PA3GYF GPS-Disciplined OCXO

Version 1

Log

Jan de Jongh – pa3gyf

jfcmdejongh@gmail.com

Last Update 20191013

1 Introduction

This document describes version 1 of pa3gyf-gpsdocxo, a project the author undertook in 2016 and for which final testing and documentation finished late 2019. The GPS-disciplined OCXO is designed around a Morion MVR89A OCXO operating at 10 MHz, a Rockwell Jupiter TU30-D140 GPS receiver with 10 kHz output, a G4RUH-designed PLL circuit in high-speed TTL, the G4HUP DA1-4 distribution amplifier, and a control board based upon the Arduino Micro with firmware designed and implemented by the author. The hardware and software designed by the author of this document is released under Apache Commons License, unless otherwise noted. In particular, the license does *not* apply to the distribution amplifier.

2 Background and History

After over a decade of absence, I decided in 2015 that it was about time to pick up my old HAM and electronics hobby again. In 2015, I focussed at the construction of a shack, and recovering/refurbishing my measurement equipment and (finished) HAM projects from the past; in the meantime thinking of what kinds of projects I wanted to work on. In 2016, I decided to focus at small-signal RF-projects design and construction and measurement equipment. This still includes HAM-related projects, though I'm no longer occupied with the concerns of getting stuff outside in my mast. Well, who knows, maybe later…

This document describes the first project I had picked up: a GPS-disciplined OCXO. The project was on my mind for several years already, and to that end, I already attached a high-end GPS antenna outside, and collected a few essential items (notably, suitable OCXOs and GPS receivers). Because I had been out-of-the-loop for so long, I decided to use/steal ideas from others to the extreme.

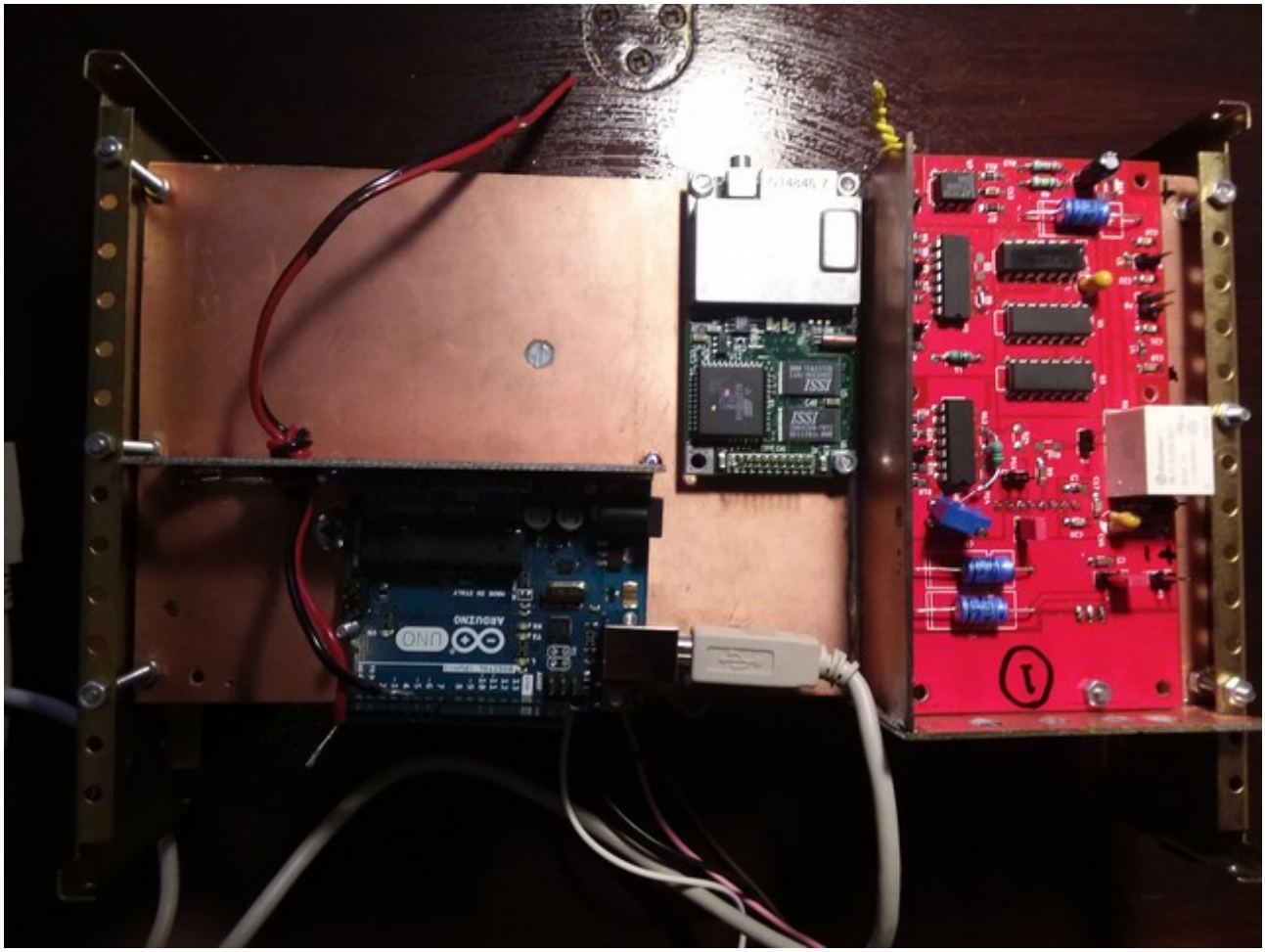
Please note that the project was mostly meant as a personal journey into updating “old-school” skills. For one, I did not care that much (well, within limits) about costs, and was not particularly striving for simplicity. I *did* however, care about flexibility (most PCBs have several implementation options, including some meant to overcome design and/or dimensioning mistakes), as well as the final “looks” of the project.

3 Photo Impression

Before going more detailed into the project, we present some construction photos. Unlike most if not all of my earlier projects, I decided to design the circuits starting with a given enclosure (one of the fantastic Velleman enclosures). Note that many pictures in this section were taken in 2016, and they do not represent the final hardware.

3.1 Bottom side with Arduino, GPS and PLL boards

Mechanically, I decided to use a double-sided blank PCB and mount it with the support holes on the sides of the Velleman enclosure. In the photo below, we actually look at the bottom side. The Arduino Uno (switched to Micro later) is visible in the lower left, the GPS is in the upper center, and the PLL board (well, an earlier version of the final one) is shown on the right. The white box is a relay switching the tuning input of the OCXO. The OCXO itself is on a separate PCB on connected to the PLL board with a SIL connector.



Clearly visible: I misjudged the size of the USB connector, but it is only needed for programming the Arduino. In the final version, I will remove both the power and USB connectors from the Arduino (re-programming is then done on a separate Arduino and by simply replacing the ATMega processor). The reason is that I need to create sufficient height for both shields (which are stacked).

