install these carefully aligned so that they are straight and at right-angles with respect to the PCB. The connector body should not protrude beyond the edge of the PCB. When the connectors are well-aligned, you will have no trouble fitting the PCB into the optional enclosure. Clip off excess leads and tabs to make sure nothing protrudes more than 1.5mm.



Assembly of the QDX kit is now almost complete!



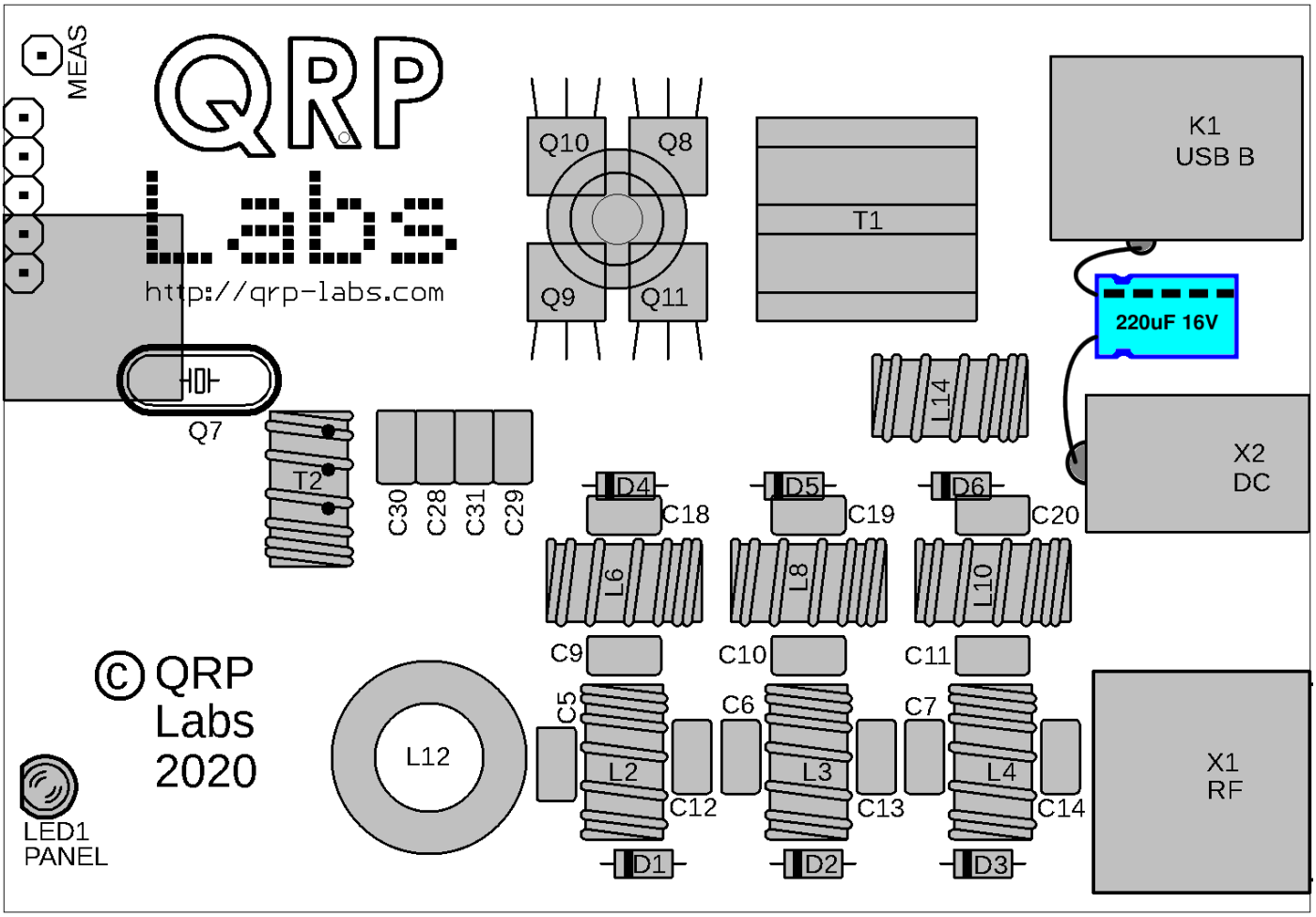
2.15 Install 220uF Power supply capacitor

As mentioned in the Design details section below, the QDX design uses the AMS1117 voltage regulators which were found to be rather weak in the QCX-mini design. The solution is to add a minimum 10uF capacitance at the DC input to the radio. It prevents rapid transitions that can destroy the AMS1117.

Once this capacitor is added, QDX is safe. Since there are no connectors where hot-plugging could be an issue, the AMS1117 is not at further risk.

A 220uF 16V electrolytic capacitor is supplied and it must be fitted at the DC input as shown below. The positive terminal of the DC input connector is the one at its rear (furthest from the board edge). A convenient ground point is the metal mounting tab of the USB connector which is grounded.

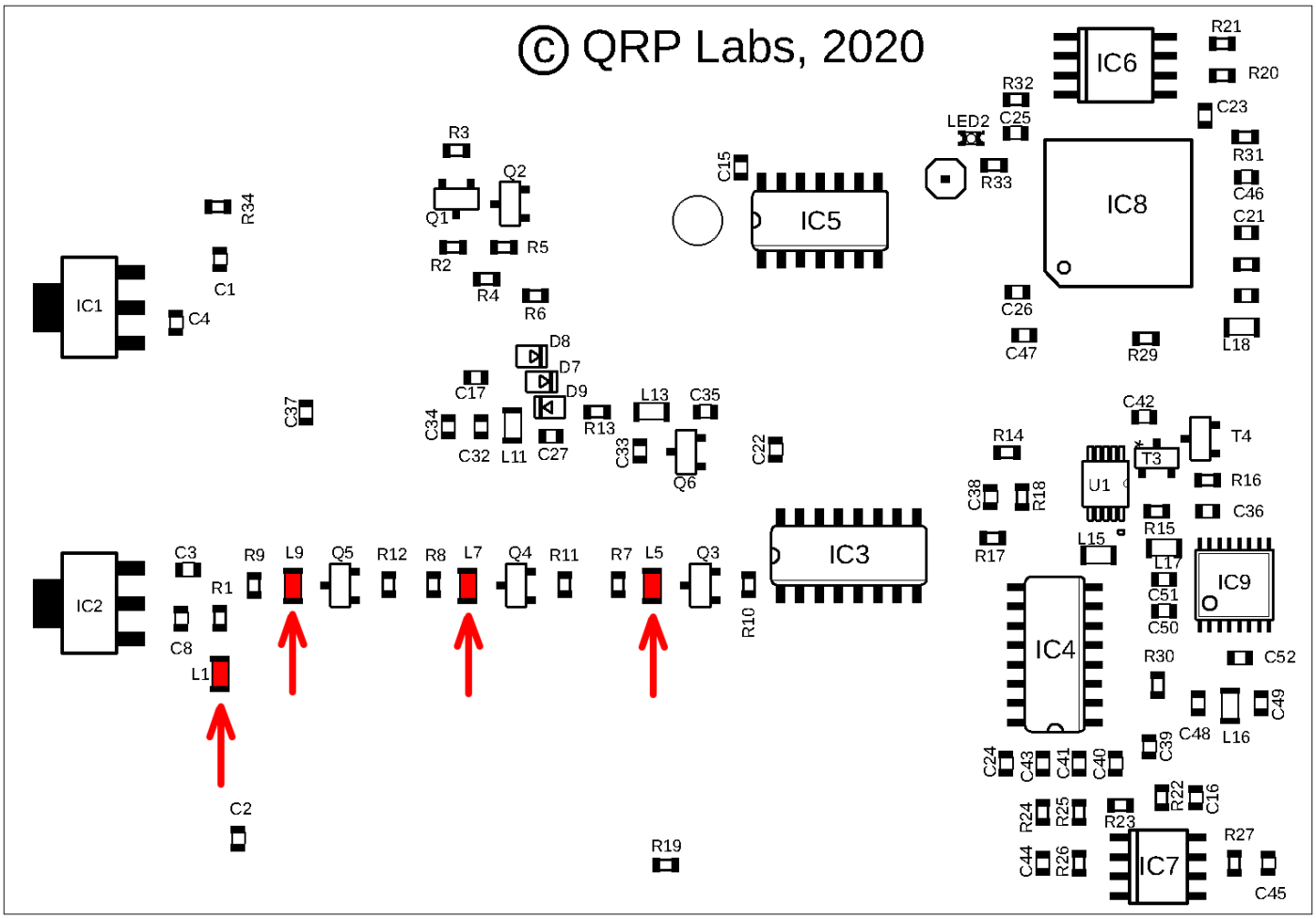
The capacitor should be soldered on the top side of the PCB, between these two points exactly as shown. Make sure that the markings on the capacitor body indicating its negative terminal, are correctly orientated so that the capacitor negative terminal is connected to ground (the USB connector body).



2.16 Replace 47uH inductors

Unfortunately after manufacture, the 47uH SMD inductors used in the PCB assembly were found to have self-resonances in the HF region. Regrettably this causes leakage past the low pass filters and degrades the attenuation of transmitter harmonics.

To solve this problem, it is necessary to remove four 47uH SMD inductors and replace them with through-hole 47uH axial molded inductors (supplied). The inductors are L1, L9, L7, L5. These SMD inductors are easy to remove with a soldering iron, but be sure to correctly identify them first! Removal is made easier by adding a small blob of solder to the iron tip.

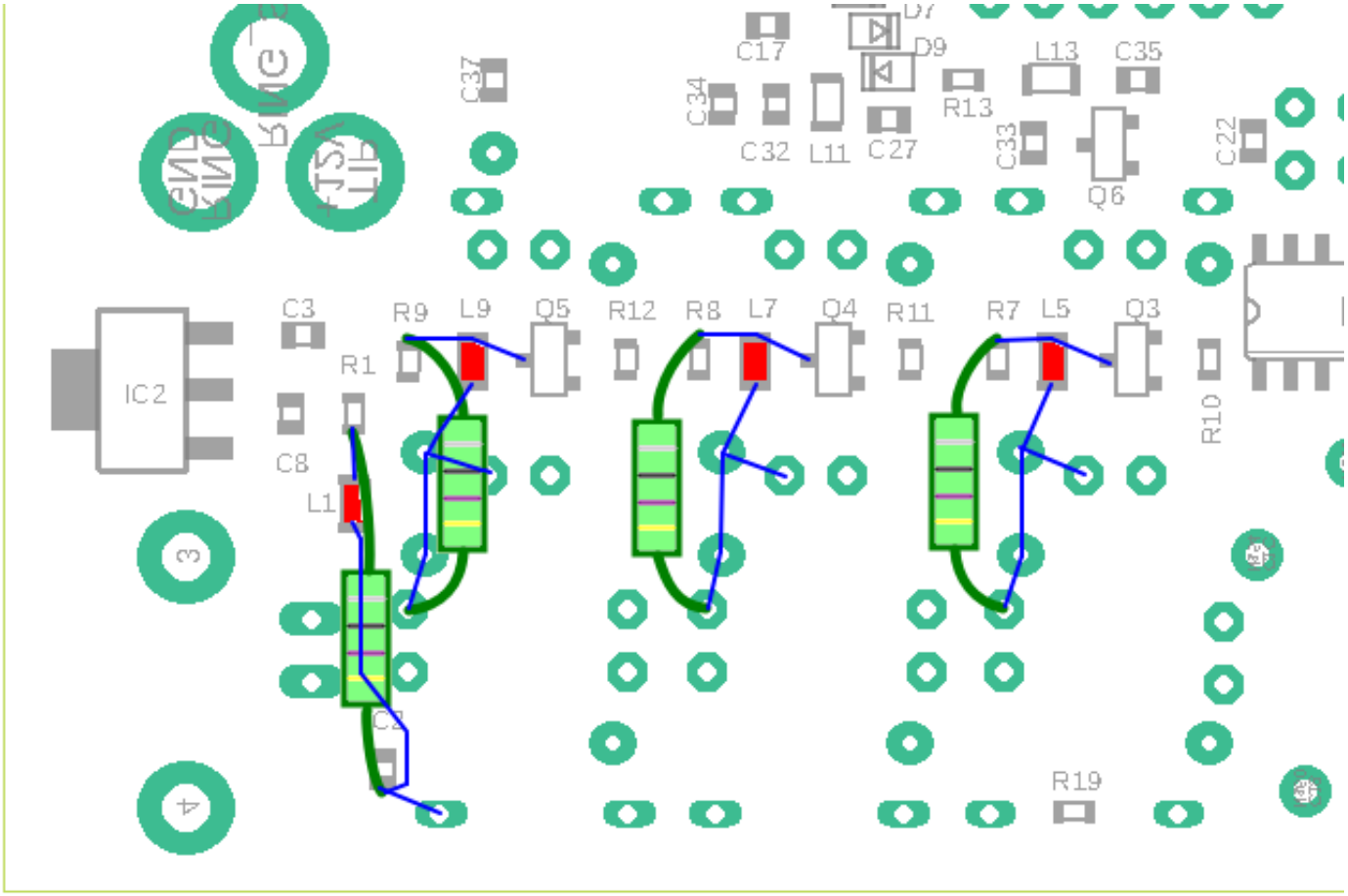


The diagram on the following page shows the best place to connect the leaded through-hole axial 47uH inductors. Do NOT try to connect the inductor leads to the tiny pads where the SMD inductors were installed – the pads are too delicate and will be ripped off too easily. Instead, solder them to a through-hole pad or a lead of an existing through-hole component.

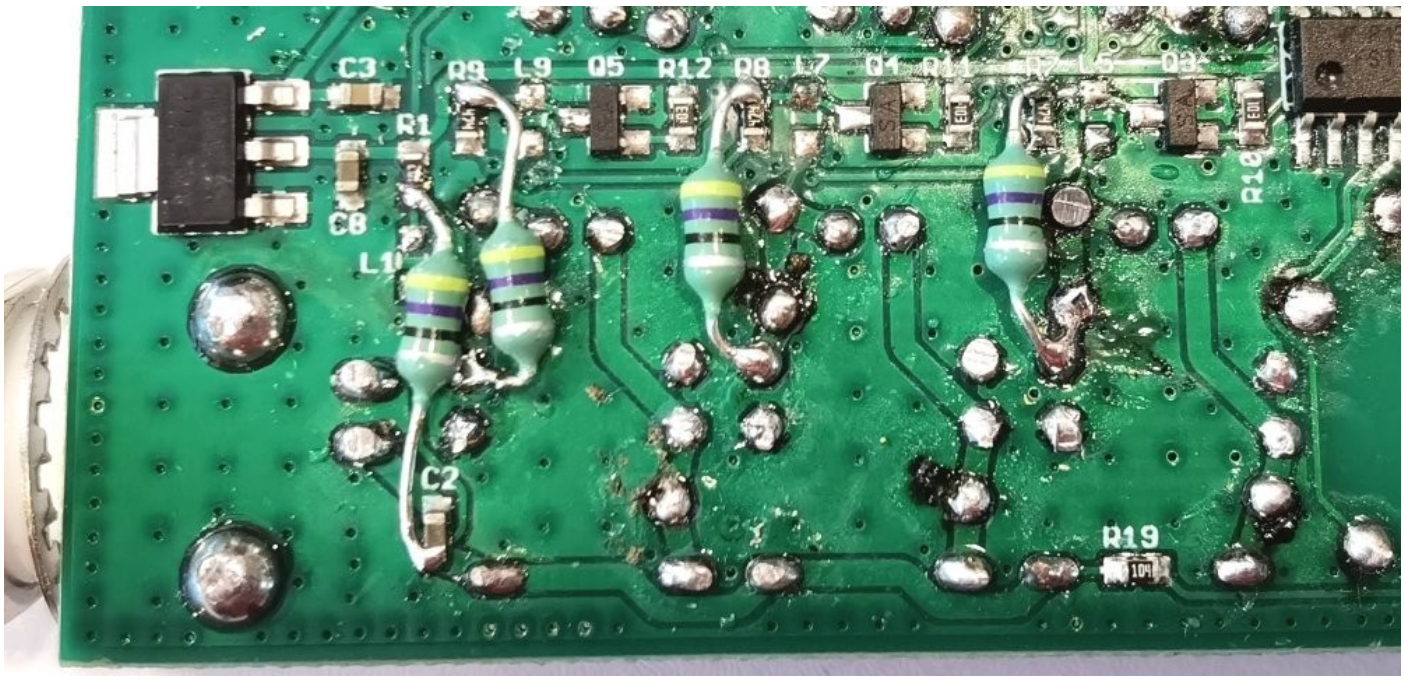
In the following diagram, the blue lines indicate where the tracks are routed, on either side of each inductor. Connecting the through-hole inductors to anywhere on the blue lines will work. I suggest the following:

L1 replacement: from the bottom side of R1, to the bottom side of C2.
L9 replacement: from the top side of R9, to the pad shown.
L7 replacement: from the top side of R8, to the pad shown.
L5 replacement: from the top side of R7, to the pad shown.

Make sure the replacement inductors are soldered to the correct parts of the circuit. Make sure the component leads do not cause short circuits. And make sure the inductors are flat against the PCB so that there is sufficient space for them to fit in the clearance between the PCB and the enclosure bottom.



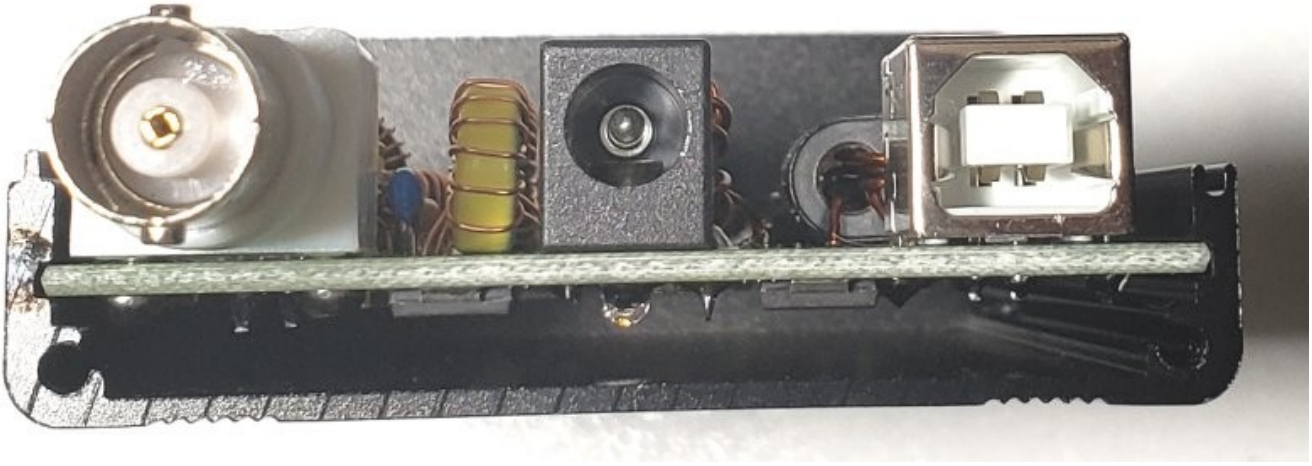
The result is something like this:



2.17 Optional enclosure

Installation of the QDX PCB in the optional enclosure is very straightforward and self-explanatory.

1. Install the screws at the bottom left and bottom right of the front panel, fixing it to the bottom half of the enclosure. Note that the enclosure top and bottom are identical.
2. If the short edges (front and back, i.e. LED status edge, and connectors edge) of the PCB feel rough, it can be a good idea to gently file them so that the PCB edge is square.
3. Gently slide the PCB into the guide-rail grooves in the enclosure bottom.



4. Install the two screws at the bottom left and bottom right of the rear panel, fixing it to the bottom half of the enclosure.
5. Fit the top half of the enclosure. Note that one side has an “I-tongue” and the other side has a “U-groove” and this is repeated on both halves (top and bottom) of the enclosure. Therefore the top half will only fit the bottom half, “one-way-round”. If it does not appear to sit correctly, rotate the top half 180-degrees. Secure the top with the remaining screws.
6. Finally you may, if you wish, fit the nut to the BNC connector. I generally leave off the washer. The nut may be omitted too, it is not really required and should in any event not be over-tightened since there is a small gap between the back surface of the rear panel, and the BNC connector body. If desired, attach the self-adhesive rubber feet to the bottom.

3 Operating instructions

Operation of the QDX transceiver is really simple. There are only three connections:

1. DC Power: use a regulated well-smoothed DC supply of not more than 12V; it should be capable of up to 1A current supply. The connector is a 2.1mm barrel type (outside diameter 5.5mm).
2. RF: a standard BNC connector to your antenna system
3. USB: USB type B connector (a standard USB type A to B cable is required) for Audio and CAT control.

Drivers

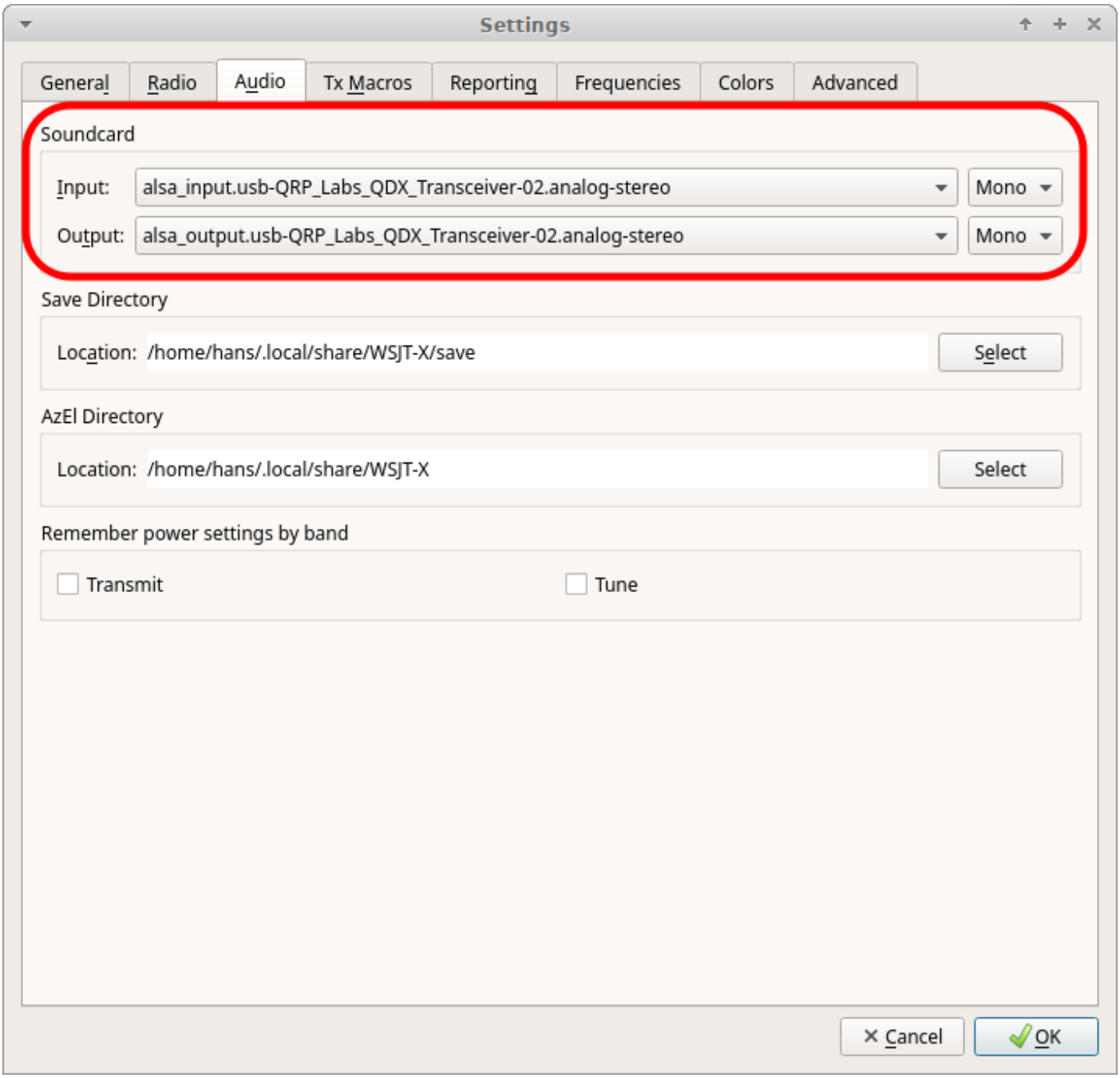
No additional drivers are required for operation with most Linux distributions, Apple Mac or MS Windows 10.

For older versions of MS Windows, it may be necessary to install a driver for the serial port because this driver is not on your computer already by default. This driver is available from the ST Semiconductor website at https://www.st.com/en/development-tools/stsw-stm32102.html and is applicable to 98SE, 2000, XP, Vista®, 7, and 8.x Operating Systems. There is a description for installation on Windows 7/8 on the QRP Labs QLG2 page http://qrp-labs.com/qlg2 so if in doubt, please check this.

WSJT-X configuration

Next it is necessary to set up WSJT-X to communicate with QDX. We will use WSJT-X as the example, because it will be what most people are using. But other software will be identical (for example JS8Call) or similar. There are two parts to the set-up – firstly to choose the right USB Sound card, and secondly to set up the CAT communication so that WSJT-X can control the QDX via the serial comm port.

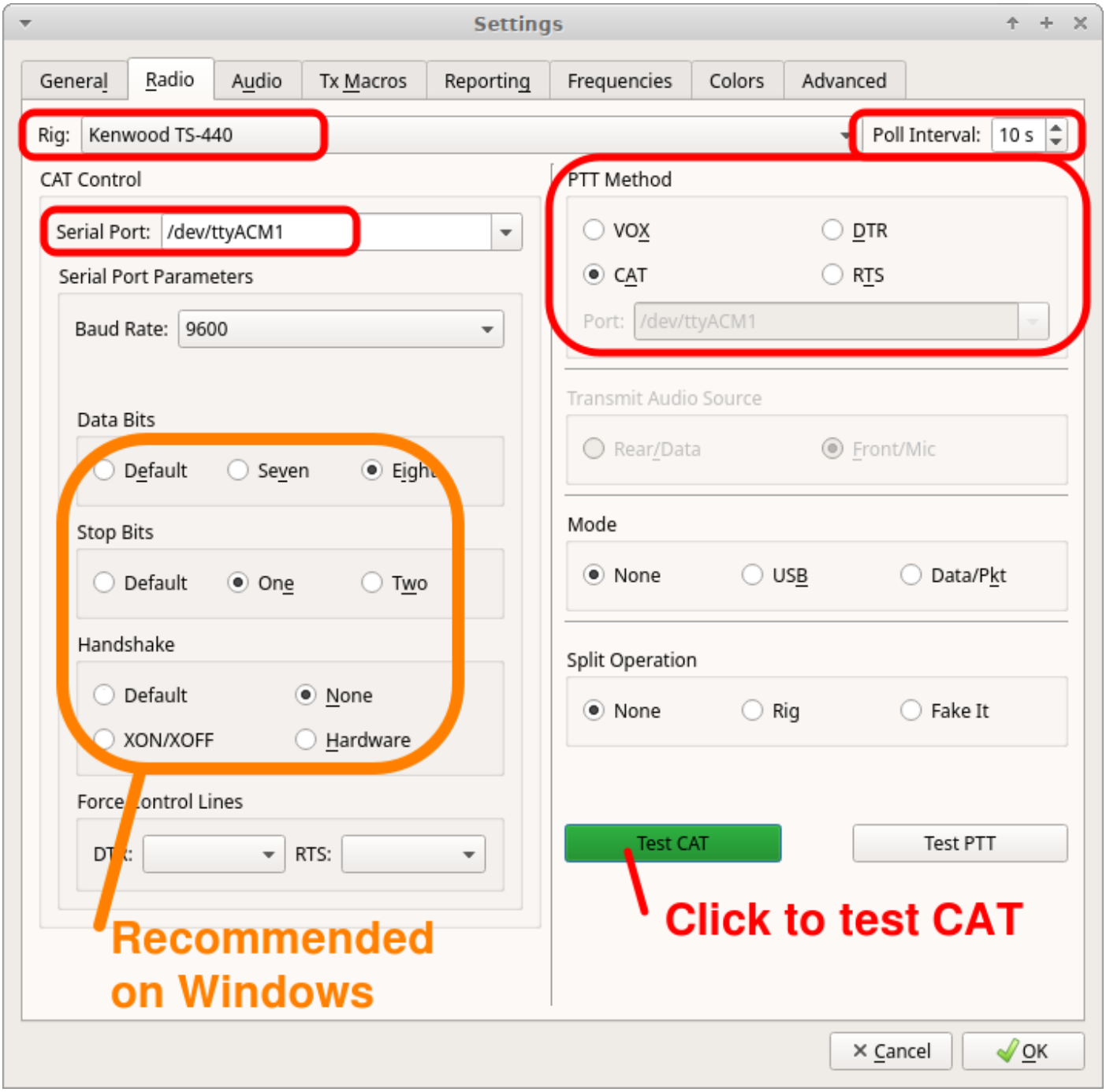
Open the WSJT-X settings window (from the File menu) and select the Audio tab. Select “QRP Labs QDX Transceiver” as the input and output sound card. The below screenshot shows how it looks on my system, which is Linux (Xubuntu 18.04). It will look different on Windows, Mac and perhaps other Linux distros but the basic idea will be the same… you should see something in the drop-down which says something about QDX, and that’s the sound card to select.



Next click the “Radio” tab in the settings window, which sets up the CAT control communication.

The following four settings need to be changed, and are illustrated in the diagram below:

- Rig will be set to None by default, click the drop down and choose “Kenwood TS-440” which should work well with QDX. On some other software, if you find that TS-440 is not present in the list or does not work properly, you could try “Kenwood TS-480”. More details about CAT and debugging any CAT problems are in another section of this manual, where the CAT test terminal screen is described.
- The Serial Port drop-down must be set to the correct port where QDX is connected. On my Linux system it is either “/dev/ttyACM0 or /dev/ttyACM1. On Windows systems it will be a COM port numbered COM1, COM2 etc. Unfortunately unlike the USB Sound, the serial port name doesn’t contain the text “QDX”. If you are unsure which port to choose for QDX, the easy way to find this is as follows. Unplug QDX. Restart WSJT-X. Look in Settings -> Radio and make a note of the list of serial devices. None of these are QDX (because you unplugged it). Now close WSJT-X, plug in QDX, start WSJT-X and again look in Settings -> Radio, and now you should see a newcomer in the list of available ports. The newcomer is QDX!
- Note that none of the Serial Port Parameters need to be changed, leave them all at their defaults. Even the baud rate 9600 is unimportant because it is irrelevant to the USB Virtual COM Port which is a virtual port over USB, not a real physical serial port.
- Change the Poll Interval to 10 seconds, the default will be rather chatty with QDX which probably is not a problem, but anyway I feel more comfortable with the less frequent polling. QDX has no capability to alter its operating frequency for example by itself, it can only do so at the command of WSJT-X over CAT; therefore the polling is actually redundant anyway.
- Change PTT Method from the default “VOX” to “CAT”. VOX means “voice operated exchange” or “voice activated transmission”; the radio will automatically switch to transmit, when incoming audio is detected. With PTT Method set to CAT, when WSJT-X wishes to start a transmission, it will send an actual CAT command to QDX informing it to start the transmission, before sending the audio. This CAT command causes QDX to switch from Receive mode to Transmit mode (and back again afterwards). “CAT” is preferable to “VOX” because if system sounds are accidentally routed to your “QDX” sound card as output, then with VOX that will enable the QDX transmitter and try to transmit the sound.
- Now click the “Test CAT” button and after a few seconds, it should turn Green to indicate successful communication with QDX.



NOTE 1: If you are using other software than WSJT-X or JS8Call, then QDX CAT commands should still work with this software. If you encounter difficulties then it is possible that your software is trying to communicate with QDX using CAT commands that are not supported by QDX. In the section of this manual on the CAT Test utility (in the QDX Terminal applications), you will find a listing of the CAT commands supported by QDX. Another useful utility is the log file, which will let you record all CAT commands received and investigate any issues. If CAT commands are missing for your application, QRP Labs can add support for them easily.

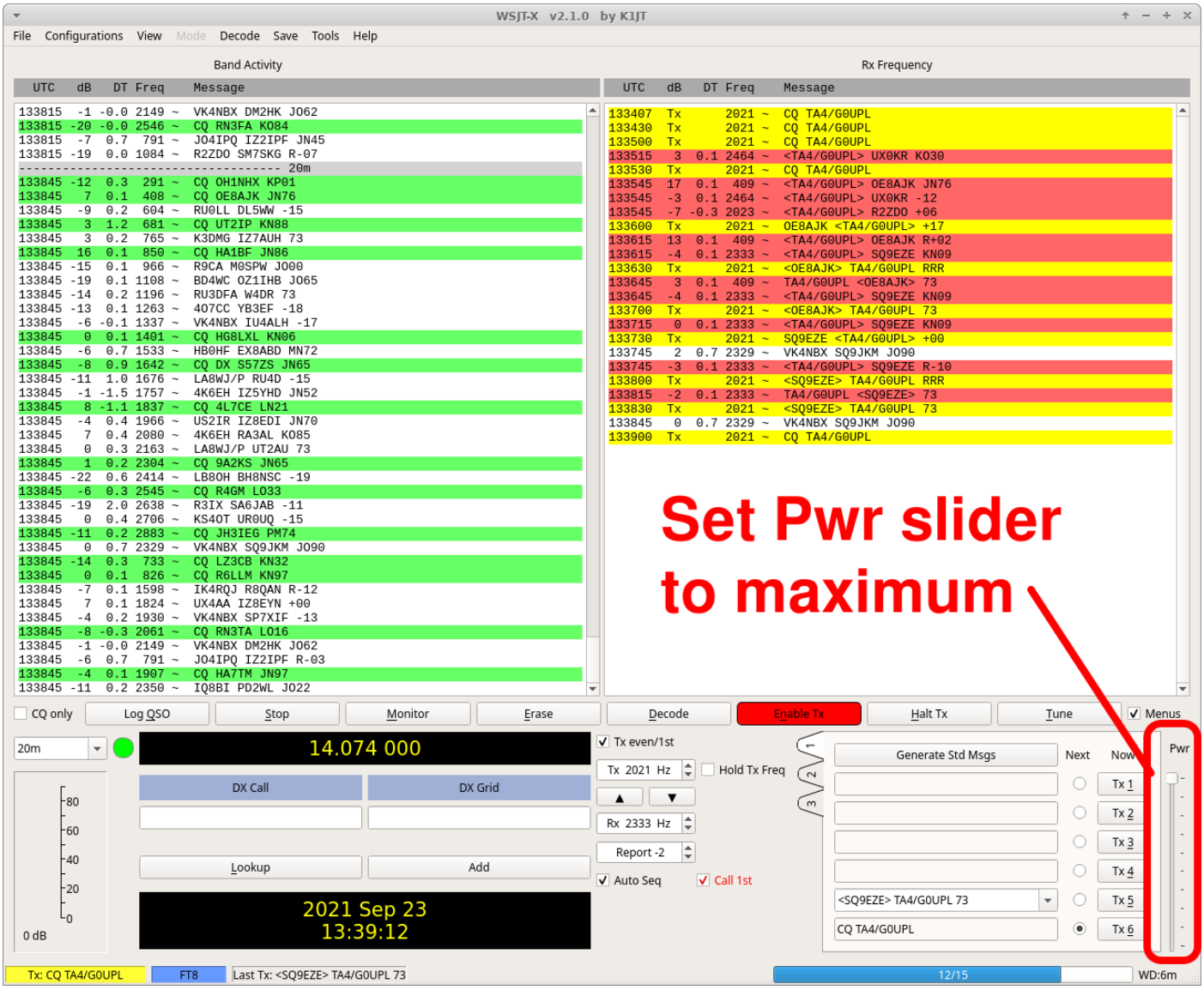
NOTE 2: As mentioned above, CAT control of transmit/receive switching is recommended. If you INSIST on using VOX, QDX can support that. For example, you may be using a software application which does not support CAT control of transmit/receive switching and can only use VOX. In that case you should change the QDX transmit/receive switching mode from CAT to VOX in the QDX terminal Configuration utility, which is described in the terminal applications section of this manual.

