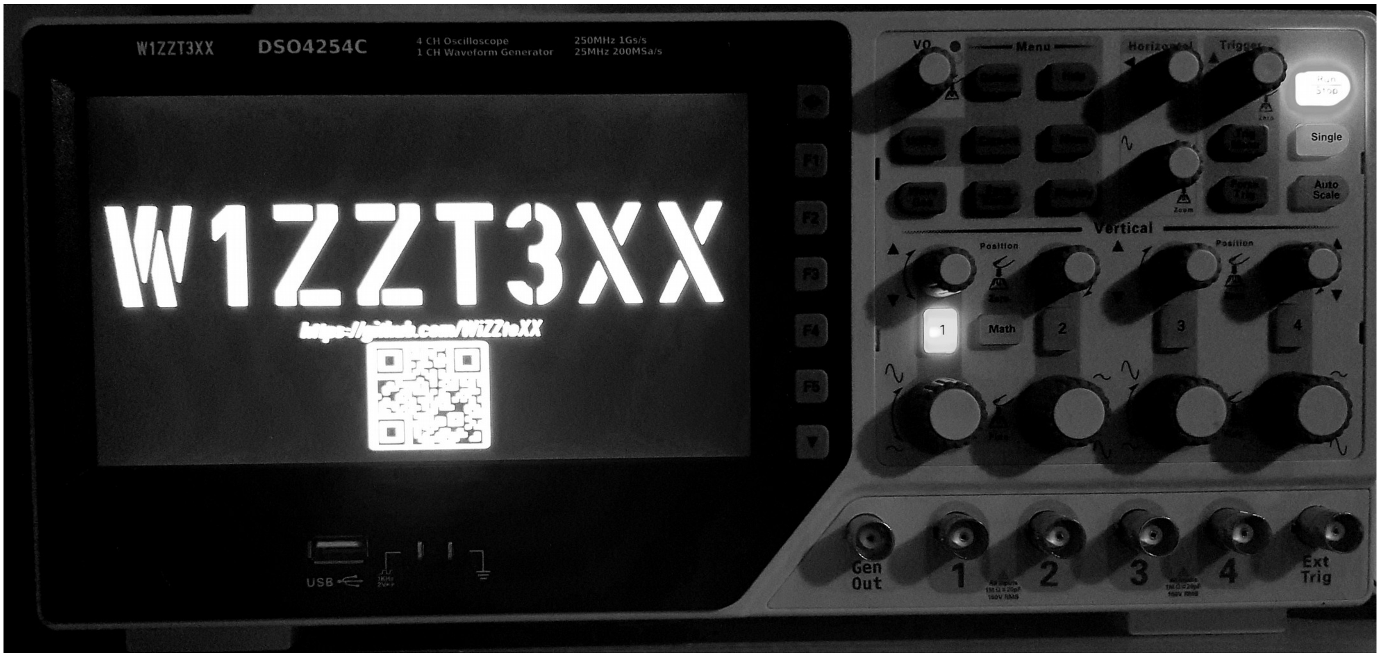
Hacking the Hantek® DSO4xx4B/C series

- A Conversion Summary -

PCB v1.05

(Probably also works for Voltcraft’s® DSO-1xx4E/F series)



rev. 2

Hacking the Hantek® DSO4xx4B/C

Table of contents

1. What is this about?

My old scope had only 20Mhz, was a vintage tube device and had no decode functions. So I decided to buy a new one. Doing this electronic stuff just for fun/hobby , I wanted to get the following:

- A table top device,
- 4 channels,
- At least 100 MHz bandwidth,
- I2C/SPI/UART decoding,
- some kind of possible data exchange,
- If possible, some kind of function generator for sine and square waves,
- Getting all of that INEXPENSIVELY!!!

So I started doing some intense research and found out, that there are some Hantek® devices out there that fulfill my wishes – the DSO4004B/C series. The C scopes have a function generator, the B’s don’t. They all sample with a rate of 1GSa per second but come in 4 speed modes with increasing prices :

• DSO4084B/C	80 MHz max.
• DSO4104B/C	100 MHz max.
• DSO4204B/C	200 MHz max.
• DSO4254B/C	250 MHz max.

The DSO4254C costs about 530 €/$, whereas the cheapest DSO4084B I found was out there for only 285€/$. There were identical types from other vendors, e.g. Voltcraft DSO-1xx4E/F, but they were more pricey.

After some more research (thanks to EEVblog – special thanks to users Microcheap and rlohmann!) They discovered that the models are identical by hardware and, in fact, hackable. So I took the offer to get the cheapest model which arrived two days later.

After two more days of fiddling out, I had enabled the spec’s I wanted:

- 4 Channels with 250 MHz bandwidth @ 1 Gs/s,
- 1 functional Channel waveform/function generator.

I payed 285€ for the scope and 28€ for some electronic parts, so compared to the DSO4254C, which is technically the same, I saved 217 €. That was worth it!