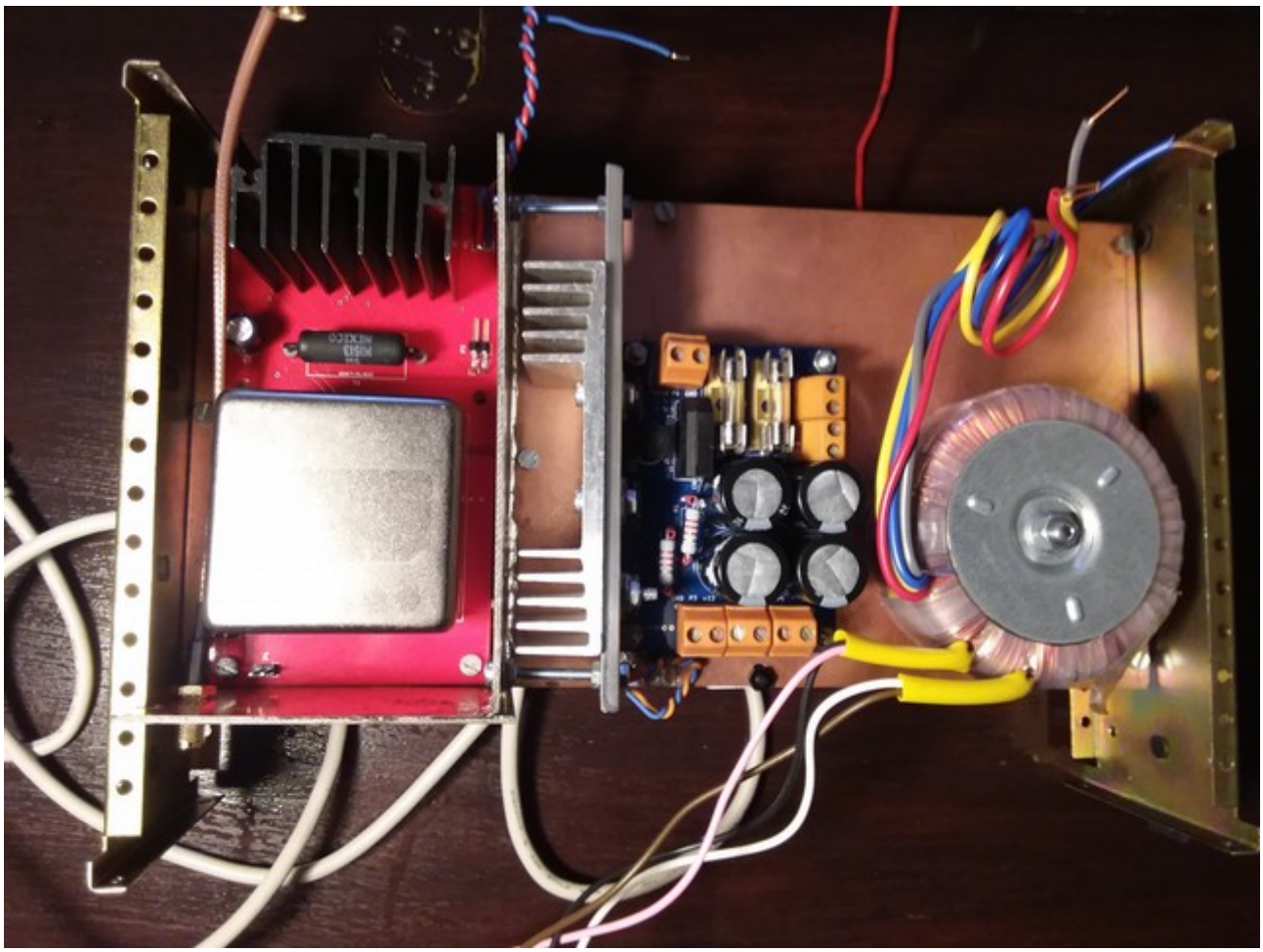
The excessive height of the Arduino/shields combo became too much of a hassle, so I decided to redesign the control board around an Arduino Micro, with a much smaller footprint.

The top-left is reserved for the DA, which is used unmodified (I even add the DA's enclosure). The GPS is also used unmodified. The MMCX/SMA pigtail is not shown, as it has been mounted already in the rear panel.

Note that at present time, I have merely started the internal wiring of the various PCBs. I'm still waiting for some el-cheapo SMA stuff, as I decided to standardize on SMA for internal wiring, even at 10 MHz. Main reason: I do not want to stock 20 different connector types, and SMA are among the smallest connectors available. Well, maybe I'll reconsider this in the future...

3.2 Top side with OCXO, PSU and Transformer

The photo below shows the top side holding the OCXO board (sandwiched to the PLL board on the bottom side), the Power Supply Unit and the transformer. The heatsink on the OCXO board is for its private 12V regulator. Between the regulator and the OCXO is the current-sense resistor.

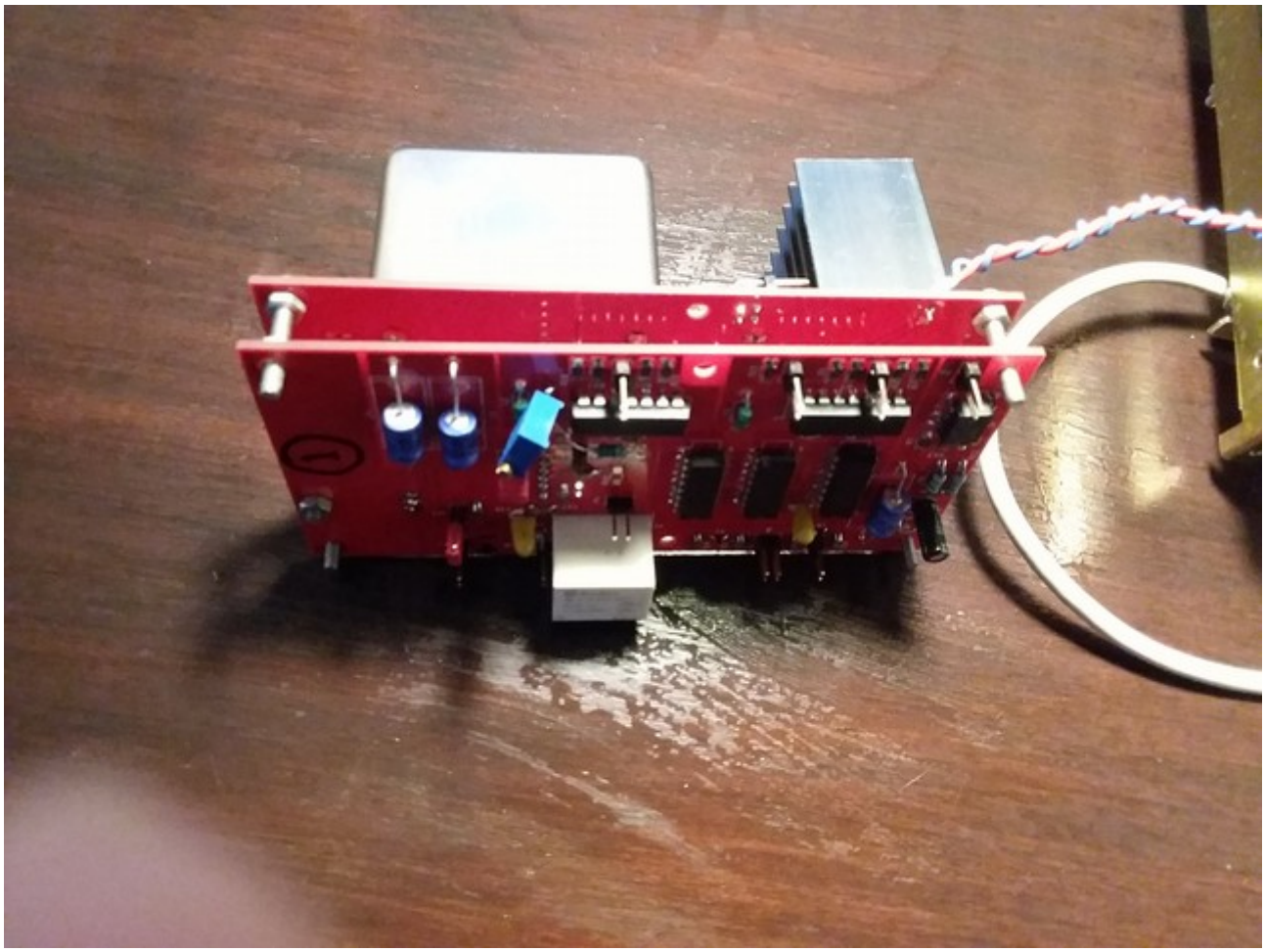


The top side of the internal construction. Contains pretty much everything that “might get hot”. The heatsink for the PSU is not mounted in its final form yet, because I need some larger M3 bolts. The current-sense resistor is a 2Ω/1%/5W, somewhat larger than needed but it also helps to thermally

offload the regulator during OCXO heat-up (current reaching 850 mA!), in view of the 21V unregulated supply.

3.3 OCXO-PLL Sandwich

In the photo below we see the sandwich construction of the OCXO and PLL boards.