NOTE 3: The Data Bits, Stop Bits, Handshake should not need to be changed; however several users have reported that changing them to the settings shown in the orange box has resolved some issues with CAT reliability on Windows Operating Systems.

WSJT-X “Pwr” Slider

The only other point to note is that WSJT-X should be operated with the power slider at the maximum setting. This point is discussed further in the QDX design section which explains that best accuracy in determining the audio tone frequency being sent by the PC, is when the Pwr is at the maximum setting. There is no point to using any setting other than maximum, because QDX only ever transmits at full power (5W), there is no way for it to transmit at a lower power output under command of WSJT-X. If you wanted a lower power output, you would need to use a lower supply voltage. Furthermore, QDX cannot be “over-driven” by too high volume, in the way that a SSB transceiver could.

Therefore the “Maximum” setting for the Pwr slider is highly recommended, it is the optimum setting for QDX operation.



QDX Status LED

The front panel of QDX contains a 3mm red Status LED. The operation of this LED is as follows:

- Quick flashing (flickering) for first 5 seconds after power up: if you disconnect power now, then next time you power up, QDX will be in firmware update mode. During this five seconds QDX is also operating normally in Receive mode. After five seconds, the LED will go to a steady on (lit) state.
- Slow flashing: QDX is in firmware update mode (refer to the Firmware Update procedure section of this manual). If you did not wish to be in firmware update mode and have entered firmware update mode inadvertently, simply power down QDX and re-apply power; QDX will now be in normal operating mode again (quick flashing for the first five seconds, see above).
- Steady LED on state: QDX is in normal Receive mode operation
- LED flashes 3 times in quick succession, repeating every 1 second: QDX is in normal Transmit mode operation

Operate!

Once CAT is configured and working, and the QDX sound card is chosen, just operate WSJT-X as you would normally! You can choose the desired band 80m, 40m, 30m or 20m from the WSJT-X screen and WSJT-X will communicate with QDX via CAT, to cause QDX to switch in the correct filters.

This QDX manual is not the place to include tutorials on various digi mode operation or particular application software such as WSJT-X, such guides are readily available and written very much more thoroughly than I could hope to achieve!

4 Design

4.1 Summary

Traditionally it was normally assumed that transmission of digital modes requires a computer, to generate the audio tones, and an SSB transceiver, to modulate those to RF and transmit them via an antenna.

Yet – the widespread assumption that an SSB transceiver is REQUIRED is in fact absolutely incorrect, at least for a wide category of digital modes transmitting a single tone frequency shift keyed signal. This class of transmissions are NOT SSB, they are a single carrier, frequency modulated. Potentially this can provide simplification of the radio transceiver design, delivering both higher performance and lower cost.

Why is this important? An SSB transmitter design is necessarily relatively complex and a non-trivial undertaking for several reasons. The audio signal must be converted to RF by an SSB exciter. An exciter may commonly take the form of a superheterodyne architecture having a modulator, IF filter (crystals) and another mixer to convert to the final RF operating frequency. It then requires a linear amplification chain to raise the signal amplitude to the required power output at the antenna port. An alternative architecture is a phasing network exciter which makes use of phasing techniques to null the unwanted sideband, and converts directly from baseband to RF. Again it is followed by a linear amplification chain. In both cases considerable attention to the design is required, to minimize the residual carrier frequency, and the unwanted sideband.

Linearity must be maintained throughout the amplifier chain to the antenna port, to avoid splattering onto adjacent frequencies. Even though the audio signal is theoretically a single audio tone, harmonics of the audio tone may be present particularly if the audio level is too high, and the carrier suppression and unwanted sideband are never perfect; so there is plenty of potential for generation of intermodulation products, hence the need for good linearity throughout. A linear amplifier (for example, Class AB) has low efficiency compared to non-linear amplifiers such as Class C. Low efficiency means higher current draw for a given power output (less battery-friendly in portable scenarios), and higher heat production, probably requiring heatsinking of the PA transistors – increasing size, cost and weight.

Double Sideband (DSB) is even worse – the designs are simpler but here we have an equal amplitude unwanted sideband emission. Half the transmitter's power is wasted on this unwanted sideband, which potentially can interfere with users of adjacent channels. On receive, we suffer a 3dB degradation in noise floor at best, and at worst, interference from signals in the unwanted sideband. With the strong double tone (both sidebands) the potential for intermodulation products due to non-linearities in the amplification chain is higher. Except under optimal conditions, and even then with difficulty, two DSB transceivers cannot QSO with each other! A DSB transceiver can only QSO with an SSB transceiver at the other end. DSB is the only mode which cannot even communicate with itself! Some would say, an abomination of a mode.

If there were a way to convert the audio tones from the computer, to RF, without going through the SSB modulation process and linear amplification, it could solve many problems.

Back in 2012 the QRP Labs Ultimate QRSS/WSPR/Digi transmitter kit was the first kit to generate and transmit digital signals standalone with no PC required, and without an SSB transmitter. It used just a microcontroller calculating the required frequency shifts and commanding an RF signal generator – then the AD9850 Direct Digital Synthesis (DDS) chip – to produce the required frequencies. Several iterations of the kit evolved over the years, to the current Ultimate3S http://qrp-labs.com/ultimate3/u3s which can transmit quite a list of digital modes in the standalone beacon application.

However for QSO digital modes, using computer (PC) software to encode and decode the audio, SSB transceivers remained the norm.

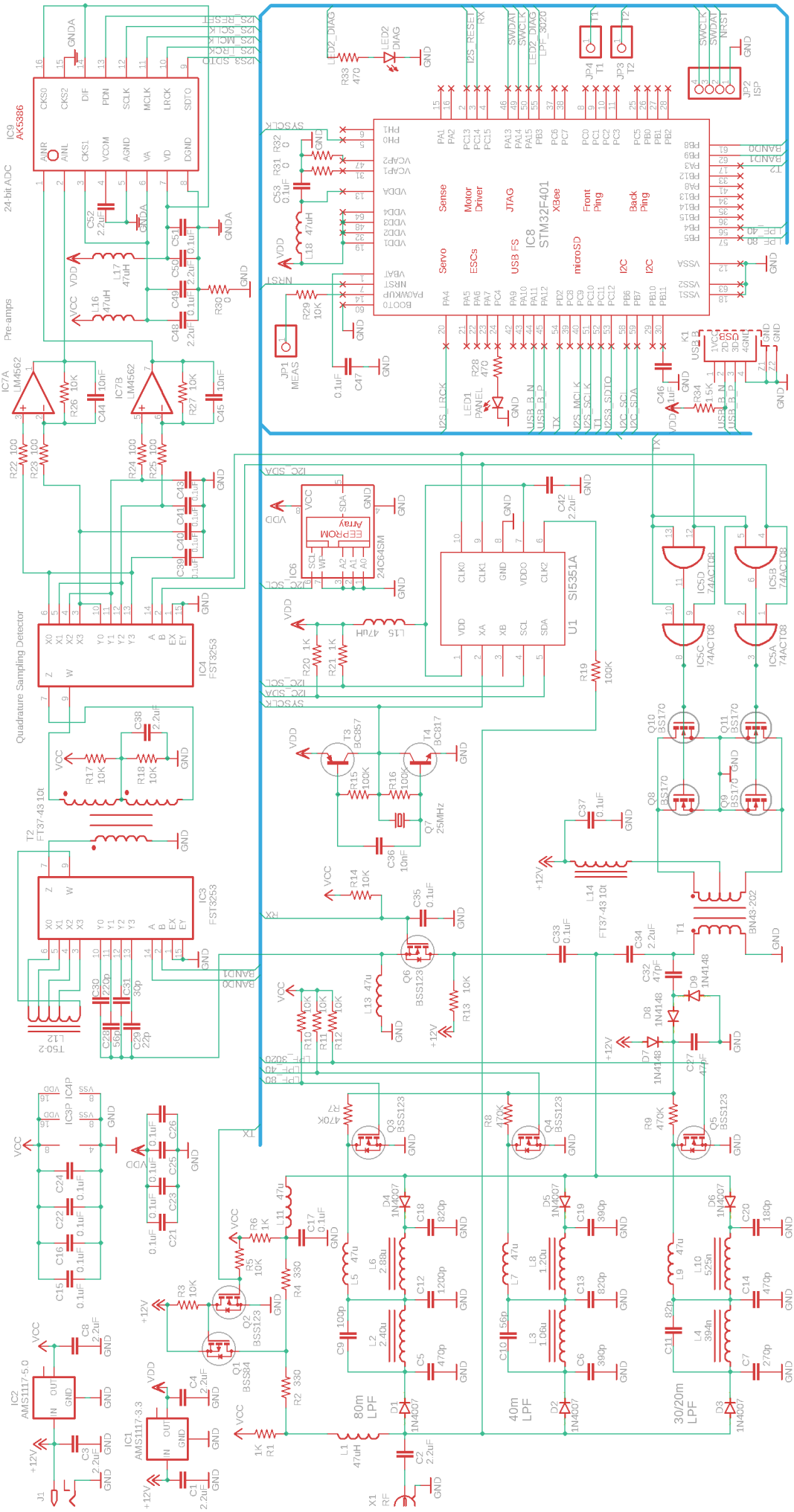
In the second half of 2019, I was thinking about FT8, and a number of simple low cost FT8 transceiver kits that were available (DSB transceivers), none of which offered a particularly good performance. I realized that there IS a better way to do this, that will dramatically increase the performance without increasing the price. Straight away I designed QDX and built the prototype.

The fact that the computer is generating the audio, does not need to stop us from generating a single RF tone from an RF signal generator, just as the Ultimate3S QRSS/WSPR/Digi kit does. A microcontroller can analyze the audio arriving from the PC, and determine the audio frequency of the tone; then add that to an RF base frequency (which on an SSB radio, we would call the “USB Dial Frequency”), to determine the correct RF carrier frequency to generate for transmission. The rest is then a matter of commanding an RF generator to produce the calculated frequency, and amplifying it for presentation to the antenna. The generated signal is absolutely clean, having only harmonically related unwanted spurious emissions which can be taken care of in the conventional way by Low Pass Filtering.

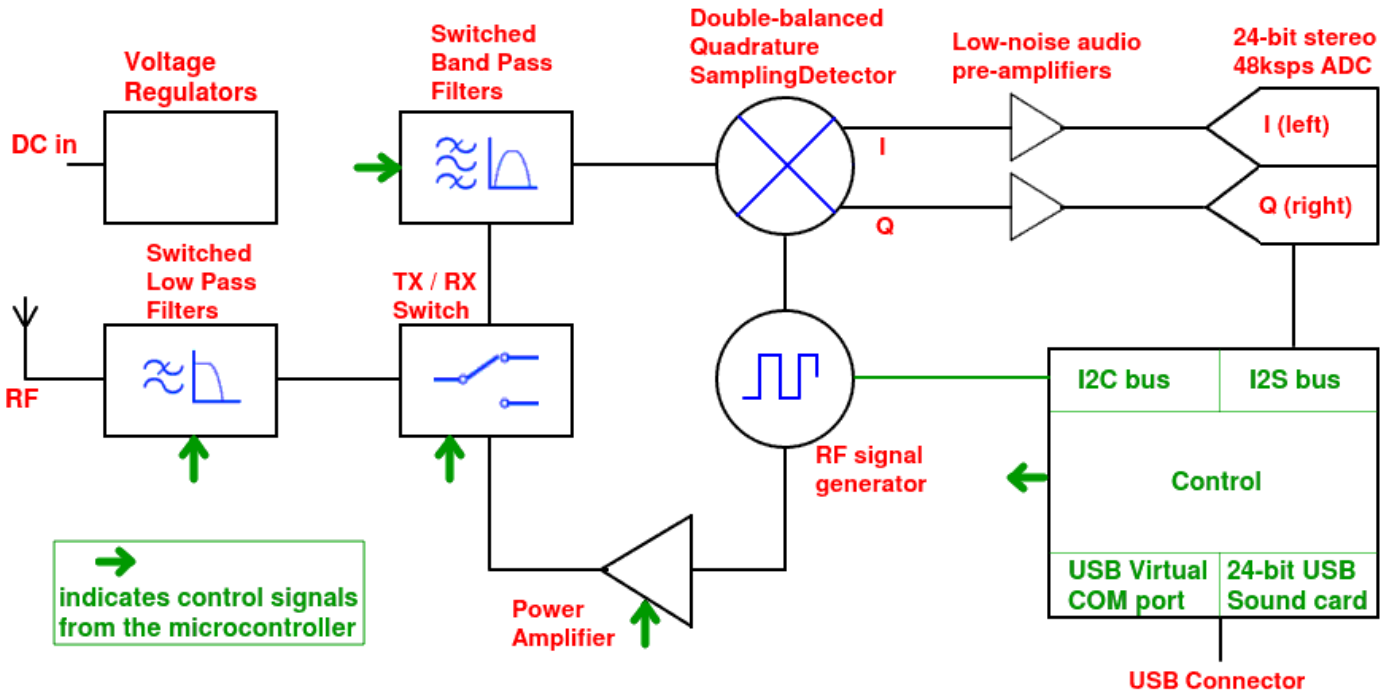
The importance of this technique cannot be understated.

- We no longer need a complex SSB modulator involving multiple mixers, IF filters (or phasing in the case of the direct method).
- No more linear amplification chain – we can use higher efficiency non-linear amplifier classes, reducing the need for heatsinking, and lowering transmit current, very battery friendly.
- The unwanted sideband simply DOES NOT EXIST. It isn’t just “there but attenuated” depending on how well the transceiver was designed and aligned - it just isn’t there at all.
- Similarly the carrier also does not exist. Carrier suppression in a modulator depends on how well the mixer is balanced and is never perfect. But here, we generate a single tone directly – and there’s NO SSB carrier at all. Zero.
- It is impossible to overdrive this design by turning up the audio volume too high on the PC. Therefore it is impossible to generate a messy signal that splatters on to adjacent frequencies.

With this in mind, the rest of the QDX transceiver design falls into place, around this central concept.



4.2 Block Diagram



QDX consists of the following circuit blocks:

1. A synthesized Local Oscillator using the famous Si5351A Digital PLL signal generator chip. A 0.25ppm 25MHz TCXO is included as standard as the PLL reference.
2. Low Pass Filtering to attenuate emissions at harmonics of the operating frequency. Filters are provided to cover 80, 40, 30 and 20m bands, and selected using solid state, PIN diode switching.
3. A solid state transmit/receive switch – no bulky, noisy, expensive, unreliable relays here!
4. A switched receiver band pass filter to provide some degree of protection to the mixer against out of band signals.
5. A high performance, double balanced Quadrature Sampling Detector to mix the incoming RF signals to baseband and produce I and Q signals.
6. A low noise pre-amplifier for the I and Q signals.
7. High performance 24-bit 48ksps (Kilo Samples Per Second) stereo Analog To Digital Converter (ADC) chip with 110dB of dynamic range.
8. An embedded Software Defined Radio (SDR) receiver which implements, digitally, a 12kHz Intermediate Frequency superheterodyne receiver with excellent performance and unwanted sideband suppression, also implementing a sharp digital filter.
9. An embedded 24-bit 48ksps stereo USB Sound card – no more noisy, cluttered audio cables!
10. A CAT control serial interface, also over the same USB cable, to allow the PC software to control the radio (frequency, transmit/receive switching etc) in a standard way.
11. The microcontroller performs single cycle frequency analysis of the audio tone, and commands the Si5351A Signal generator to produce the necessary RF frequency.

12. A Class-D Push-Pull power amplifier that is small, low cost, high efficiency, and produces very low even harmonic output levels, reducing the demands on the Low Pass Filter.

13. Voltage regulation and supply decoupling.

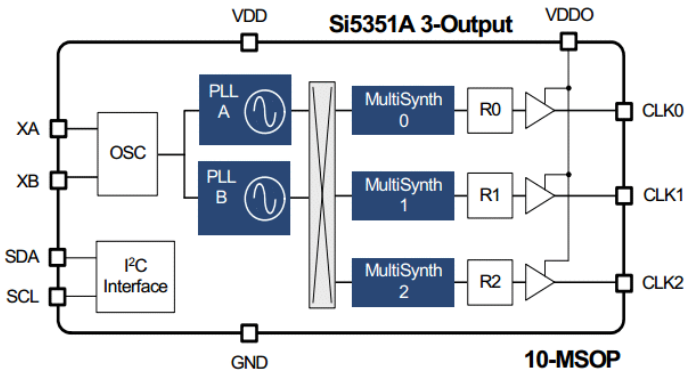
In subsequent sections, each of these blocks will be described in detail.

4.3 Synthesized local oscillator

Generating a stable, precise oscillator signal was one of the most challenging aspects of transceiver design. Modern semiconductors however, make it one of the easiest building blocks of the radio.

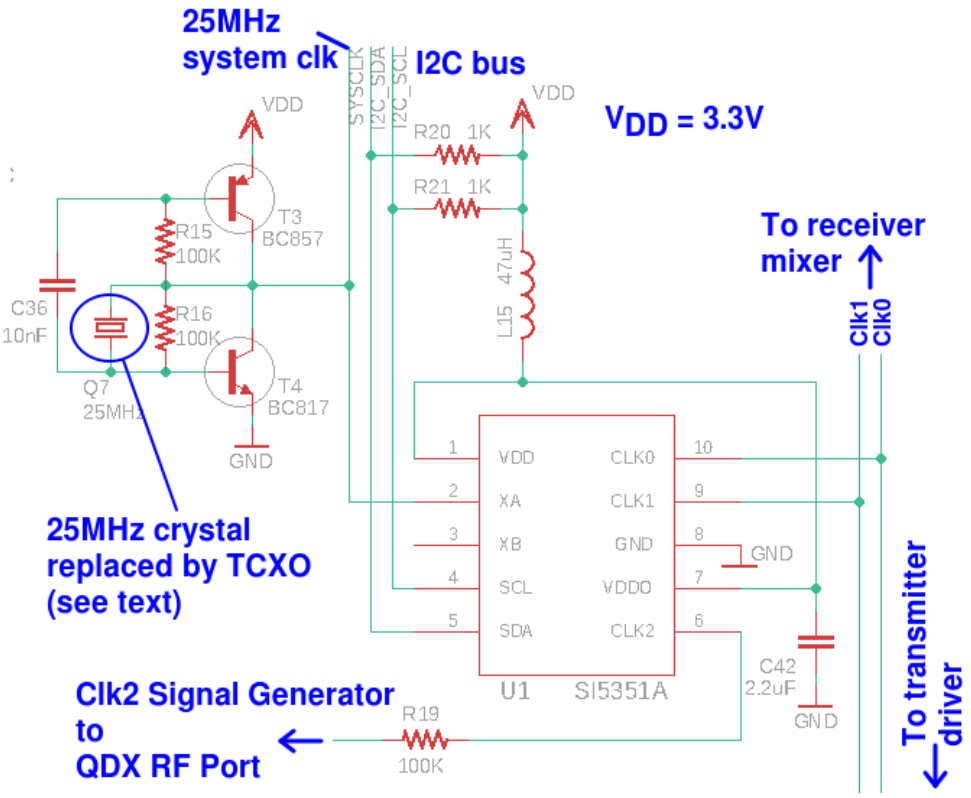
The synthesizer used in the QDX is the same as in the QCX 5W CW transceiver, and various other QRP Labs products: the Si5351A. This is a Digital Phase Locked Loop (PLL or DPLL) synthesizer which provides three separate frequency outputs, each having a frequency range spanning 3.5kHz to 200MHz. The frequency stability is governed by the a crystal reference.

The block diagram (right) is taken from the SiLabs Si5351A datasheet. Briefly, the 27MHz reference oscillator is multiplied up to an internal Voltage Controlled Oscillator in the range 600-900MHz (the PLL), then divided down to produce the final output frequency. The multiplication up and the division down are both fractional and so the frequency resolution is extremely finely controlled. The chip has two PLLs and three output divider units.



For best jitter performance, the Si5351A datasheet recommends the use of even integer dividers (no fractional component) in the MultiSynth dividers and in the QDX transceiver design, this recommendation is followed.

The synthesizer section of the circuit diagram.

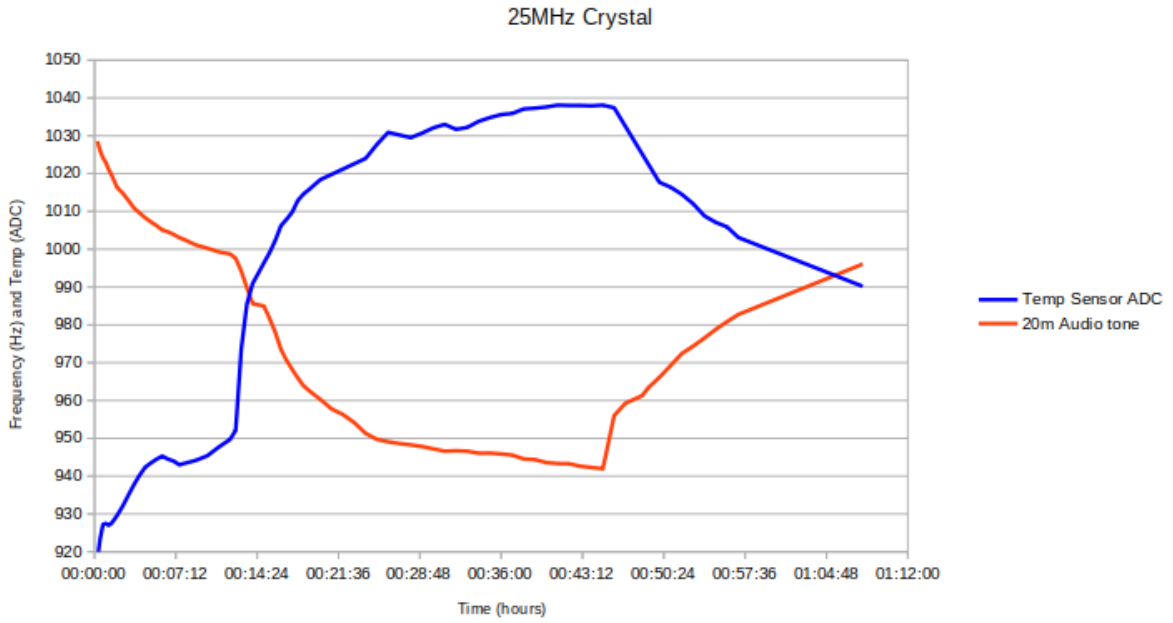


The Si5351A datasheet dictates the use of a 25 or 27MHz crystal or oscillator. In the QDX a 25MHz crystal was originally used, together with an unusual NPN/PNP transistor oscillator circuit, which produces a near rail-to-rail (3.3V peak-peak) output that is perfect for driving both the Si5351A and the STM32 processor.

However during testing it was found to provide an unsatisfactory level of frequency stability, which was

very disappointing and boards had already been ordered. In the graph below, an example operating session is shown. QDX was operated on the 20m band and the operating frequency was monitored as an audio tone on a separate receiver. 20m was used as it is the highest frequency of operation of QDX and would therefore illustrate the worst-case scenario. The nominal operating frequency was 1kHz audio and was not altered during the test.

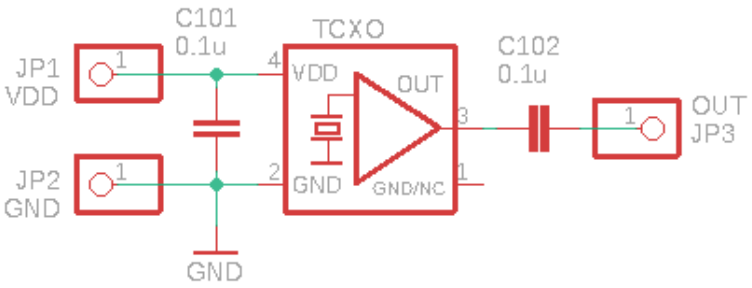
In the graph, the red line is the audio frequency. The blue line is the output of the STM32's internal temperature sensor ADC input. It is uncalibrated (to temperature), and just indicates relative temperature changes.



The QDX was switched on at 00:00 on the left side of the graph. An initial warm-up drift of 30Hz downward drift can be seen (red line). At around 12 minutes, FT8 operations were commenced, having the usual 15 second slots with alternating transmit and receive cycles, resulting in a duty cycle a little under 50% (FT8 transmissions are a bit less than the 15 second cycle). FT8 operations ceased at around 47 minutes, by which time the enclosure had warmed up considerably and the frequency had dropped almost 90 Hz from power-on! As the unit cooled down, the frequency drifted back upwards again. The temperature curve (blue) gives a relative indication of the temperature of the PCB.

FT8 has a bandwidth of about 50Hz. You can see that a 90Hz drift in half an hour of operation is rather unacceptable! Anybody who has tried FT8 operation will know that the band is usually full of QSOs and you struggle to find a clear spot to transmit in, often slotted in with other stations closely on either side. With such a bad drift rate, you would drift right across the other stations' transmissions – or to avoid that, have to keep making little adjustments to the operating audio frequency in WSJT-X to counteract the drift! Horrible!

The problem was resolved by designing a neat little PCB containing a 25MHz TCXO that is fit to the QDX during assembly, INSTEAD of the 25MHz crystal. The tiny PCB has three pads which line up precisely with the crystal terminal that is connected to the base of Q4, and to two pins at the board edge which are used to supply Ground and +3.3V.



The output amplitude of the TCXO is 1.1V peak-peak which is enough for the Si5351A, but insufficient to drive the STM32 oscillator input. It is therefore buffered using the original NPN/PNP transistor oscillator circuit, which results in a rail-to-rail (3.3V peak-peak) near-square wave.

After fitting the TCXO in place of the original 25 MHz crystal, the drift is reduced to +/- 1Hz or so which is unnoticeable and entirely acceptable.

The Si5351A has a large number of internal 8-bit registers to control the synthesizer configuration and output frequency, and these are programmed by the microcontroller using the I2C serial protocol. 1K resistors R20 and R21 are pull-ups required for the operation of the bus at 400kHz.

There are three outputs of the Si5351A synthesizer and these are all used to good advantage. The Clk0/1 outputs are used to drive the Quadrature Sampling Detector (QSD) during receive.

A feature of the Quadrature Sampling Detector is that either the RF input, or the LO input, must provide two paths in 90-degree quadrature. This is normally applied at the Local Oscillator where it can be easily controlled for best performance. So, two oscillator signals are required, with the same frequencies but a precise 90-degree phase offset. Generating this quadrature Local Oscillator signal is always difficult. Analogue phase shift circuits have limited accuracy. Often a divide-by-4 circuit is used, to produce quadrature oscillator outputs from an oscillator input at 4x the reception frequency. This also creates challenges particularly as you try to increase the reception frequency to cover higher bands. For example, on 10m e.g. 30MHz, a local oscillator at 120MHz is required and the divide-by-4 circuit must be able to operate at such a high frequency. Devices such as the 74AC74 can do so, but pushing it higher into the 6m band cannot be done with the 74AC74.

The Si5351A has a phase offset feature, which is not really very clearly described in the SiLabs documentation. However, QRP Labs has perfected the technique to put two of the Si5351A outputs into precise 90-degree quadrature, which is maintained without tuning glitches as the frequency is altered. It’s a nice development because it eliminates one more circuit block (the 74AC74 divide-by-4 circuit), again reducing complexity and cost. To the best of my knowledge this is the first time the Si5351A has been implemented in a product directly driving a QSD with two outputs in quadrature (no divide-by-4 circuit).

The Clk0/1 Si5351A outputs are also routed to the transmit driver. During transmit, Clk0 and Clk1 are configured with a 180-degree phase difference, this allows them to cleanly drive the QDX RF power amplifier, which is a Push-Pull configuration. It’s a very neat way to obtain the necessary push-pull driver signals in a highly symmetric and low parts-count way.

Finally the Clk2 output is used as a separate signal generator, it is routed back to the QDX transceiver’s RF port via the 100K resistor R19. This signal generator can be used by the QDX diagnostic and self-performance measurement tools, to sweep the audio filter and sweep the RF bandpass input filter. Clk2 is normally switched off, and when these functions are needed, it is enabled in conjunction with Clk0/Clk1 operating the receiver mixer.

An additional function of the 25MHz TCXO and its NPN/PNP transistor buffer amplifier, is to provide the precise 25MHz system clock for the STM32. The STM32 microcontroller contains its own internal PLL clock generation systems that are highly configurable. In QDX, the 25MHz reference oscillator is fed to the STM32 external clock input, and the STM32 is configured to provide:

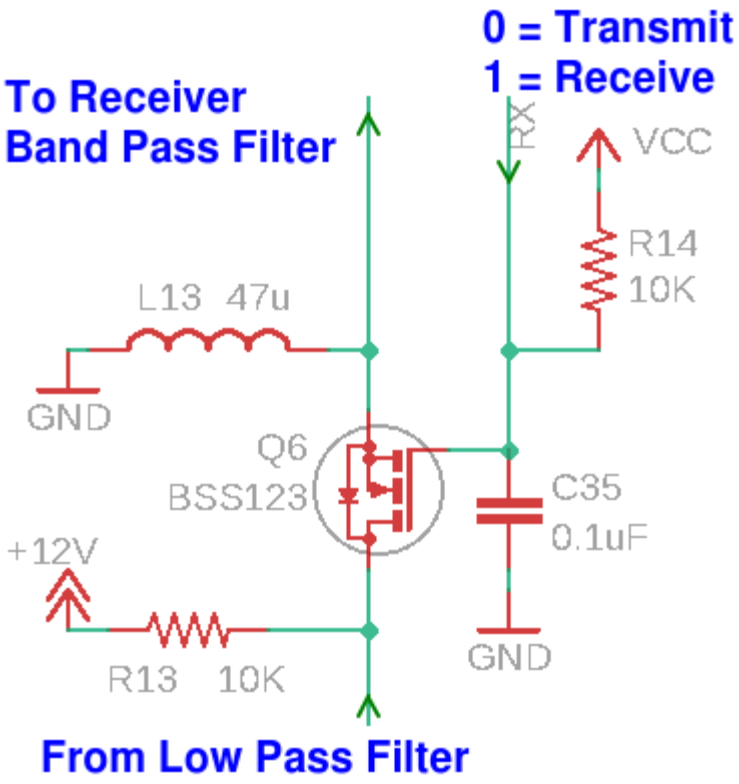
- 24.576 MHz clock for the 512 fs input to the 24-bit stereo I2S ADC chip (24.576 / 512 = 48,000) – fs is the number of clock pulses for each analog to digital conversion.

- 48 MHz clock for the STM32's USB peripheral to enable it to provide USB functionality for the USB sound card and the CAT Virtual COM serial port.
- 72 MHz for the main CPU clock

None of these signals are absolutely precise but they are very close to the target values and well within specified allowable errors for the protocols involved (USB, I2S). The 72 MHz CPU clock is a little below the maximum allowable 84 MHz for this processor, but this is acceptable since the performance at 72 MHz is plenty adequate to achieve all desired functionality for QDX.

4.4 Solid state transmit/receive switch

During transmit, the receiver must be disconnected from the RF port of QDX because a 5W signal is 45V peak-peak which would be too much for the receiver input mixer IC3.



A spectacularly high "Off" isolation is not required, only sufficient attenuation to protect the mixer. During transmit, the Digital Signal Processing of the I & Q channels is suspended and there is no audio output of the receiver. In the QDX application a totally clean click-free transition between Receiver and Transmit states is not needed – a welcome relaxation compared to the demands of a CW transceiver.

The transmit/receive switch is implemented by a single BSS123 MOSFET. The source is at DC ground (via 47 uH inductor L4). The control signal from the microcontroller switches the MOSFET on or off. Capacitor C35 close to the MOSFET gate is found to be necessary to prevent inductive pickup of the 5W RF from partially switching on the MOSFET.

An interesting additional detail here is that the microcontroller output can only drive the MOSFET gate to around 3.3V as this is the supply rail of the microcontroller. When "On" the processor output pin is put into a high impedance state and the 5V via R14 switches on the MOSFET. When "Off", the processor output pin is in a low impedance state and pulls the gate to near 0V. Given the much lower VGS (Gate Threshold Voltage) of the BSS123 compared to the BS170 used in through-hole designs this 5V "on" potential may not be needed in this application.

During receive, the transmitter drivers are all at zero potential, so the transmitter power amplifier transistors are in a high impedance state and do not affect the receive operation.

This simple transmit/receive switch very conveniently avoids the need for relays. Relays may be great for some applications but if a relay was used here for transmit/receive switching it would add bulk, weight, cost, and reduce reliability.

4.5 Switched receiver band pass filter

In any receiver, performance is improved by excluding strong out of band signals using a band pass filter. In QDX a rather simple series resonant L-C filter is used, which although limited, still provides some useful protection to the mixer. The mixer is a Quadrature Sampling Detector which is a high dynamic range, high IP3 mixer and therefore has excellent intermodulation performance. Together, the BPF and QSD provide a high performance receiver front end.

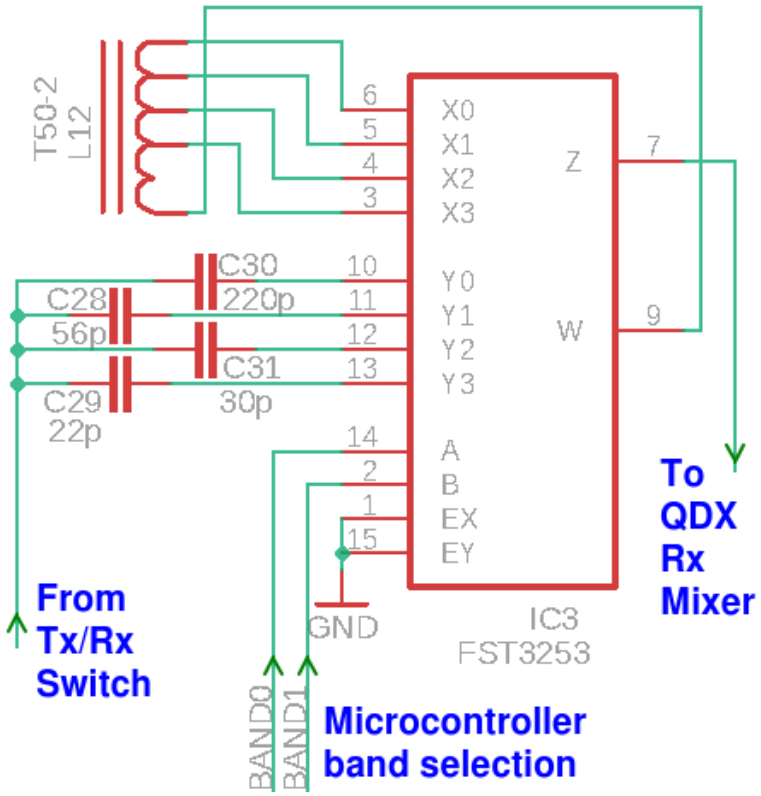
The band selection switch is a FST3253 dual 1:4 analog multiplexer IC, containing in other words, two 4-way switches.

The switch connects one of Y0 to Y3, to the W pin; and at the same time one of X0 – X3 to the Z pin. The selection of which of the four possibilities is connected, is made by two control lines named A and B (pins 14 and 2 respectively) which are connected to output pins of the STM32 microcontroller.

The Y-W switch connects one of capacitors C29, C31, C28 and C30 (for 20, 30, 40 and 80m respectively) to pin 9 (W).