To make it some easier for You folks out there – this is how I did it.

2. What do You need?

Sometimes it’s better to check what You need before You start…

so here are the lists of required equipment and parts.

2.1 Equipment

For the Bandwidth hack:

- Torx / star bits (T6, T10, T15, T20) to open the casing,
- A PC, preferably Linux, but Windows and Mac should work, too,
- USB-to-UART-Bridge (Google for CP210x or PL2303, ≈2 €/$) with Dupont connector, 3.3 Volts and fitting Dupont cables (preferably male/female),
- A fat32-formatted USB flash drive with at least 512 MB memory,
- A plain text editor (Windows: editor.exe, Linux: geany,…)
- A shell client (I personally prefer PuTTY).

Additionally for the installation of the Function Generator:

- Soldering equipment (small tip iron, desoldering wick, tin, flux),
- A magnifying glass or even better a linen tester – SMD parts are small!,
- Tweezers, Linemans’s pliers, diagonal pliers,
- some ethanol, thinner or acetone to clean the PCB,
- a small shard hobbyists’ utility knife or scalpel like this:
- a multimeter

Additionally for the relabeling:

- A printer,
- White paper,
- Clear decal film,
- Matt clear adhesive tape,
- Double-sided tape,
- Scissors or ruler and a knife (see above)
- Software (gimp , a hex editor(Linux: e.g. Bless, Windows: e.g. hxd),

2.2 Parts

For the Bandwidth Hack (= 0€/$):

- Nothing 😊

For the Assembly of the Function Generator (≈30 €/$):

- 1 pc. Metal 50 Ω female BNC Connector (THT, straight, with nut)
  - e.g. Amphenol **B6251G3-NPP3G-50**
- 1 pc. D/A-IC **DAC902E** (TSSOP-28)
- 1 pc. Double Diode **BAV99** (SOT-23)
- 1 pc. Single OpAmp EL5166IS/**EL5166ISZ** (SO-8)
- 4 pcs. Resistor Network 4 x 22 Ω , 4 x 0603 equals 1206
  - e.g. **DR1206-22R-4/8**
    - I desoldered them from old DDR2-RAM
    - 20 to 30 Ω should be fine
    - You could even use 4 single 0603 Resistors each

For adding an external UART Port (≈0-2 €/$)

- A small piece of Breadboard (approx. 35 mm by 20mm (1 by ½ “))
- A 3-pin Dupont Connector or some other (12 by 5 mm (½ by ¼ “) max.)
- 2 PCB Spacers with Nuts and Screws (M3 max.)
- 2.5mm or 3mm Drill
- Some Wire
- A permanent Marker

For adding internal storage (0-10 €/$)

- A fat32-formatted microSD® card (can be old, small and slow)

3. Extending the Bandwidth to 250 MHz/s

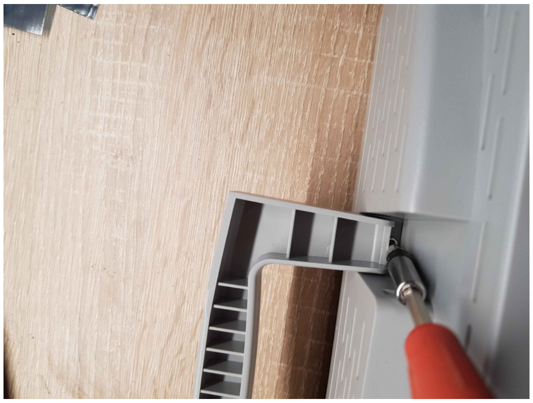
It does not matter, which DSO you own – internally they are all the same. At least as long as they have the same parts inside.

All You have to do is to lead to believe the software, that the device is licensed as the 250 MHz version… Here's how it's done.

3.1 Opening the Casing

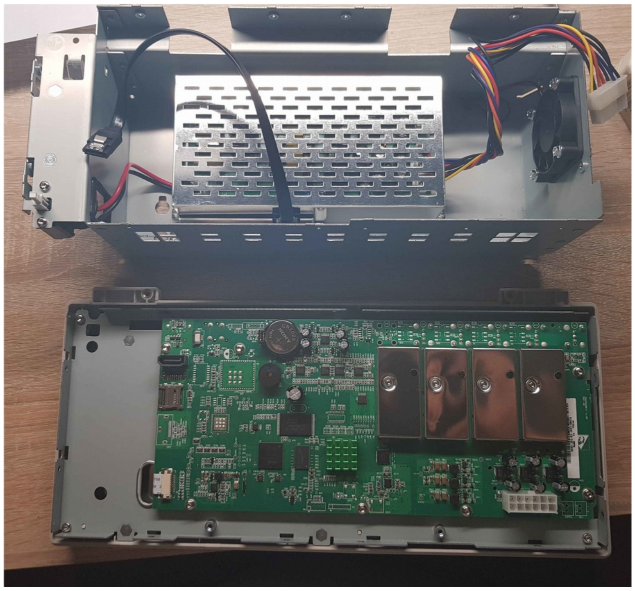
1. Pull out the power knob with some pliers or tweezers. Use tissue to prevent it from scratching.