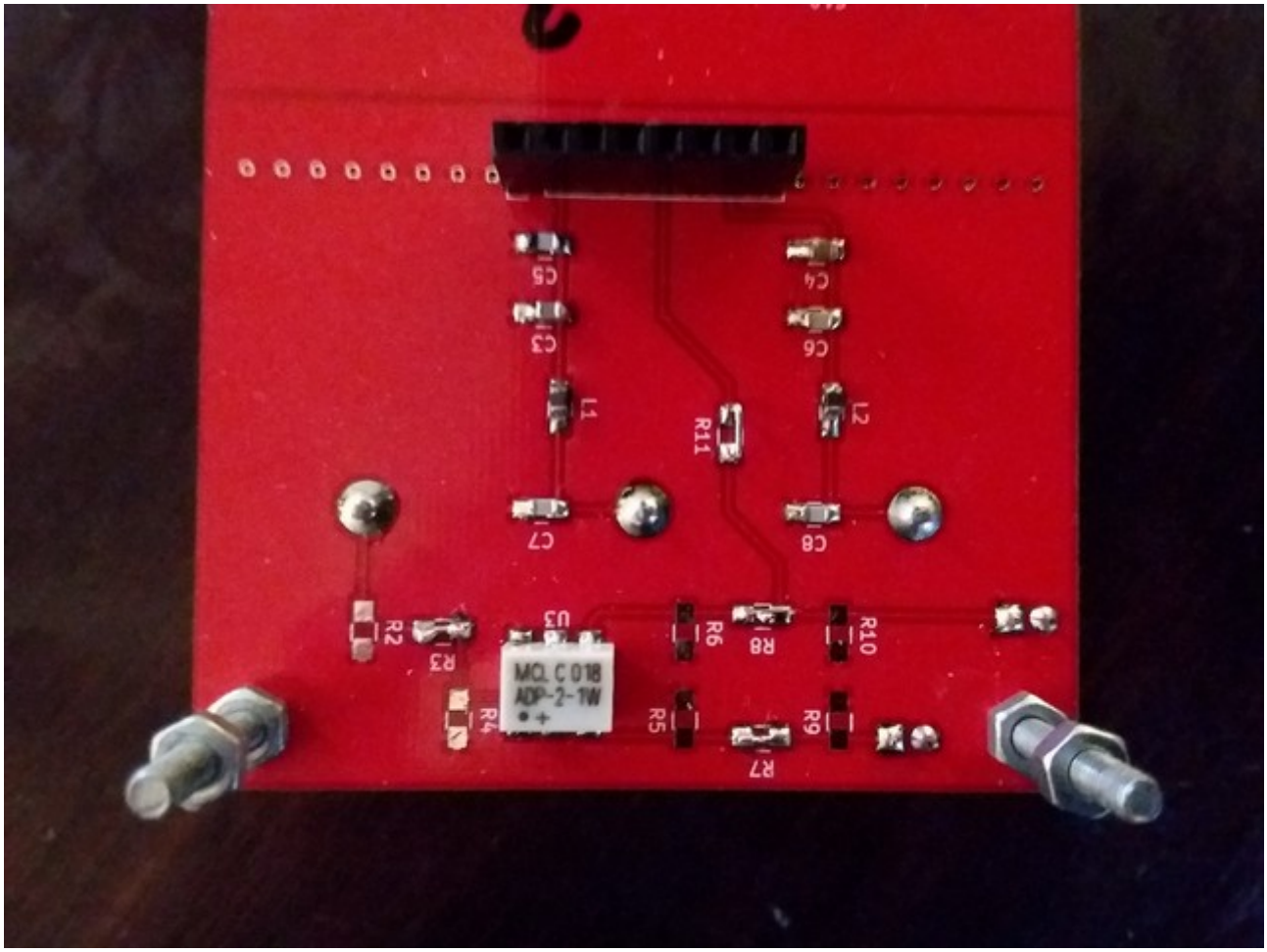


Note the botch circuit providing +2.5V DC bias to the input of the Schmitt-trigger. In the final version, the bias circuit will be added to the (separate) LNA. The LNA will be a low-noise single-rail 5V-capable OPAMP (like OPA65 and the like). For now, I guess I'll use something from the junk box…

The white box is a 5V relay switching between manual and PLL mode. The (optional) 7805 regulator is mounted horizontally on the rear side of the PLL PCB.

3.4 OCXO PCB Rear-Side Detail (e.g., splitter, note 0805 use)

In the photo below we show the most important part of circuit on the OCXO board.

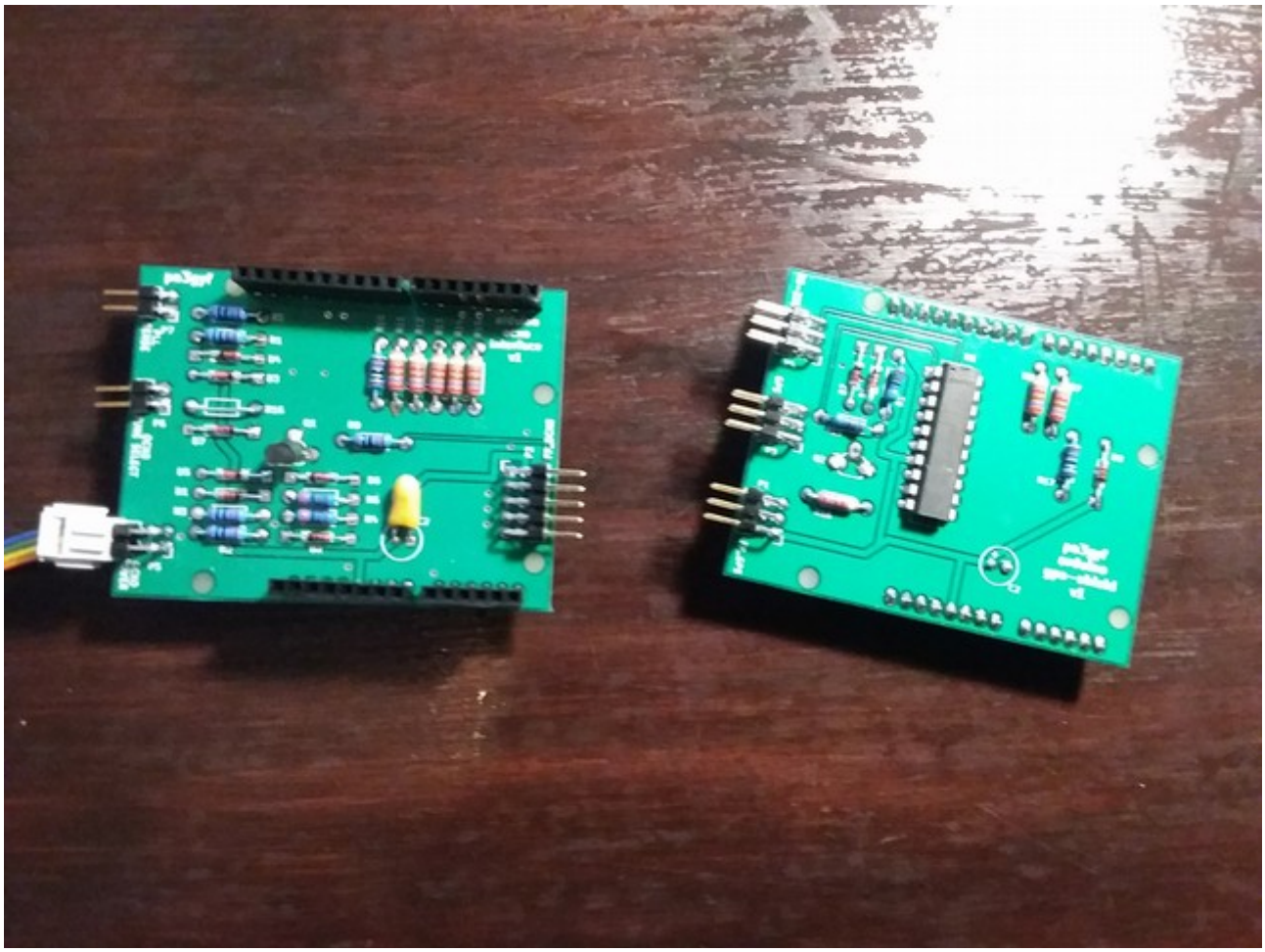


No attempt has been made to design a 50Ω circuit, which I somewhat regret. For the moment, both signals are available on the front side of the board through 2-pin headers (i.e., no coaxial connection). In the final version, I intend to directly solder RG316/SMA pigtails on the board.

3.5 Arduino Shields

Arduino Uno

In the photo below, we show the two Arduino shields for the Arduino Uno., one shield for interfacing to the GPS receiver, and for interfacing to the OCXO/PLL combo. Note the excessive use of classic through-hole components; at this stage, I was also working on gaining experience with PCB design and reluctantly moving towards SMD components.