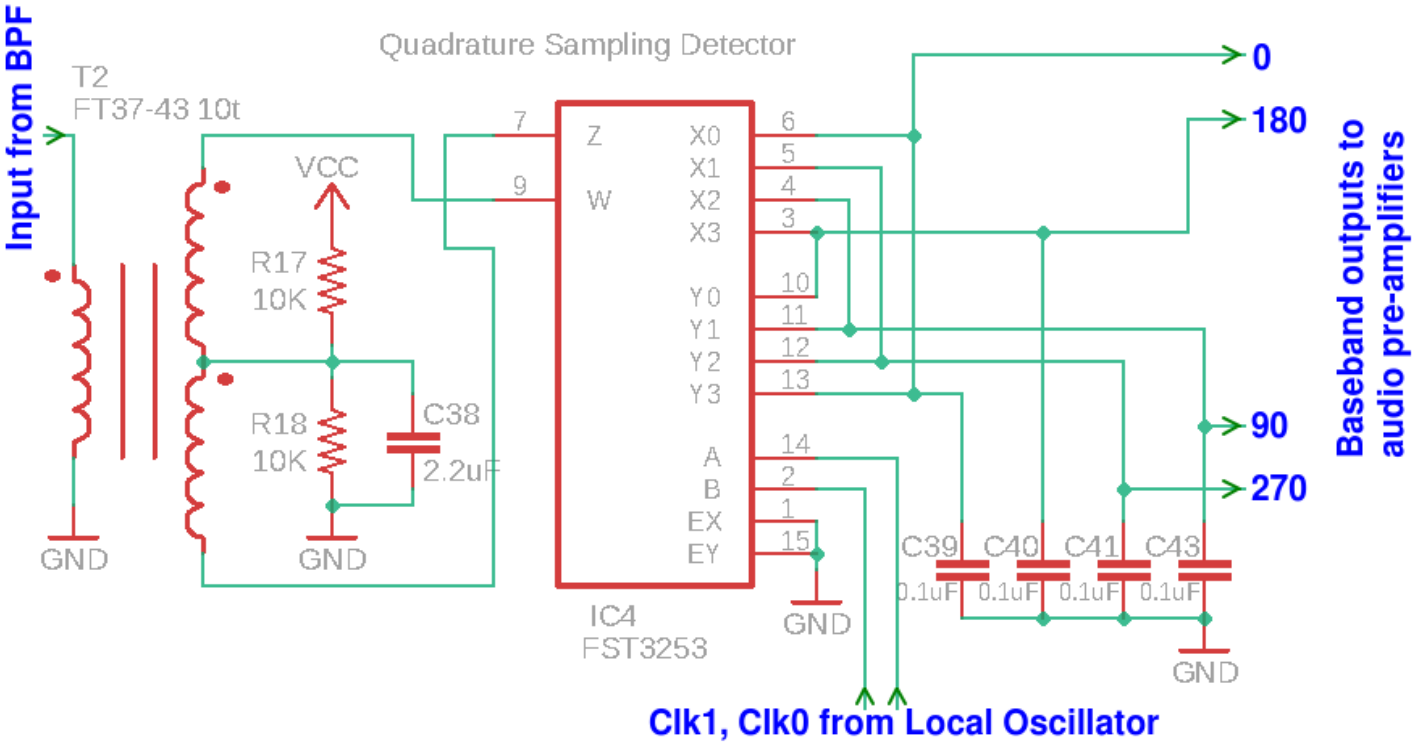
The second switch X-Z then selects one of four taps of inductor L12.

Inductor L and capacitor C values are chosen such that the LC resonance is centered on the digi modes section of each band.



More sophisticated Band Pass Filters could be used but would come at the expense of complexity, board area, and cost.

4.6 Double-balanced Quadrature Sampling Detector



This circuit implements a double-balanced Quadrature Sampling Detector, mixing from RF to baseband. The FST3253 is a dual 1:4 multiplexer which is often seen in QSD circuits. It has fast switching times and very low on resistance of only a few ohms. The input signal is switched by the quadrature LO to each of the four integrating capacitors C39, C40, C41 and C43 in turn, for 90-degrees of the RF cycle each. The result is that the audio difference (beat) between the RF input and LO input appears across each of the four integrating capacitors, with four phases at 0, 90, 180 and 270 degrees.

The 0.1uF capacitors and the low source resistance results also provide a relatively fast roll-off of the audio response. This is effectively a narrow band pass filter since any incoming RF more than a few 10's of kHz away from the LO frequency is greatly attenuated. The QSD is inherently a very high performance mixer design with high third order intercept and dynamic range, and low loss (0.9dB).

The FST3253 dual switch is often connected with the two switches simply paralleled together (which does halve the switch ON resistance). But I prefer the double-balanced mixer configuration which provides higher performance. The double-balanced configuration requires two RF inputs 180-degrees out of phase (opposite to each other). This is provided here by trifilar-wound transformer T2.

R17, R18 and C38 provide a 2.5V DC bias to the RF signal through the mixer and audio pre-amplifiers. This simple bias does not source or sink any significant current due to the balanced nature of the system, therefore no buffering is required.

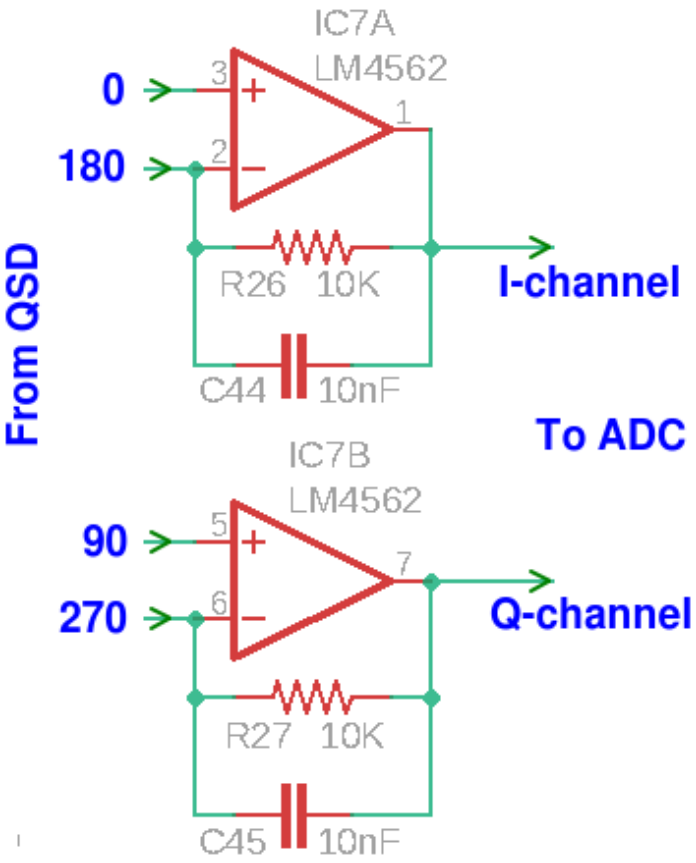
4.7 Low noise pre-amplifiers

The same LM4562 dual operational amplifier chip is used in QDX as is used in the QCX-series CW transceivers and QRP Labs receiver module. This is a low noise op-amp providing an excellent combination of high performance and low cost.

Low noise is important for the pre-amplifier because it determines the overall sensitivity of the receiver (in conjunction with losses of the RF front end).

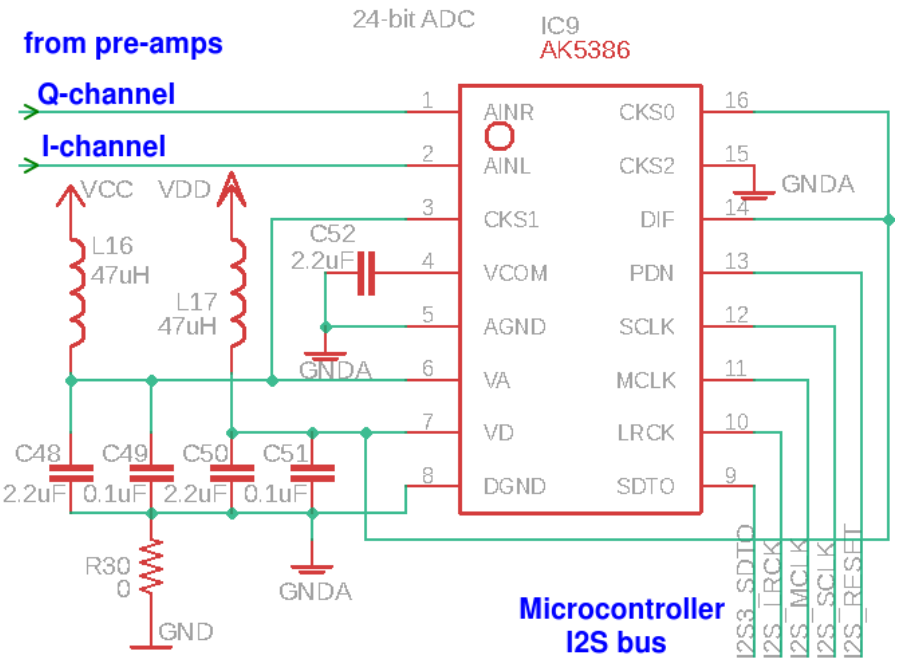
The amplifiers are used in a difference configuration to provide common mode rejection. The 0 and 180 degree signals are summed to produce the I-channel, with 90 and 270 degree QSD outputs producing the Q-channel.

The I and Q channel contain all necessary phase and amplitude information to allow the following SDR to demodulate any mode, although only single sideband is used in QDX.



4.8 Analog to Digital Converter

The AK5386 Analog to Digital Converter (ADC) chip is a high performance stereo ADC chip. Its purpose is to convert the analog I and Q-channel signals to a digital representation to allow digital signal processing by the microcontroller, which implements a Software Defined Radio (SDR). The AK5386 specifies a 110dB dynamic range and is used at 48 ksps (kilo samples per second).



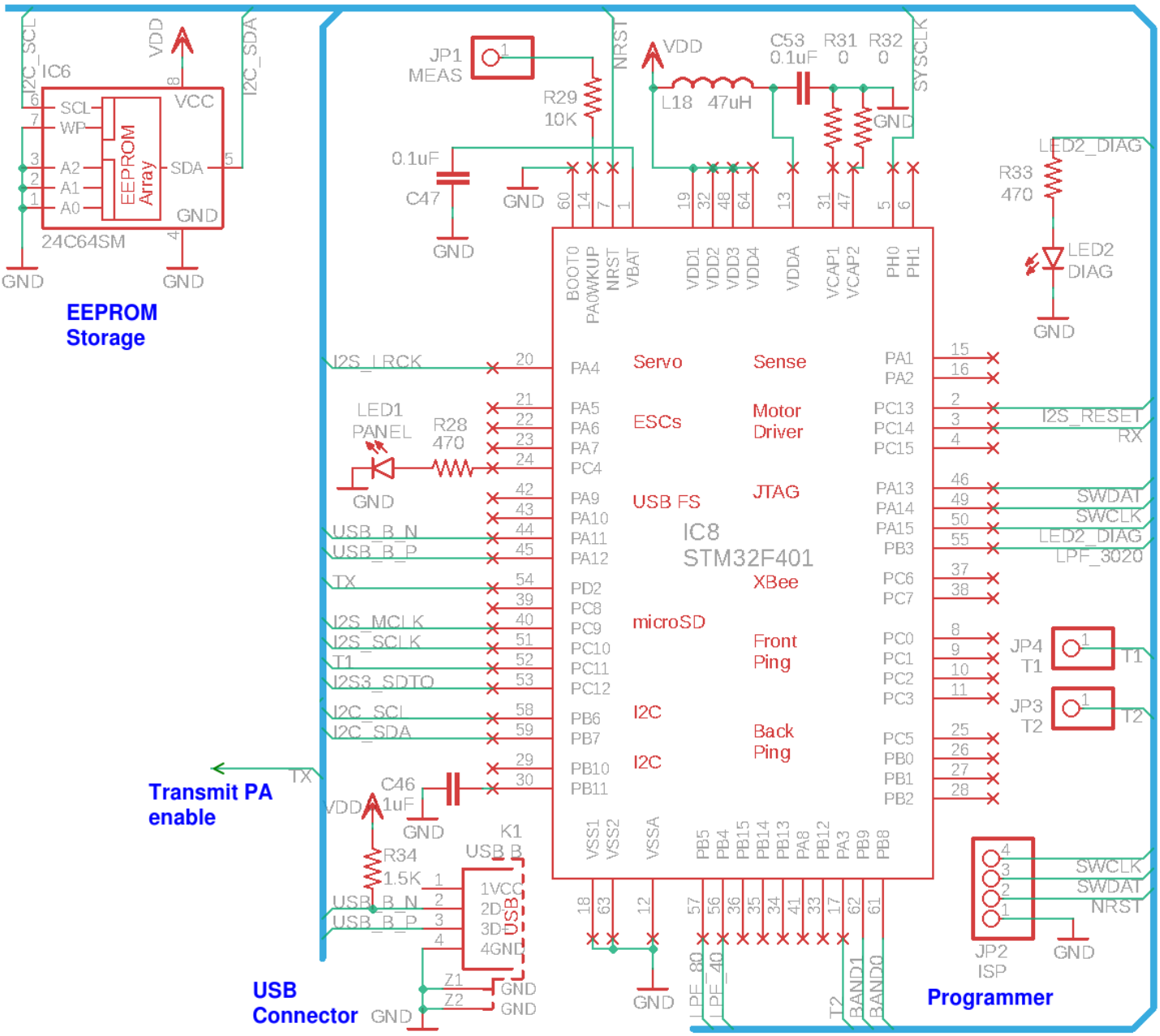
Particular care is required in the PCB layout to achieve the specified dynamic range, with separate ground planes for the analog and digital sections. The zero-ohm resistor R30 is the single point interface between the two grounds.

In this design, the AK5386 chip acts as Master of the I2S interface and the STM32 I2S peripheral acts as Slave. The STM32 generates a 24.576 MHz clock signal (I2S_MCLK) which is 512 x the sample rate (i.e. 48,000 x 512 = 24,576,000). From this, the AK5386 generates the bit clock (I2S_SCLK) at 3.072 MHz (64 bits per sample – two 32-bit fields, for Left and Right channels, of

which 24-bits are populated). The processor also generates the Frame clock (LRCK) at 48 kHz. The I2S_RESET signal is supplied by the microcontroller and must be activated once at power-up to reset the ADC.

4.9 Embedded Software Defined Radio

This schematic fragment contains the whole microcontroller section around the STM32F401RB microcontroller.



The microcontroller section of the schematic includes a 24C64 (8K x 8-bit) serial I2C EEPROM chip. Unlike the ATmega328 processors used in many other QRP Labs kits, the STM32 used here does not contain internal EEPROM so for storage of configuration parameters it is necessary to either use Flash storage or EEPROM. The problem with Flash storage is a limited 10,000 write-cycle specification which may be a little low. Therefore an external EEPROM chip is used.

The 4-pin programming header at the bottom right of the diagram is for factory use only to install the QFU bootloader on the QDX board. An onboard diagnostic LED is also used only for factory validation. LED1 is a front panel LED on the QDX which shows basic status information such as whether or not the unit is transmitting or receiving, or in bootloader (firmware update) mode. A transmit-enable output switches on the transmit driver to enable the RF power amplifier.

Other than that, all the magic goes on in the firmware, which is not open source.

The Software Defined Radio receiver (SDR) implements digitally a superhet receiver having a 12kHz Intermediate Frequency. The reason for doing this rather than processing directly at baseband as a direct conversion receiver, is that it eliminates any issues with power line hum – the harmonics of 50Hz or 60Hz (depending on your country) and other noise which exists around 0Hz.

The receive signal path processing contains the following stages implemented in the Digital Signal Processing (DSP):

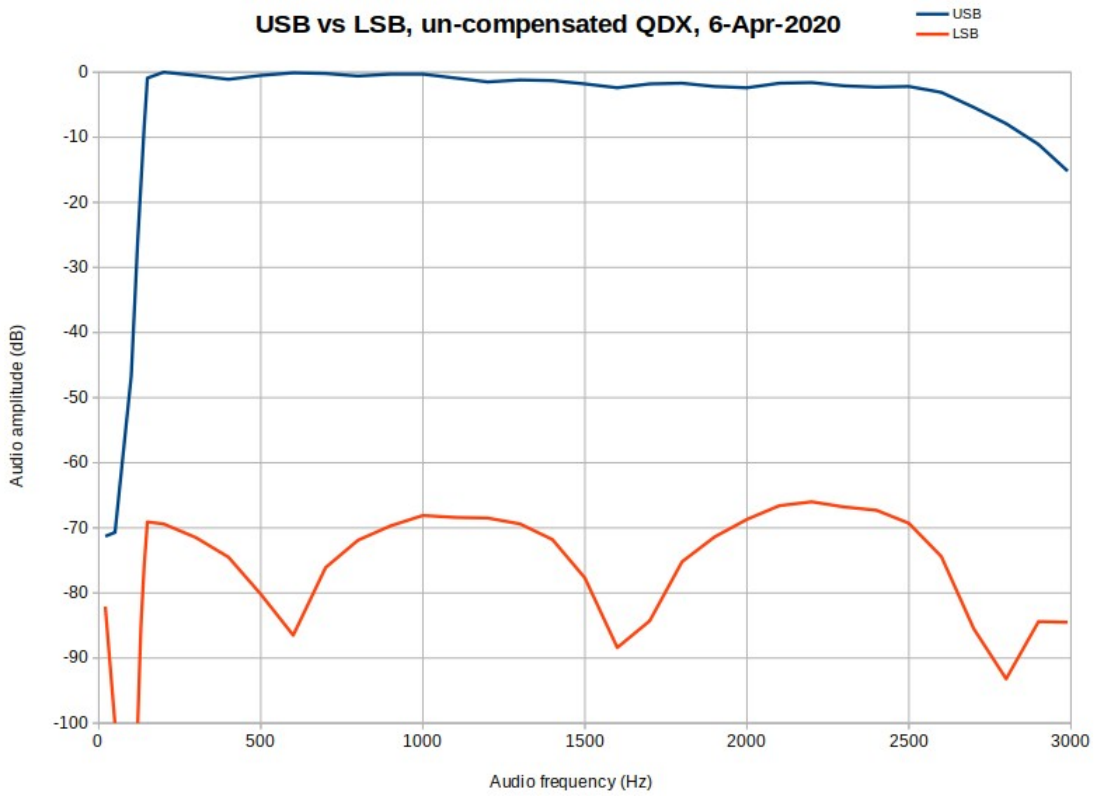
- Retrieve a block of 32 24-bit audio I and Q samples from the ADC chip via the I2S bus
- “Mix” to 12 kHz Intermediate frequency, including a Low Pass Filter in the factor-of-four decimation process; the reason for the choice of 12 kHz as the intermediate frequency is that the mathematics becomes relatively easy, as it is exactly one quarter of the 48 kHz sample rate.
- A Hilbert Transform applies a 90-degree phase shift to one of the I and Q paths relative to the other; this is the equivalent of the analog all-pass phase shift network implemented using op-amps in the QCX CW transceiver.
- Sum or subtract the two paths to produce either the Upper or Lower sideband; QDX is capable of demodulating either sideband but digital operations normally use upper side band (USB) which is therefore the default operating mode.
- Apply a digital audio filter, which has a passband from 150Hz to 3.2kHz
- Interpolate back to 48 ksps audio
- Send the 48ksps audio samples to the PC over the USB Digital Audio link.

The performance of the receiver was tested using the internal signal generator to inject a signal into the QDX input at a defined offset; Argo software was used on the PC to determine the received signal amplitude.

Two different measurement runs were performed, with quite some elapsed time between them (17 months!). In both cases, there is no change to the firmware but the measurements were performed on different prototypes. There is no attempt at amplitude or phase compensation, which does not appear to be needed (the results are already excellent).

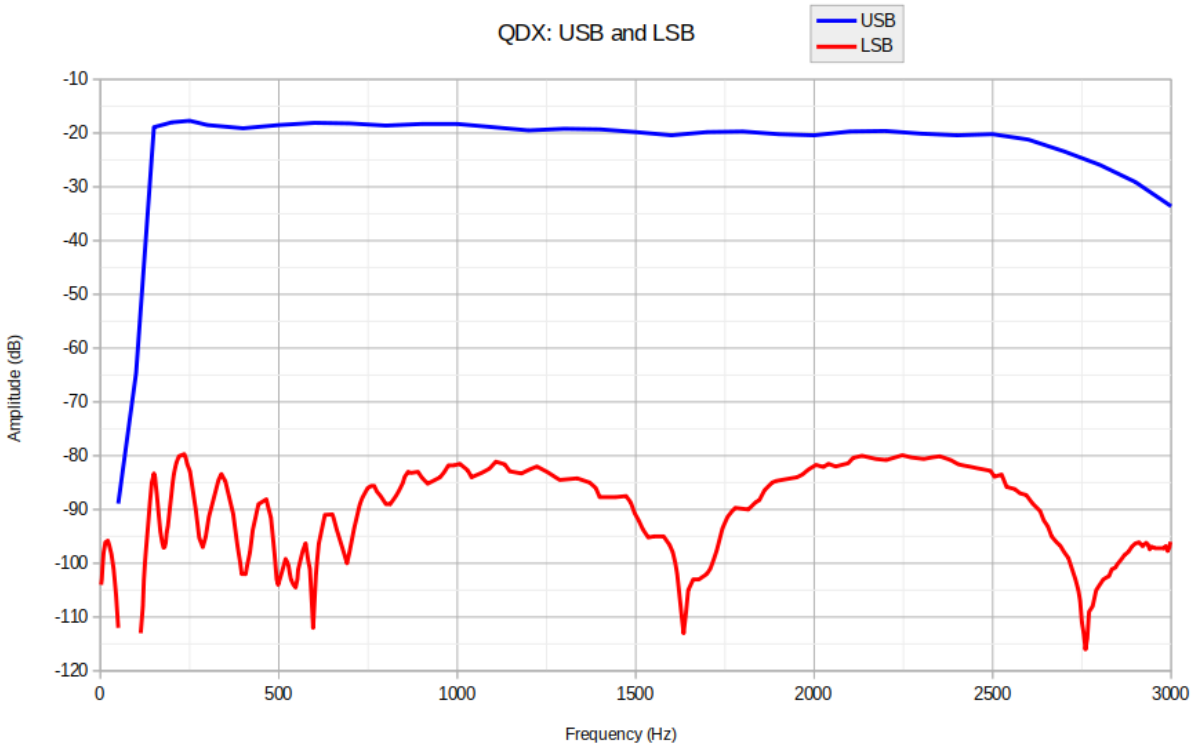
The vertical scale in dB has an arbitrary zero reference; the important point is the difference between the upper sideband (blue line), and the unwanted lower sideband (red line).

In the first example (shown below) dating from 06-Apr-2020, the worst case unwanted sideband suppression is at -64dB relative to the wanted sideband, which occurs at about 2,200Hz audio.



The second measurement (September 2021) shown below, shows a worst case unwanted sideband -60dB relative to the wanted sideband, again at about 2200Hz audio. The difference of 4dB between the two prototypes will have been due to component tolerances and build-style variations. In both cases, this is really good performance!

The second measurement was painstakingly made with much finer resolution (lower frequency steps) than the earlier measurement, which shows more clearly the behaviour of the Digital Signal Processing.



4.10 Embedded 24-bit 48ksps stereo USB Sound card

QDX has its own embedded 24-bit 48ksps stereo USB sound card! But there is not a real physical little piece of hardware strapped inside, like the PCB from a sound card dongle or something. Instead, the USB sound card is actually coded in firmware directly by the STM32 microcontroller, the same single microcontroller that is also doing everything else, in the design.

This is an extremely important design feature of QDX.

A radio transceiver having an audio input has a number of disadvantages:

- An audio cable is one more cable to have to plug in between your radio and your PC.
- There may be level mis-matches between the PC generating the audio signal, and the audio input of the radio. The PC will need to be carefully set up to set the volume such that the radio produces the expected RF proper power level, but does so without distortion (clipping, intermodulation distortion, splatter, however you want to refer to it). With the built-in QDX sound card, no level adjustment is necessary; set the volume to the max and rest assured that the QDX will never be overdriven (in fact it CANNOT be).
- Audio cables are a recipe for picking up noise, particularly in most of our lab environments where we power our equipment from grid power. You will often see 60 Hz (US) or 50 Hz (most of the world ex-US) harmonic lines on spectrum waterfalls. Ground loops can be hard to eliminate. With QDX and its built-in sound card, numeric audio sample data is transferred back and forth between the PC and QDX with zero loss, zero distortion, no hum or other noise. The quantization error due to sampling at 48ksps is minimal but is in any case matched by WSJT-X so a higher sample rate would not achieve an increase in performance. The benefit of direct digital numeric transfer between PC and QDX, without any conversion errors or noise, is enormous!

You can buy cheap USB dongles and connect them between your radio and your PC. But the dynamic range IS limited, and not just by the 6dB/bit theoretical relationship between bit depth and dynamic range, but also by the actual noise floor of the converter (the least significant bits are in other words, just noise).

In contrast, QDX uses a 24-bit stereo ADC chip with 110dB dynamic range specification. If you look at the specification of any expensive USB external sound card, and the price of getting anything like 110dB, you will appreciate the true significance of this.

All Digital Signal Processing in QDX is carried out at full floating point resolution, not limited to 16-bits as many SDR implementations do. In its communication with the PC this is rounded to 24-bits and transferred via the USB Digital Audio device class. WSJT-X only uses 16-bits so the PC sound system will automatically re-sample but still, the 24-bit USB sound is sent by QDX to support any future PC applications which use a higher resolution.

The result is a very high performance radio receiver, and equally, an excellent performance transmit side due to the zero-loss, zero-noise transfer of audio digitally from the WSJT-X synthesis straight to the QDX frequency analysis.

4.11 CAT control serial interface

QDX includes an emulated virtual COM (serial) port, all in the same STM32 processor that does everything else. The STM32 processor therefore actually includes THREE USB device classes:

1. Compound device class (a container for more than one other device classes)
2. Digital audio
3. Virtual COM (serial) port

The introduction of the compound device class makes QDX effectively, contain a virtual USB hub, to which is connected a Digital Audio USB device (the USB sound card) and a Virtual COM Serial Port. This allows a single USB cable to be used between QDX and the PC, the USB cable transfers both digital audio and serial data seamlessly between the two.

An important benefit of the CAT control interface is that it allows QDX to have a flexible tunable VFO and band switching, all without any actual physical controls. No physical controls are at all necessary because everything is done under the control of the WSJT-X software as CAT controller. Elimination of display and physical controls is not just a matter of lowering costs, it also makes the QDX extremely easy to use. Want to change bands? Simply click the drop-down in WSJT-X and QDX will automatically change bands. Want to use a different frequency (for example, change mode to WSPR, or perhaps even change software application to JS8Call)? The software application will automatically switch you to the correct frequency for the mode you select.

Another advantage of CAT control is that WSJT-X contains an option to effect the transmit/receive switch either by VOX (Voice activation, or in other words, the presence of audio) or by an actual CAT command to enable transmission. QDX also supports both methods, there is an entry in the configuration screen that allows VOX to be enabled or disabled. The default setting is “disabled”, which means a CAT command is necessary to switch from Receive to Transmit (and back).

The danger of VOX is that it is all too easy to accidentally have the QDX selected as the sound card for system noise outputs etc… then when a WhatsApp message from someone arrives, and the PC does its fancy “pling” sound, well that’ll be faithfully converted to a series of audio tones by QDX and transmitted accordingly. So disabling VOX and insisting on a proper CAT command to switch on to transmit, is the recommended modus operandi.

QDX also has a rich user interface via the Virtual COM port, by connecting a terminal emulator on a PC, to the virtual COM port. This is a whole blissful and enormous topic by itself, it allows interaction with all the built-in analysis tools in QDX as well as the configuration screen and features such as factory reset and firmware update.

4.12 Audio frequency analysis

Determination of the audio frequency arriving from the PC software (for example, WSJT-X) is key to the whole operation.

My first thought was a Fast Fourier Transform (FFT). However that quickly turns out to be a lot more difficult than it at first sounds. For the FT8 mode for example, the audio range is a 3kHz band. It isn’t so trivial to set up a FFT with 6.25Hz buckets, that wide, and would require a significant amount of processing power, such that I would need to use a more powerful (and expensive!) microcontroller than I’d really like. Furthermore WSJT-X modes are heavily optimized such that the period is the inverse of the tone spacing, so I might need to do TWO interleaved FFT analysis runs offset by half the tone spacing. Then there’s the problem of the frequency slide when

a tone changes. Furthermore, the QDX would have to know which mode was being transmitted, in order to configure the size (and hence speed) of the FFT buckets. I started to shudder and shake!

I soon realized that FFT would be the sledge hammer approach and there is a much simpler and faster way, which also offers higher accuracy.

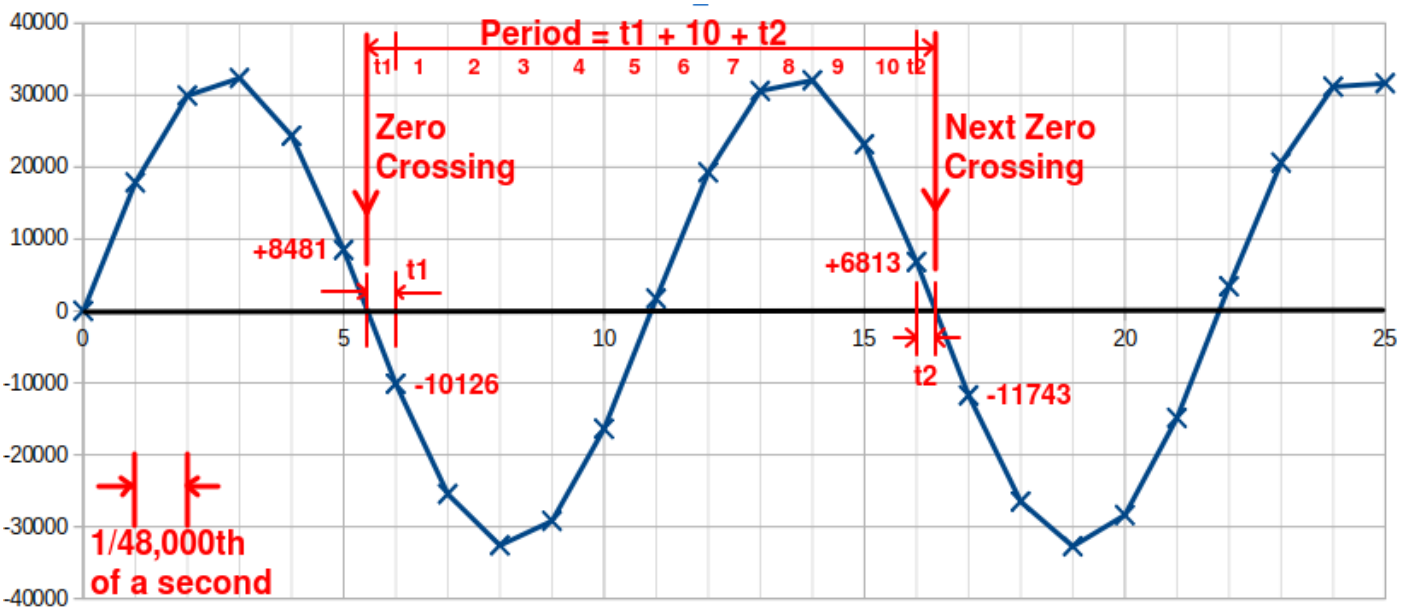
The simple approach is to detect the zero crossings of the sinewave, and carefully time the period between zero crossings. Frequency is the reciprocal of Period, so once you have the period, a moment later you know the frequency too, and the problem is solved. It turned out to be very easy to implement this method, and the accuracy is rather impressive. Experimentally, a SINGLE CYCLE measurement of the frequency at 1500Hz audio frequency leads to frequency accuracy of around +/- 0.05 Hz. More than enough!

NOTE: 1500Hz is an unfair frequency to measure at, because each cycle fits into an exact number of audio samples; 32 audio samples at 48,000 cycles per second make up one 1500Hz audio cycle. This sets us up for an extremely low error cycle measurement. So to make things fair, I actually offset the frequency for the test, to around 1520 Hz, so that the cycle doesn't contain an exact number of samples.

In actual fact we don't really need to update the frequency of the Si5351A Synth 1500 times per second… that's excessive. QDX has configuration options that allow you to specify an averaging period, where a specified number of sound samples AND a specified number of audio cycles must both be exceeded, in order to terminate the audio cycle measurement. The default parameters are set for a minimum of one cycle, but a minimum of 480 samples. 480 samples is 0.01 seconds and since we never go as low as 100Hz, the dominant criteria is therefore a measurement of at least 0.01 seconds.

With this default configuration, we get 100 audio frequency measurements per second, and experimentally the frequency accuracy is better than +/- 0.002 Hz. Amazing.

As for the details of the analysis, study the following hypothetical sinewave:



Remember that we have a built-in 24-bit 48ksps stereo USB sound card. Accordingly we get 48,000 samples per second. In this example I am assuming WSJT-X is used; WSJT-X outputs 16-bit samples, so the extent of the sinewave data is from -32,768 to + 32,767. The sound samples are received from the PC via the USB cable and are absolutely FREE of any noise. It's a lossless and zero noise transfer between the WSJT-X software on the PC, and the QDX's firmware. The

sequence of sample numbers received are EXACTLY what the WSJT-X software generates in its audio software-DDS.

We also know that the sound samples arrive at exact equal intervals of one 48-thousandth of a second. This is effectively the time base of our period measurement. It is not necessary to run any kind of actual timer inside the microcontroller, because in this instance we already have the luxury of knowing we get exactly 48,000 samples per second from the PC.

In the graph, the sine wave is shown starting at sample 0, with amplitude zero. In general, except in special cases (such as 1500Hz as mentioned above), we won't have a subsequent zero crossing indicated by a sample with value zero. We will have a sample ABOVE zero, followed by a subsequent sample BELOW zero. It is easy to detect this situation in the firmware code, just by looking for two successive samples, the first one being positive and the second one negative.

In QDX the firmware detects positive-to-negative zero crossings but it could equally well be done on negative-to-positive zero crossings, it is of no importance. It would even be possible to measure the frequency of half-cycles, with half the accuracy. But this kind of extremism doesn't seem warranted, unless in future some kind of mode with very fast measurements is needed or perhaps, we need to quickly measure the frequency of very low frequency audio signals.

Once we have a positive value followed by a negative value, and we know that they are 1/48000th of a second apart, we now want to find the exact time of the zero-crossing itself. This is also quite easy with a straight-line (linear) approximation. For small angles (in radians) the well known approximation holds: x = sin(x).

The period of the measurement is then given by the fractional times calculated by interpolation, at the start and end of the sample, plus the number of complete sound samples that occur BETWEEN the zero crossings. This is a "number of samples", and we know that there are 48,000 samples per second, so now it is easy to calculate the time in actual seconds which is the Period of the audio cycle; then the frequency is 1/Period, and we're done.

An example

To give a worked example, consider again the above simulated sinewave samples (generated in a spreadsheet). For ease of viewing, there are not many samples per cycle, in other words, this is a much higher frequency than we will be dealing with in practice; but it serves to make the principles relatively easy to understand.

We see audio sample #5, which happens to have a value of +8481 (according to the spreadsheet). Now the next sample, audio sample #6, has a value of -10126. We want to know the fractional interval (in samples), that is at the start of this sinewave cycle. I have called this time t1 in the diagram. Time t1 (as counted in samples), is a fraction 10126 / (8481 + 10126). This is a straightforward linear interpolation between the values at sample #5 and sample #6. The result calculates to be 0.544203794 samples.

The next zero crossing occurs between sample #16 (value +6813) and sample #17 (value -11743). Now we want to know the time t2, which is the fraction of a sample that occurs between the time of sample #16 and the zero crossing. I hope that you can see that this fraction is given by, in this case, 6813 / (6813 + 11743) = 0.367158870.