2. The casing is screwed together with 4 screws. 2 are in the front bases (T-20) , 2 more are under the handle.

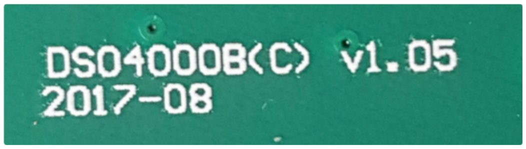


3. After screwing out the 4 screws the casing back can be taken off.

4. Remove all screws (T-15), that hold the two enclosing parts together and remove the back part.



5. I You want so, remove power cord and the SATA cable, which serves as connector for the rear USB connector.

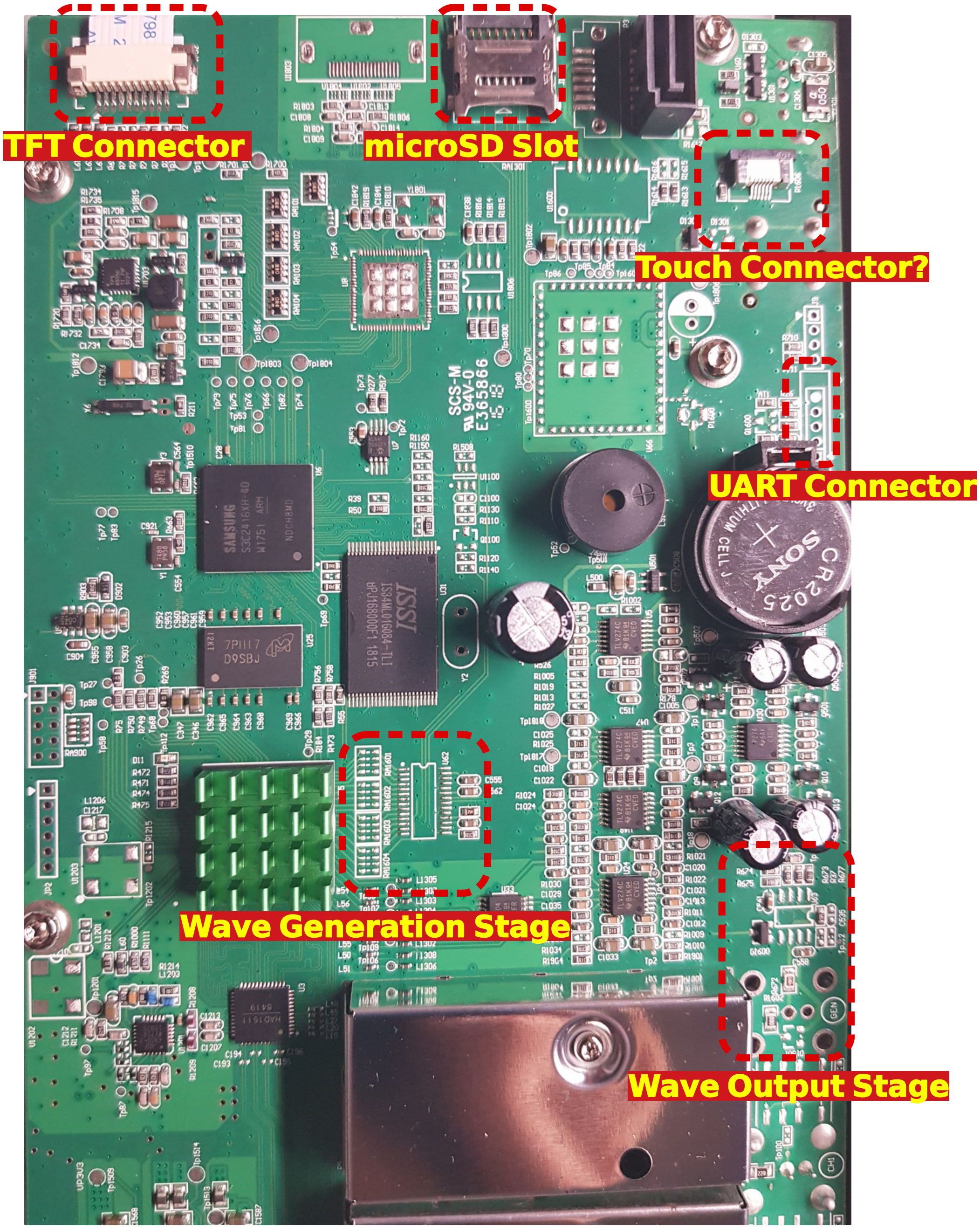


DSO4000B(C) v1.05
2017-08

Board Type

3.2 PCB Overview

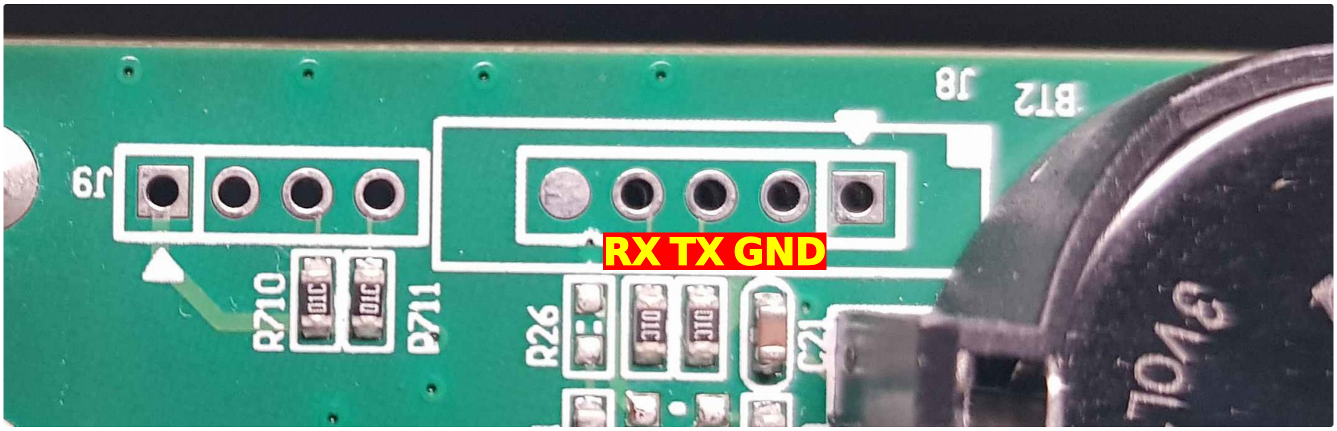
Only parts mentioned in this document are marked.



3.3 Getting Control over the Device

The device can be controlled over an internal UART. It serves as a super user terminal, so all files can be accessed and changed. Therefore, all You need to communicate with the scope are a USB-UART-Brigde (see 2.1). and a terminal on a host device (like PC, Notebook or tablet PC).

As seen in 3.2, the board’s UART connector is 5-pin row (J8).



Now connect Your bridge as mentioned in the picture and plug it into Your computer. Run a shell client and connect the serial interface on the corresponding USB port. The configuration is as follows:

- Baud: 115.200 bits per second
- 8 Data Bytes
- 1 Stop bit
- No parity
- No flow control

After turning the scope on You should see a start log. After hitting Enter, you’ll see the prompt `[root@Hantek ~]#`. You are root now.

If You are not familiar with the UNIX shell, feel free to read http://linuxcommand.org/lc3_lts0020.php for some basics.

3.4 Extending the Bandwidth by Software

1. Plug Your USB flash drive into the scope, you should see the notification *'Storage device is connected'* on the scope screen.

2. In the shell type **`cp /config/root/system.inf /mnt/udisk/`**. This copies the device configuration file to Your flash drive.

3. Type **`cp /config/root/system.inf /config/root/system.infBackup`** in order to create a local backup file.