The board on the left is the OCXO/PLL interface to the Arduino; the one on the right its GPS interface with the MAX433 RS232 ↔ TTL level-converting IC. I still have to drill the M3 mounting holes, although they are not (that) essential in the final construction. The shields are exactly the same size as the Arduino Uno R3, but that should be considered an error, because the lowest board (i.e., closest to the Arduino) will “clash” with the Arduino's power and USB connectors. Also, due to impatience (:-)), I ceased my attempts to put both interfaces on a single board. In retrospect, that would have saved me from the 'excessive height problem'.

As mentioned earlier, I later decided to redesign the entire control system and use an Arduino Mico instead.

Arduino Micro

<tbd>

3.6 Front Panel

The photo below shows the front panel (as it also appears in the final product).

With BNC terminators already in place. Both panels (front and rear, see below) were ordered at Schaeffer.

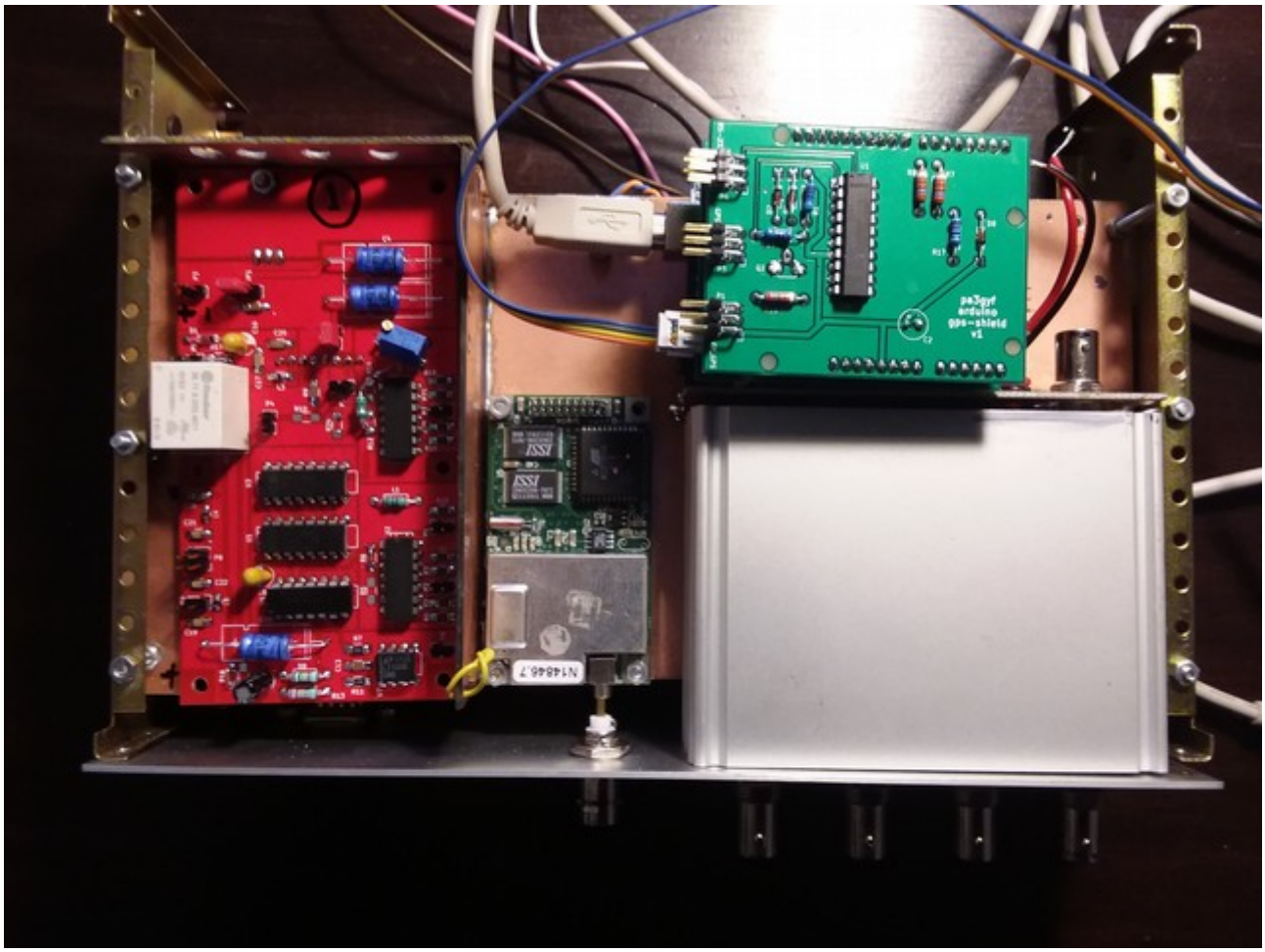
3.7 Rear Panel

The photo below shows the rear panel. The mains connector should really be in the lower-left corner, but I did not have sufficient room on the inside of the box to hold the filter unit (as it extends quite deep).

One of the four outputs of the DA is to be connected to a separate BNC with direct connection to the front-panel. This saved me the trouble of having to create an extra output on the DA, or sacrifice on of its outputs permanently.

3.8 Bottom view with DA installed

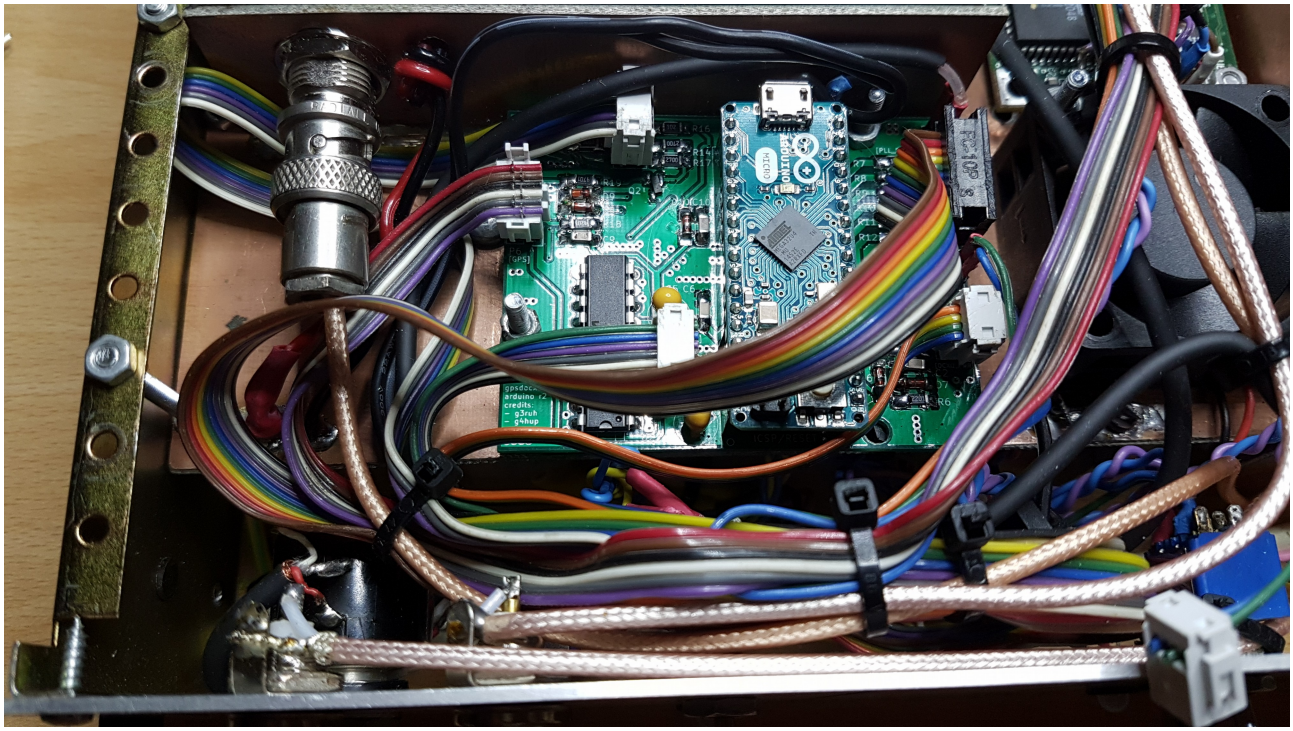
In the photo below we show the bottom side again, but this time with the Distribution Amplifier in place. It also painfully reveals the trouble I had with the Arduino shields for the Arduino Uno.



The BNC in the upper right is the 10 MHz input to the DA. The mounting holes in the double-sided copper board in the upper left are used for (“el-cheapo”) SMA/RG316 pigtails for the 10 MHz input, the 10 MHz TTL output, the 10 kHz TTL input and the 10 kHz TTL output (optional, I intend not to implement this).