Next we can see that there are 10 complete sample periods in the cycle, between our two fractional pieces t1 and t2. The complete period, measured in audio samples, is therefore t1 + 10 + t2, which is 0.544203794 + 10 + 0.367158870 = 10.911362664.

By now, we have determined that our audio cycle being measured, has a period of 10.911362664 audio samples. We also know that there are 48,000 samples per second. Therefore the Period of the cycle in seconds, is 10.911362664 / 48000 = 0.00022732.

The frequency of the audio tone is therefore 1 / 0.00022732 = 4399.08 Hz.

All without anything too complex, no complicated DSP, Fast Fourier Transform, etc. Love it!

Error considerations

Now like any good scientist, we should consider the possible sources of error in this measurement technique and try to understand if it is good enough for our purposes.

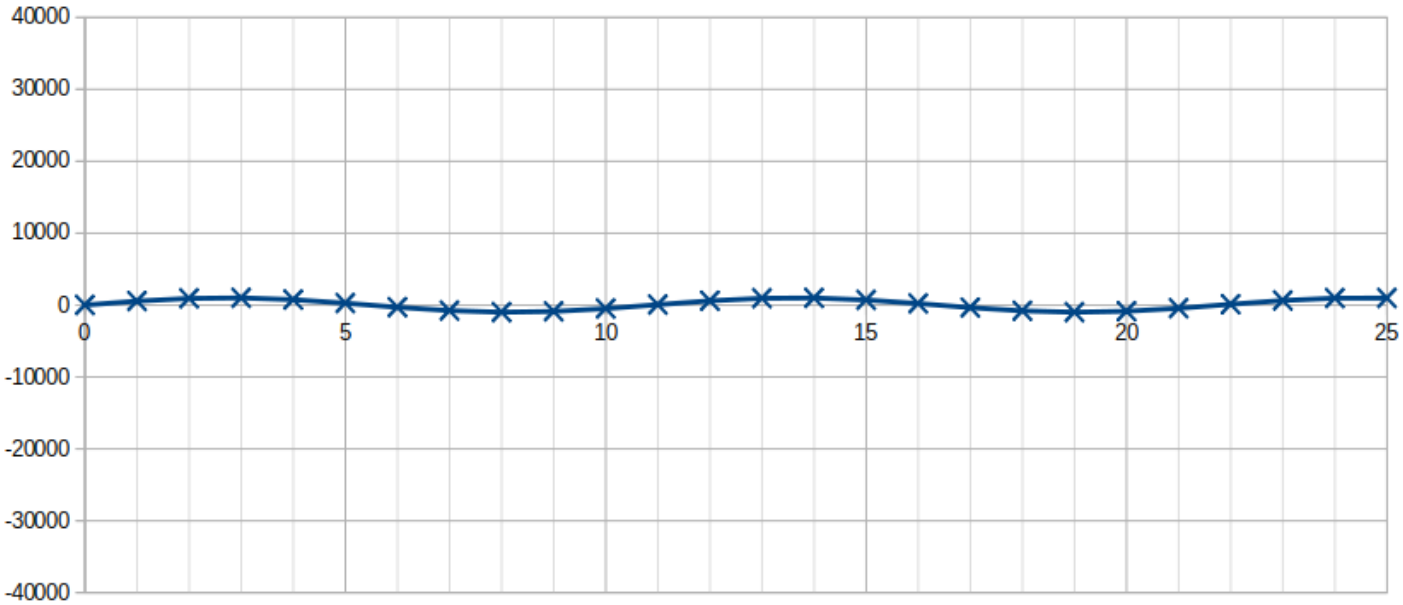
Thinking about it, I could envisage three possible sources of error: audio amplitude error, timing error, and interpolation error (the straight line linear interpolation). Let's consider each of these in turn.

1) Audio amplitude error:

WSJT-X produces samples with 16-bit resolution. Other digimodes software may offer 24-bit resolution but I have not come across any popular modes or software that do this. Neither do I think there are any that provide BELOW 16-bit resolution. Since most users will use WJST-X, and other software anyway has similar facilities, I will use WSJT-X for the purposes of discussion.

A 16-bit resolution provides 65,536 amplitude values per sinewave cycle. That's a theoretical (and practically achieved, since we have the built-in USB soundcard and therefore a lossless, noiseless transfer mechanism) 96dB dynamic range in the information transfer between WSJT-X and QDX, a very accurate transfer indeed.

And yet – there are two scenarios where this can be dramatically reduced.



The first scenario is where the operator uses the output amplitude slider at the bottom right of the WSJT-X screen, to something less than maximum. On a more traditional digital modes station involving a SSB transceiver, this would be done in order to avoid over-driving the SSB

transceiver's audio input, causing splatter. There is NO POINT to doing this on the QDX, since the QDX RF output simply cannot be overdriven. The QDX RF output simply **is**, it is the perfection of a RF signal without any spurious residual carrier or attenuated unwanted sideband, without any audio harmonics due to over-driving and clipping.

But perhaps the operator wants to reduce the transmission output power, and uses the amplitude slider to do this? Again there is NO POINT, because QDX does not transmit at reduced output power. It's all or nothing. There is no in-between.

Therefore the operator should be advised for best accuracy, to leave the volume slider at the maximum value, as there is no point to any lower value choice anyway. I found I was able to move the slider to very low volumes anyway before the frequency measurement accuracy significantly deteriorated. So this is not a particularly serious concern. But the recommendation still remains, just leave the slider at maximum.

The second scenario where lower resolution sinewaves can be generated by WSJT-X, is during the key-down and key-up instants, where WSJT-X applies (I believe), a raised cosine amplitude envelope to the beginning and end of the transmission. For the sake of design simplicity and because it is much less critical in digital FSK modes, than on/off keyed modes such as CW, amplitude envelope shaping has been omitted from the QDX design.

So there is no point to WSJT-X sending reduced amplitude sine waves at the beginning and end of the transmission key-down, since QDX cannot implement that anyway. Of course, WSJT-X does not know this, and there is no way to turn it off, so it will happen anyway. We can't "advise" WSJT-X software in the same way as we could advise the operator to leave the slider at maximum setting, in the previous discussed scenario.

At very low amplitudes, there would be a much more significant quantization error due to the limited amplitude resolution, that has the potential to increase the frequency estimation error substantially at very low amplitudes such as when the raised cosine envelope shape starts. This could cause a frequency estimation with a significant amount of error, and the Si5351A Synthesizer to start producing RF at an incorrect frequency.

To avoid this possibility completely, the QDX has a configuration ("Rise threshold") which allows it to ignore any audio cycles until a certain percentage of full amplitude has been reached. This can be set to a high percentage value such as 80 or 90%. This will ensure that there is no initial frequency value with a high error. Transmission will not occur until the 80% threshold (for example) is reached.

As a side-benefit, this will also FORCE the operator to keep the volume slider at the maximum value, or there will be no RF output! The "advice" will become binding.

The threshold should not be set to a very high value such as 99% because at high frequencies, where the cycle is made up of a smaller number of sample values, there may be no sample value near the actual peak of the sinewave, and so it may take several cycles for perchance, a value to exceed the threshold.

There is a similar "Fall threshold" setting to switch off the transmission at a defined amplitude threshold, before the raised cosine trailing edge of the amplitude envelope generated by WSJT-X takes the amplitude to a very small number and causes a potentially inaccurate frequency measurement.

"Rise threshold" has a default value of 80% and "Fall threshold" has a default value of 60%; it is unlikely the operator would ever want or need to change these from the default values. The default values are not critical but they should be neither very small, nor very large; and furthermore the

falling edge threshold should be somewhat smaller than the rising edge threshold, otherwise key-up will be effected prematurely.

When the operator uses the maximum volume setting, and thresholds are implemented to avoid the possibility of high calculation error at very low amplitudes which could occur during the raised cosine leading and trailing edges, the effects of amplitude error are practically speaking, of no consequence.

2) Timing error:

USB audio devices transfer data to and from their PC host at an agreed rate. In the case of QDX, it appears to the PC as a 48ksps (48,000 samples per second) 24-bit stereo USB sound card. However, in common with all such USB audio systems, there is no facility for synchronization between the PC USB host, and the USB device. In other words, there is no 48kHz clock signal between the two. In the Full Speed USB protocol, the host and device communicate via data packets sent 1000 times every second.

Now it can get all rather complicated… because the host may send data to the device at not exactly 48kHz, and the device may send data to the host at not exactly 48kHz either. There are specifications in the USB standards that define the worst allowable error percentage. Now both ends of the connection have to try to deal with the fact that there is ALWAYS going to be a difference between them. Even if both 48kHz crystals derive their sample rates from a quartz crystal reference, the crystals will not be at exactly the same frequency.

In the case of QDX, there is a 25MHz TCXO that operates as the CPU clock input. All signals are derived from this clock, including the 72.something CPU clock, and the 48MHz USB peripheral clock (well, CLOSE enough to 48MHz), and the I2S peripheral master clock which is at 512 times the sample rate (each sample has 512 clocks); or 48kHz x 512 = 24.576 MHz.

This 24.576 MHz signal is derived internally in the QDX, by a PLL multiplication factor of 117, then division factors of 17 and 7. So 25 * 117 / (7 * 17) = 24.579832 MHz. Dividing by 512 gives an audio sample rate of 48007.5 samples per second! As well as this synthesis accuracy limitation there's also the smaller imprecision in the 25MHz TCXO oscillation value.

Luckily, this small 7.5 samples per second difference is well within the allowable percentage error in the USB sound device specification. On the PC side, the built-in USB driver has to deal with this difference, in addition to the fact that its own master clock will also not be truly at precisely 48kHz; it does so, I believe, not by digitally re-sampling to what it thinks is exactly 48ksps, but by some kind of software phase lock mechanism.

On the incoming data from the PC over the USB interface, we're also not going to be getting data samples at exactly what WE believe is 48ksps (our belief also being slightly inaccurate, itself). So what do we do… certainly I'm not going to dabble in software PLL clock recovery; an option could be do occasionally drop a sample if there are too many, or to duplicate a sample from time to time when there are, in my judgement, too few.

But in the end, after all this waffle, none of it is necessary. It is perfectly acceptable to do NOTHING about the lack of synchronization, just nothing at all! It is fine to pretend that we get exactly 48,000 samples per second, and analyze them all perfectly as though everything was exact. If there is any sample rate error, it will simply produce in the calculation result, a proportionate error in calculated frequency. Even a 1Hz error, if it happened (and frankly, I have never seen even a 0.01Hz error), would be of no consequence to operation of the transceiver, alongside much more significant errors such as the precision of the RF synthesis.

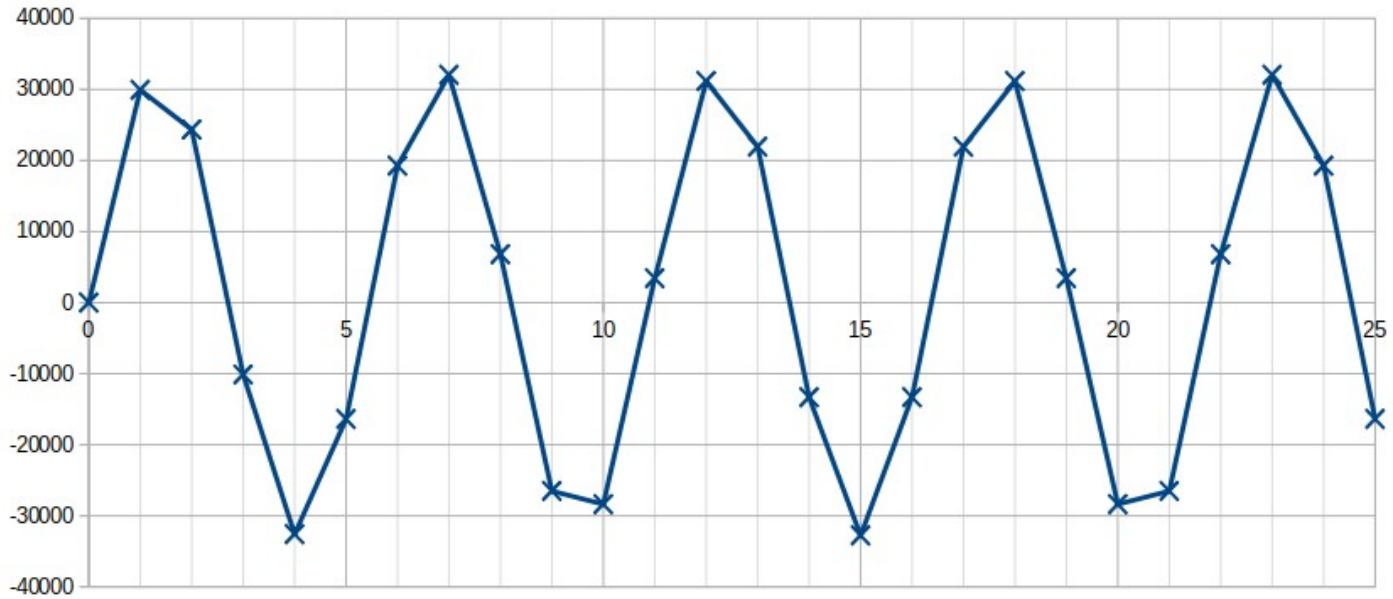
In the end therefore, the conclusion is that timing errors are of no consequence.

3) Interpolation error

Remember the approximation we have used for the zero-crossing interpolation that determines the exact moment of the zero-crossing event? Mathematically, the approximation is that for small values of x, then sin(x) equals x. Put another, less mathematical way, we can draw a straight line between the two points we want to interpolate, which makes the calculation easy.

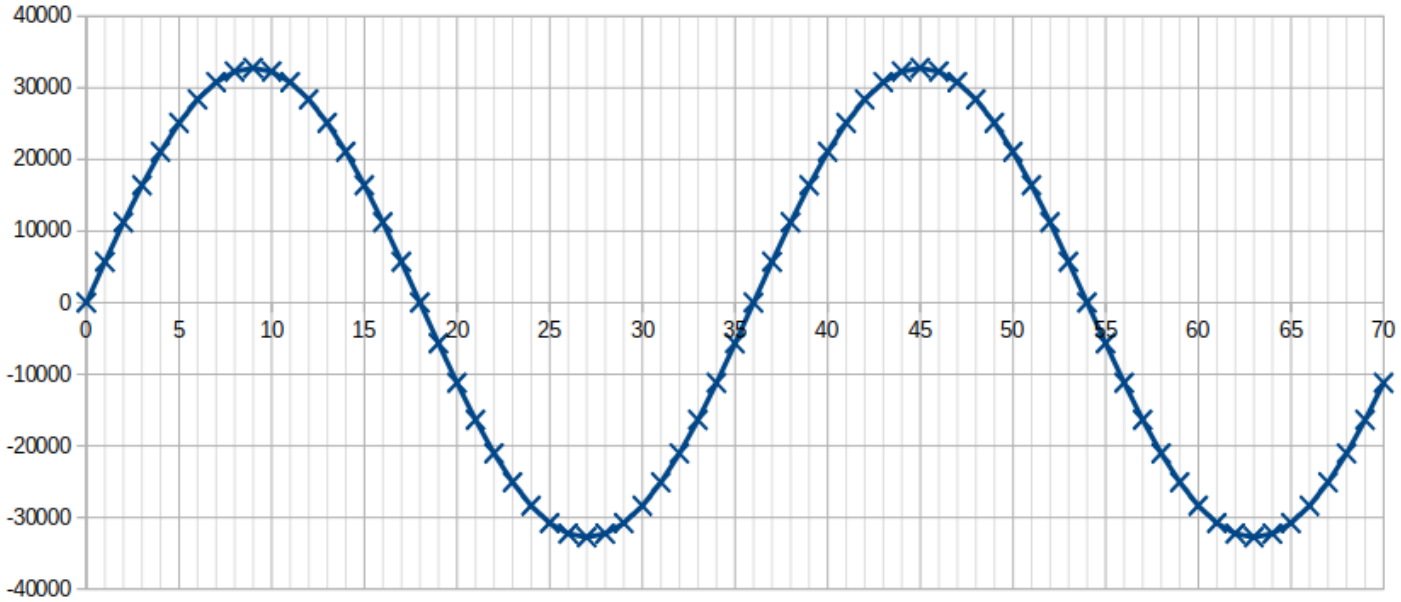
What can cause this to generate error? Well, when we are no longer dealing with “small values of x”. Practically speaking in our case, this occurs when there are a lower number of samples per audio cycle.

Consider an extreme example, for the sake of demonstration – here is a an 8.4kHz sinewave (much higher than we will ever be dealing with):



You can see that the sample points, indicated by the X’s, are quite far apart. The zero-crossing time, or the time between the two samples adjacent to the zero-crossing, is a relatively large fraction of the cycle period. Our “x” is therefore not so small as it was before. The straight lines that previously didn’t look at all bad, now look a lot less like the sinewaves that are being approximated.

Now consider this sinewave which happens to be about 1.27 kHz:



Obviously it LOOKS a lot better but we can also see that using straight lines at the zero crossing, will be a very much better approximation to the sinewave, than the high frequency case.

We can therefore say that when the sample rate is fixed at 48ksps, a low frequency audio sinewave is a lot more accurately interpolated, than a higher frequency audio sinewave. In other words, the errors get bigger, and quickly so, as we increase the audio frequency. This can also easily be seen experimentally by generating a signal in WSJT-X and then observing the measured frequencies (this can be done using QDX's own built-in tools, as described elsewhere in this manual).

The error in frequency measurement is also reduced by measuring over more than one cycle. Since we are adding a fraction of a sample at the beginning, and a fractional of a sample at the end of each cycle, and counting the number of samples in between – any error on the fractional estimations (due to linear interpolation error) will be reduced by counting multiple samples. For example, if we measure for 2 cycles, the error will be halved. If we measure for 10, the error will be reduced by a factor of 10.

As mentioned earlier, it is quite unnecessary to count the frequency so often as a SINGLE cycle. If we are measuring say, an audio signal of around 3kHz – do we really need to measure it 3000 times per second? Not practically speaking, no.

So the QDX has a configuration built in, which specifies a number of cycles that must be measured, and a number of samples that must be measured. BOTH of these parameters must be satisfied, before the measurement activity completes. The default values are 1 cycle, and 480 samples. 480 samples is a time duration of 0.01 seconds. In this configuration, a very low frequency signal, below 100Hz (0.01 seconds period) would be measured properly. And at any higher frequencies than 100Hz, the minimum samples criteria will dominate, and ensure that the averaging period is automatically increased for higher frequencies! For example at 3kHz, 480 samples will contain 15 cycles, so the error in the fractional period estimation due to interpolation inaccuracy, will be reduced by a factor of 15. In this way, high frequencies do more averaging than low frequencies; we get 100 frequency measurements per second.

Again these configuration parameters are not particularly critical, and again, the default values will suffice for all imaginable purposes and most likely the QDX operator will never want or need to change these, other than perhaps for his own curiosity of investigation.

Sinewave interpolation

As an aside – for interest only – I did investigate the possibility of more advanced interpolation methods than linear (straight line). I spent some enjoyable time experimenting with a method which measures the amplitude of the sinewave (looking at the incoming samples) and then fits the two samples either side of the zero crossing to an actual sinewave rather than a straight line.

It did provide some improvement, some observable reduction in error. However, the error reduction was not as great as I had expected, and I suspect this is due to limitations in the floating point accuracy of the arc-sin implementation, perhaps (speculation) a limitation to the number of terms of a polynomial expansion? Furthermore, although the STM32 processor used has a floating point unit, the calculations were still too slow at the 72MHz CPU clock speed used. In experimental measurements the calculations were too slow to be able to perform single cycle frequency estimation at the upper end of the frequency range, which is exactly where we most need the improved accuracy; multiple cycles were therefore necessary and it was found that the improvement in accuracy of averaging simple straight line interpolation over several cycles, was more than the improvement in accuracy of the slow sinewave interpolation!

Furthermore other inaccuracies could creep in, such as the estimation of the maximum amplitude; or we'd have to assume the maximum amplitude sinewave, which would work less well on the rising and falling keying edges.

I toyed briefly (very) with the idea of trying to implement a three (or more) point sinewave fit, but dismissed it as it would involve even more calculation and be even slower. Polynomial fits were another possibility considered.

But frankly – in the end – the simple linear interpolation is very reliable at the frequency range of interest, and is already SO good, that the idea of trying to pursue more advanced approaches, seemed, particularly after a day's worth of investigation, to constitute an unnecessarily frivolous indulgence in mathematical decadence.

Errors conclusion

The frequency estimation by cycle timing method employed in QDX is exceptionally accurate, far more precise than needed for any digi modes likely to employed on HF bands during QDX operation (I cannot imagine any exceptions). There are a few boundary cases where accuracy could conceivably be impaired, and though this is actually still unlikely to be of any practical significance, configuration parameters are provided to mitigate and control the errors, with suitable default values that will be excellent for all conceivable use cases.

As a final word on audio frequency estimation, I want to also mention that WSJT-X is said to transition evenly between two frequency symbol tones, by implementing a raised cosine falling envelope shaping on the finishing tone, at the same time as a raised cosine envelope shape on the newly starting tone. As discussed previously, QDX just doesn't do envelope shaping. However, I believe that to all intents and purposes, that this fancy raised cosine tone envelope shaping is equivalent to a gentle "frequency slide" between the old and new tone frequencies. Perhaps it is even exactly mathematically equivalent, I don't know.

At any rate – for any mode WSJT-X generates, QDX will continuously measure the frequency and update the transmitter output frequency accordingly; at 100 updates per second (the default configuration) this is plenty sufficient to achieve smooth transitions between tones; additionally the frequency synthesis method used in the Si5351A itself causes a "slide" between frequencies not a sudden discontinuous transition, because the PLL takes a finite time to settle on the new value.

QDX contains its own test and analysis tools built-in; elsewhere in this manual is a description of how you can observe these frequency transmissions yourself, using the built-in tools.

Si5351A fine control

Historically QRP Labs products have set up the Si5351A with an integer frequency, except for the Ultimate3S QRSS/WSPR transmitter which has an algorithm during WSPR operation to optimize the frequency steps for very accurate 1.46 Hz tone spacing.

The problem with QDX is that the QDX must faithfully transmit any frequency given to it. The tone spacing may be very fine, or it may be coarse. QDX does not know. QDX knows nothing of the transmission mode. All QDX does is receive audio tones from a PC and effectively modulate them to RF. The tones may be 1.46 Hz spaced in the case of WSPR, or 6.25 Hz in the case of FT8, or many other possibilities. Therefore a fixed algorithm for WSPR such as Ultimate3S has, or an integer frequency configuration such as used in the QCX CW transceiver, will not be optimum here.

As an example, suppose we want an output frequency of 14,097,002.123 Hz and the firmware has to decide the parameters to configure the Si5351A chip. So let's say we chose a MS Synth divider of 62 (abiding obediently by the even integer MS Synth divider rule). Now that means the internal VCO must operate at 870,014,131.626 Hz. The multiplication factor from 25 MHz is therefore 870,014,131.626 / 25,000,000 = 34.960565265039996. Now that has to be represented as a + b / c where all are integers, b is in the range 0 to 1,048,575 and c is in the range 1 to 1,048,575 (b and c are 20-bit numbers – so the max value is 2 ^ 20 – 1).

The question therefore is the choice of b and c to get the fraction 0.960565265039996. Conventionally one may simply set c to 1,048,575 and set b to 0.960565265039996 x 1,048,575 = 1,007,224. Now we can calculate the output frequency:

25,000,000 x (34 + 1,007,224 / 1,048,575) / 62 = 14,097,001.845 Hz

The error in the output frequency is 14,097,002.123 – 14,097,001.845 = 0.228 Hz. It's rather larger than we might like; remember also that the error will be different every time and could be better or worse than this.

Well it turns out we took the cowardly way out by choosing the denominator (c) to be 2 ^ 20 – 1 (1,048,575) and that there are better choices which will bring the ratio b / c very much closer to the desired fraction 0.960565265039996. In fact the choice b = 416,333 and c = 433,425 works very well… look, now the output frequency is:

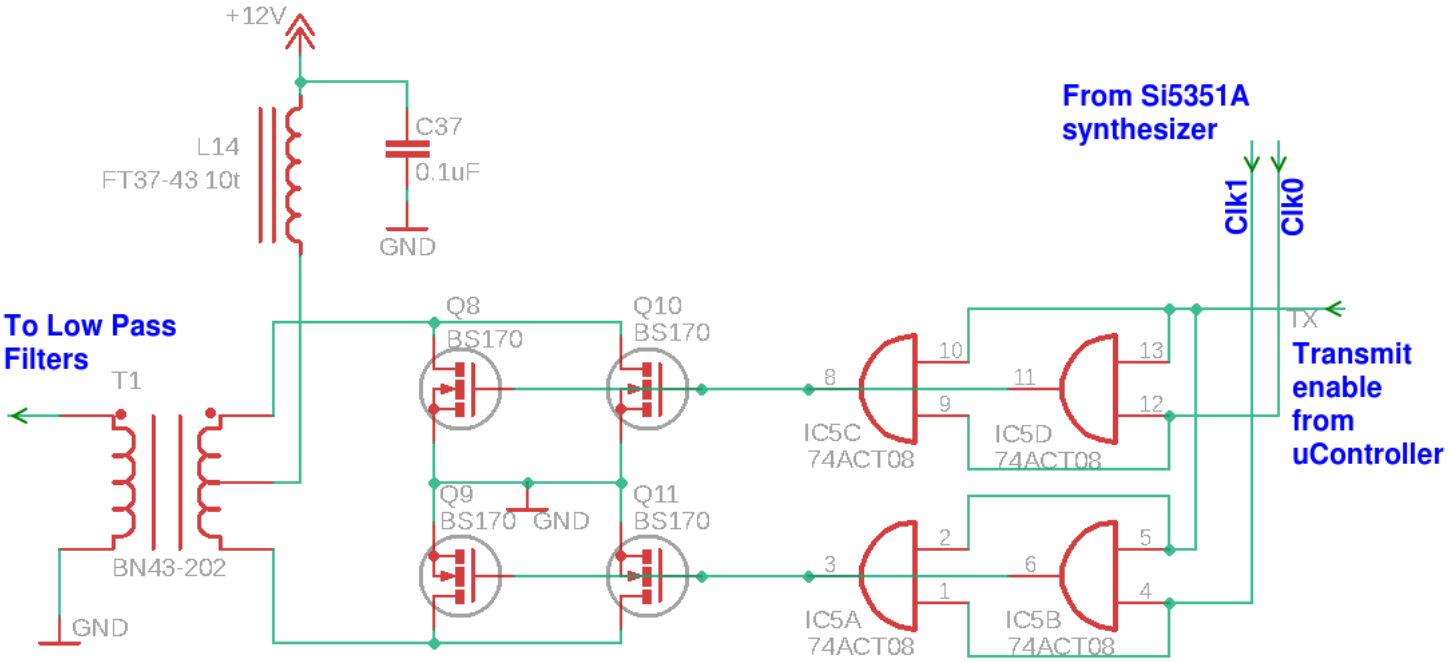
25,000,000 x (34 + 416,333 / 433,425) / 62 = 14,097,002.123 Hz

The error, to three decimal places, is now 0.000. Bingo!

However the question is STILL, how to choose b and c to get this magical and wonderful result. The answer is "best rational approximation" which delves into an area of mathematics called "continuing fractions". Rather than spoil (and lengthen) this manual with my own garbled interpretation, allow me to just refer you to Google and Wikipedia https://en.wikipedia.org/wiki/Continued_fraction .

The implementation in QDX is a simple and fast iterative algorithm based on the information found here, which rapidly converges on values for b and c which provide a very optimal ratio.

4.13 Class-D Push-pull Power amplifier



An interesting power amplifier circuit was designed specially for this transceiver. There are four BS170 transistors. Two are connected in parallel for the “push” side, and two connected in parallel for the “pull” side. Each side is driven by two AND gates of a 74ACT08 quad AND-gate IC, in parallel to ensure strong and fast drive.

The AND gate drivers are clocked from two outputs of the Si5351A Synthesizer which are arranged to have 180-degree phase shift. This provides the necessary phasing for the “push” and “pull” sides of the amplifier, without requiring an input transformer. All the 74AC08 AND gates are enabled by a “transmit” or in other words, a “key down” signal from the microcontroller; only when this signal is high, are the oscillator signals passed on to the MOSFET “gate” connections – this is the operation of the logical AND function.

The outputs of the “Push” and “Pull” sides are combined by transformer T1.

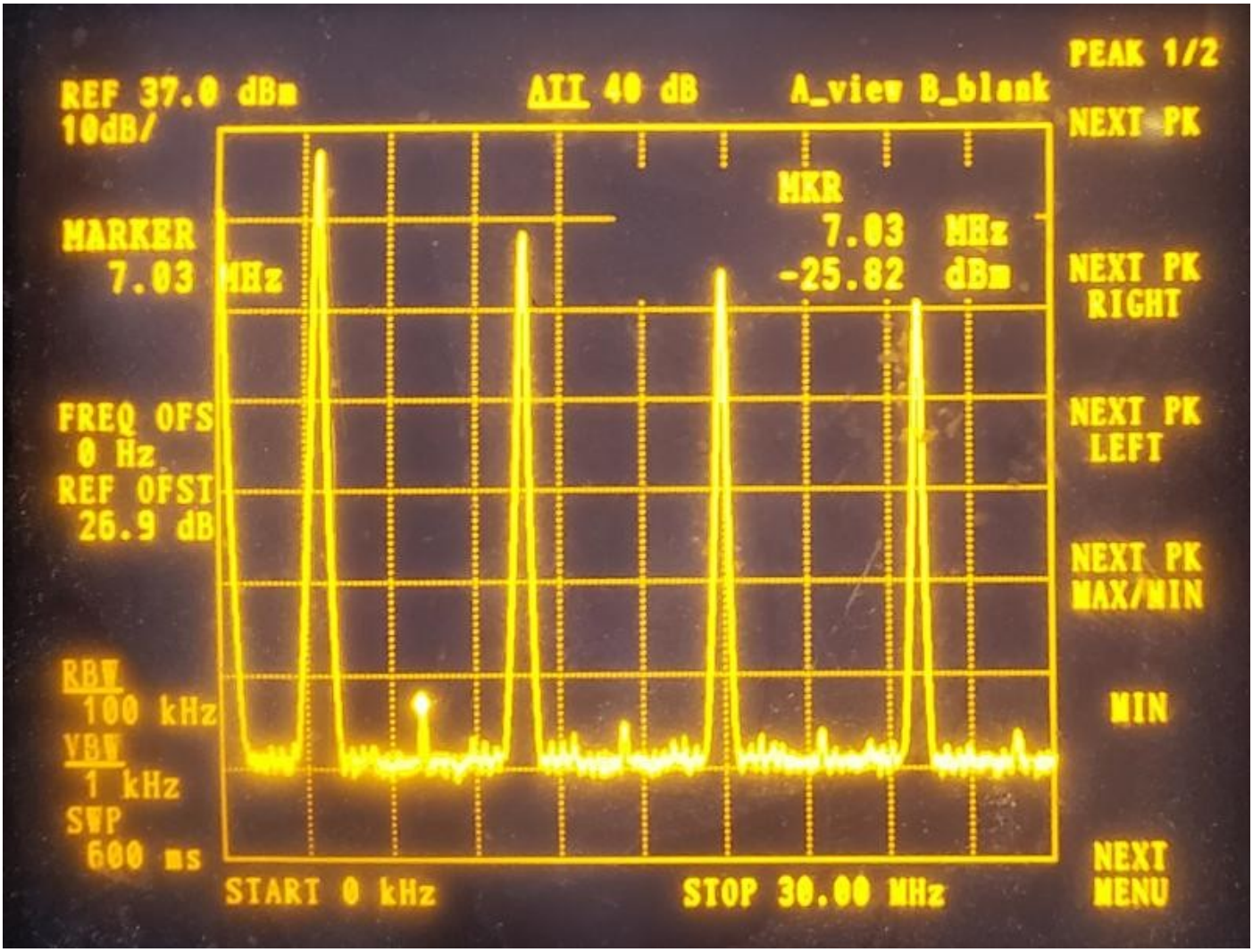
This power amplifier design provides a number of useful features:

- Class-D operation is efficient compared to Class C, or even worse, Class A/B linear which as discussed previously, is overkill for this application. Class-D is not as efficient as Class-E (as used in the QCX CW transceiver), but it better suited to a multi-band transceiver, since a Class-E amplifier has a resonant load circuit and this must be switched by band.
- Push-pull operation provides 5 W output power at a lower voltage than would be needed in a single-ended amplifier (9 V supply produces about 5 W output).
- Push-pull operation cancels even harmonics which are therefore at a very low level (due only to imperfect cancellation due to slight differences in transistor characteristics). Importantly the 2nd harmonic level is very low, which greatly simplifies the demands on the Low Pass Filters that will subsequently clean up the signal after the amplifier.
- Push-pull operation allows us to use four BS170 transistors to spread the load, but they are only connected in pairs so the driver only has to drive two paralleled gate capacitances not four.

- Using an antiphase drive (the precise 180-degree out of phase signals from the Si5351A) eliminates the need for an input transformer that is normally required in push-pull power amplifiers.
- All components are low cost and easy to replace in the unlikely event of failure.

As an example spectrum analyzer output, consider the following trace showing an 80m QDX. For this measurement, a 50-ohm dummy load was connected directly at the output of transformer T1, with no Low Pass Filtering.

The second harmonic is seen, about -70 dBc (70 dB below the fundamental) due to the cancellation effect of the Push-Pull amplifier. The subsequent even harmonics are even lower amplitude. Only the odd harmonics are problematic and require filtering.



4.14 Switched output Low Pass Filters

Every radio transmitter always requires low pass filters at its output to attenuate any out of band spurious emissions to sufficient level to comply with regulatory requirements (for example FCC in United States).

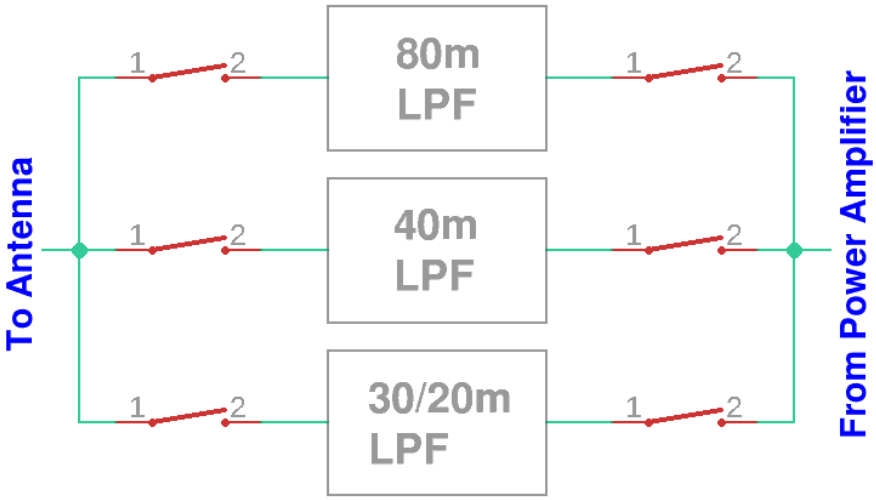
In QDX the Low Pass Filters are carefully optimized to provide excellent performance with low parts count. This can be achieved because the Class-D push-pull power amplifier design produces a very low level of even harmonics.

It is very important to reiterate that Low Pass Filter and Power Amplifier design must be considered carefully, at an overall SYSTEM LEVEL. In fact this applies to the entire transceiver design, but nowhere more importantly than the transmitter output stages, where the performance is critical to your regulatory compliance. The requirements on Low Pass Filtering are determined by the level of spurious emissions produced by the final Power Amplifier as well as levels in earlier stages feeding into the power amplifier. One cannot simply transplant a circuit block from one transceiver design to another, and expect the performance to be satisfactory. System level design is critical.