4. Plug Your USB flash drive into the PC and open the file with a plain text editor. It should look like this:

```
[machine]
        Model=80M$DSO4084B
        Vendor=Hantek
        Product=DSO
        Manufacturer=hantek
        Serial=XXXXXXXXXXXXXXX
[version]
        Pcb=101.001.001.000.000.000.000.000
        Keyboard=1
[language]
        Lans=163190
        Language=0
[add]
        Start=4
        Update=0
```

5. To change the bandwith to 250MHz, simply change the **Model** value from Yours to **250M$DSO40254C** (or **250M$DSO40254B** if You do not want to add the Function Generator functionality). The firmware will automatically recognize this value and set the maximum bandwidth up.

6. Once You edit the file, You could also change the **Pcb** value from **101.001...** to **501.001...** .This enables the waveform generator signal output and the software accessibility over the **Wave Gen** front panel button.

7. You could also add German and Polish language by changing the **Lans** value from **163190** to **193193**. Later you can select the language by pressing the front panel **Utility** button an pressing **F1** several times.

8. Save the file. Leave the editor. Plug out the flash drive and back into Your scope.

9. In the shell, now type **`cp /mnt/udisk/system.inf /config/root/`**. This copies the edited file back to the scope. Carry on with the steps on the next page.

☼ ***HINT:*** *If You are familiar with the UNIX shell, You can also edit the file in-shell via* ***vi****.*

3.4.1 Updating the EEPROM

As I found out, at least the Bandwidth and the language options are also coded in the devices' EEPROM. That is an 8K x 8 I²C device by Microchip (Datasheet). There are Linux I²C Tools and some extensions to communicate with I²C devices. Stefano Barbato programmed a tool, that makes it really easy to change the EEPROM values – eeprog. I pre-compiled the tool and uploaded it to my github.