3.9 Close-Up of Control Section in Final Product

In the photo below, we show the control section in the final product. Unlike the earlier Arduino-Uno based version, we used an Arduino Micro on top of a single interface board in the final product. The BNC connector in the upper left is the input feed of the distribution amplifier. Note that the view is from below, i.e., this is the bottom side of the unit. Also visible on the right is a small fan that was included in the final redesign because certain components on the PSU board became quite hot.



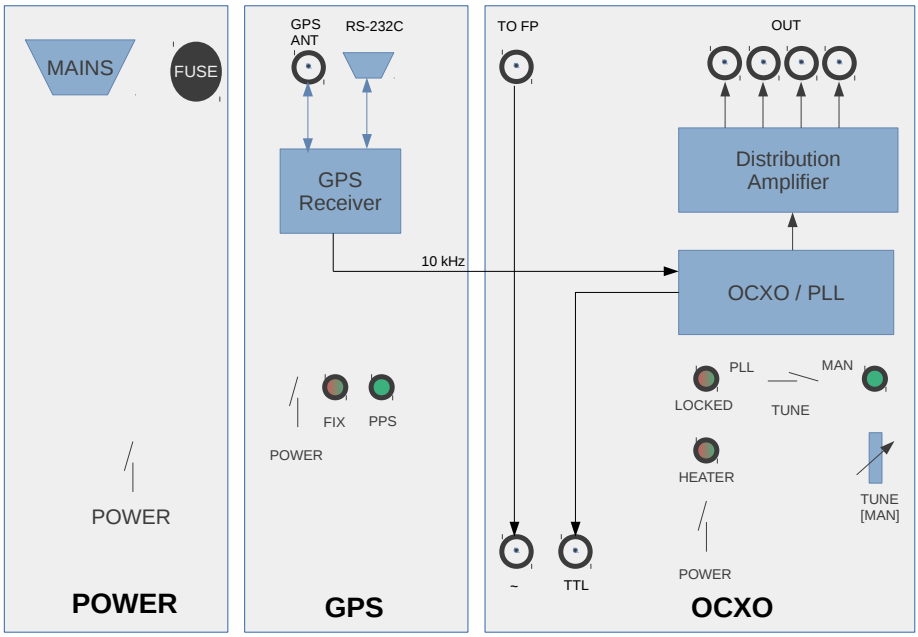
4 User Manual

In this section we describe the operation of the unit.

4.1 Functional Overview

The figure below provides a functional overview of the unit. (It is also available as a separate file from github.) It consists of three main blocks:

- **POWER:** Controls the power to the (entire) unit.
- **GPS:** Controls the GPS receiver and the 10 kHz reference signal to the PLL.
- **OCXO:** Controls the OCXO (and the PLL).



4.2 Front Panel

In the photo below, we show the front panel for reference later in this section.

The front panel features the following groups of controls, indicators and connectors:

POWER

- **POWER:** Turns the unit on or off. Note that there are no provisions to keep heating the OCXO while the unit is off.

GPS

- **GPS/POWER:** Toggles the power to the GPS subsystem.
- **GPS/FIX:** GPS satellite acquisition status:
  - **RED:** GPS receiver not (yet) heard, or GPS messages received that are unknown or could not be decoded, or no (appropriate) reports received from the receiver (ERROR condition).
  - **RED/GREEN BLINKING:** The GPS receivers sends status reports, but is still in the process of acquiring more satellites in order to obtain a fix. Two cases:

        - **75% RED, 25% GREEN:** There is communication with the GPS but, as of yet, none of its messages could be decoded.
        - **25% RED, 75% GREEN:** There is communication with the GPS and it sends messages that could be decoded properly. There is, however, not yet a 3D FIX.
    - **GREEN:** The GPS generates 3D FIXes (and thus a highly stable 10 kHz reference signal for the PLL *and* a precise PPS signal on the RS-232 connector at the rear which you can use for time synchronization).
- **GPS/PPS:** Pulse-Per-Second indicator. Note that this LED will always blink at 1 Hz when the GPS subsystem is enabled, even if the receiver does not have a fix (yet).

OCXO

- **OCXO/POWER:** Toggles the power to the OCXO and (de)activates the PLL. The PLL is only active (if at all, see below) if the OCXO is switched on.
- **OCXO/HEATER:** OCXO heater status:
    - **RED:** The OCXO is heating up, but is not yet close to its operational temperature.
    - **RED/GREEN [dual]:** The OCXO is still heating up, but is coming closer to its operational temperature; the “greener” the LED, the closer.
    - **GREEN:** The oscillator has reached its operational temperature, and is ready for use with the PLL
- **OCXO/TUNE:** This switch determines the ways of (fine) tuning the OCXO:
    - **PLL:** The OCXO is part of a PLL loop (at 10 Khz) attempting to lock the OCXO frequency to that of the GPS 10 Khz reference signal.
    - **MAN:** The OCXO is in free run mode, in which the frequency can be fine-tuned (+/- a few Hz) manually.
- **OCXO/LOCK:** PLL lock-status indicator:
    - **RED:** The OCXO is out-of-lock (with GPS) because either the GPS is inactive or has no fix (yet), or the OCXO is in free-run mode, or because of an error condition.
    - **RED/GREEN BLINKING:** The PLL is attempting to lock the frequency of the OCXO to the GPS reference of 10 KHz.
    - **GREEN:** The OCXO is locked in frequency to the GPS reference signal (10 kHz). This implies that the GPS system is active and has a FIX and that the OCXO/TUNE switch is set to PLL. The estimated obtained accuracy while in lock is better than 10E-9, but this improves even further over time.
- **OCXO/MAN:** LED indicating, when GREEN, that the OCXO is in free-run (MAN)

mode and that the OCXO frequency can be fine-tuned with the MAN TUNING KNOB (see below). If the LED is OFF, the frequency of the OCXO *is not* affected by the setting of the MAN TUNING KNOB.

- **OCXO/MAN TUNING KNOB:** Tuning knob (10 turn) for the fine-tuning of the OCXO frequency in free-run mode. The OCXO can only be pulled a few Hz away from its frequency. Note that the knob has *no* effect when the TUNE switch is set to PLL.
- **OCXO/10 MHz OUT:** BNC connectors carrying the output signals:
  - **~:** The 10 MHz sine wave at 5-10 dBm level into 50Ω.
  - **TTL:** The 10 MHz output signal as TTL square wave. Note that the duty cycle of the TTL signal is *not* specified nor even guaranteed to be 50%!.

Notes

- The presence of the 10 MHz sine wave on the front panel requires a patch cable on the rear of the unit, leading the signal from one of the rear outputs of the Distribution Amplifier to the front connector.
- In theory, a PLL lock can be obtained *before* the OCXO has reached its operational temperature. The OCXO heater status, although maintained, does not play a role in determining PLL lock status by the software.
- Depending on environmental temperature, OCXO isolation and probably various other factors, OCXO heating to operational temperature may take up to 30 minutes. Once the OCXO is at its operational temperature, occasional periods of heating can and probably will occur, but these should not affect the LOCK status of the PLL.
- The 10 MHz OCXO output signal (both sine and TTL) is *always present if the OCXO is enabled*. In other words, output is not blocked when GPS FIX or PLL LOCK is lost.

5 Design

5.1 Introduction

5.2 Starting Point – Initial Design

As mentioned earlier, the starting points for the design were

- The available MV89A and STP2145A OCXOs on Ebay;
- The Jupiter GPS TU30-D140 with 10 kHz GPS-locked output;
- The design of G3RUH for a 10 kHz PLL using these OCXOs.