In QDX there are three Low Pass Filters. Each is made up of a 5-element filter, with a 6th element as resonant trap at the 3rd harmonic. The three filters are:

- 80m
- 40m
- 30/20m

The intrinsically low levels of 2nd harmonic is what lets us share the third filter between 30m and 20m.



There is a PIN diode switch either side of each Low Pass Filter. So at any one moment, one pair of switches is activated, to switch the appropriate Low Pass Filter into the circuit.

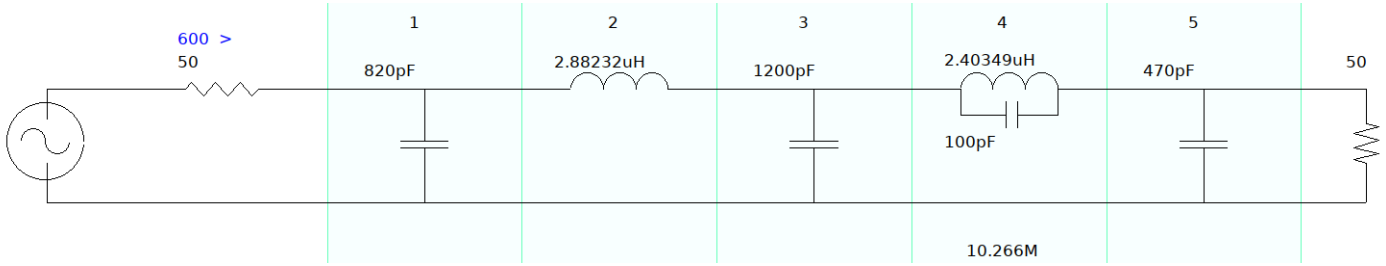
The discussion of this block of the circuit divides naturally into a two parts: the LPF design, and the PIN diode switches design.

Low Pass Filters

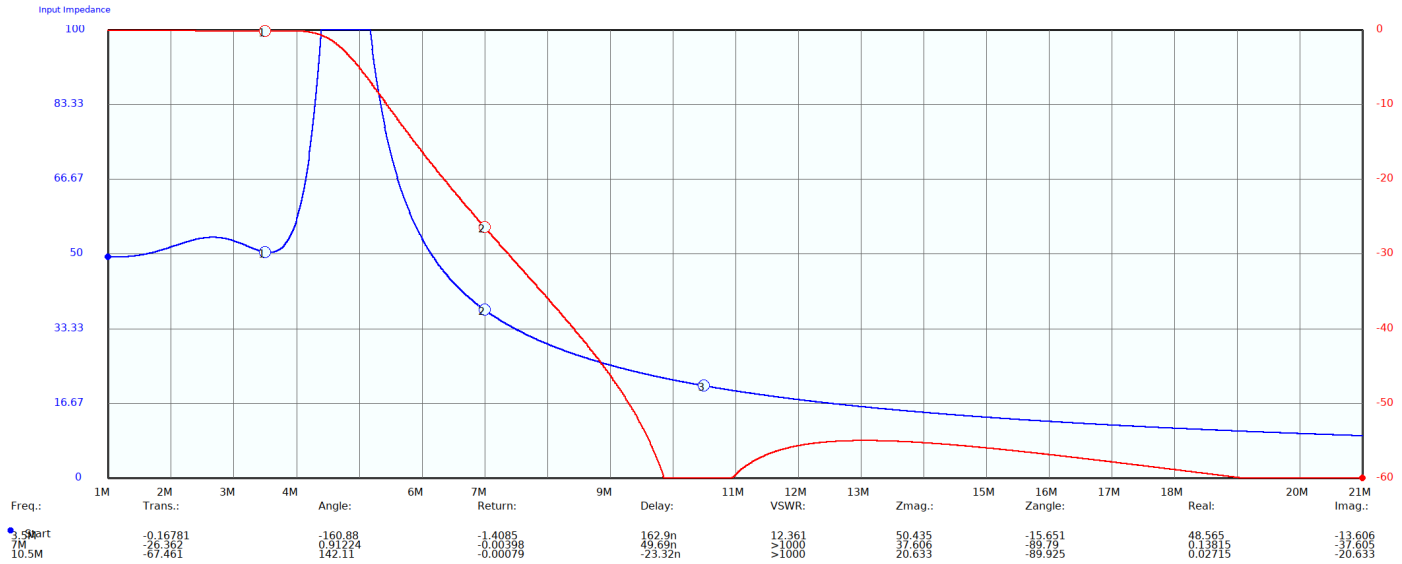
The Low Pass Filter design was done using the Elsie program by Tonne Software. It is freeware and can be downloaded from here: http://tonnesoftware.com/elsie.html

80m:

For 80m this is the final schematic, having a resonant trap at the third harmonic:



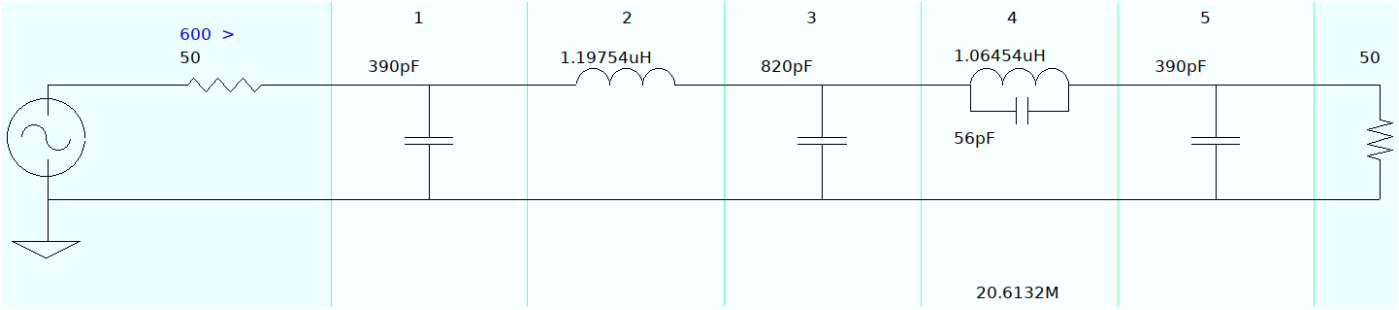
The simulation shows:



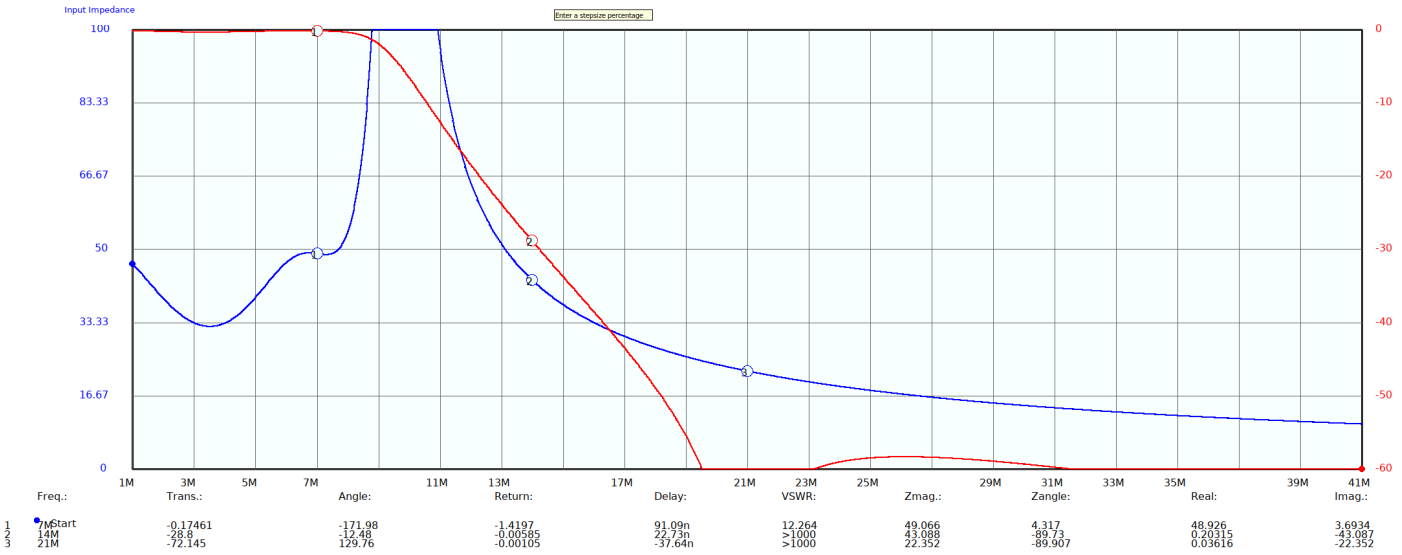
Fundamental:	3.5 MHz	-0.17 dB
Second harmonic:	7.0 MHz	-26.4 dB
Third harmonic:	10.5 MHz	-67.5 dB

40m:

For 40m this is the final schematic, having a resonant trap at the third harmonic:



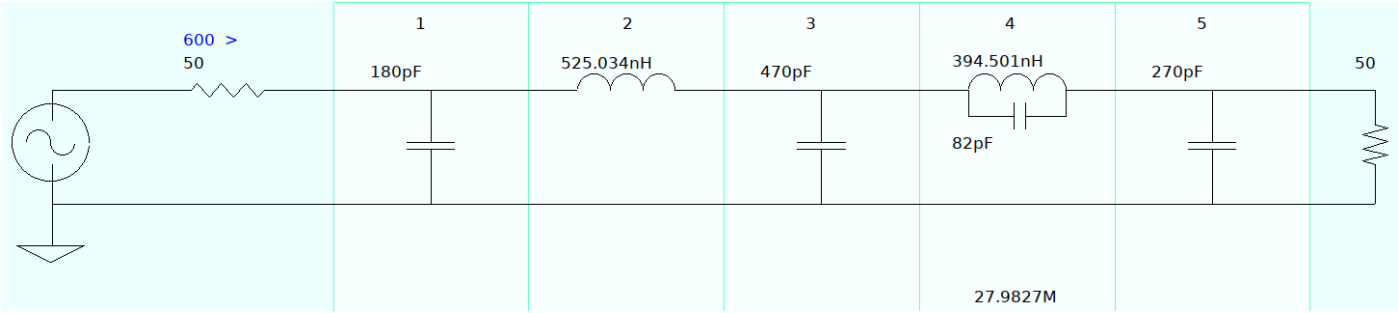
The simulation shows:



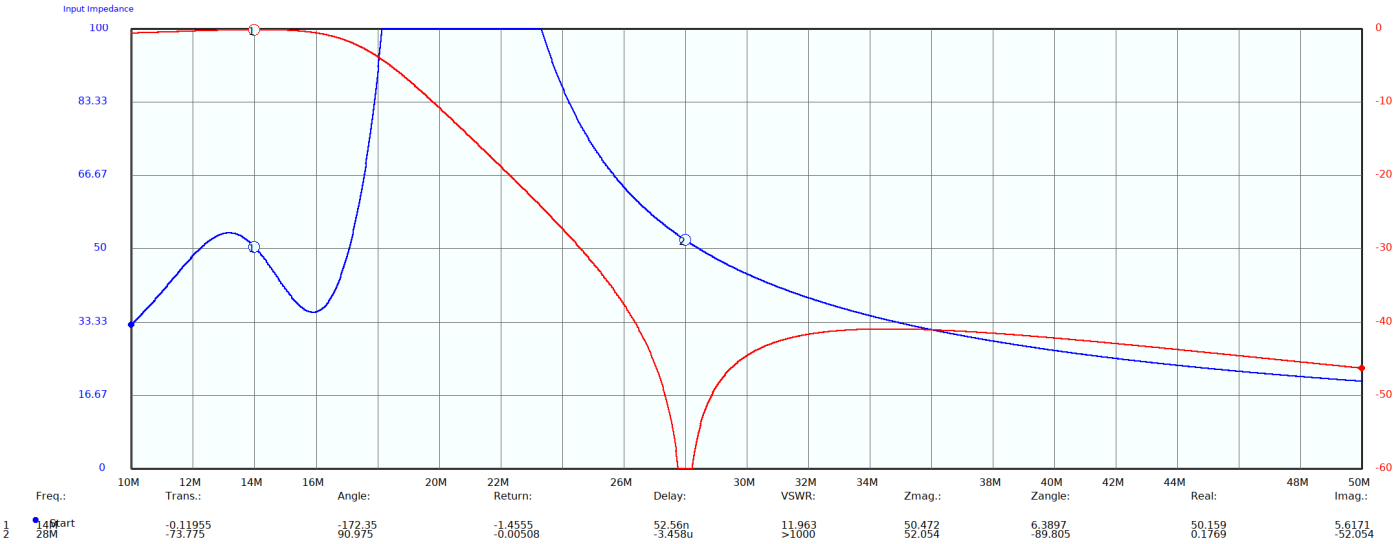
Fundamental:	7 MHz	-0.17 dB
Second harmonic:	14 MHz	-28.8 dB
Third harmonic:	21 MHz	-72.1 dB

30/20m:

For 30/20m this is the final schematic, having a resonant trap at the 20m second harmonic, 28MHz. This is a departure from the 80 and 40m designs, which have the trap at the third harmonic. The reason for this is that the 20m second harmonic is close to the 30m third harmonic. This was found to represent an optimal compromise between 30m and 20m performance.



The simulation shows:



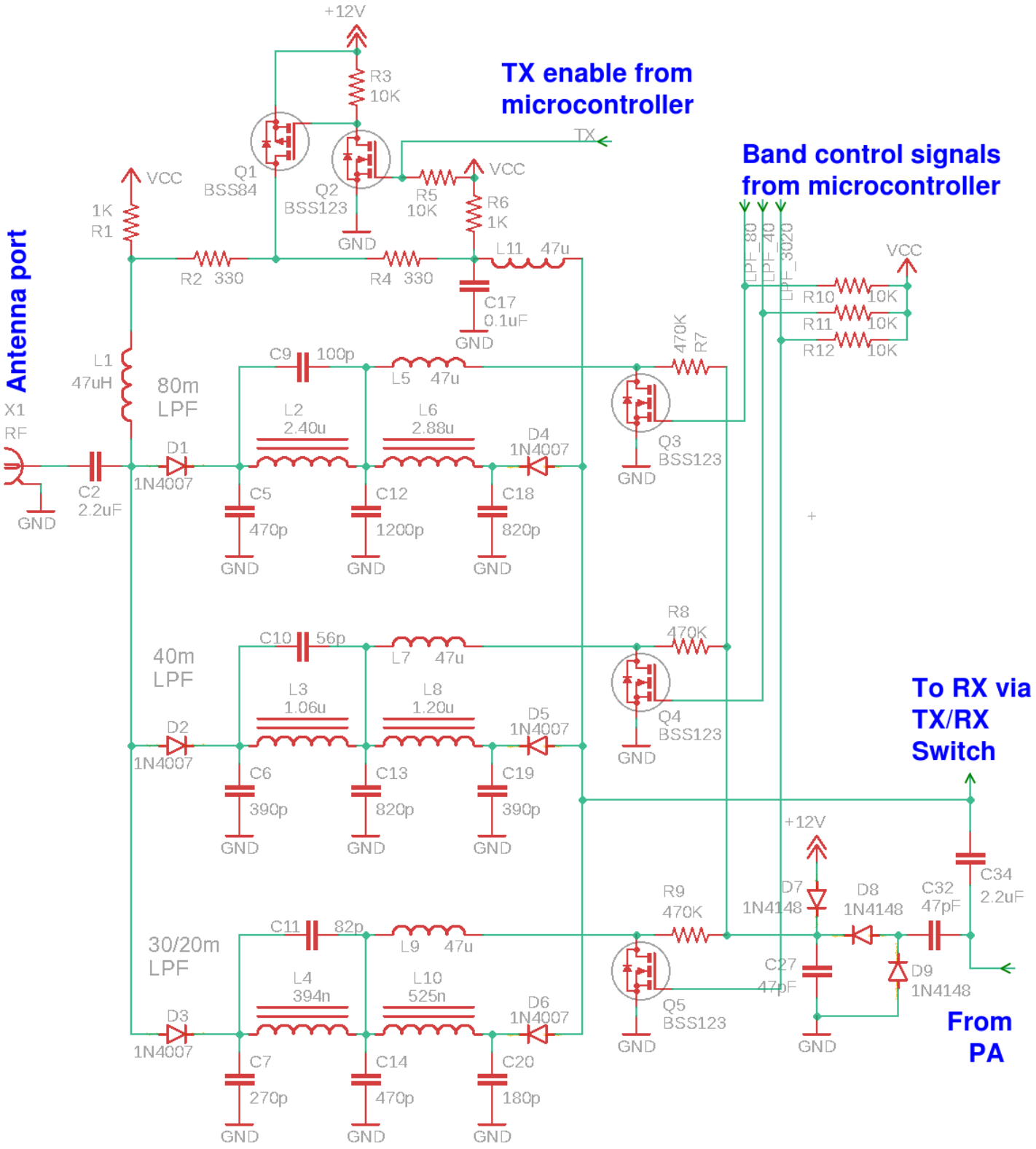
20m:

Fundamental:	14 MHz	-0.12 dB
Second harmonic:	28 MHz	-73.8 dB

On 30m, while the second harmonic (20.2 MHz) attenuation is only about 12dB, the even harmonics are in any case at a very low level; the more important harmonic is the 3rd harmonic and for 30.3 MHz the attenuation is about -45 dB which is very satisfactory.

Low Pass Filter switching

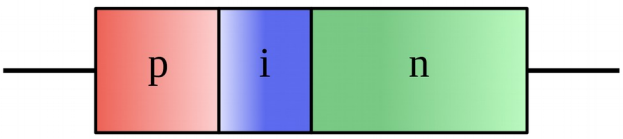
The schematic of the Low Pass Filters and switching arrangements is shown below.



Very often relays are using for Low Pass Filter switching. However relays have several disadvantages:

- Bulky
- Noisy
- Expensive
- Potentially unreliable (mechanical moving parts)
- Poor isolation at RF, can be important in some applications (unless expensive relays used)

Cost and bulk are of critical importance here. A better solution comes in the form of diode switching. Not just any old diode, PIN diodes. This is a fascinating topic and I did a great deal of research and experiment to perfect the circuits used in the 50W PA kit http://qrp-labs.com/50wpa and many of those ideas are carried over to the QDX design.



Ordinary diodes cannot be used for switching because they could act as Rectifiers, switching on and off in time with the RF cycle. A PIN diode, however, is different. It is a diode with a wide undoped “intrinsic semiconductor” region between the usual P-type and N-type doped semiconductor layers of a regular diode (Wikipedia image).

The important point is that this undoped “intrinsic layer” acts like a storage reservoir. When incoming forward-biased current arrives, this “intrinsic layer” fills up with electrons. When the diode is reverse biased, it takes time for all these electrons to empty out of the intrinsic layer. At high enough frequencies, the time available is too short! So the diode never “turns off”. It remains conducting through the whole RF cycle.

PIN diodes behave like variable resistances. Their resistance is determined by the amount of DC forward bias current. They are near-perfect variable resistors which add very little distortion to the signal. In our case, we don’t care about variable resistors, all we want is either a very low resistance (On), or a very high resistance (Off).

A problem is that PIN diodes tend to be expensive, harder to find items. However, a solution comes along, the 1N4007 rectifier diode! This is the big brother of the 1N400x series of rectifier diodes, and has a reverse voltage rating of 1,000 V. By good fortune, its internal construction in order to achieve this very high PIV rating, is very similar to a “real” PIN diode. Hence the 1N4007 is often referred to as the “Poor man’s PIN diode”!

No doubt the 1N4007’s characteristics are not repeatable enough, or its resistance not linearly controllable enough, or some such other imperfection, that it is not suitable for all applications. But in our application where all we need is an On/Off switch at HF, a 1N4007 costing mere cents functions just as well as a “real” PIN diode costing 50x the price!

1N4007 functions well enough as a PIN diode switch across the whole of HF to 10m band, and even down to 500kHz.

What about dynamic range and third order intercept point IP3? A properly handled 1N4007 should not impact the dynamic range or IP3. An excellent set of measurements by Claudio IN3OTD see https://www.qsl.net/in3otd/electronics/PIN_diodes/PIN_diodes.html and https://www.qsl.net/in3otd/electronics/PIN_diodes/1N4007.html confirms the excellent characteristics of the 1N4007, which at HF in this switching application are similar to other PIN diodes. Other experimenters have determined similar results. Broad conclusions are that a properly biased 1N4007 at HF (1.8 MHz-30 MHz) has the following characteristics:

- “On” insertion loss less than 0.1 dB for 10 mA forward bias current
- “Off” isolation at least 30 dB (at worst case, 30 MHz frequency)
- IP3 > +50 dBm

Therefore the use of 1N4007 PIN diode switches should result in very little loss of transmitter output power, very little loss of sensitivity on receive, and will not degrade the excellent IP3 (and dynamic range) characteristics of the QDX transceiver with its high performance double balanced Quadrature Sampling Detector front end.

What of “proper biasing of PIN diodes”?

In my opinion the most excellent reference for the use of PIN diode switches for HF Transmit/Receive is the work of Don W6JL. Don has a very informative page on QRZ.com https://www.qrz.com/lookup/w6jl and a long article “Homebrew PIN diode QRO QSK system” in Funkamateur, March 2016 which is available as a PDF on the internet see https://www.funkamateur.de/tl_files/downloads/hefte/2017/w6jl_improved_qsk_system_mar_2016.pdf. Study of W6JL’s work should be the first task of anyone wishing to experiment with HF PIN diode switching. Don has a separate homebrew transmitter and receiver. The transmitter is 600W and is permanently connected to the antenna and the receiver, via his solid state PIN diode switching system. Sidetone is via the receiver monitoring the transmitter directly! The switch provides 137dB of isolation in the “Off” state. Amazing! We don’t need anything so extreme (or complex) here – but the principles are well described by W6JL and his work is inspiring.

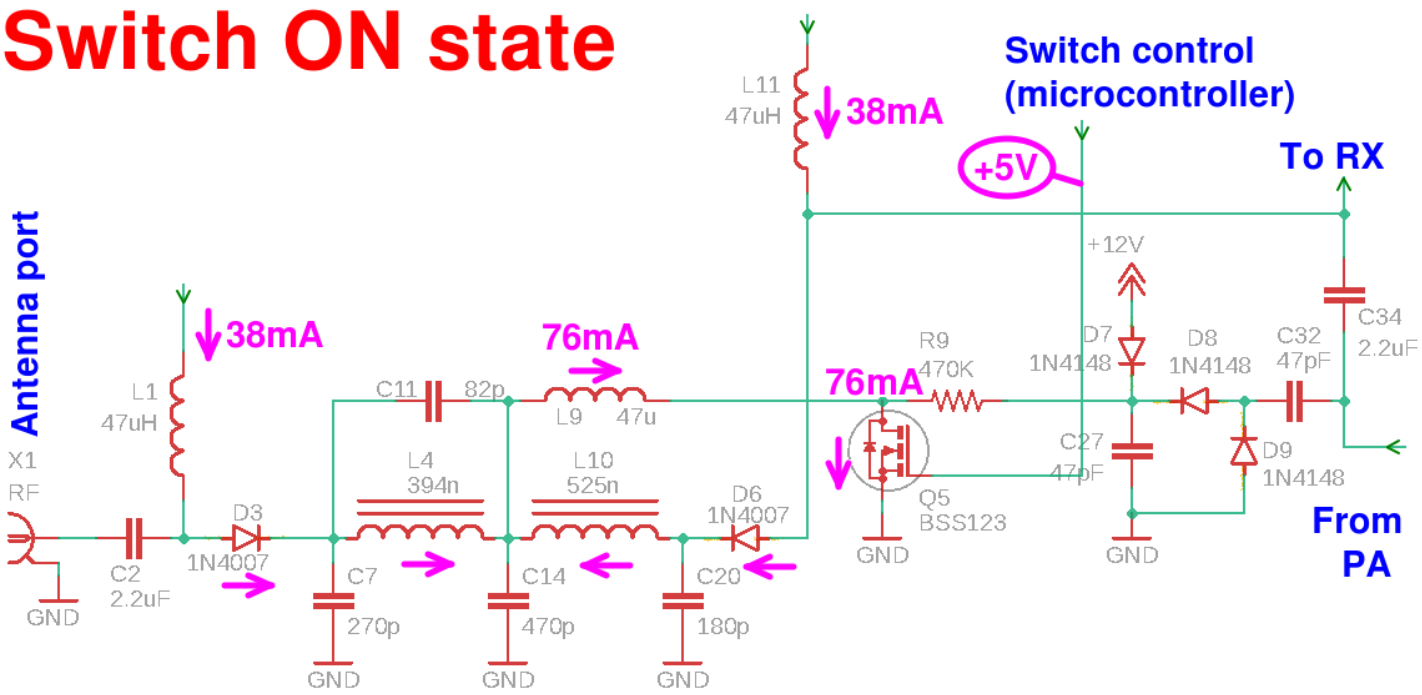
It is possible to distill the whole topic into two important rules:

- To switch the PIN diode Off, apply a large reverse bias VOLTAGE. The voltage should be higher than the peak-to-peak RF being switched.
- To switch the PIN diode On, apply a forward bias CURRENT; the more current, the lower the insertion loss. At 10 mA forward bias, the insertion loss is only about 0.1dB. One can increase it a bit further, to be on the safe side.

When the two “rules” are followed carefully, the performance of the PIN diode switch is excellent; furthermore it is low cost and reliable to implement using inexpensive common components like the 1N4007. In the QDX transceiver the switching task is considerably simplified compared to the 50W PA kit, because here there is no need to support fast QSK (full break-in) operation or provide a receive bypass path.

To understand the operation of the switching, consider just ONE of the Low Pass Filter circuits, the 30/20m circuit. At any time, one set of the three Low Pass Filter switch pairs are ON, and the other two are OFF.

Let’s consider first, the switch in the ON state. The partial schematic is:



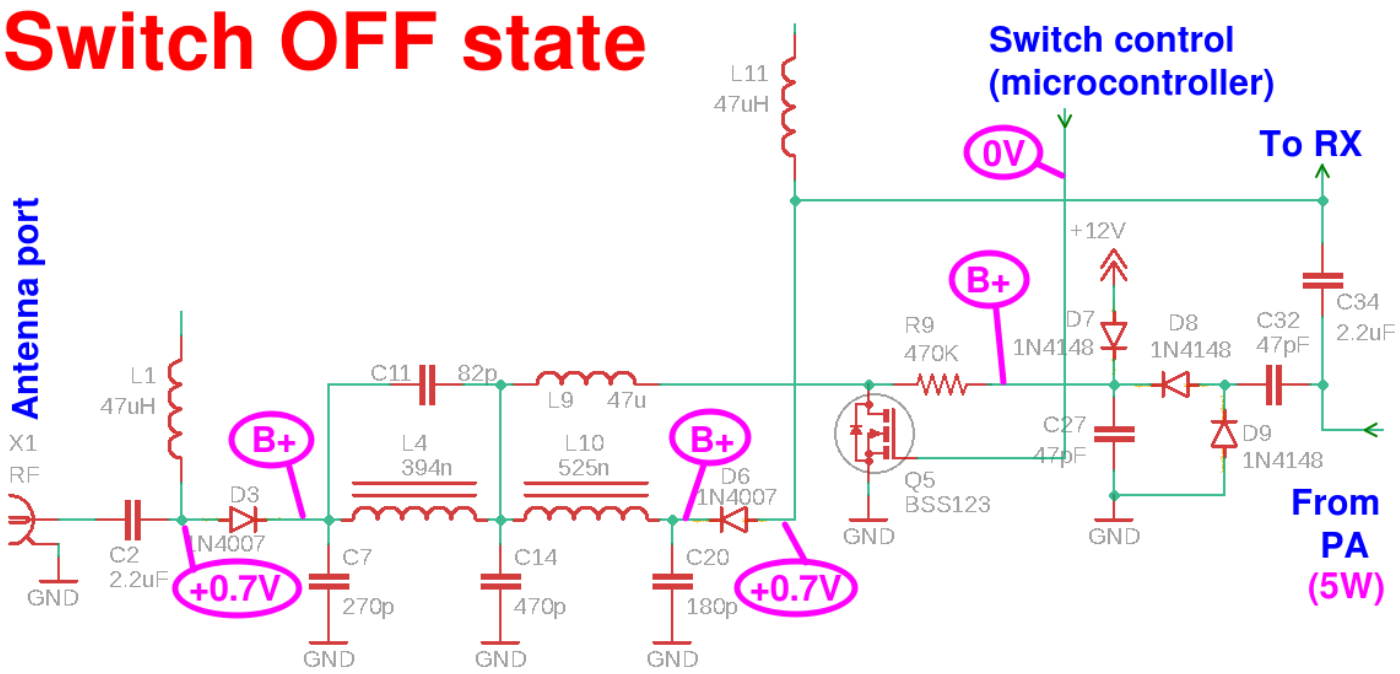
Remember that for a diode to be ON, it should be passing at least 10 mA current. In QDX, the forward current is set at 38 mA during transmit, this is determined by the bias circuit around Q1 and Q2 which will be discussed in a moment. Why so high, you may wonder? The reason for 38 mA is that the diode, if biased at lower forward currents, increases the level of even order harmonics due to its own distortion. 38 mA switches it ON so well that the distortion level is very low.

Remember also that capacitors block DC, but allow the passage of RF; whereas inductors block RF, but allow the passage of a bias current (Direct Current, DC).

In the ON state, the microcontroller I/O pin controlling the Low Pass Filter is set to high impedance, resistor R12 (10K) to +5V then pulls the potential at the gate of MOSFET Q5 to +5V, switching ON the Q5 MOSFET. The bias circuit can now drive 38 mA through each of L1 and L11, through the respective diodes D3 and D6, through L4 and L10 respectively, the sum of the currents (76 mA) through L9 and through the Q5 transistor to ground. These flows are indicated by the pink arrows on the diagram. The various capacitors block DC, only permitting the bias current to flow through the components indicated by the pink arrows.

Now that a sufficient forward bias current flows in D3 and D6, RF will see a low resistance all the way from the PA, through C34 and D6 to the Low Pass Filter; and on the other side of the filter, RF will pass easily through D3 and C2 right to the antenna. L1 and L11 block RF from escaping into the bias current circuits.

In the ON state, we think about current (forward bias current). In the OFF state, we must think about voltage (reverse bias voltage). Here's the same schematic, labeled for the OFF state.



Remember the first rule of PIN diodes? "To switch a PIN diode off, apply a reverse voltage that is larger than the peak-to-peak voltage of the applied RF". Now, 5W of RF is 45V peak-to-peak (assuming 50-ohms impedance) and therefore we need more than this, as reverse bias, to switch off the diodes. Obviously this is not a voltage that is available from any supply rail! While Don W6JL had available 350V from the screen grid of the valves (tubes) in his transmitter – we have no such luxuries available here.

The very neat solution is to “borrow” a little energy from the PA output itself! Then voltage double and rectify it. The result is a nice clean DC which is always more than the peak-peak of the signal, and plenty enough to reverse bias the diodes so they are always perfectly and completely OFF.

This doubler section of the schematic is formed by D9, D8, C32 and C27. The resulting voltage, we shall call B+ in honour of valve (tube) circuits of old. Or HT if you prefer. It’s exact value will depend on the impedances and other factors but don’t worry, it is sufficient for the proper diode OFF condition.

Now this voltage passes through resistor R9, through inductor L9, and through inductors L4 and L10, right to the cathodes of diodes D3 and D6 respectively. The inductors don’t impede DC; resistor R9 would impede DC but the current involved is near zero so practically speaking there is no significant voltage drop across R9. The anodes of diodes D3 and D6 are at a potential of about +0.7 V. Why? Because whilst this Low Pass Filter is OFF, one of the other two LPFs is switched ON, and 38 mA current is flowing through each of its diodes, resulting in a voltage across the diodes of about 0.7 V.

So with B+ on the cathode and 0.7 V on the anode, the diodes see a large reverse bias voltages and satisfy the conditions for the diodes to be OFF. They present a high resistance to the RF, which cannot then flow through this Low Pass Filter. It is OFF.

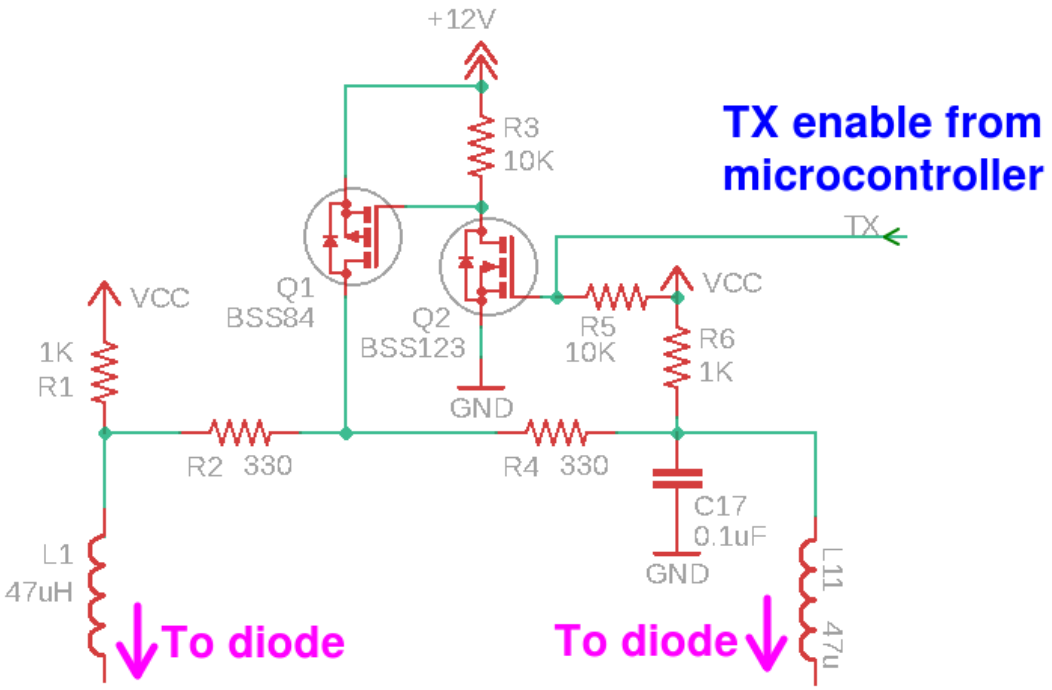
The next thing to worry about is what happens when we are NOT transmitting. The signal during receive, still needs to pass through the proper Low Pass Filter for the band of operation. Its switches still need to be ON, and the other two LPFs OFF. Now there is no large 5 W (45 V peak-peak) signal to rectify and double to make a large B+ voltage. On the other hand, we also no longer need such a large B+ voltage, because now the Low Pass Filters (and their switches) are only seeing the small signals picked up by the antenna. To provide some voltage to reverse bias the PIN diodes, D7 connects +12V (the QDX supply voltage) to B+ during receive, when the doubler is not producing rectified RF (during transmit).

The other minor detail to consider, is that during Receive we aren’t handling anywhere near such a large amplitude signal so we don’t really need such a large current as 76 mA (total for two diodes) forward bias through the ON switches. A smaller current will suffice and will reduce the overall QDX receive current, making it more battery friendly.

Here’s the bias circuit that generates the appropriate forward bias currents.

When in receive mode, the TX signal from the microcontroller is at 0V; this keeps Q2 MOSFET switched off.

Q1 is a P-channel MOSFET and its gate is pulled to +12V when Q2 is off via 10K resistor R3, meaning that Q1 is also switched off.



In this configuration current flows from the Vcc supply rail (+5 V) through the 1 K resistors R1 and R6, and the inductors L1 and L11, to the respective two diodes. The current in each diode is given by Ohm's law I = V / R. The voltage across the resistor is 4.3 V because the voltage across the diode is 0.7V approximately. Therefore the current flow in each leg is 4.3 V / 1000 ohms = 0.0043 A, which is 4.3mA.