How to change the bandwith in the EEPROM:

1 .Download the compiled i²c tools from my github link above and save them to a flash drive.

2. Once You are connected to the UART, run the following instructions

```
cd /mnt/udisk/usr/local/sbin
cat 250M | ./eeprog /dev/i2c-0 0x50 -f -w 0x0006 -16
./eeprog /dev/i2c-0 0x50 -f -w 0x0000:16 -16
```

(This will change the bandwidth value from your value to 250M. The file 250M just contains the string "250M" without any control signals what makes it easier than an echo instruction)

4. The last output should now show the hex values for 250M like this:

```
0000| 00 00 00 00 00 00 32 35   30 4d 00 00 00 00 00 00
```

5. Type **reboot** to restart the device. Done.

3.5 Adding an external UART Port

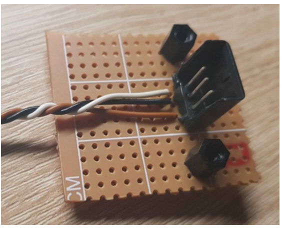
I am a lazy guy when It comes to working steps that have to be carried out repeatedly. Let’s call it ‘efficient’…

So I made the decision to make the UART port accessible without having to dismantle the casing for testing and research purposes.

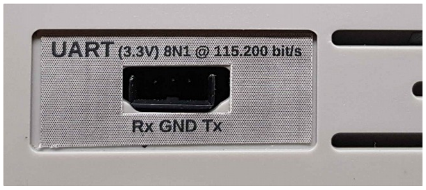
The device’s backside holds an empty LAN port punching and one more port being hidden under some adhesive film (probably HDMI). It is punched out of the casing and the sheet metal.

It was like made for my aim.

Howto



1. Cut out a piece of breadboard to approximately 35 mm by 20mm (1 by ½ “).
2. Solder a three-wire pin-row or small connector to the middle of the board.
3. Remove the plastic film, that is located over the punching hole.
4. Put the rear metal sheet and the rear casing together. Place the breadboard with the connector in the punching hole and make sure that it is centered. Hold it tight to it’s later position and remove the plastic part. Now mark the holes on the breadboard with the permanent marker.
5. Drill the holes and drive the spacers into the drill holes. Tighten them with a nut.
6. Solder the wires 1-by-1 to the connector and remember their colors and location. Twist the wires and cut them to a length of about 30 cm / 12 “.
7. Screw the preassembled breadboard into the metal sheet casing. Don’t mind the screw heads – there’s enough space for them.
8. Solder the wires to the scope’s PCB in Your preferred orientation. See pt. 3.3 for connector’s details.
9. Consider to mark the pin connections on the port. As an example how to make it visually appealing refer to pt. 5.2.2.

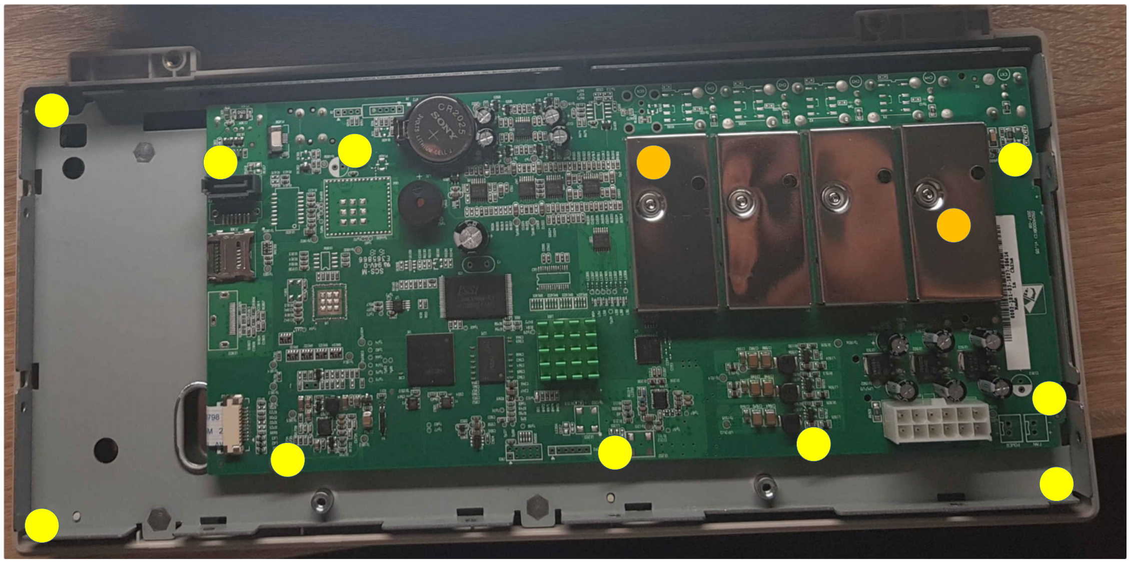


4. Adding the Function Generator

A Function/ Signal / Waveform Generator / DDS is nice for testing and prototyping purposes. The integrated one gets up to 25MHz for sine waves and up to 10 MHz for square waves. It has the option to add own arbitrary waves to be output.