5.3 Problems Encountered in the Initial Design

With the initial design (basically, the G3RUH circuit with a suitable OCXO), we ran into several problems:

- From several (undocumented) experiments, we found that many OCXOs do not provide sufficient output power to properly trigger the Schmitt-Triggers in the G3RUH design.
- With AC coupling between the OCXO and the Schmitt-Triggers, we lose a lot (half) of the sensitivity.
- Let's take an example: a 5 dBm output signal from the OCXO is equivalent to 3.16 mW and yields approximately 400 mVrms into a 50 ohms load, which amounts to 1.125Vpp, or with AC coupling, between -562 mV and +562 mV. But if we study the 74AC14 datasheet, we soon realize that this just barely, if at all, enough for proper Schmitt-Trigger operation, even with ideal DC bias. Without going into too many details, the 74AC14 has typical hysteresis of 1.0 V, with *maximum* positive threshold of 3.2V to 3.9V depending on supply voltage (4.5V to 5.5V) and *minimum* negative threshold of 0.9V to 1.1V (Vcc from 4.5V to 5.5V). This means that in a more or less worst case scenario (Vcc=5.5V), we need 3.9V of input voltage (from below) to make the output "0" (the 74AC14 is inverting), and 1.1V of input voltage (from above) to make the output "1", in other words, a signal of 2.8Vpp with accurate DC bias, or almost triple the range we have with +5 dBm output signal.
- Note that, in the previous discussion, the duty cycle of the signal is not really a concern, since we will divide to 10 kHz.
- The GPS receiver will issue a 10 kHz signal even in absence of a GPS FIX or even in absence of a GPS antenna at all.
- The OCXO will output a 10 MHz signal (or thereabout) even in absence of a "proper" lock with the GPS 10 kHz signal.

5.4 Concerns in the Initial Design

- The output of the OCXO should be independent from the load.

- The output from the OCXO should be distributed to multiple sources, with sufficient isolation between the sources, as well as (see previous point) sufficient isolation between the individual sources and the OCXO.
- Dissipation of the regulators and bridge rectifiers in the PSU.
- Dissipation of the regulator and current-sense resistor on the OCXO Board.

5.5 Wish List for the Final Design

- We want to monitor the (expected) heater current for the OCXO, in order to assess whether or not the OCXO has reached its operation temperature.
- We want the GPS receiver to use as a (PPS) time reference source with chrony/ntp.
- We do *not* want to integrate a time server, and therefore, we want a RS232-C input/output with PPS support.
- We want to use Arduino if possible (user base; libraries).
- LED-monitoring at the front panel.
- Manual operation of fine-tuning the OCXO (without PLL support).
- Power to the GPS Antenna.

5.6 Non-functional Requirements for the Final Design

- Sufficient test points.
- Support for learning experience.
- Modular design.
- Parts availability and second-source.
- Isolation.
- Spectral Purity.
- Front and Rear Panel designs.
- Extensibility.
- Adaptability: Support for other OCXOs; other filters designs; other PLLs; other GPSs.
- Future-proof.
- Ethernet support.
- Initial design with SMDs and 3rd -party PCB manufacturing.
- License; FOSS.
- Open-source PCB design.
- Thermal considerations.

- Long-term operations.
- Safety.
- Fits into enclosure.

5.7 Final Design

- **Enclosure:** Was given in advance; Velleman L860: 80x250x180 mm. Beware that these are no longer produced by Velleman.
- **Mains Circuit:** Filtered Euro chassis connector (IEC 60320/C14), primary chassis-mounted fuse holder, and a dual-pole mains switch with integrated neon indicator, all from the junk bin.
- **Transformer:** Amplimo 17061, primary 2x115V, secondary 2x7.5V 2A (each).
- **Power-Supply Unit (PSU): Own design;** needs 21V unregulated for the OCXO power supply and current-sense circuit. This requires the secondary windings of the transformer to be put in series. The PSU also provides +12V (Arduino, Distribution Amplifier, Fan) and +5V (PLL, GPS and front-panel LEDs). Actually the Distribution Amplifier is fed from a dedicated regulator, isolating it from the Arduino (and Fan) feed. A small 12V Fan is part of the PSU; it provides air flow to the regulators in the PSU and the bridge rectifiers.
- **GPS:** Jupiter GPS TU30-D140.
- **OCXO:** Morion MV89A from e-bay (bought three which all work OK).
- **OCXO Board: Own design;** provides a splitter/isolator for the OCXO 10 MHz output signal, and a 12V power supply with a current-sense resistor before the regulator.
- **PLL Board: Own design;** provides the Schmitt-trigger(s) for the OCXO signal
- **Distribution Amplifier:** G4HUP, DA1-4L. I bought the unit pre-built. However, I do not remember, nor seem to have documented, the build options chosen...
- **Arduino:** The V1 release is built around the Arduino Micro. However, we initially designed the circuit around the Arduino Uno.
- **Arduino Interface: Own design;** provides interfaces to GPS (serial), front-panel LEDs, rear-panel RS-232C
- **Front Panel: Own design;** Front-Panel Designer (Schaeffer), 246.5x78 [mm] (WxH).
- **Rear Panel: Own design;** Front-Panel Designer (Schaeffer), 246.5x78 [mm] (WxH).
- **Firmware: Own design;** controls front-panel, monitors GPS (UART communications and FIX status), monitors OCXO current (heater status) and PLL lock status.

5.8 PSU Circuit and PCB Design

The PSU Board circuit and PCB in the V1 release have been designed in KiCAD as a stand-alone project named *pa3gyf-gpsdocxo-2016-psu-r3*. In the figure below (on a separate page) we show its circuit diagram.

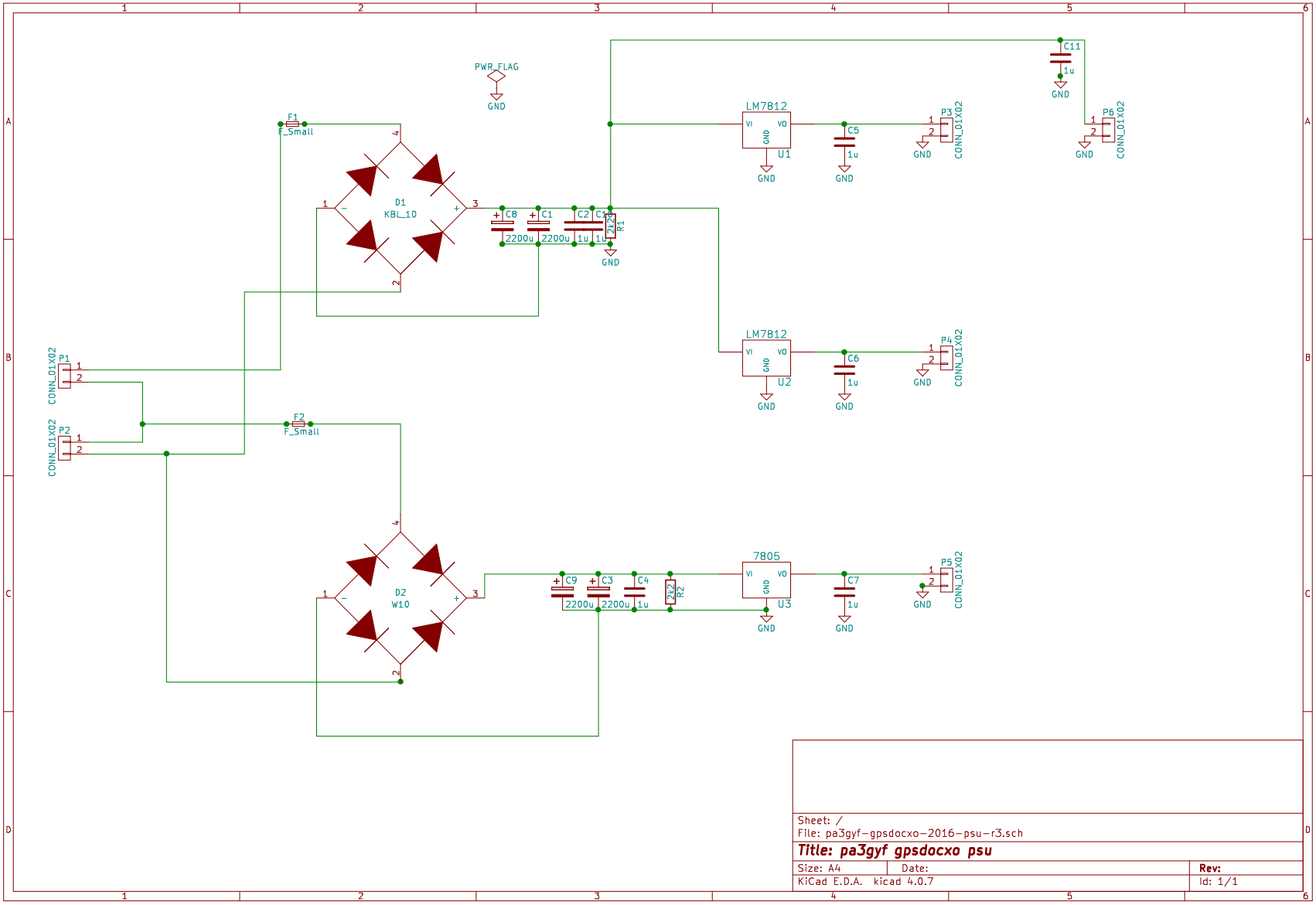
The two secondary windings are put in series and the upper part of the diagram shows how the +21V unregulated and twice +12V voltages are created into P6, P3, and P4 respectively, using a heavy-duty KBL-10 bridge rectifier and a whopping 2x2200 µF with dual 1 µF RF decoupling in a linear design. The 2k2 resistor (R1) acts as a bleeder. The perhaps surprisingly high capacitance value was chosen because the initial heater current is quite high: < 1.5A according to specifications, but about half (800 mA) in all the measurements I took.