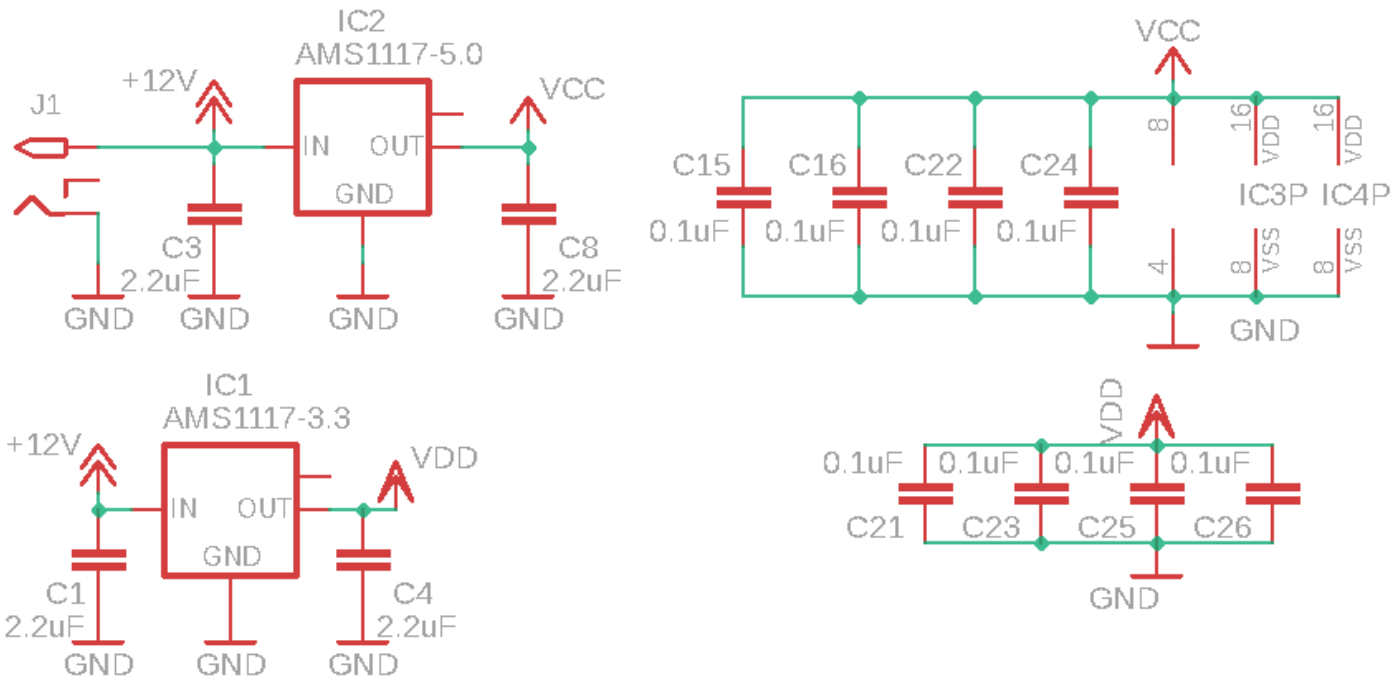
So in receive mode, the forward bias in each of two diodes in the ON Low Pass Filter, is 4.3mA for a total switch current of 8.6mA, which is not unreasonable.

When the TX signal from the microcontroller goes high, transistor Q2 switches on, pulling the gate of Q1 to near ground potential, thereby switching Q1 on. Now 12 V flows through Q1, and through each of R2 and R4, and on to the diodes. The current contribution through each resistor is given by Ohm's law I = V / R. The voltage across the resistor is approximately 11.3 V because the voltage across the diode is 0.7V approximately. Therefore the current flow in each leg is 11.3 V / 330 ohms = 0.0342 A, which is 34.2 mA. This sums with the current from the +5 V supply rail, to give a total of 34.2 mA + 4.3 mA = 38.5 mA.

So in summary, due to this biasing circuit:

- In Receive, total switch current is 8.6 mA approximately
- In Transmit, total switch current is 77 mA approximately

4.15 Voltage regulation and supply decoupling



The two fixed voltage regulators in the QDX design produce +3.3V and +5.0V. The voltage regulator type is AMS1117. This voltage regulator became infamous in the QRP Labs QCX-mini CW transceiver. It was found to be unstable in the case of intermittent power connection such as can occur when a power supply cable is hot-plugged. Furthermore, hot-plugging other connections could cause a temporary short circuit of the +5V rail to ground, which also destroyed the AMS1117. The failure mode is particularly unpleasant because rather than fail open-circuit, it fails short-circuit, and passes the input voltage (for example +12V) to the +5V rail, destroying everything on the +5V rail.

In the QCX-mini first batch, the problem was partially mitigated by addition of a 10uF electrolytic or tantalum capacitor right at the power input to the radio. This prevented the issue with unstable power supply but did not address the issue with hot-plugging to the PTT plug.

In subsequent batches of QCX-mini, the voltage regulator was changed to 78M05 which is practically indestructible.

Unfortunately the QDX boards were manufactured before the delicate nature of the AMS1117 became known. Therefore the 10uF capacitor addition at the input power connector is also required on the QDX. The issue with hot-plugging other connections will not arise since the only other two connections on QDX are the USB connector and the RF connector, neither of which have any connection to the +5V or +3.3V rails. Therefore provided a capacitor of at least 220uF is added, and the supply voltage is kept below 12V, the AMS1117 will not cause an issue.

QDX has NO reverse polarity protection!

Make sure you connect the power correctly. The coaxial DC power supply connector must have +12V on its center pin and Ground on its outer barrel.

REVERSE POLARITY will kill your QDX!

5 Firmware Update procedure

On occasion QRP Labs may make available updated firmware for QDX, in order to deliver bug fixes or functionality enhancements.

QDX contains a new firmware update procedure for STM32-series microcontrollers, called QFU (**Q**RP Labs **F**irmware **U**pdate) which provides the following features:

- **Easy** – anyone can do the firmware update
- **No additional hardware required:** only a standard USB A-B cable (or micro-USB cable if you have installed a micro-USB connector)
- **No additional software required:** just the standard file manager application that is already available on any PC
- **No drivers:** no special drivers need to be installed, the existing drivers on any modern PC operating system are used
- **Works on any PC Operating System:** and in the same way: Windows, Linux, Mac
- **Secure:** firmware files are published on the QRP Labs website and are encrypted using 256-bit AES encryption technology

Entering bootloader (firmware update) mode:

QDX provides two possible ways to enter firmware update mode:

1) On powering up QDX, you will see the QDX status LED flickers quickly for 5 seconds after power up, then illuminates solidly. If you power down QDX during this 5 seconds, then apply power again, QDX will now boot up in QFU (Firmware Update) mode.
2) Select the “Update firmware” menu option in the QDX Terminal (see subsequent section of this manual). QDX will then enter firmware update mode.

In firmware update mode, the status LED flashes slowly.

Exiting bootloader (firmware update) mode:

QDX provides two possible ways to exit firmware update mode:

1) Update the firmware! After updating the firmware, QDX will automatically reboot in normal operating mode.
2) Power down QDX, and re-apply the power again. QDX will reboot in normal operating mode.

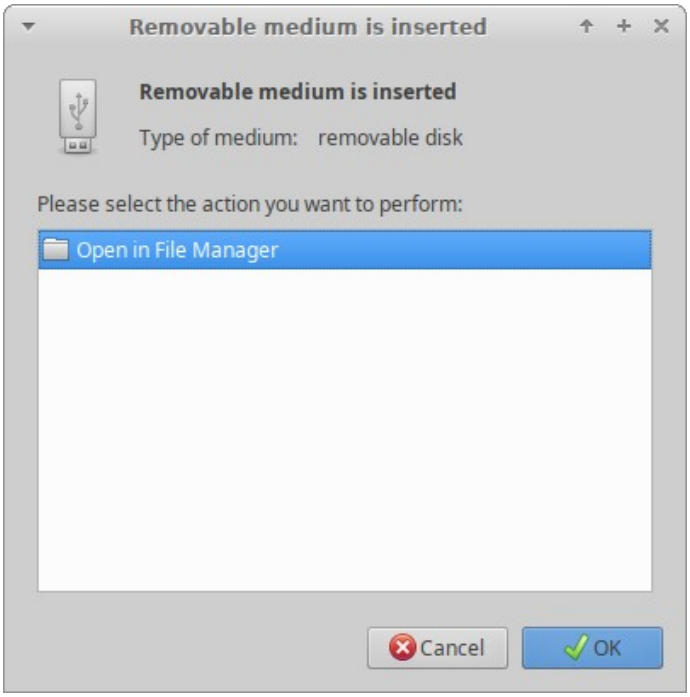
USB Flash memory stick emulation:

In the firmware update mode, the QDX pretends to be a USB Flash memory stick, having a 4MByte capacity and implementing a FAT16 file system. This virtual “Flash stick” contains two files:

1. the firmware program file of the QDX microcontroller. You may read the file from QDX, or write a new one, just by dragging files in your file manager application.

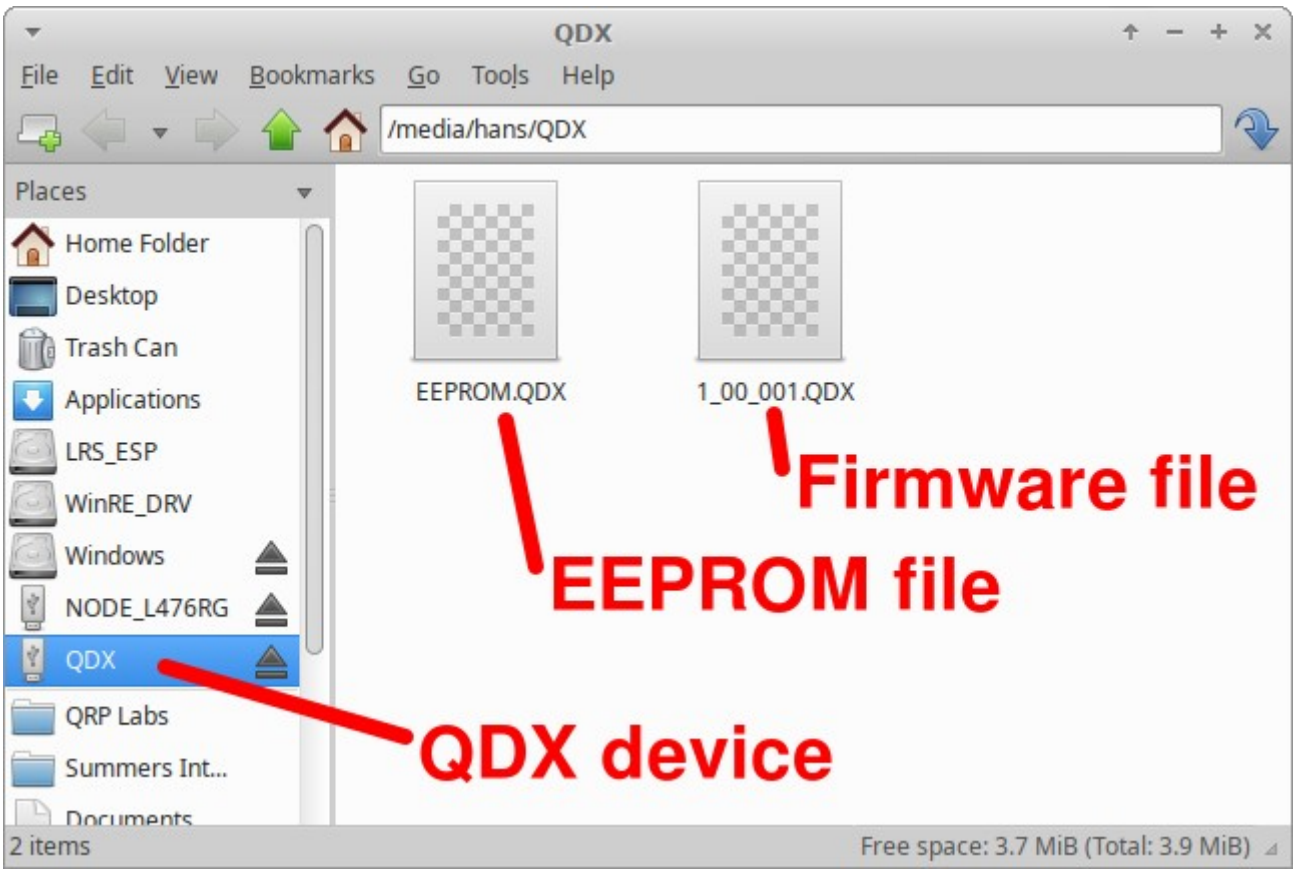
2. EEPROM contents: the QDX configuration and log file (if enabled). Again, you can read the file from QDX or write a new one to QDX, simply by dragging files in your file manager application.

On entering the Firmware update procedure, a pop-up window should appear on your PC. On my system (Linux XUbuntu 18.04) it looks like this:



Click the OK button.

The File Manager window will then open, and on my system looks like this:



QDX appears as a removable USB Flash device named “QDX”, and the folder two files. The firmware file in this example is named shows a single file which is the firmware version file, 1_00_001.QDX in this example. The EEPROM file is always named EEPROM.QDX. You can read and write EEPROM files in order to make and restore backup copies of your configuration etc.

The firmware file name must not be longer than 8 characters, and cannot contain punctuation or spaces; the file extension must be no more than 3 characters. This is because the file system emulation is FAT16 and these are the specifications of the FAT16 format.

You may check the properties of the file and will note that it is a 119.5K file. QDX firmware images are always a 119.5K file. The creation date and modification date etc. have not been set, because it was important to minimize the size and complexity of the QFU bootloader, in order to maximize the space available to the application firmware.

You may copy the existing firmware file to another directory of your computer. Crucially, to do the firmware update, all you need to do is copy the new firmware file to this QDX “Flash disk”.

Download the new firmware file from the QRP Labs website, unzip it, and simply drag it into the folder where the existing firmware file version is shown. Or copy and paste it, however you wish.

The file on the QRP Labs website is a ZIPPED file, please be sure to unzip it to get the .QDX file before copying it to QDX.

As soon as you copy the new file to the QDX QFU “flash drive”, the QDX QFU bootloader erases the current program from its memory and installs the new one.

The QDX firmware is 256-bit AES encrypted and this means:

- The encrypted QDX firmware file will only work on a QRP Labs QDX board, it cannot be installed on any other board, even one containing the same processor.
- No other firmware file will work on the QRP Labs QDX board except an official QRP Labs encrypted QDX firmware file.

The procedure will vary slightly for different Operating systems but in all cases is just a simple matter of copying the new firmware file to the emulated QDX QFU USB Flash drive.

The above firmware update procedure works on ANY modern OS because the QFU bootloader emulates a USB Flash memory stick with the USB Mass Storage Device (MSD) class, for which drivers are already present.

The QFU bootloader implements a USB device stack (Mass Storage Device class), emulated FAT16 file system, Flash erase/write, and 256-AES encryption.

6 Terminal Applications

QDX provides a suite of applications which may be accessed via a Terminal emulator running on your PC. These applications provide configuration utility, and various self-test tools. It is very educational and interesting to experiment with these tools.

However, **most QDX users need never use the terminal applications, it is not necessary for ordinary operation of QDX with WSJT-X etc**. The terminal applications are for the interested user, or if you need to set up a particular configuration.

The terminal applications display everything as ASCII text in a 80 x 24 character window. It's not as polished as a dedicated graphical user interface software application for QDX would be. However, it has the advantage of requiring no special software or drivers, and all the variations that would have to be supported for different PC Operating Systems such as Mac, Windows and Linux, software installation procedures etc. Instead, all the applications are hosted and coded in the QDX itself. The terminal emulator is only used to display the results. This keeps things simple and low maintenance. After all, the terminal applications are useful bonus features rather than core QDX functionality.

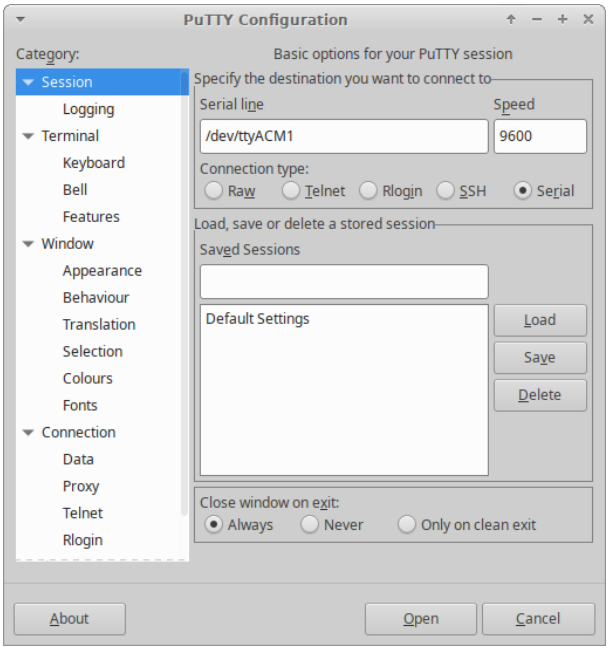
6.1 PC terminal emulator

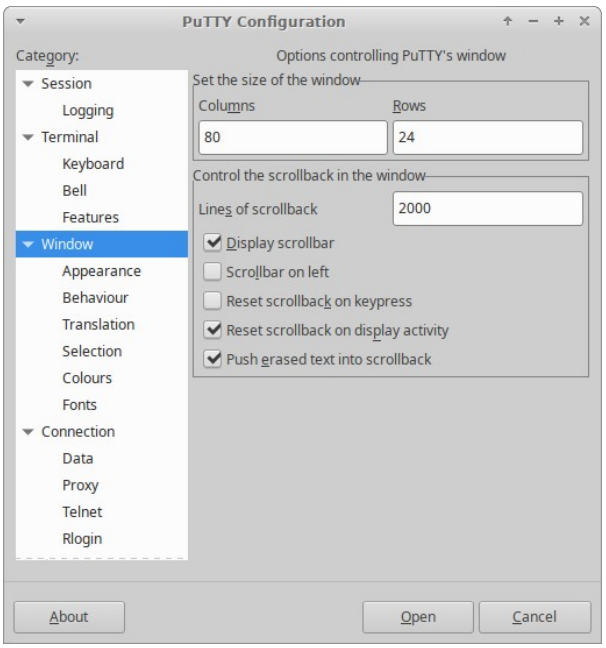
I use Linux (XUbuntu 18.04) and I'm using the PuTTY terminal emulator. There are numerous other terminal applications which will work fine. You may have your own favourite. They are all capable of correct operation with QDX in its terminal mode.

I start PuTTY using command line "sudo putty" then connect to QDX on /dev/ttyACM1. Again as before (refer to the Operating Instructions for connecting to QDX using CAT on WSJT-X) it is necessary to know which serial port is being used by QDX. There is also a guide to identifying the serial port at http://qrp-labs.com/qlg2 (scroll down the page); or you could use WSJT-X.

Make sure WSJT-X is NOT running, when you connect to the QDX serial port using the terminal emulator. Only one PC application at a time can connect to Virtual COM Serial ports.

Set the size of the terminal window to 80 columns and 24 rows.





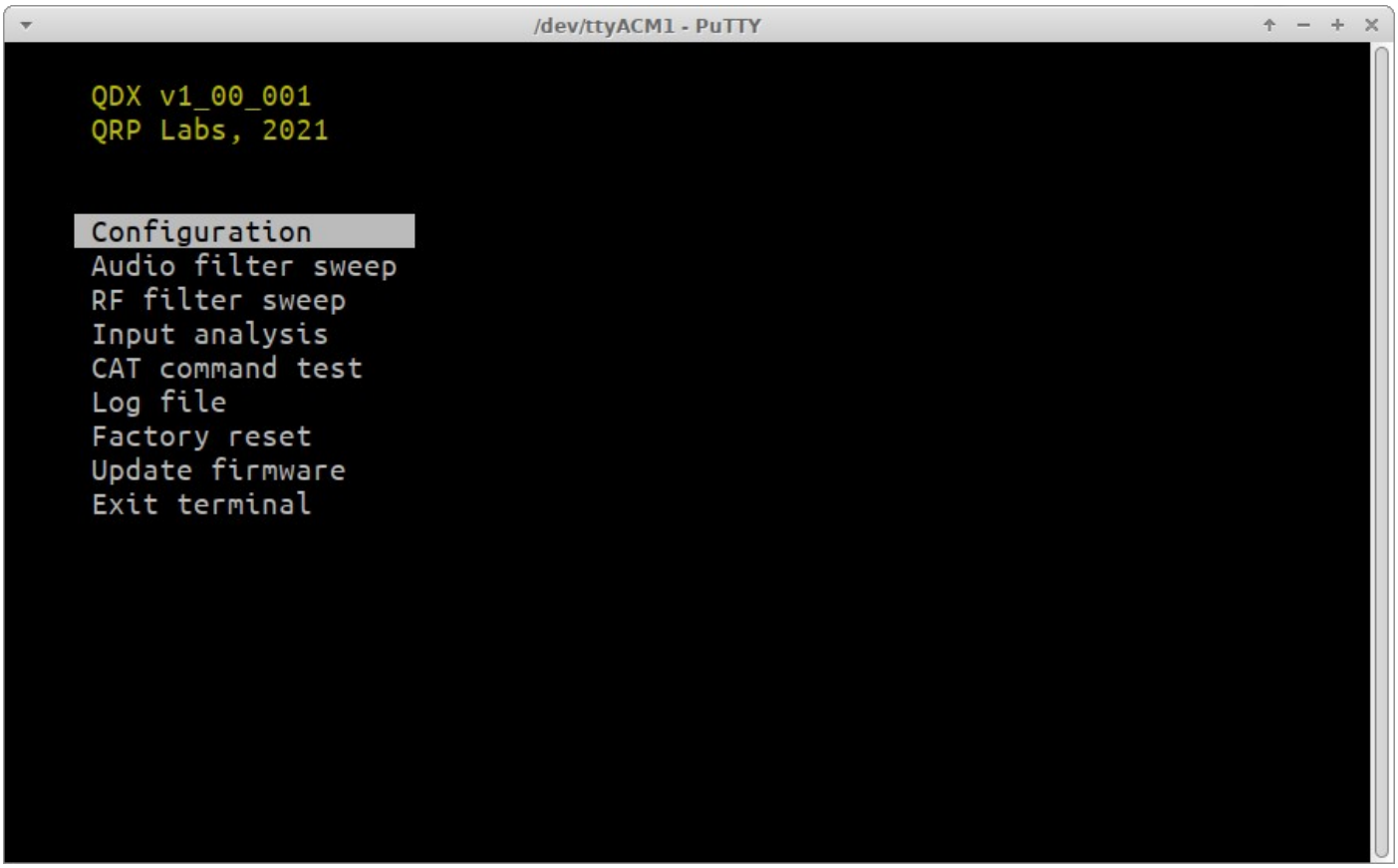
6.2 Entering terminal applications mode

Normally the QDX serial port is connected internally to the QDX's CAT command interpreter. CAT commands include text and numbers, each CAT command is terminated by a semicolon. CAT commands never include a carriage return (enter).

Note that when the terminal is connected, you can actually type on your keyboard to send CAT commands. For example, try typing FA; (just those three characters – no Enter at the end). The text FA00007074000; will appear on the terminal. FA is the command to read or set VFO A, and the result is 7.074 kHz, the default QDX startup frequency. HOWEVER, this is not a very convenient or easy way to try out CAT commands, there is a CAT command testing application which is much easier to use.

To switch to terminal applications mode, simply press the Enter key on your keyboard. Now the terminal applications mode will appear in your terminal emulator window, as shown below. The screen shows the QDX firmware version (coloured yellow at top left), and a list of applications coloured white. You may use the cursor keys to move the highlighted application up and down in the list, and press Enter to select an application. Any application can be quit by pressing Ctrl-Q which returns the terminal to the main menu screen.

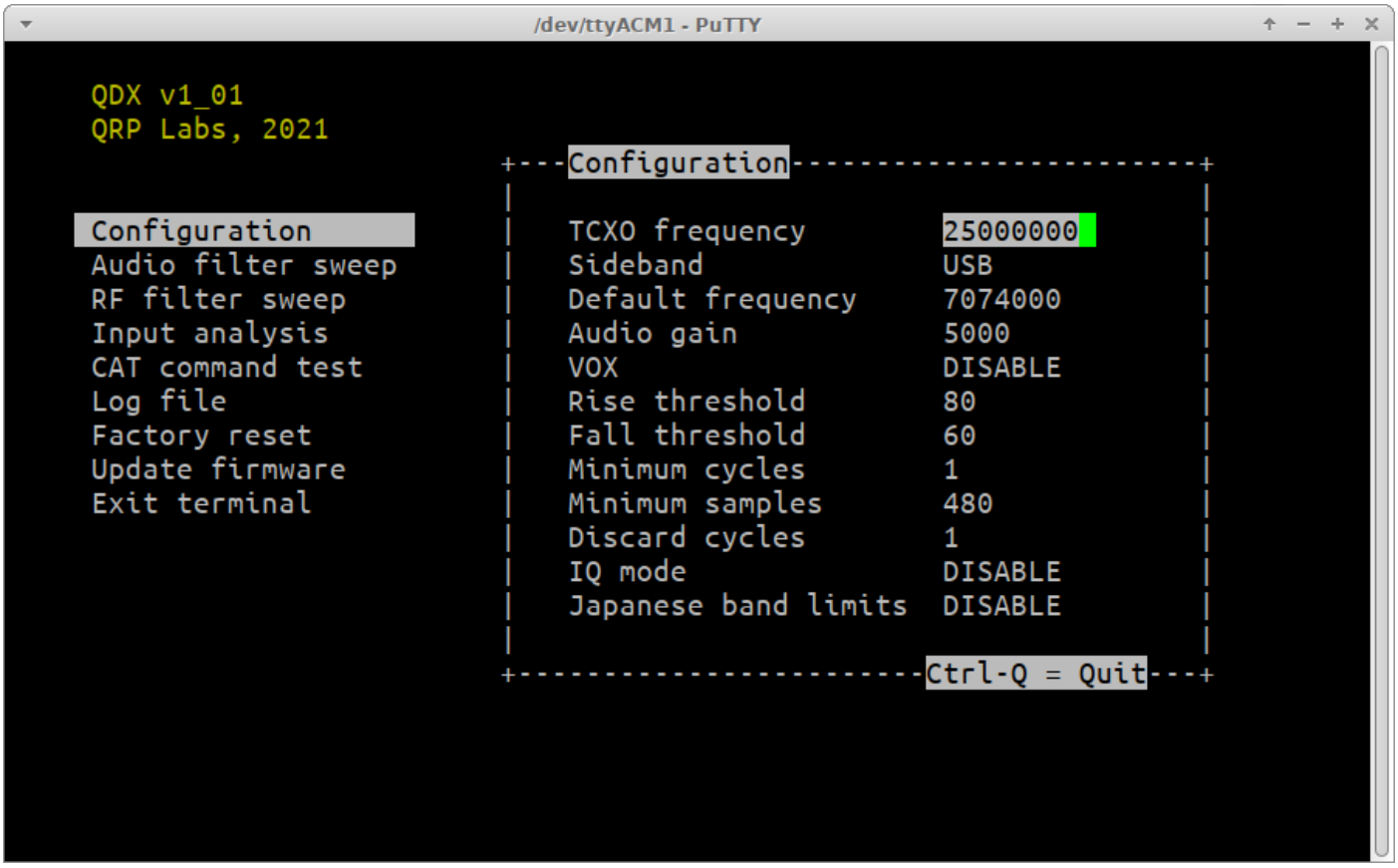
Operation of each of the applications will be described in detail in subsequent sections.



6.3 Exiting terminal applications mode

When exiting terminal applications mode, do not simply close the terminal emulator window. Doing so will leave the QDX in terminal applications mode and it will not accept CAT commands. To switch it back to CAT command mode, use the cursor keys to scroll down to the "Exit terminal" option at the bottom of the list, and press Enter. The screen is now cleared, and QDX is back in CAT command mode. Only then should you close the terminal emulator window.

6.4 Configuration



The configuration screen allows entry of various QDX configuration parameters. The default values are suitable for the vast majority of operating use. The following paragraphs describe each configuration setting in turn. Use the up/down arrows to select the item you wish to edit. The cursor is positioned at the last character of the value. Numeric values have a defined field length. You can press the backspace key to delete the current entry in whole or in part, and type in the new value. For non-numeric configuration parameters, you can use the left and right arrow keys to choose between the available values.

TCXO Frequency

Default is 25000000 (25 MHz). This is the oscillation frequency of the QDX TCXO (Temperature Controlled Crystal Oscillator) and is used for calculating Si5351A parameters for setting the desired QDX operating frequency.

The supplied TCXO is a high precision component and will normally be found to be within (a one standard deviation error of) +/- 5 Hz of the specified 25 MHz value. It is not normally particularly necessary from an operating perspective, to have a more precise operating frequency than this. Remember that the error is also scaled to the operating frequency. So a 5 Hz error at 25 MHz will translate to a 2.8 Hz error at 14 MHz.

However the perfectionists among you may wish to calibrate your operating frequency precisely – and this menu entry is for you!

To configure the correct TCXO reference frequency, you will need to measure your operating frequency, deduce the error amount, and apply a correction to the TCXO frequency configuration parameter.

As an example, suppose your transceiver is set to a USB “Dial Frequency” of 14.0956 MHz and WSJT-X is set up to transmit WSPR at 1500 Hz audio offset. This should result in a transmission frequency of 14.097100 MHz. But let’s suppose that you measure it accurately, and you find that it is 3 Hz high, at 14.097103 MHz. Now what?

There’s an error of +3 Hz in your operating frequency. To work out the required correction to the TCXO reference frequency configuration, calculate 3 Hz multiplied by a ratio of 25 MHz / 14.0971 MHz, which results in 3 Hz x 1.77 = 5.3 Hz.

Therefore you should increase the reference frequency by 5Hz. So move the highlight to the TCXO frequency using the up/down cursor keys, then press the backspace key once, and type 5. That will change the reference frequency to 25000005.

How about if you don’t have an accurate way of measuring your operating frequency? I have developed tools for QRP Labs website to help you to use the WSPRnet reporting network to determine your operating frequency quite accurately. To use these tools, simply use WSJT-X and QDX to operate as a 20m WSPR reporter (receiver) for several minutes, then look at this page:

https://qrp-labs.com/images/wsprnet/rxerror.html

Look for your callsign in the list, which shows the error in your reception reports (operating frequency error).

Alternatively, you may operate as a WSPR transmitter using WSJT-X and QDX, and the following page will show your actual transmitting frequency:

https://qrp-labs.com/images/wsprnet/txfreq.html

Both of these QRP Labs pages are updated every two minutes. The analysis loads the last 5 minutes (approximately) of 20m WSPR reports from the WSPRnet website database. It cross-references all the reports, analyzing the error of receiver stations by cross-referencing against reports of the same transmitters by other stations. In this way calibration errors of all receiving stations in the network are averaged out. The accuracy is generally within 1 or 2 Hz.

Sideband

This setting determines the demodulation sideband. Normally Upper Sideband (USB) is used for all digital modes, and is the default setting. If you wish to use Lower Sideband (LSB) for some reason, you can change it here. Use the left and right arrow keys to change between LSB and USB.

Default frequency

This is the default start-up frequency of QDX at power-on. It defaults to the 40m FT8 frequency 7.074 MHz. If you would like it to power up on a different frequency, you may edit this configuration setting.

Audio gain

According to the WSJT-X documentation, the software operates optimally when the level meter on the bottom left side of the screen is at 30dB when no signals are present. It is not particularly critical and as long as the signal level bar remains green-coloured, the application will function well. However if you find that you need to change the default gain of the QDX, you may do so by editing this parameter. Note that this parameter can also be set and retrieved using the CAT command AG; though the value set and retrieved by AG is scaled by a factor of 256. For example, at the default audio gain of 5000, a CAT command AG; will return the result AG19;

VOX

If you want to use VOX (Voice Operated Transmission) then set this parameter to ENABLE. Any incoming audio from the PC will then operate the Transmit/Receive switch and be transmitted. When the audio stops, QDX will switch back to Receive automatically. The problem with this is that any system sounds on your PC, if the PC is configured to deliver these to the QDX USB sound card, will operate the transmitter and be transmitted.

The default setting “DISABLE” requires a CAT command from the PC host application (WSJT-X for example) in order to enable the transmitter. This is discussed in this manual in the operating manual section, on setting up WSJT-X for QDX.

If you wish to use software that does not support CAT Transmit/Receive switching, this may be one reason why you would want to enable VOX.

Rise threshold

This is a percentage signal level of maximum, above which the transmitter will be keyed down (switched on). Its purpose is to ignore very low amplitude audio signals at the start of a raised cosine keying envelope, whose audio tone could be decoded inaccurately due to quantization error. This is discussed further in the Design section of this manual in the Audio Frequency Analysis section. The default value of 80% should be fine for all purposes. The value should not be set too close to 99%, since higher frequency audio where the number of samples per cycle is small, may not contain a value sufficient to trigger this threshold in every cycle.

Fall threshold

This is a percentage signal level of maximum, below which the transmitter will be keyed up (switched off). Its purpose is to ignore very low amplitude audio signals at the end of a raised cosine keying envelope, whose audio tone could be decoded inaccurately due to quantization error. This is discussed further in the Design section of this manual in the Audio Frequency Analysis section. The default value of 60% should be fine for all purposes. The value should not exceed (or be close to) the Rise threshold parameter, otherwise the transmitter will be repeatedly keyed on and off falsely.

Minimum cycles

This parameter specifies the minimum number of audio cycles to use, in the measurement of audio cycle period, for audio frequency calculation. This parameter is used in conjunction with the Minimum samples parameter: both conditions must be satisfied in order for an audio frequency measurement to be completed. This parameter is discussed further in the Design section of this manual in the Audio Frequency Analysis section. The default value of 1 should be fine for all purposes.

Minimum samples

This parameter specifies the minimum number of audio samples to use, in the measurement of audio cycle period, for audio frequency calculation. This parameter is used in conjunction with the Minimum samples parameter: both conditions must be satisfied in order for an audio frequency measurement to be completed. This parameter is discussed further in the Design section of this manual in the Audio Frequency Analysis section. The default value of 480 should be fine for all purposes.

Bearing in mind that there are 48,000 audio samples per second, a value of 480 specifies a minimum 0.01 second audio measurement period. In other words, there will be 100 measurements of the audio frequency, per second, in this default configuration. This is sufficient to ensure that high audio frequencies are measured accurately. In the unlikely event that frequencies below 100Hz need to be measured, the “Minimum cycles” value (1) will ensure that a longer measurement period is used, to measure one cycle.

Discard cycles

This parameter specifies the number of audio cycles (zero crossings) which are ignored, when audio is first detected. The reason for this parameter is that in conjunction with the “Rise threshold” parameter, it can be seen that the first audio cycle after the threshold is passed, is not a complete cycle. The following zero crossing therefore needs to be discarded because its period measurement will be too short. The default value of 1 should be fine for all purposes.

IQ Mode

When IQ mode is enabled, the raw I and Q channels from the ADC are fed to the USB soundcard directly, without any demodulation. This is suitable for people wishing to experiment with using QDX as an SDR front end, with PC SDR software to demodulate I and Q channels. Remember that there is a 12kHz IF offset applied to the VFO.

IQ Mode is not suitable for use with WSJT-X and other Digi mode programs.

Japanese band limits

When this setting is enabled, QDX will refuse to go into transmit mode if the specified synthesis frequency is outside the Japanese band limits as specified in the JARL bandplans document https://www.jarl.org/English/6_Band_Plan/JapaneseAmateurBandplans20200421.pdf. This setting is useful for Japanese license regulations compliance.

6.5 Audio filter sweep

QDX contains its own internal signal generator, which can be used to sweep the audio passband of the receiver, checking the audio response and the unwanted sideband cancellation.