4.1 Mounting the BNC Connector

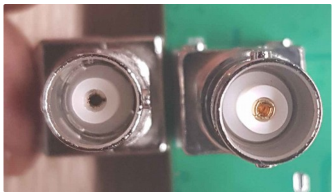
After opening the casing (see pt. 3.1), you will have to disassemble the PCB and it's holder in order to get to the front panel's backside. First disconnect the TFT connector by pulling the lid towards the ribbon cable and take the cable out. There are 10 T-15 screws visible. Two more are hidden under the two outer metal sheet shields, which can be easily dismantled once the T-6 screws are driven out.

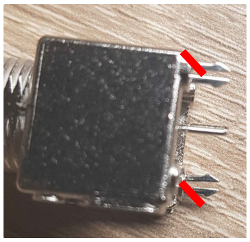


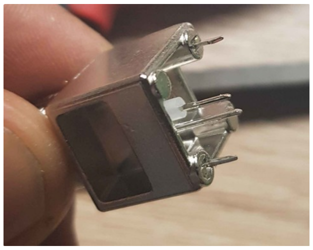
Screw Positions

Take away the plastic part. Remove the connection cable. Unscrew the existing BNC connectors from the sheet metal with some fitting pliers. Disassemble the PCB.

Finding a completely fitting PCB connector was not successful for me. So I came up with buying one that fits in the concerns of part's height and depth, visual appearance, and technically (50 Ohms). Unfortunately, the part's width did not fit. Cutting and honing down the outer pins was quite straight forward in order to get a matching part.







To get the connector in straightly later, carefully cut out the punching hole from the front adhesive label. As it can be seen on the picture, the punching hole already exists in the plastic part. So the best way for me was to keep the label in place and cut the hole for the BNC connector while using the plastic part as a cutting guide.

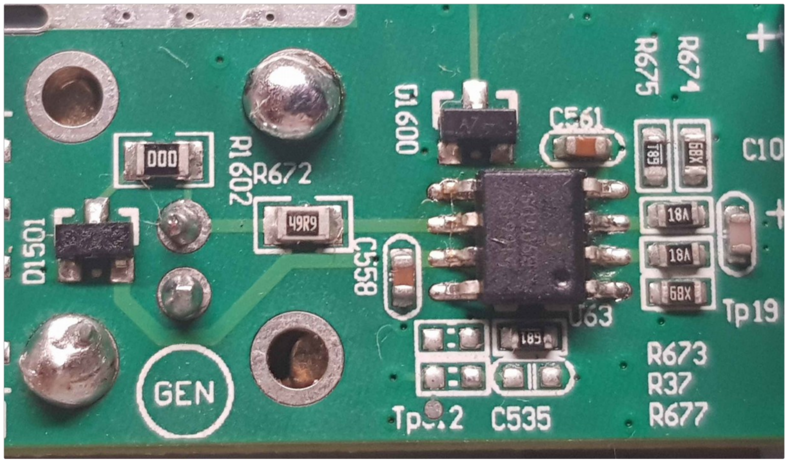
Afterwards, screw the connectors' nuts tight and stuck them through the front panel. By then, after once again checking for correct centering, solder the connector to the PCB. Finally, put the screws back in place.

4.2 Mounting the missing Parts

The next steps demand You to solder some really small-as-fuck SMT parts. The probably simplest solution would be to do it with an Hot Air Soldering Gun. But if You don't own one, you'll also be able to have it done successfully with a usual soldering iron. The trick is to solder them just like THT parts while adjusting them with tweezers. After the part is in place, - depending on the package size - either solder the single pins or solder them in a row and suck up the excess tin with desoldering wick. This may not be perfect, but will finally give You good results. You'll just have to use a bit more flux then when soldering THT, what makes it inevitable to clean the PCB when finished soldering.

There are two seperated stages of the waveform generation on the board. The generation is done via a 12-bit digital-to-analog-converter (DAC902E, Datasheet), that gets a 200MHz clock signal on pin 28. It outputs the current on pins 21 and 22 as complementary current to the Output stage. This second stage consists of an Operational Amplifier EL5166ISZ (Datasheet), which amplifies the current and passes it on to the recently installed BNC connector.

Let's start with the assembly of the output stage, because the parts are a little greater. The Output stage is located right of the BNC connector. Solder the BAV99 double diode to marking **D1501** and the EL5166IS to **U63**. Afterwards it should look like (or better than) this.