The lower part of the diagram shows how the +5V supply is generated using only one the “lower half” of the transformer and another bridge rectifier. However, both capacitor banks are connected to GND on their negative side. This implies that the current while charging one of the two banks of capacitors may actually have a return path through the *other* bridge rectifier. **This is a design error.** The Vishay KBL_10 has maximum rating of 4A average and maximum 1.1V forward voltage per diode (when in forward state). For the Multicomp W10 these values are 1.5A and 1V, respectively, so there a good change that the W10 takes far too much of the total (return) current.

Both upper and lower circuits are equipped with their own (glass) fuse in the secondary circuit mounted on the PCB. **<which value?>**

Although not shown in the diagram, a (+12V) fan for the regulators and bridge rectifiers completes the circuit of the PSU.

Circuit	Voltage	Measured Current	Remarks
Fan	5V		
Arduino Uno shield-gps shield-ocxo	12V	50-60 mA	Somewhat dependent on gps/ocxo settings on front panel. Including gps/ocxo interfacing. Excluding pll interfacing. No LED driving voltage.
Arduino Micro + Board	12V		
Distribution Amplifier	12V	206 mA	Adding 50Ω termination has little effect. (However: measured WITHOUT signal on input).
GPS GPS Antenna FP-LEDs	5V	220 mA <= 10 mA 0 mA 250 mA	GPS and Antenna on/connected OCXO heating (LEDs) GPS+OCXO off GPS/Antenna/OCXO on with 3 LEDs burning and 1 (PPS) flashing.
OCXO	21V unreg	0 mA 810 mA 500 mA 290 mA 280 mA	OCXO off OCXO Heating (RED) OCXO Heating (first signs of GRN) ALMOST GRN GRN


PWR_FLAG
GND
F1
F_Small
D1
KBL_10
C8
2200u
C1
2200u
C2
1u
C4
1u
2x2
R1
LM7812
U1
C5
1u
P3
CONN_01X02
C11
1u
P6
CONN_01X02
LM7812
U2
C6
1u
P4
CONN_01X02
P1
CONN_01X02
P2
CONN_01X02
F2
F_Small
D2
W10
C9
2200u
C3
2200u
C4
1u
2x2
R2
7805
U3
C7
1u
P5
CONN_01X02
Sheet: /
File: pa3gyf-gpsdocxo-2016-psu-r3.sch
Title: pa3gyf gpsdocxo psu
Size: A4
Date:
Rev:
KiCad E.D.A. kicad 4.0.7
Id: 1/1

5.9 OCXO Circuit and PCB Design

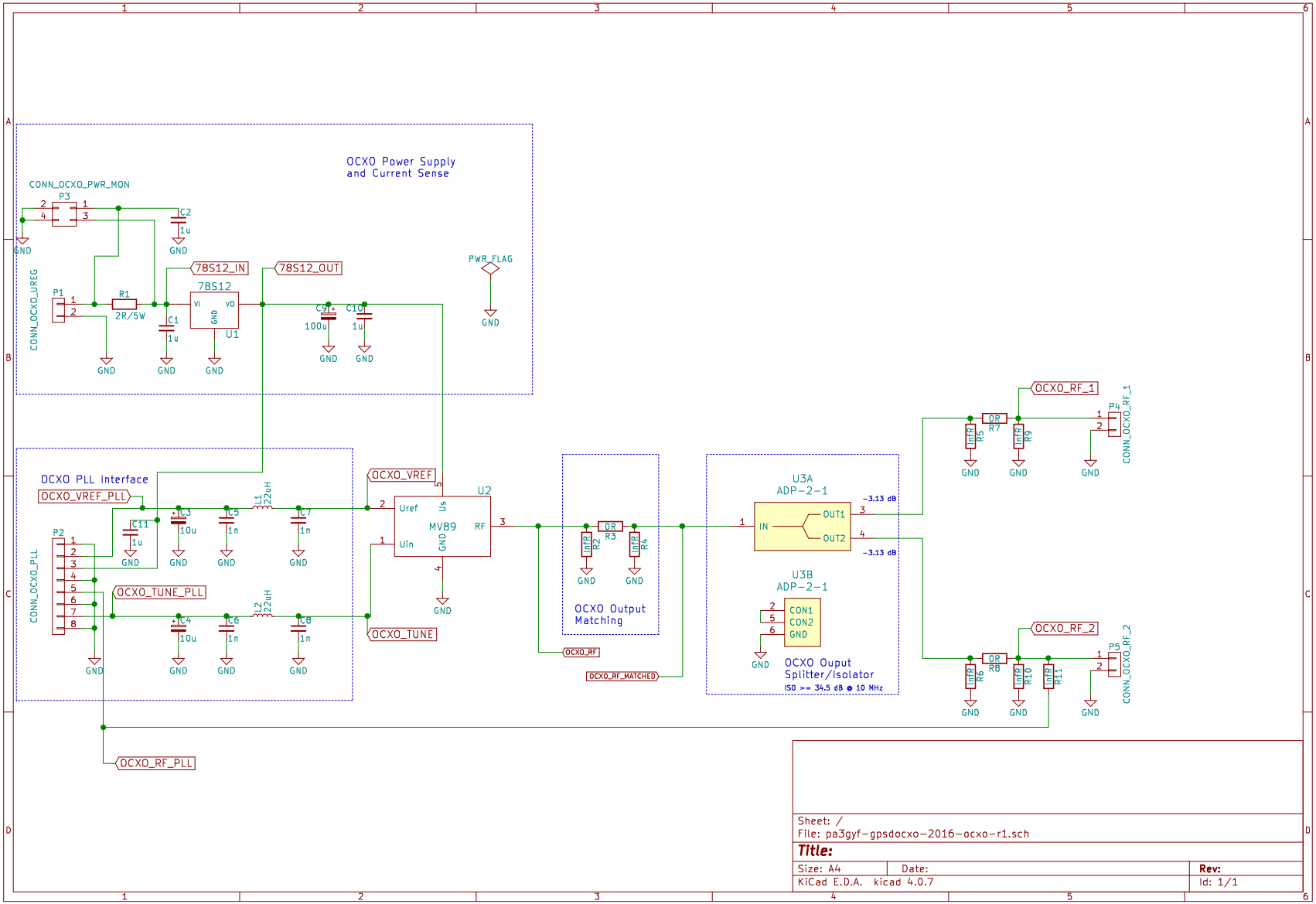
The OCXO circuit houses the OCXO itself, its dedicated 12V power supply from unregulated 21V, its current sensing circuit, and the splitter/isolator circuit of the 10 MHz signal. The OCXO Board circuit and PCB in the V1 release have been designed in KiCAD as a stand-alone project named *pa3gyf-gpsdocxo-2016-ocxo-r1*. In the figure below (on a separate page) we show its circuit diagram.

The circuit receives unregulated 21V power on P1, which is subsequently fed into a 78S12 regulator (U1) after current-sense resistor R1, which is dimensioned at 2R/5W. By specification, the maximum current drawn from the OCXO is 1.5A (maximum peak power consumption at T=25C during warm up), which means the current-sense resistor will hit at most 3V@1.5A = 4.5W, only just within limits. From measurement, however, we found significantly lower heater currents, on the order of 850 mA average heater current, so we may assume that we are safe on the dissipation in R1, also given the fact that the heater typically takes only 15 minutes to heat the OCXO, after which the current drops to a much lower value (typically 300 mA max, @25 degrees C).