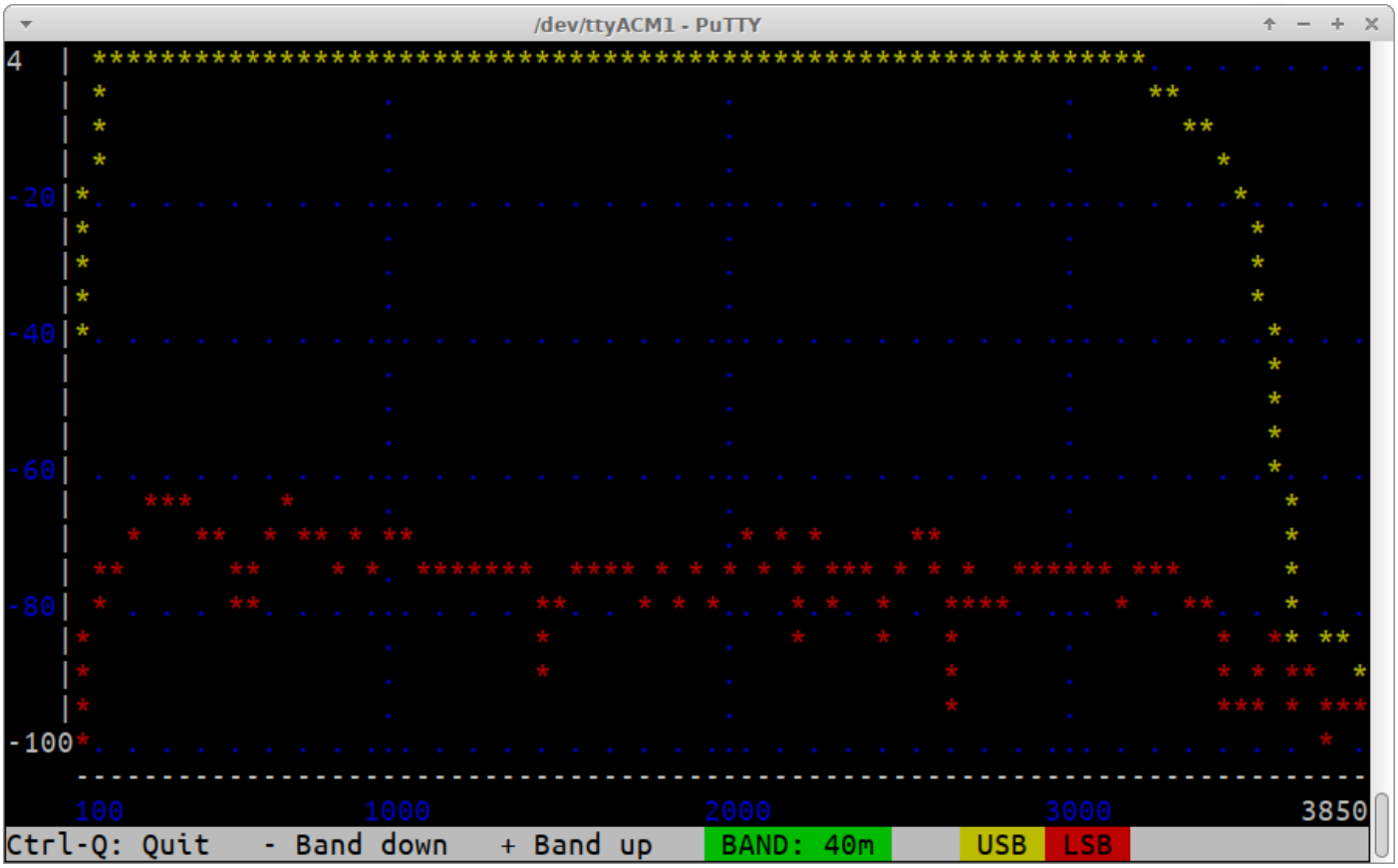
Connect a dummy load for best results.

On entering the application, a sweep automatically commences. The percentage completion of the frequency sweep is indicated in the bottom right corner of the screen.

The sweep starts at 100 Hz (injected signal is 100 Hz higher than the “USB Dial Frequency”, then proceeds in 50 Hz steps up to and including 3850 Hz. The RF frequencies (“USB Dial Frequency”) used for bands 80, 40, 30 and 20m are 3.573, 7.074, 10.136 and 14.074 MHz respectively. The vertical axis shows audio decibel (dB) level. The offset is arbitrary. Gridlines axis are displayed in blue. Vertical gridlines are every 20dB, and horizontal axis gridlines every 1000 Hz. Gridline labels are displayed in blue; the minimum and maximum audio level (in dB) displayed in white, and the maximum audio sweep value also displayed in white.

The chart contains two lines, the line of yellow asterisks is the Upper Sideband (USB) and the red line is Lower Sideband (LSB). In the default demodulation mode (USB), we expect to see the yellow line nice and flat between the filter cut-off points 150 Hz to 3.2 kHz. The red line is the unwanted sideband and we expect to see it around 60dB below the wanted sideband.

The – and + keys can be pressed to move down and up to adjacent bands. Generally the result should not differ much from one band to the next.



6.6 RF filter sweep

QDX contains its own internal signal generator, which can be used to sweep the receiver input Band Pass Filters, checking its response and center frequency. Though the difference in performance is small, the perfectionist may wish to adjust (by squeezing turns) the Band Pass Filter inductor L12 to optimize the center frequencies of the filters.

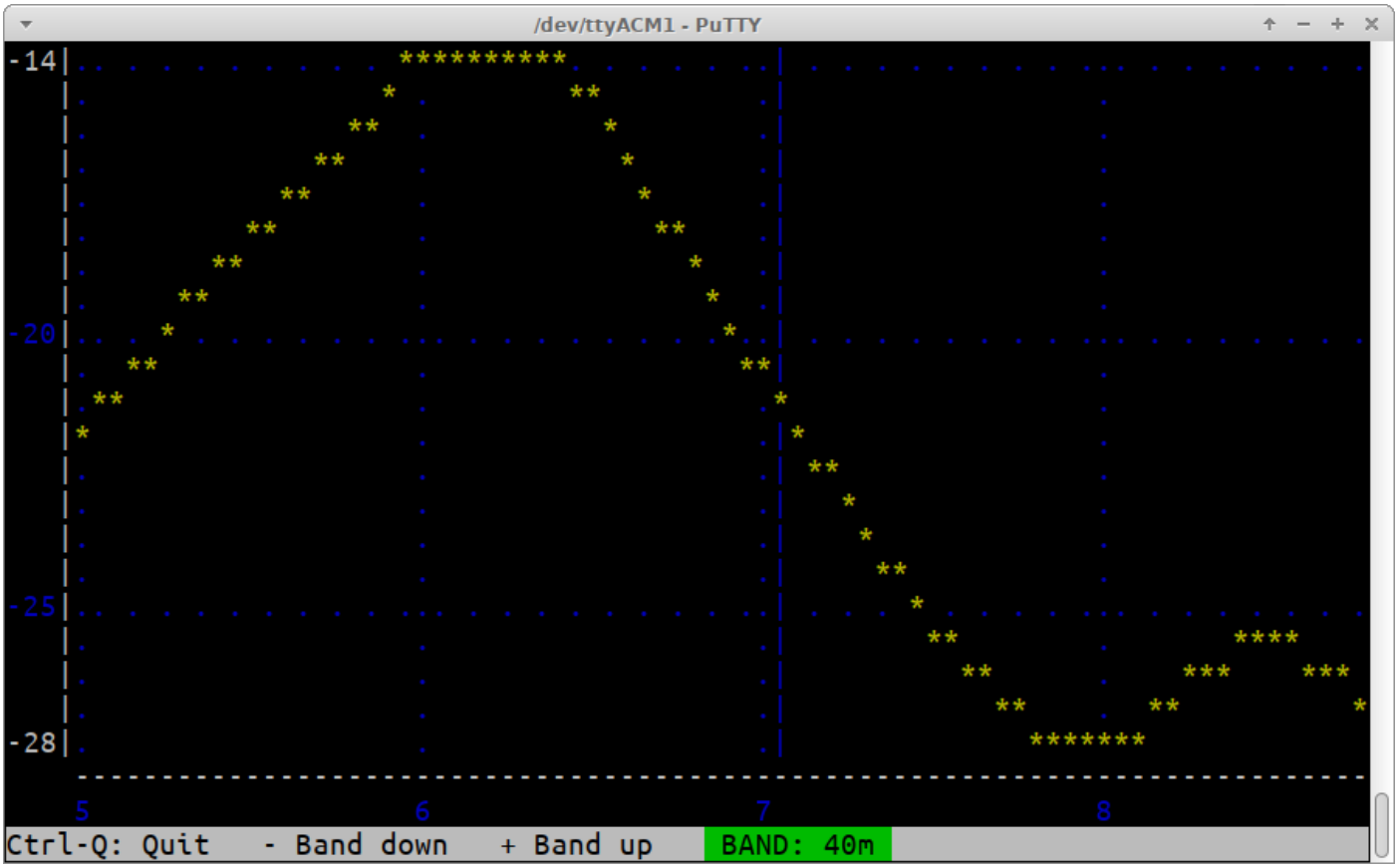
Connect a dummy load for best results.

On entering the application, a sweep automatically commences. The percentage completion of the frequency sweep is indicated in the bottom right corner of the screen.

The center frequency and sweep width are defaulted appropriately per band, to show several MHz of frequency sweep cleanly, and are not adjustable. The vertical axis shows audio decibel (dB) level. The offset is arbitrary. Gridlines axis are displayed in blue. Vertical gridlines are every 20dB, and horizontal axis gridlines as appropriate to the band. Gridline labels are displayed in blue; the minimum and maximum audio level (in dB) displayed in white.

A vertical line is shown at the position corresponding to the default operating frequency in the current band. Ideally the peak of the response (line of yellow asterisks) should coincide with the center frequency of the band. The filters are not very very sharp so the performance degradation by being slightly off frequency is not severe. In the example screenshot (40m), the center frequency is incorrectly somewhat below the 40m operating frequency.

The – and + keys can be pressed to move down and up to adjacent bands.



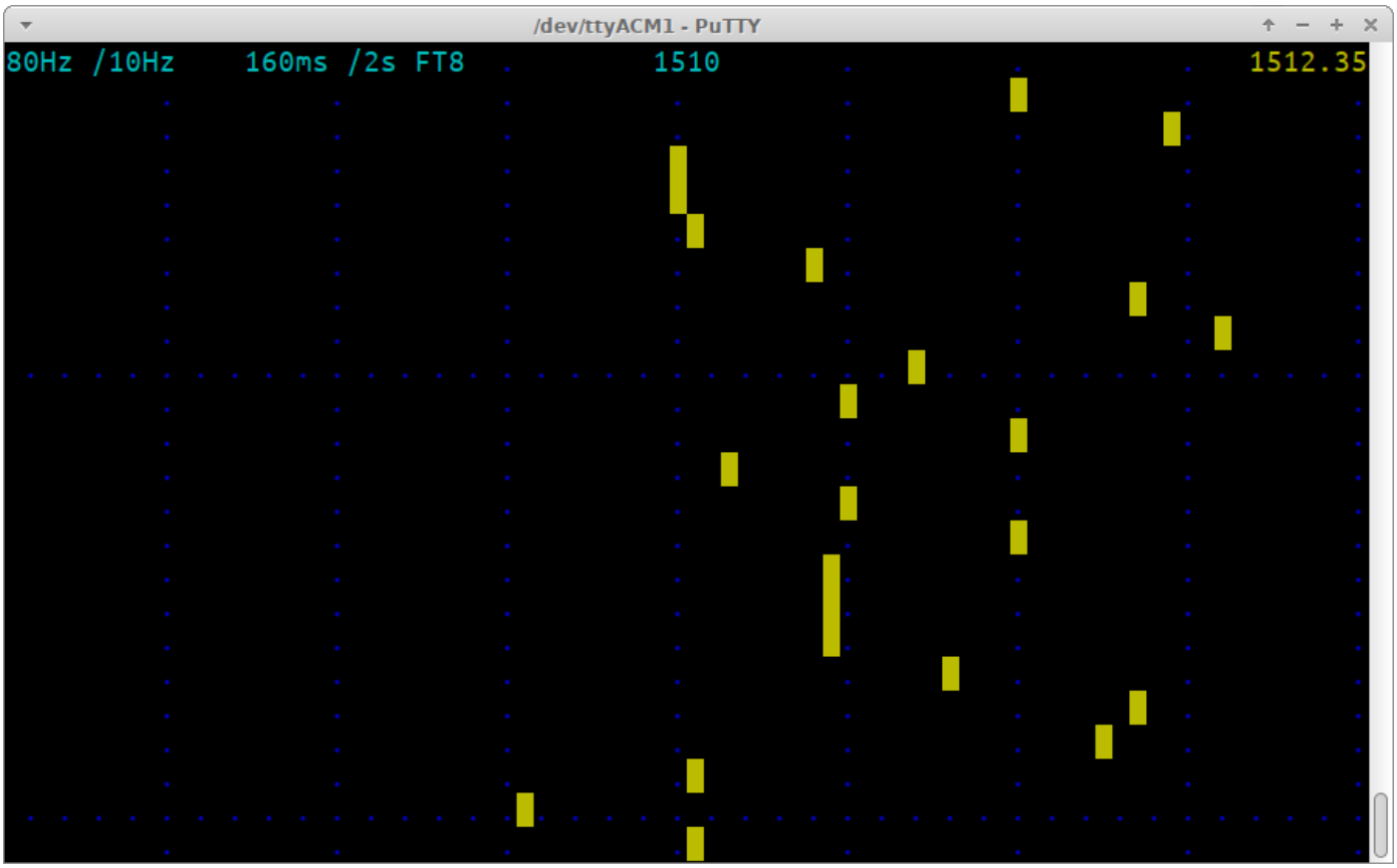
6.7 Input analysis

The Input analysis application is one of the most interesting screens in the QDX Terminal application suite. It shows a “waterfall” which scrolls upward, with the measured audio frequency shown as a yellow block in each row, with its horizontal position determined by its frequency. The scroll rate, and the screen width, are adjustable. The application can be used for checking the accuracy of the frequency analysis (for example when changing configuration parameters “Minimum cycles” and “Minimum samples” at various incoming audio frequencies).

Keyboard controls

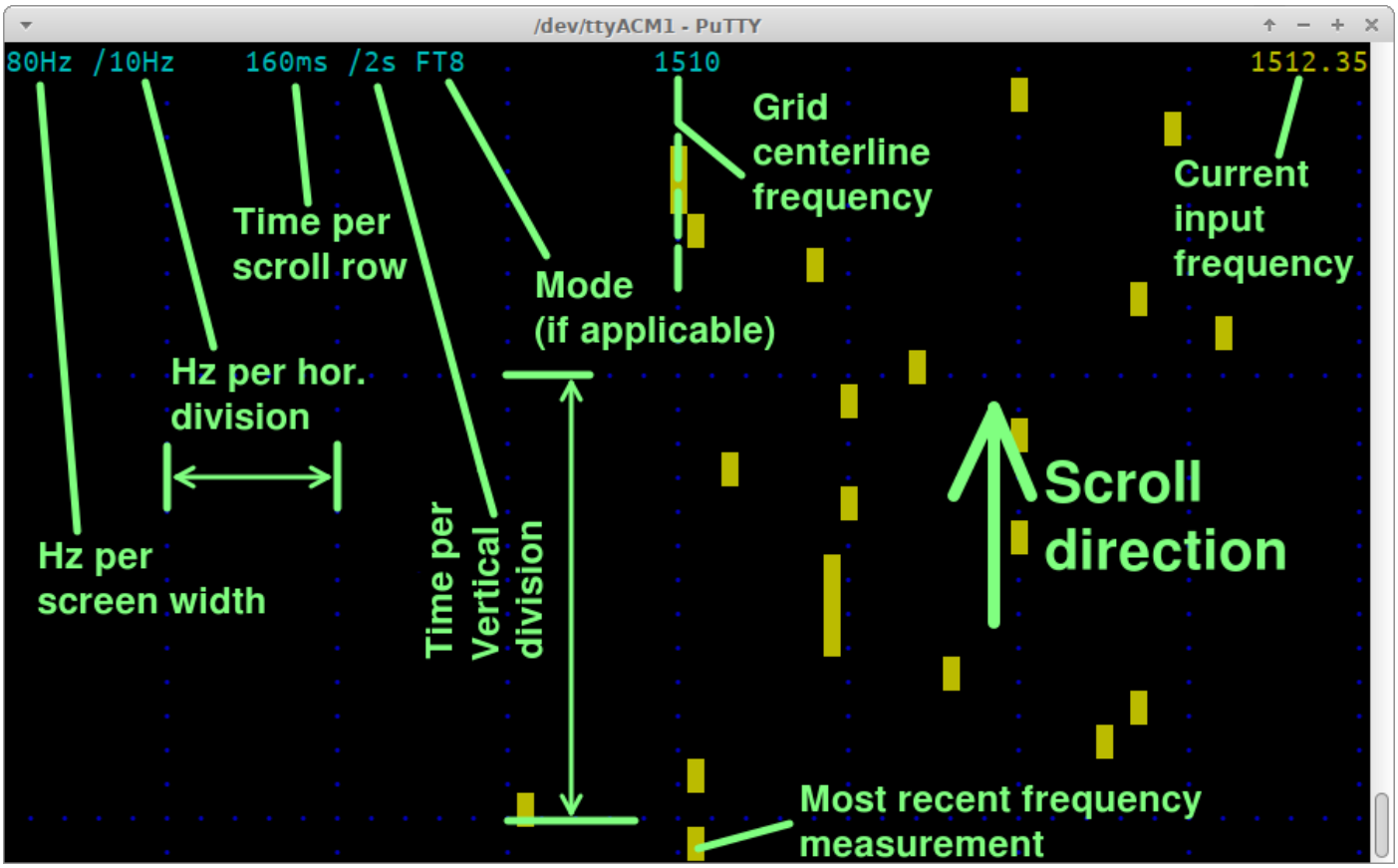
During operation, the following keyboard buttons may be pressed to control the operation of the waterfall:

CTRL-Q:	Quit the input analysis application
Right arrow:	Increase the screen width (higher number of Hz per horizontal division)
Left arrow:	Decrease the screen width (lower number of Hz per horizontal division)
Up arrow:	Scroll faster (shorter time interval per vertical division)
Down arrow:	Scroll slower (longer time interval per vertical division)
Space bar:	Pause or un-pause the display; when paused the top right corner text shows “PAUSED” in inverted yellow text
. (dot):	Change the colour of the background gridlines. The default colour is blue. Each time the dot is pressed, the colour changes; it cycles through: Blue, Magenta, Cyan, White, Red, Green, Yellow and then blue again.



Display elements

This annotated screenshot illustrates the various elements on the display.



The display elements on the top row are:

- **80Hz:** Total 80-column screen width, in Hz. In this case an 80 Hz wide screen means each block is a 1 Hz Waterfall bucket
- **/10Hz:** Hz per horizontal division. There are always 8 horizontal divisions on the screen. This parameter displays the number of Hz per division.
- **160ms:** the update rate of the screen, or put another way, time per horizontal scroll row.
- **/2s:** The time per vertical division of the screen (distance between the horizontal blue dot grid lines)
- **FT8:** The mode matching this update rate; various popular modes coded and when an update rate matches a mode, the mode name is written here. See below for the list of supported update rates.
- **1510:** Center frequency of the waterfall
- **1512.35:** The most recent frequency measurement
  NO SIGNAL: displayed at the top right when no audio signal is detected
  PAUSED: displayed at the top right when the screen is paused by the space bar

NOTE: At very fast update rates, which are useful for displaying some features, the screen update is too slow to display all the information and still be capable of updating fast enough. Therefore information is dropped. At 20ms per row (50 row updates per second) the only text on the display is “20ms” in the top left corner; the grid lines are still displayed. At 10ms (100 row updates per

second), even the grid lines are dropped; only “10ms” is written in the top left corner, and the yellow block drawn for the measured frequency.

Update rates:

The available update rates (selected by the up/down arrow keys) are chosen to match various popular modes. For example, the symbol duration for FT8 is 160 ms. So “FT8” is written next to the vertical division rate, on the top row of the display.

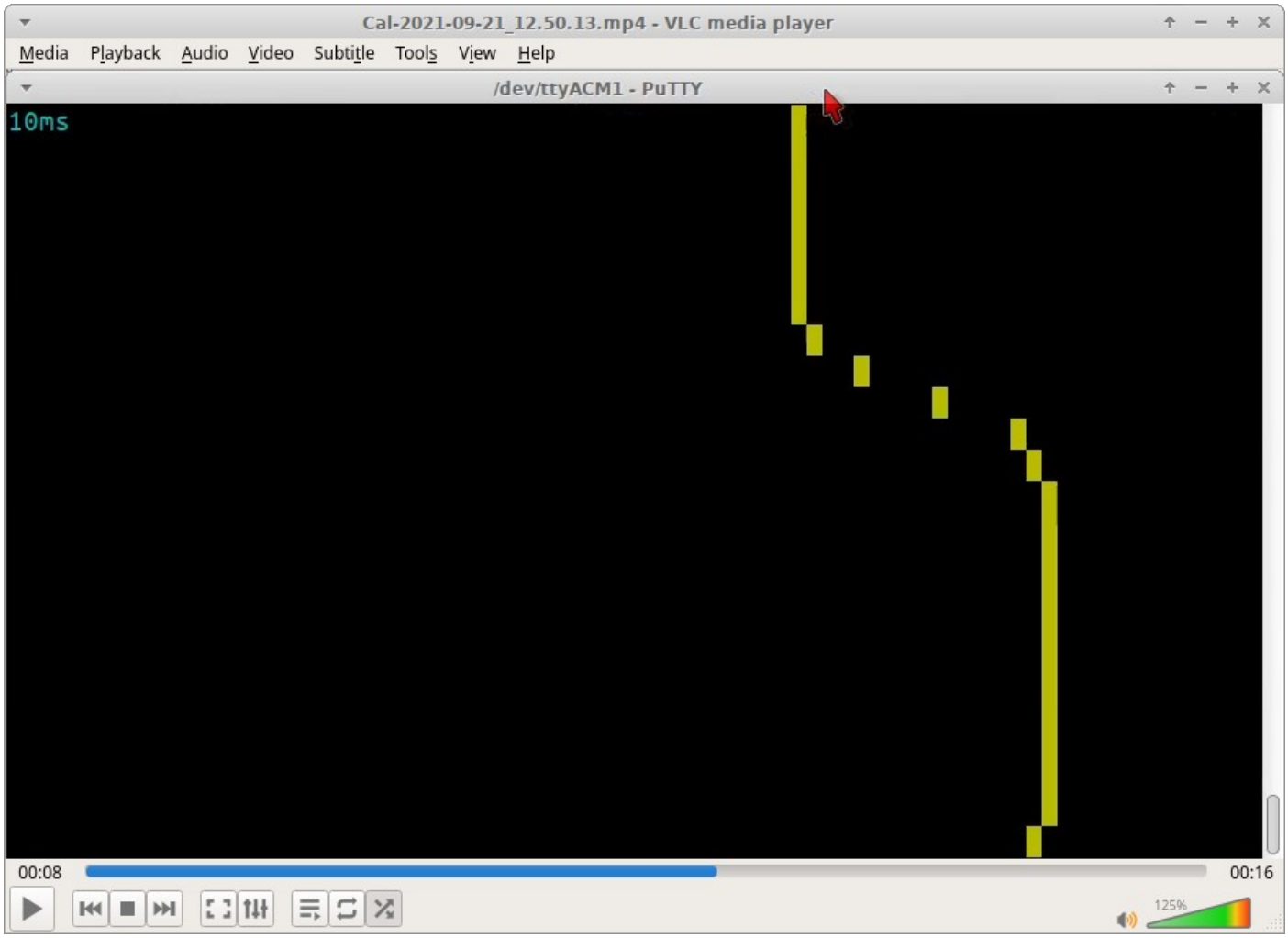
Update rate (ms)	vertical Div (s)	Matching mode	Comment
10	1		Only display “10ms” in top left
20	1		Only display gridlines and “20ms” in top left
40	1		
80	1		
160	2	FT8	
227	2	JT4	
361	5	JT65	
576	5	JT9	
683	5	WSPR	

Horizontal size (Hz):

The available horizontal screen size (selected by the left/right arrow keys) are chosen to cleanly fit the 80 columns display width.

Screen width (Hz)	Horizontal Div (Hz)
12	1.5
20	2.5
40	0.5
80	1
160	2
320	4

Demonstration of sliding frequency changes



A fun and useful demonstration of the capabilities of QDX and the Input Analysis application, is to investigate the FT8 tone change transition. We know according to the WSJT-X documentation that to avoid splatter onto adjacent frequencies, WSJT-X makes a smooth transition between one tone symbol and the next. It's nice to be able to check this using the Input Analysis screen.

Here the screen width is 80Hz so each horizontal column is a 1 Hz bucket. The FT8 symbol change event is 2 tones (a frequency change of 12.5 Hz since the FT8 tone spacing is 6.25 Hz. The scroll rate is sped up the fastest rate, 10 ms per horizontal row (100 frequency measurements per second).

We see clearly that at the moment the FT8 tone changes, the measured frequency smoothly changes in the expected way from one tone to the next.

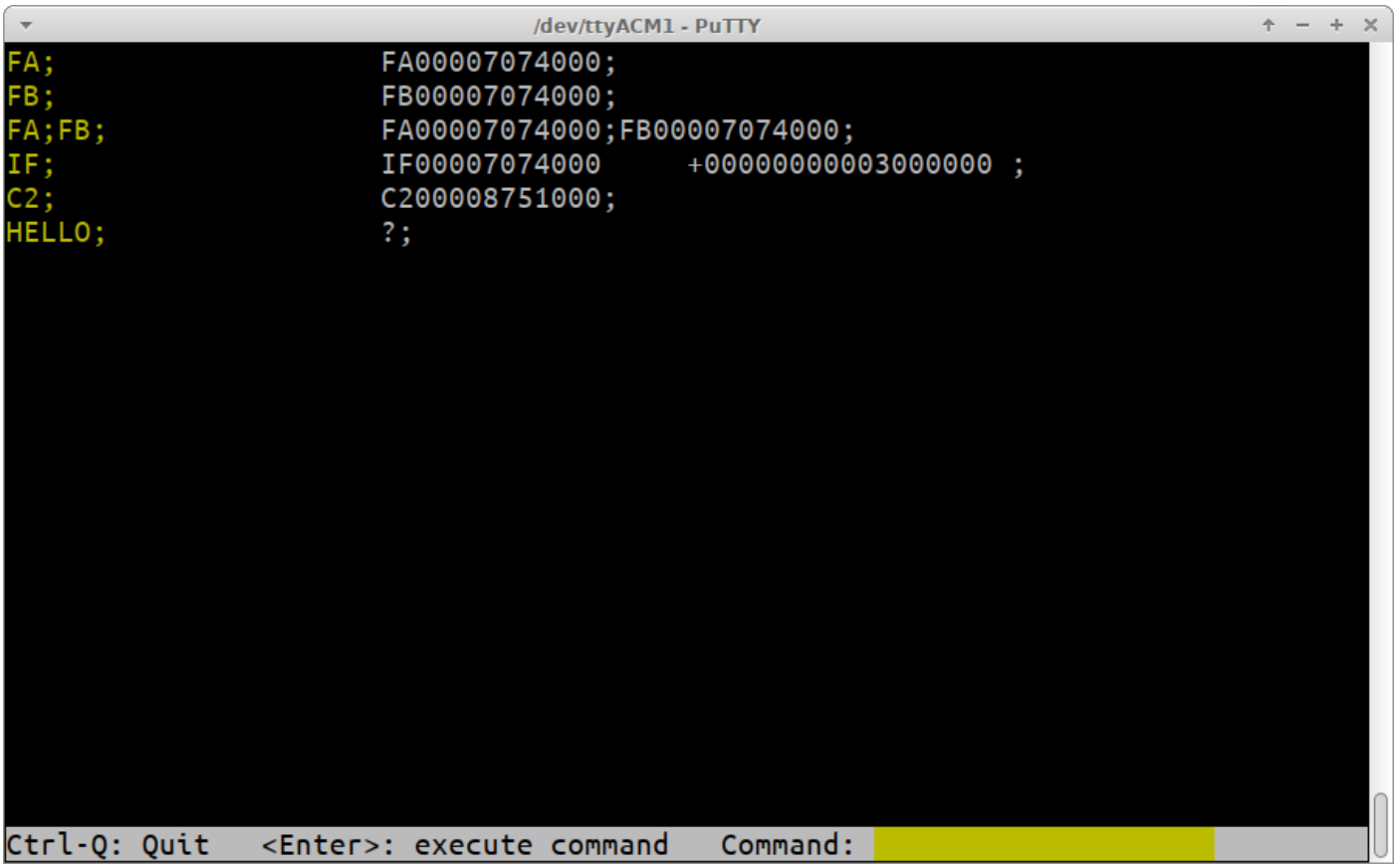
Note that an even smoother transition could be made, by setting the "Minimum samples" parameter to 240 samples – this would result in slightly lower accuracy measurement but a smoother transition in tones.

We're talking here about tradeoffs in different aspects of performance; either way the performance is excellent and QDX keeps up with the changes in input frequency! So this is a nice confirmation of the excellent performance characteristics of QDX.

6.8 CAT command test

The CAT command test application is a simple screen which allows you to test CAT commands. An annoying thing about typing CAT commands directly into a terminal emulator when QDX is in the normal operating mode, is that you cannot see the command you are typing, nor can you edit it if you made mistakes.

Here in this simple application, you can type the command (or commands) you wish to send to QDX in the yellow area of the bottom row. If you forget the trailing semicolon, the application adds one automatically for you. Then simply the command is shown in yellow in the left column, and the command result in yellow in the right column. When the screen is full, it will scroll automatically. Naturally unrecognized CAT commands simply return an error ? as per the CAT specification.



Cat command set

QDX implements a subset of the Kenwood TS-480/TS-440 CAT command set which is an old standard, but contains all the necessary commands for WSJT-X and other software to be able to control QDX, and is old enough that it is widely supported by most software packages.

Additionally QDX provides an extended set of CAT commands Q0, Q1, Q2 and so on until QB. These commands allow a host program to set and get all of the parameters in the QDX configuration screen. Except for the default start-up operating frequency, all of these commands have effect only for the current operating session, they are not stored in EEPROM. If you need to permanently store the configuration parameters in EEPROM please use the Terminal Configuration screen. These commands could be useful for example, if you wish to provide a particular start-up script of CAT commands in your PC software to configure your QDX a certain way.

The following lists the commands and responses in alphabetical order:

AG: Get/Set AF Gain

Set: Sets the audio gain. The actual gain setting is the supplied gain multiplied by 256. So for example command AG21; will set the gain to 21 x 256 = 5376. Note that the gain can also be set in the Terminal Configuration application.

Get: Returns the audio gain, divided by 256. The default audio gain in QDX is 5,000 so the returned value is 19.

C2: Get/Set Signal Generator frequency

Set: Sets the Si5351A Clk2 output frequency (the signal generator).

Get: Gets the frequency of Si5351A Clk2 (the signal generator)

FA: Get/Set VFO A

Set: Sets VFO A value. Example: FA7030000; sets VFO A to 7.030MHz

Get: Returns the VFO A contents as an 11-digit number. Example: “FA;” returns “FA00007030000;”

FB: Get/Set VFO B

Set: Sets VFO B value. Example: FB7016000; sets VFO B to 7.016MHz

Get: Returns the VFO B contents as an 11-digit number. Example: “FB;” returns “FA00007016000;”

FR: Get/Set Receive VFO Mode

Set: Set VFO Mode: 0, 1, 2 correspond to VFO A, VFO B or Split respectively. This is the case for both the FR and FT commands (which are nominally Receive and Transmit VFOs) because in the QCX+ the VFO mode use does not correspond exactly to TS-480.

Get: Get Receive VFO Mode: 0 means VFO A is used for receive (could be due to VFO mode being VFO A, or VFO Mode being Split); 1 means VFO B is being used for receive (must be VFO Mode B).

FT: Get/Set Transmit VFO Mode

Set: Set VFO Mode: 0, 1, 2 correspond to VFO A, VFO B or Split respectively. This is the case for both the FR and FT commands (which are nominally Receive and Transmit VFOs) because in the QCX+ the VFO mode use does not correspond exactly to TS-480.

Get: Get Transmit VFO Mode: 0 means VFO A is used for transmit (must be VFO Mode A); 1 means VFO B is being used for transmit (could be due to VFO mode being VFO B, or VFO Mode being Split)

FW: Get filter bandwidth

Get: Always returns 3200 meaning, 3200Hz (the QDX’s filter bandwidth).

ID: Get radio ID

Get: Always returns 020 (Kenwood TS-480)

IF: Get transceiver information (TS-480 format).

Get: Returns a composite information string containing the state of the transceiver, as follows (excluding command ID and ; terminator character):

- 11-digit operating frequency (VFO A or B, according to the VFO mode setting and transmit/ receive state
- 5 spaces
- 5-digit RIT frequency, as +/-9999Hz e.g. RIT up 200Hz returns “+0200” in this field
- RIT status: 0 = RIT OFF, 1 = RIT ON
- XIT status: always 0 because QCX+ has no XIT
- Memory channel bank number: always 0
- Memory channel number: always 00
- Transceiver status: 0 = RX, 1 = TX
- Operating mode: always 3 (CW)
- Receive VFO: 0 = VFO A, 1 = VFO B
- Scan status: always 0
- Split: 0 = Simplex operation (VFO mode A or VFO mode B), 1 = Split
- Tone: always 0
- Tone number: always 0
- Space character

MD: Get/Set operating mode

Set: Set to 1 to set Lower Sideband (LSB) or any other value to set Upper Sideband (USB)

Get: Returns 3 (CW) if the mode is Upper Sideband (USB) which is the normal mode. Returns 1 if the mode is Lower Sideband (LSB).

Q0: Get/Set TCXO reference frequency

Set: Set TCXO reference frequency. Only values between 24999000 and 25001000 are allowed. The setting is for the current operating session only and is not written to EEPROM.

Get: Get TCXO reference frequency.

Q1: Get/Set Sideband

Set: Set sideband; 1 sets Lower Sideband (LSB), any other value sets Upper Sideband (USB). The setting is for the current operating session only and is not written to EEPROM.

Get: Get sideband; 0 = USB, 1 = LSB

Q2: Get/Set default operating frequency

Set: Set default operating frequency. The value IS written to EEPROM.

Get: Get default operating frequency.

Q3: Get/Set receiver Gain

Set: Set receiver gain. The setting is for the current operating session only and is not written to EEPROM.

Get: Get receiver gain. The default value on Factory Reset is 5000.

Q4: Get/Set VOX Enable

Set: Set VOX enable: 1 enables VOX, 0 disables VOX. The setting is for the current operating session only and is not written to EEPROM.

Get: Get VOX enable; 1 = Enabled, 0 = Disabled

Q5: Get/Set TX Rise Threshold

Set: Set TX Rise Threshold, which should be a percentage number between 0 and 99 (see description elsewhere in this manual). The default setting of 80 is normally appropriate. The setting is for the current operating session only and is not written to EEPROM.

Get: Get TX Rise Threshold value

Q6: Get/Set TX Fall Threshold

Set: Set TX Fall Threshold, which should be a percentage number between 0 and 99 (see description elsewhere in this manual). The default setting of 60 is normally appropriate. The setting is for the current operating session only and is not written to EEPROM.

Get: Get TX Fall Threshold value

Q7: Get/Set Cycle Min parameter

Set: Set Cycle Min parameter (see description elsewhere in this manual). The setting is for the current operating session only and is not written to EEPROM.

Get: Get Cycle Min parameter value

Q8: Get/Set Sample Min parameter

Set: Set Sample Min parameter (see description elsewhere in this manual). The setting is for the current operating session only and is not written to EEPROM.