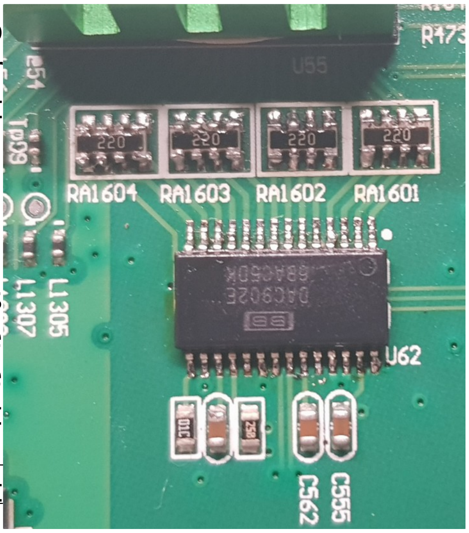


Let’s continue with the generation stage.

First solder the four resistor networks **RA1601** to **RA1604** in place using either resistor networks or single 0603 resistors, continue with the DAC, that goes to **U62** as shown in the following picture.

At this point all parts are assembled.

Check Your connections with a magnifying glass and a multimeter. Check for connection between pads and pins and for possible short circuits between the pins. Even if You are done now… keep the device open if You have to make some adjustments or just to see a <u>clear 200MHz signal on pin 28 of U62 as a proof, that Your scope is now really capable of that</u> 😊

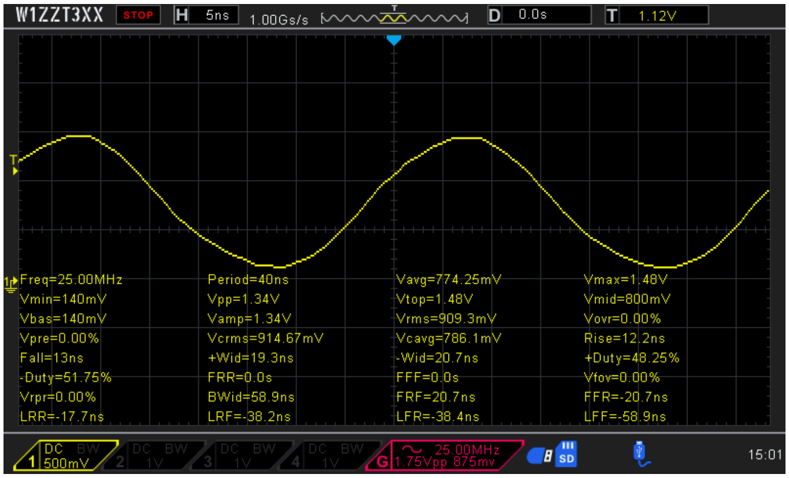


4.3 Adapting the Software

Changing the first digit of the **Pcb** value in the **system.inf** from 1 to 5 does the trick. Additionaly, change the **Model** value last digit from B to C. See pts. 3.3 and 3.4 about how to do that. After rebooting, You should be able to activate the Function Generator by pressing the front button **Wave Gen**.

4.4 Final Steps

1. Connect Gen Out connector to Channel 1 of the scope. If You don’t have a BNC-BNC-Cable, a paperclip will also do the trick, but make sure not to short-circuit between ground and signal.
2. Set some values for the Function Generator. Press **Auto Scale** . You should now see a signal, that does not match the entered values. The problem is the lacking calibration.
3. Calibrate the Generator over the Wave Gen menu 2/2, “Calibrate”.
4. Enjoy the new function.



```
Freq=25.00MHz      Period=40ns        Vavg=774.25mV      Vmax=1.48V
Vmin=140mV         Vpp=1.34V          Vtop=1.48V         Vmid=800mV
Vbas=140mV         Vamp=1.34V         Vrms=909.3mV       Vovr=0.00%
Vpre=0.00%         Vcrms=914.67mV     Vcavg=786.1mV      Rise=12.2ns
Fall=13ns          +Wid=19.3ns        -Wid=20.7ns        +Duty=48.25%
-Duty=51.75%       FRR=0.0s           FFF=0.0s           Vfov=0.00%
Vrpr=0.00%         BWid=58.9ns        FRF=20.7ns         FFR=-20.7ns
LRR=-17.7ns        LRF=-38.2ns        LFR=-38.4ns        LFF=-58.9ns
```

5. Adding more internal Storage

You could consider to add some extra internal memory to the 64MB NAND by adding a microSD® to the designated slot. It will afterwards be accessible as mounted device in **`/mnt/sd`**.

I did it in order to write some backup files on it. In future I am probably also going to use it for some extra programs.

It is important, that the microSD® is formatted as fat32. Otherwise the device will hang up at boot!

6. Relabeling the Device

Having only adapted the internals was far too amateurish, as I also wanted to have at least the Gen Out and UART ports labeled. Furthermore, the front label which tells about the specs was just wrong now. That was not compatible with my German values ☺.