Get: Get Sample Min parameter value

Q9: Get/Set Discard parameter

Set: Set Discard parameter (see description elsewhere in this manual). The setting is for the current operating session only and is not written to EEPROM.

Get: Get Discard parameter value

QA: Get/Set IQ Mode

Set: Set IQ Mode: 1 enables IQ Mode, 0 disables IQ Mode. In IQ Mode, the raw I & Q samples from the ADC are streamed directly to the USB Sound card. The setting is for the current operating session only and is not written to EEPROM.

Get: Get IQ Mode; 1 = Enabled, 0 = Disabled

QB: Get/Set Japanese Band Limits mode

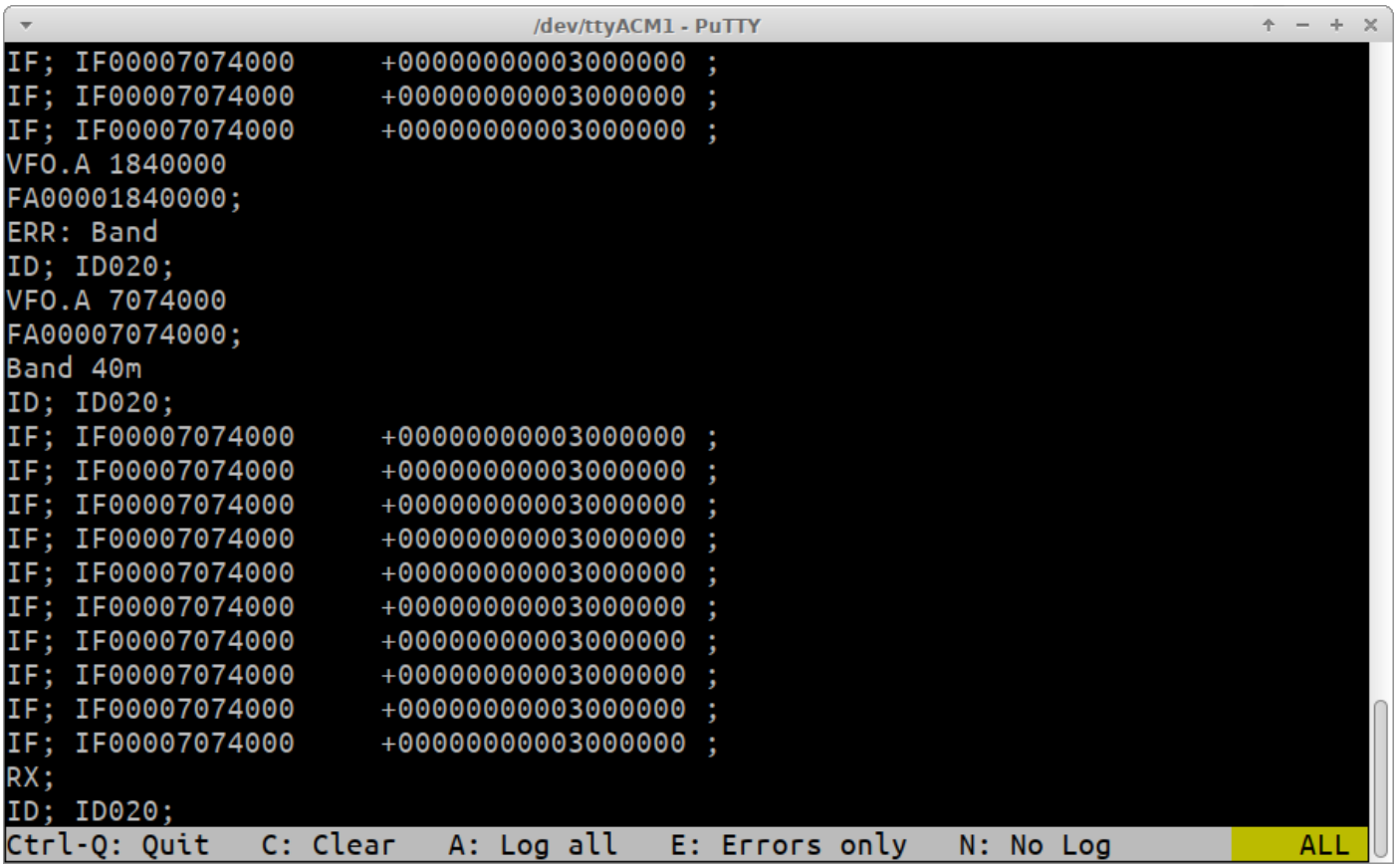
Set: Set Japanese Band Limits mode: 1 enables Japanese band limits, 0 disables Japanese band limits. In Japanese Band Limits mode, QDX cannot transmit outside the allowed Japanese band limits. The setting is for the current operating session only and is not written to EEPROM.

Get: Get Japanese Band Limits mode; 1 = Enabled, 0 = Disabled

RD: Set negative RIT offset amount
Set: Sets negative (down) RIT; for example “RD-200;” sets the RIT to -200Hz
RT: Get RIT status
Get: Returns RIT status: 0 = RIT off, 1 = RIT on
RU: Set positive RIT offset amount
Set: Sets positive (up) RIT; for example “RD150;” sets the RIT to 150Hz
RX: Set the radio into Receive mode immediately
Set: Command RX; immediately puts the radio into receive mode. It is equivalent to TQ0;
SP: Get/Set Split mode
Set: Sets Split mode: 0 = OFF, 1 = ON. For example “SP1;” switches QDX to split mode
Get: Returns the Split state: 0 = OFF, 1 = ON.
TQ: Get/Set transmit state
Set: Sets transmit state: 0 = RX, 1 = TX. For example “TQ1;” switches QDX to transmit mode
Get: Returns the transmit state: 0 = RX, 1 = TX.
TX: Set the radio into Transmit mode immediately
Set: Command TX; immediately puts the radio into transmit mode. It is equivalent to TQ1;

6.9 Log file

This application is very useful for debugging CAT problems. The log file is stored in the EEPROM chip. The EEPROM chip capacity is 8 KB (Kilobytes) or 8,192 characters. However the configuration parameters also reside in EEPROM, so the available space for the log file is a little less: 8,064 characters, to be precise! When the log file is full, no further characters are written to it.



On opening the Log file application, the existing contents of the log file are displayed. When the screen is full, it will scroll automatically; you can use the scroll bar of the window to see text that has scrolled off the top of the screen if you wish. The current logging status is displayed in yellow in the bottom right corner. In this example, the status is “ALL” which means that all events are being logged.

The available keyboard commands are:

CTRL-Q:	Quit the log file application screen (logging remains active, if enabled)
C:	Clear the log file
A:	Log all events
E:	Log only error events
N:	No logging at all (logging is switched off)

In the log file, each event is written as one line. There are several times of log file entry:

- Errors have the line prefix “ERR:”
- CAT commands are shown (semicolon terminated) followed by the command result, if any. An unrecognized or invalid CAT command would be prefixed by the error code “ERR: “.
- Radio state changes are indicated by the item being changed, followed by the new value.

Various examples from the screenshot above:

VFO.A 1840000	sets the VFO A frequency to 1.84 MHz
FA00001840000;	CAT command from WSJT-X MHz
ERR: Band	Error message “Band”

In this example, a CAT command was received from WSJT-X, to set the QDX to frequency 1.84 MHz. The CAT command is “FA00001840000;” - Note that the entries in the log file are not quite in chronological order, since the VFO update event is printed BEFORE the CAT command event; the reason for this is that the CAT evaluation must be completed in totality before the CAT command log event takes place, because only then do we know if the CAT command was successful. The final entry is an error message “Band” because 1.84 MHz is in the 160m band and this is not one of the bands currently supported by QDX.

VFO.A 7074000	sets the VFO A frequency to 7.074 MHz
FA00007074000;	CAT command from WSJT-X MHz
Band 40m	Sets the band to 40m

The operator then realizes his error setting QDX to 160m in the WSJT-X drop-down and chooses 40m instead; this time the CAT command successfully sets frequency and band.

IF; IF00007074000 +00000000003000000 ;

Here the “IF;” or Information request CAT command is received from WSJT-X and QDX replies with the CAT information string in the prescribed format.

For reasons known only to itself, WSJT-X likes to ask for the information screen multiple times in quick succession even though the result is the same.

The complete list of log events is as follows:

Event	Explanation	Example	Comment
Power up	Occurs every power up	Start 1_00_001	Shows the start event and version number
Set VFO	VFO frequency change	VFO.A 7074000	VFO A is set to 7.074 MHz. VFO B is also available
Band change	Band is set to a new value	Band 40m	An error is generated for bands not supported.
TX	Radio switched to Transmit	TX	Can be either by CAT command or by VOX
RX	Radio switched to Receive	RX	Can be either by CAT command or by VOX
VFO Mode	VFO mode set to A, B, or Split	VFO Mode A	An error is generated for any attempt other than A, B or Split
Split ON/ OFF	CAT Split on/off command SP	Split ON	
Clk2	Signal generator frequency is set	Clk2 7075000	
Gain	Set audio gain	Gain 25	Audio gain is set by the AG; CAT command
RIT Down	RIT down CAT command	RIT DN 500	
RIT Up	RIT up CAT command	RIT UP 500	
CAT	Every CAT command	IF;	Every CAT command generates a log entry event

6.10 Factory reset

Selecting Factory reset at the main terminal menu, then answering the question “Are you sure?” with “Y”, results in the EEPROM configuration resetting to the factory default state, including restoring the TCXO reference frequency to 25000000 and erasing the log file (if any). Pressing any other key cancels the factory reset.

6.11 Update firmware

Puts QDX into the QFU Bootloader mode, for firmware update. Please refer to the section of this manual describing the firmware update procedure.

6.12 Exit terminal

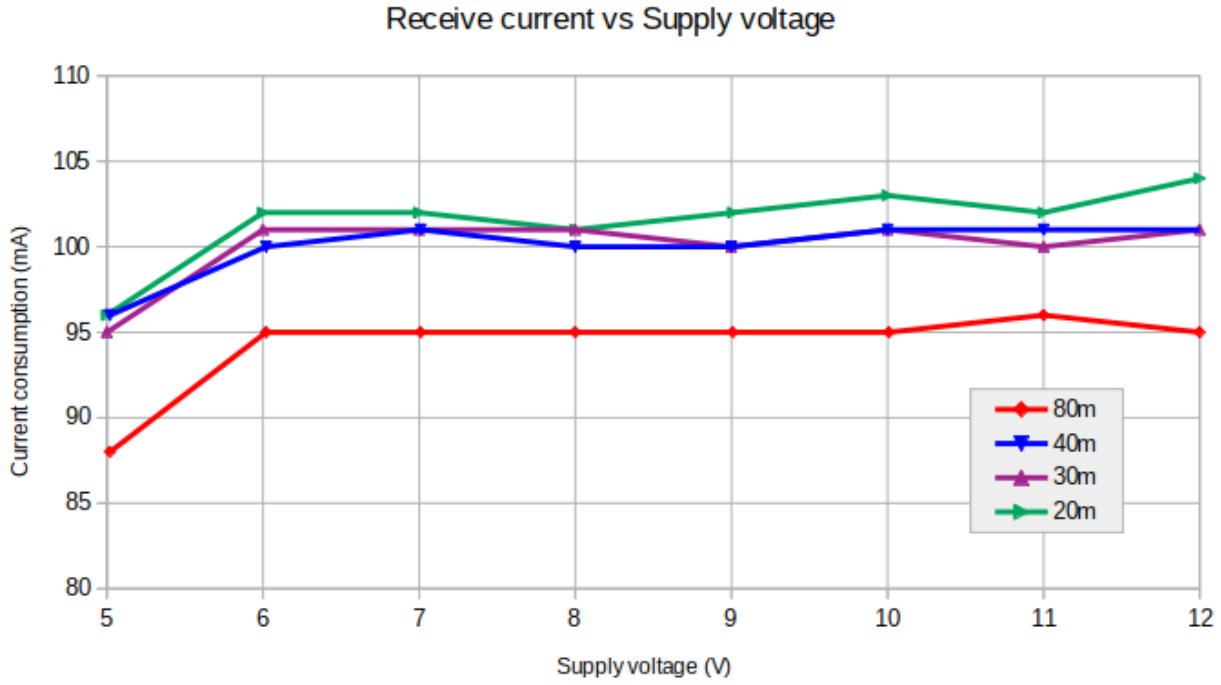
As mentioned previously, exit terminal returns QDX to normal operating mode, processing incoming CAT commands as usual. The terminal should also be disconnected (or closed) so as to free up the serial port for use by WSJT-X or whatever software you are using – remember that only one PC software application can use the serial port at a time.

7 Performance measurements

Several QDX units have been measured; there are of course variations from one to another, due to component tolerances, toroid winding style, etc. The results presented here are for a sample QDX; all measured QDX were reasonably similar.

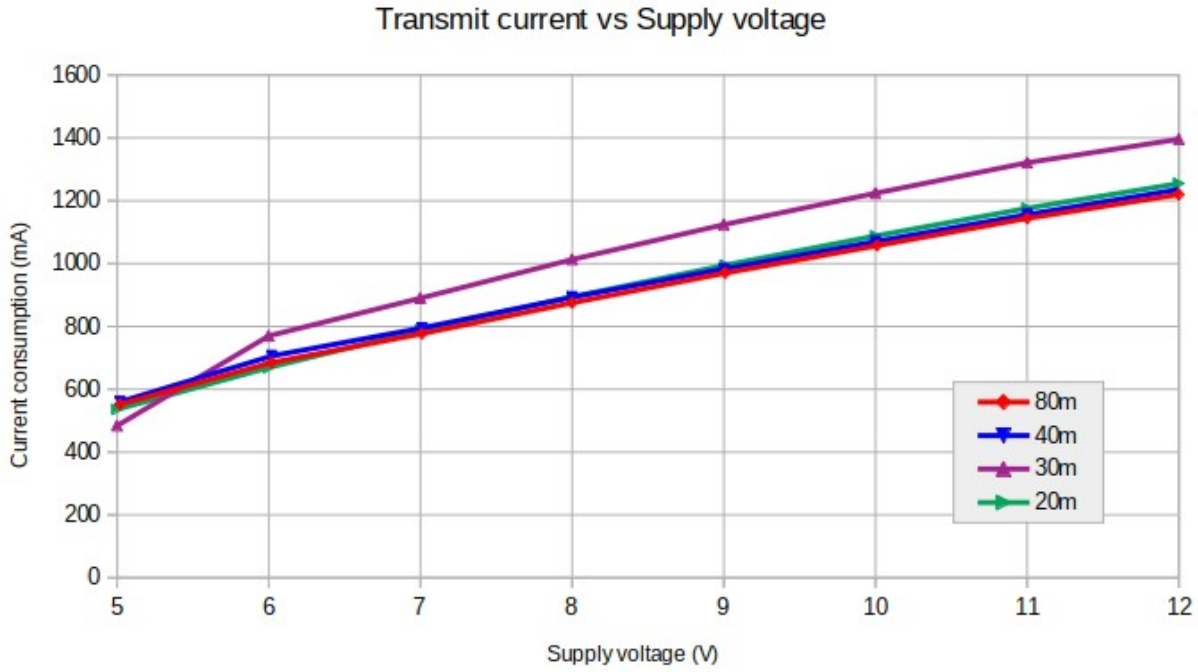
7.1 Receive current consumption

The following chart shows receive current vs supply voltage, per band. There is not much variation and it is fair to quote 100mA as the receive current.



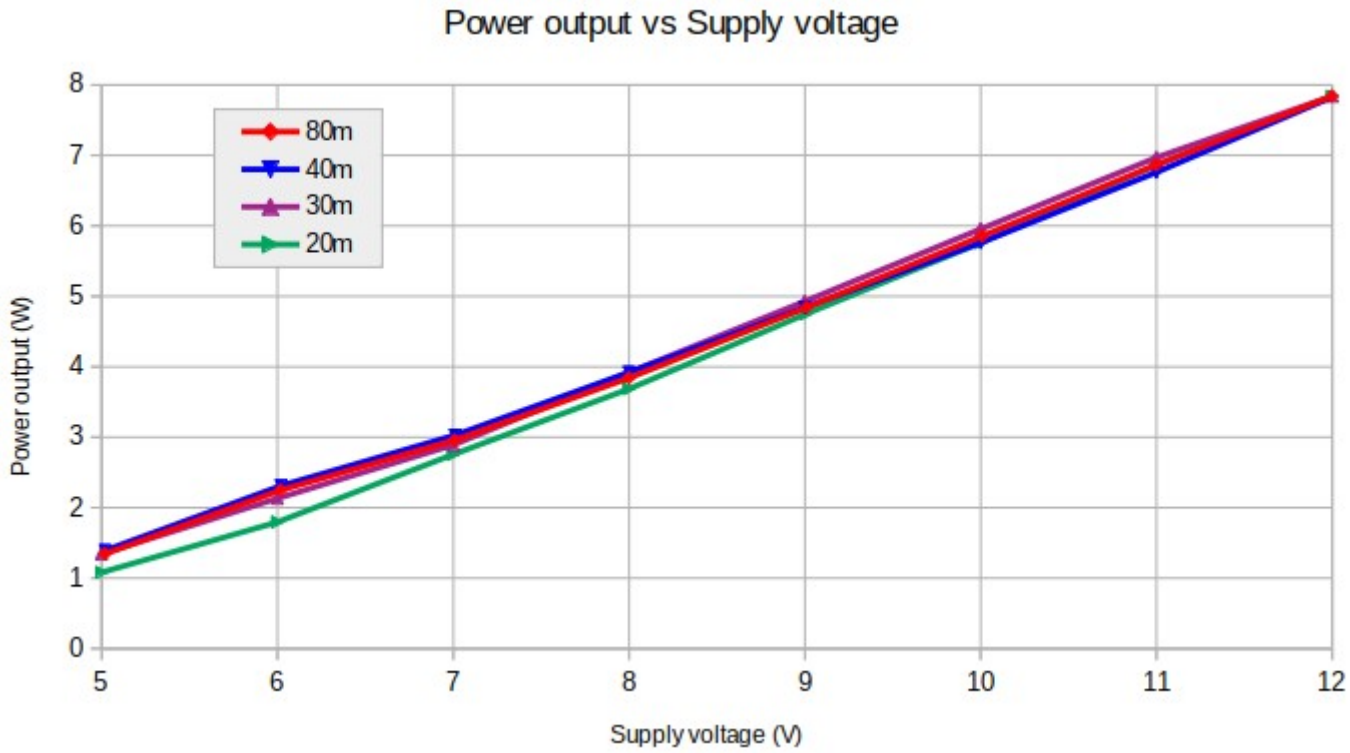
7.2 Transmit current consumption

The following chart shows transmit current vs supply voltage, per band.

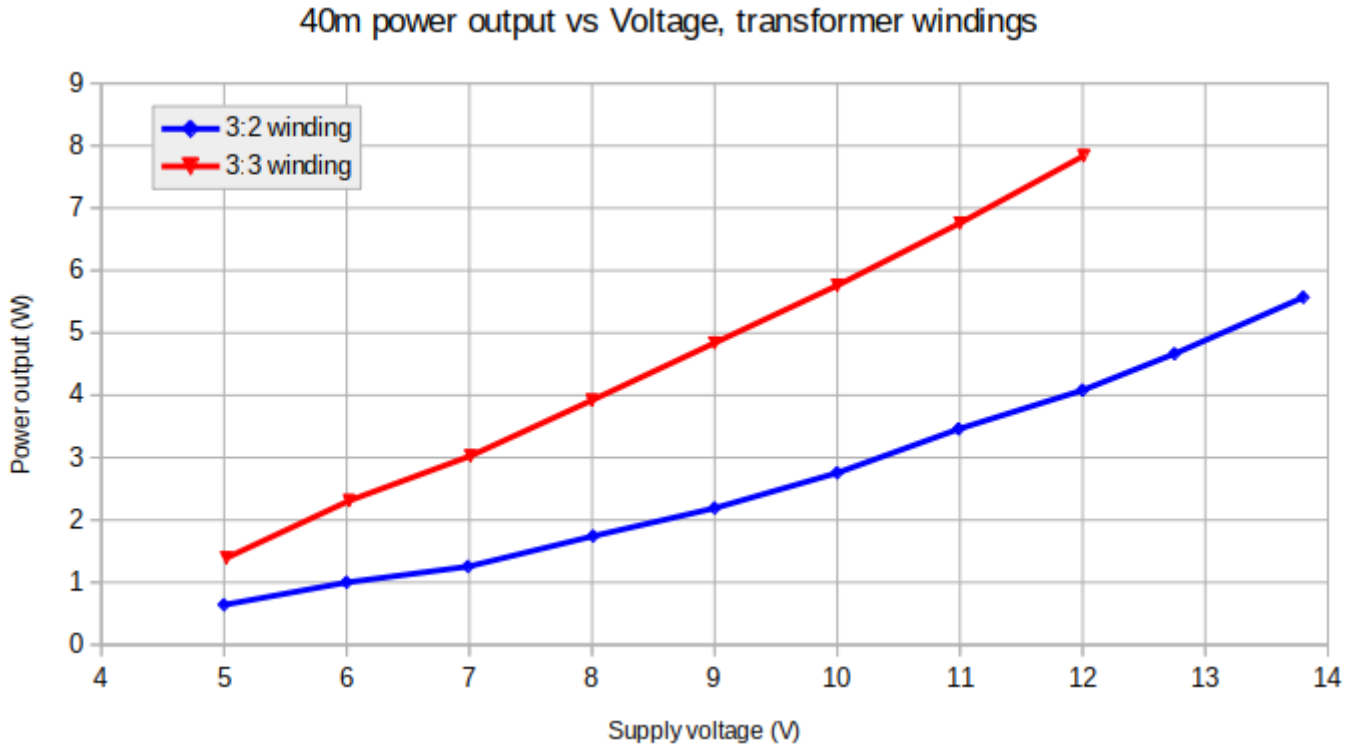


7.3 Power output vs supply voltage

Power output is consistent across bands. **Operation at more than 6W output is NOT RECOMMENDED and may destroy the power amplifier transistors.**



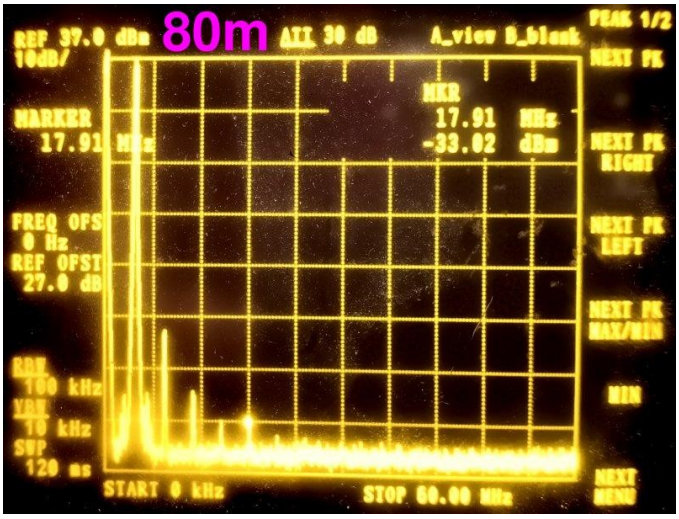
The following graph shows power output vs supply voltage for 40m operation, with the two alternative output transformer winding arrangements. 3:2 is less efficient but more suitable if you need to operate from a 12 – 13 V supply.

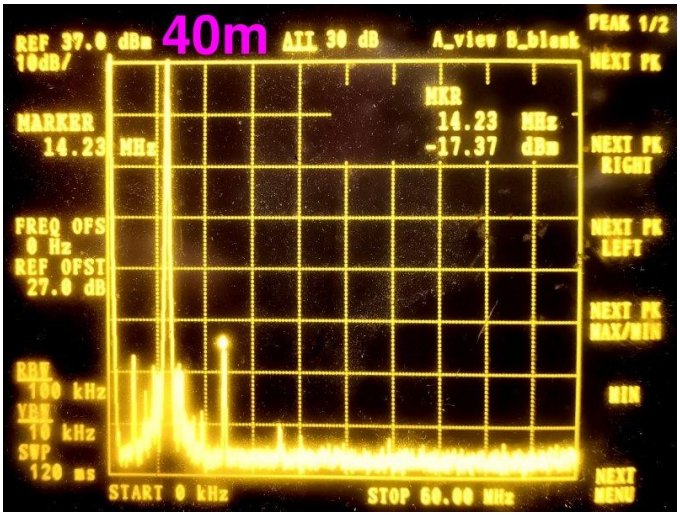


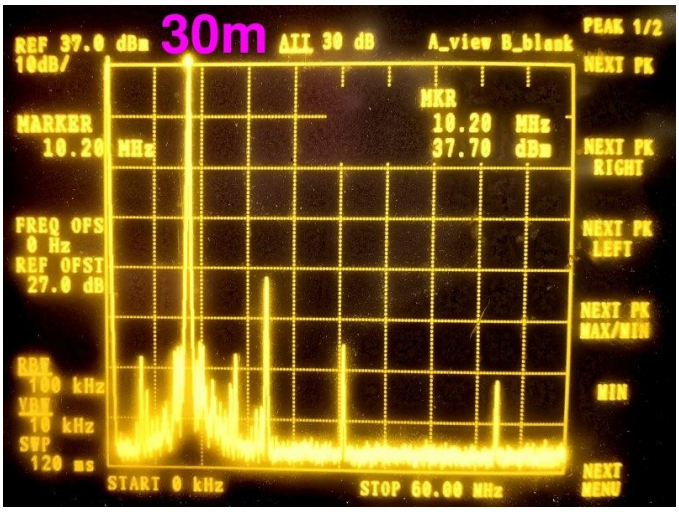
7.4 Output harmonic content

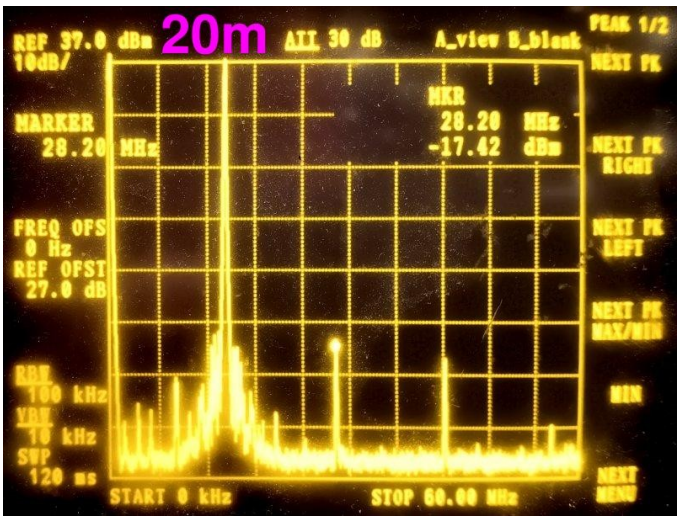
Equipment:
QDX > QRP Labs 50-ohm dummy load kit > attenuator > Advantest R3361C Spectrum Analyzer.

Band	80m	40m	30m	20m
2nd harmonic	-53 dBc	-55 dBc	-43 dBc	-55 dBc
3rd harmonic	-64 dBc	-71 dBc	-56 dBc	-57 dBc
4th harmonic	-70 dBc	Undetectable	Undetectable	-70 dBc
5th harmonic	-70 dBc	-73 dBc	-62 dBc	Undetectable



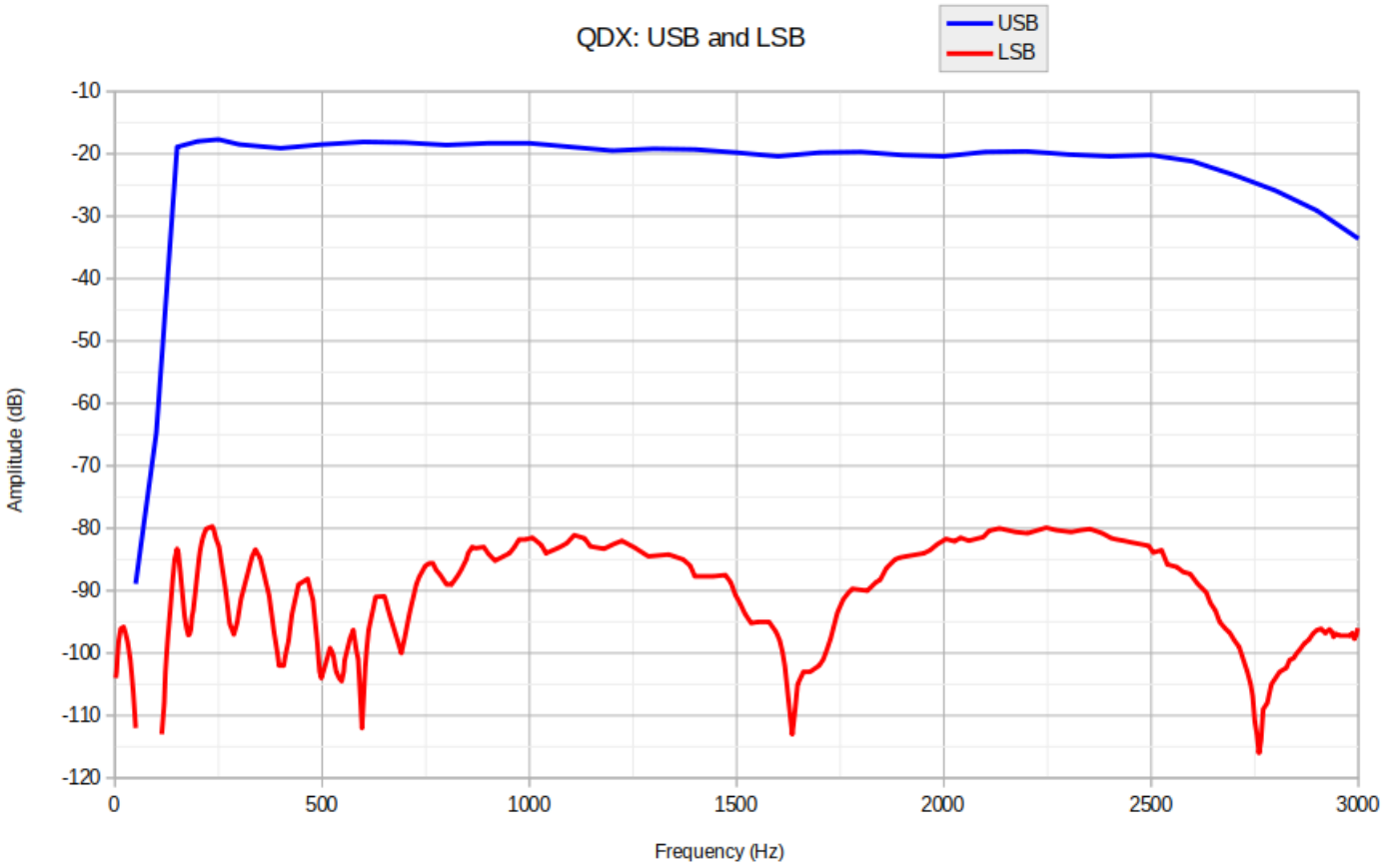




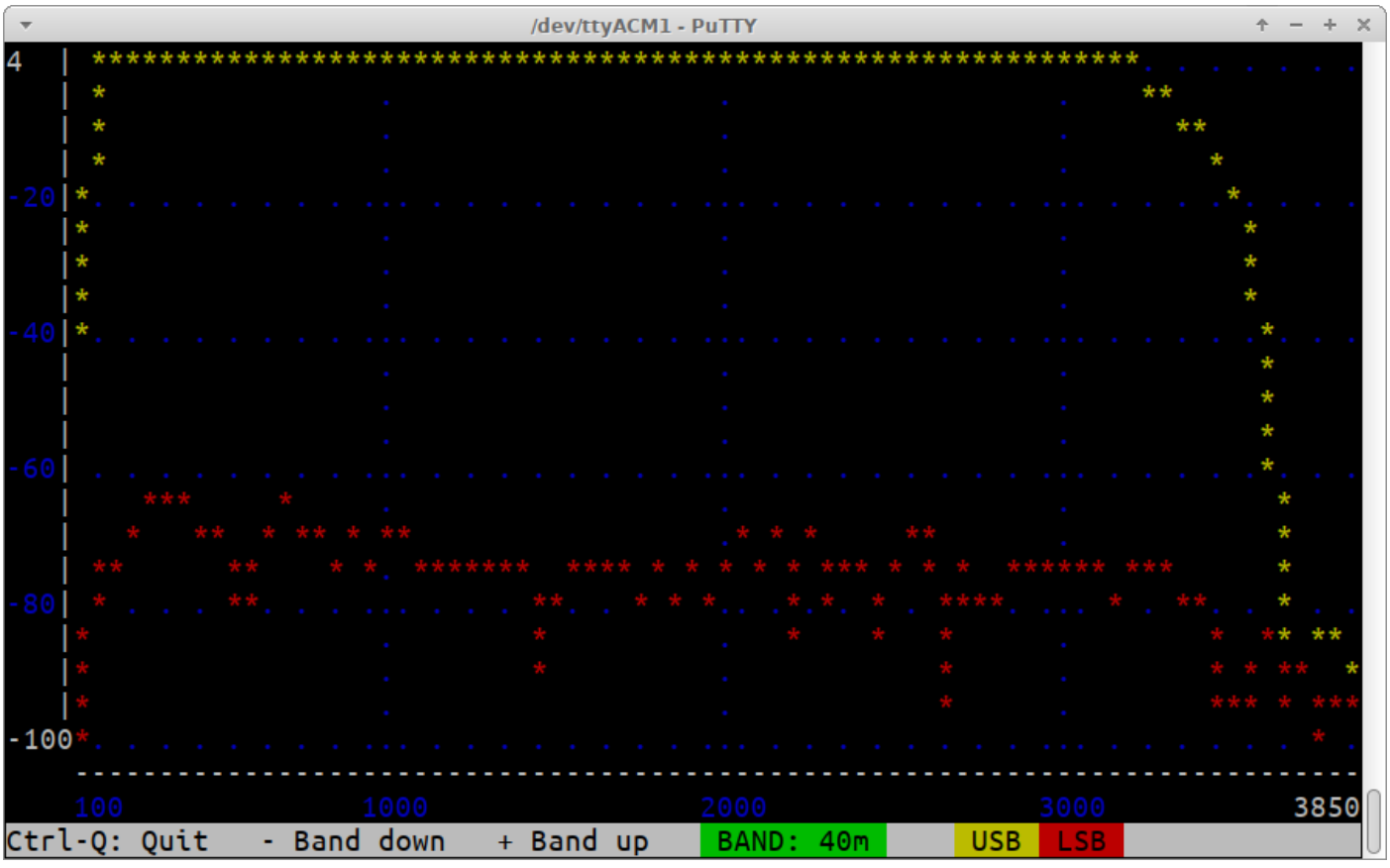


7.5 Unwanted sideband suppression

Measured using the QDX's internal signal generator; unwanted sideband is 60-70 dB down.



This is also seen on the QDX's own terminal application, Audio sweep tool:



8. Resources

- For updates and tips relating to this kit please visit the QRP Labs QDX kit page http://qrp-labs.com/qdx
- For any questions regarding the assembly and operation of this kit please join the QRP Labs group, see http://qrp-labs.com/group for details

9. Document Revision History

1.00	09-Oct-2021	First draft version version 1.00
1.01	27-Oct-2021	Lots of corrections; and update for firmware 1.01
1.02	27-Oct-2021	Added note about IQ Mode not being applicable to WSJT-X and digi modes software; and that firmware update files downloaded from the QRP Labs website need to be unzipped before installation
1.03	28-Oct-2021	Added 0.33mm wire specification to sections 2.11 and 2.12
1.04	28-Oct-2021	Clarified T1 diagram 3:3 for 9V operation; and added notes to the text about remembering to wind 2 turns secondary for 12V operation.
1.05	08-Nov-2021	Minor typos corrected
1.06	10-Nov-2021	Added recommended comms port settings for Windows, which seem to make CAT communications more reliable according to some user reports.