6.1 Relabeling the Exterior

6.1.1 Gen Out Port

Print out APPENDIX I on clear decal film, cut it out with scissors and affix it centered under the **Gen Out** port at the height of the **Ext Trig** Labeling.

6.1.2 UART Port

Print out APPENDIX II on white paper, affix the upside with clear matt adhesive tape. Fit up double-sided tape to the bottom and cut out all 3 layers together along the thin black lines with a sharp scalpel-like utility knife. Stick it to it’s place.

6.1.3 Model Label

Check out APPENDIX III for a label fitting Your desires. The original is light gray, but I prefer white because the other labels are white-lettered, too. Prepare it like the UART label and maybe even add a layer of aluminum tape between the paper and the double-sided tape for stability reasons.

Cut it out. Cut the corners round – scissors may work best for that.

Stick it on.

6.2 Relabeling the Interior

Having the exterior relabeled I thought *'Why not to keep straight and change all the labels'.* The drawable skin files are saved in **`/dso/app/res/drawable/`** as **`*.ico`** datatypes. In fact they are raw data RGB-Transparency format. The first 4 bytes define the width, the next 4 bytes the height. They are little endian coding.

6.2.1 How to edit the ico-filetype

1. Download the directory mentioned above via the shell command **`cp –R /dso/app/res/drawable/ /mnt/udisk/`** to a USB flash drive and safe them on Your PC.
2. Find the wanted file, e.g. the manufacturer logo (in my case it's **`hantek.ico`**)
3. Rename it to **`hantek.ico.data`**
4. Open the file in a hex editor in order to get the values for width and height. In this case the 8 first bytes are **`5D 00 00 00 12 00 00 00`**, what means a width of 5D^h (=93 decimal) and a height of 12^h (=18 decimal). Remember those values and close the hex editor.
5. Open the file with gimp and fill the shown user form to load a picture from RAW data with the following values:
   - Picture Type: RGB-Transparency
   - Offset: 8
   - Width: 93
   - Height: 18

   Leave the rest unchanged. You should now see the image in the preview.
6. Edit the image in gimp as You want.
7. Export it as hantek.ico.data. Don't make any changes in the upcoming dialog. Close gimp.
8. Open the new file in a hex editor. Paste the 8 size bytes You read out at 4. in front of the raw image data. Save it. The file size now matches the original's.
9. Save the file to a USB flash disk. Then copy it back to the scope via shell command **`cp /mnt/udisk/hantek.ico /dso/app/res/drawable/`**.If You used a windows PC, you will have to add the file access rights via the command **`chmod 777 /dso/app/res/drawable/hantek.ico`**.
10. Restart the scope.

HINT: *The boot screen is called* ***welcom.ico***

7. Notices and small Tweaks

7.1 Enabling German language

See 3.4.7.

7.2 How to hack the firmware updates?

1. Make a Backup!
2. Download the latest firmware from Your vendor (Latest Hantek Firmware) .
3. Unzip the download file. You will get a file called dso4kb_xxxxxxxx.upk.
4. Decrypt the file via command `gpg dso4kb_xxxxxxxx.upk`. The password is **dso4000bc**.
5. This will create a file **dso4kb.upk.tar.gz.** Unzip with `tar -xf  dso4kb.upk.tar.gz` This will create a file **dso4kb.upk.tar.** Unzip this file with `tar -xf dso4kb.upk.tar`
6. You will get a directory named **package** and a plain text file called **upend**. The text file corresponds to the file's and directory's names in **package**. If You add or delete something, You must add or delete them in the upend, too.
7. Change whatever You want to change.
8. Pack the files and make sure You type the real password when reencrypting. If You are a Windows user, you have to add the line `chmod 777 /*` *in the* **package/do_update.sh** right before **sinc**.
9. Update as usual.

8. Disclaimer

Whatever You do – You do it at Your own risk! Do only what You are capable of!

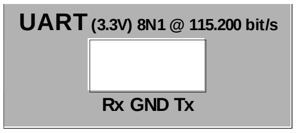
I am not responsible for faulty hardware.

APPENDIX

I Gen Out Port label

Gen
Out

II UART Port



III Model Name Labels

(cut the corners round)

W1ZZT3XX DSO4254C 4 CH Oscilloscope 250MHz 1Gs/s
1 CH Waveform Generator 25MHz 200MSa/s

W1ZZT3XX DSO4254C 4 CH Oscilloscope 250MHz 1Gs/s
1 CH Waveform Generator 25MHz 200MSa/s

DSO4254C 4 CH Oscilloscope 250MHz 1Gs/s
1 CH Waveform Generator 25MHz 200MSa/s

DSO4254C 4 CH Oscilloscope 250MHz 1Gs/s
1 CH Waveform Generator 25MHz 200MSa/s