The RF output from the OCXO is fed into an optional matching circuit (π) around R2, R3 and R4. I did not construct the circuit and simply used a small piece of wire as R3. (Also, I do not have the equipment required to measure the impedance matching.) The signal is then fed into the Splitter/Isolator U3A, half an APD-2-1. Both outputs of U3A are then fed to optional π filters for impedance matching or (for instance) band-pass or low-pass filtering. I did not construct these filters. They are available on P4 and P5, respectively.

Both the OCXO (internal) reference voltage and the tune input (Uin) are filtered and fed to the PLL connector P2. Optionally, through R11, the second output (P5) can also be routed to P2. However, I decided to use coaxial cabling and SMA connectors to bring the OCXO signal to the GPS board. Apart from not really feeling comfortable with routing the critical 10 MHz signal through a SIL connector, this approach has the additional advantage of being able to cut the connection and measure the output from the APD-2-1, or test the PLL circuit with an alternative 10 MHz VCO signal.

Finally, the 12V output supply from the 78S12 (U1) is also fed into a pin on P2 (as well as, obviously, GND). This allows the PLL board to be fed from the OCXO board (see further, the PLL board can host its private 7805 regulator using the 12V from the OCXO board). I did not use this option and fed the OCXO and PLL boards separately from the PSU.


OCXO Power Supply
and Current Sense
CONN_OCXO_PWR_MON
P3
C2
1u
GND
78S12_IN
78S12_OUT
78S12
PWR_FLAG
P1
R1
2R/5W
C1
1u
U1
C9
100u
C10
1u
CONN_OCXO_UREG
OCXO PLL Interface
OCXO_VREF_PLL
OCXO_TUNE_PLL
OCXO_RF_PLL
P2
CONN_OCXO_PLL
C11
C3
10u
C5
1n
L1
22uH
C7
C4
C6
L2
C8
OCXO_VREF
OCXO_TUNE
U2
MV89
Uref
Uin
Us
RF
GND
OCXO Output
Matching
R2
R3
R4
OCXO_RF
OCXO_RF_MATCHED
U3A
ADP-2-1
IN
OUT1
OUT2
-3.13 dB
U3B
ADP-2-1
CON1
CON2
GND
OCXO Ouput
Splitter/Isolator
ISO >= 34.5 dB @ 10 MHz
OCXO_RF_1
R5
R7
R9
P4
CONN_OCXO_RF_1
OCXO_RF_2
R6
R8
R10
R11
P5
CONN_OCXO_RF_2
Sheet: /
File: pa3gyf-gpsdocxo-2016-ocxo-r1.sch
Title:
Size: A4
Date:
Rev:
KiCad E.D.A. kicad 4.0.7
Id: 1/1

5.10 PLL Circuit and PCB Design

The PLL circuit houses various circuits related to providing a tune voltage to the OCXO, including signal-shaping, the frequency dividers for the 10 MHz signal, the phase-detector, the loop filter, lock-monitoring circuitry as well as provisions for manual tuning. The PLL Board circuit and PCB in the V1 release have been designed in KiCAD as a stand-alone project named *pa3gyf-gpsdocxo-2016-pll-r2*. In the figure below (on a separate page) we show its circuit diagram.

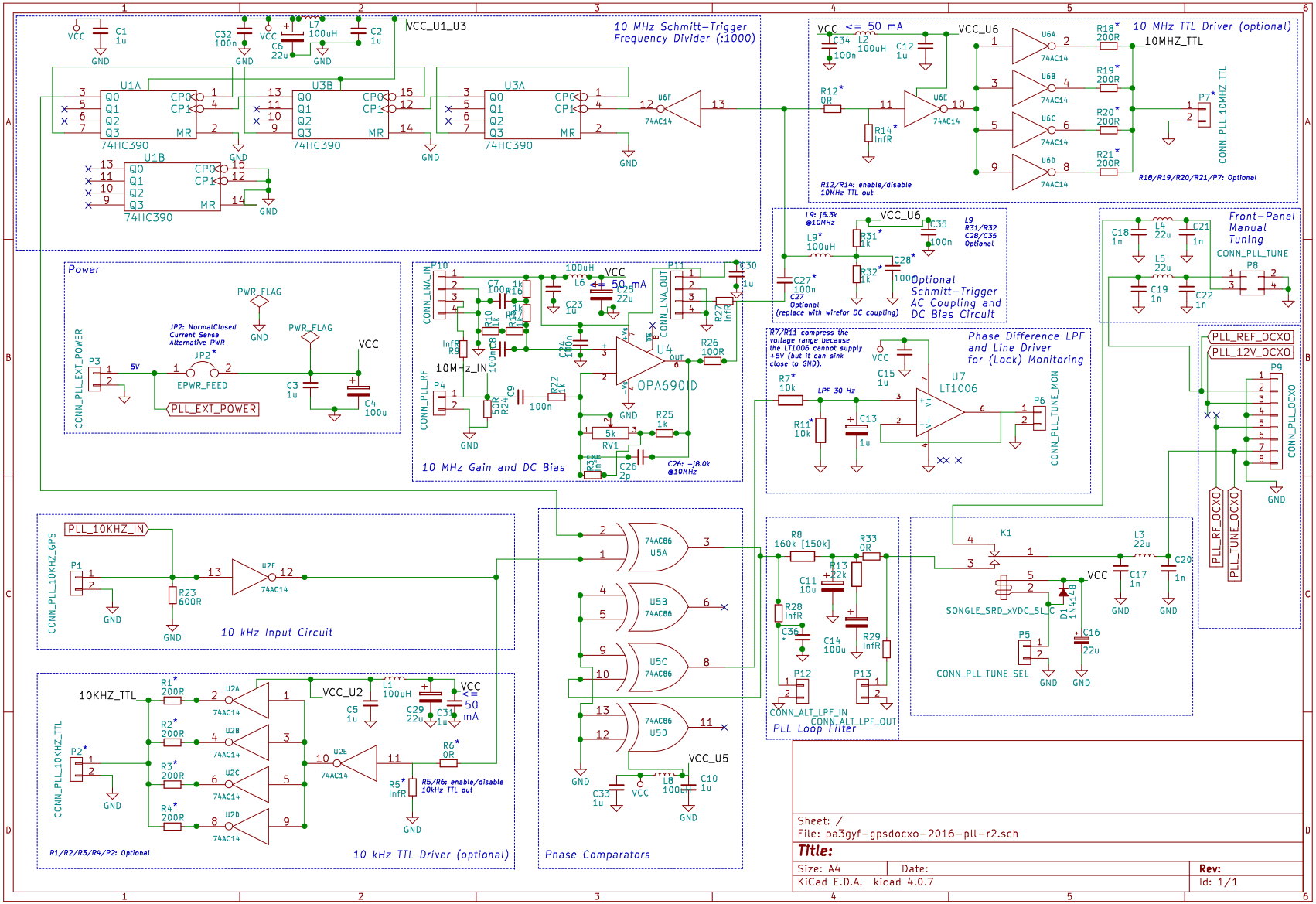
The circuit is basically a heavily beefed up version of the G3RUH design. U5A is the phase comparator and in a way, the core of the circuit. Its output goes into the loop filter R8/C11/R13/C14. The loop-filter circuit features the option of an alternative loop filter through P12 and P13 and supporting components. The output of the loop filter, i.e., the tune voltage for the OCXO goes through K1 and the C17/L3/C20 filter to P9, the main connection with the OCXO board. K1 is controlled from the Arduino control circuitry, and switches between PLL and manual tuning. In manual tuning, the reference voltage from the OCXO, available on P9, is fed through P8 to the panel-mounted 10-turn resistor and the voltage on its slider is fed back through P8 and via K1 and P9 fed to the OCXO tuning input.

At one input of U5A we find the 10 kHz reference signal from the GPS receiver arriving at P1 and buffered by U2F. At the other input we find the 10 kHz signal from the OCXO, obtained by the divider and shaper circuit around U1, U3 and U6F. At the input of U6F is an optional bias circuit R31/R32/C28/L9 and the LNA circuit for the OCXO 10 MHz signal around U4. Optionally, the LNA can be replaced with a suitable circuit on P10/P11. The 10 MHz from the OCXO is fed into P4 and the 50Ω terminator R24. The PCB-mounted RV1 control the gain of U4. Note that RV1 may be replaced with a fixed resistor R30. With given values, the voltage gain of U6 is selectable from x1 to x6, but this can be increased through increasing R25 and/or RV1. Note that the output DC bias is 0.5xVcc, or 2.5V.

The boards supports two TTL-level outputs. The optional circuit around U6A through U6E provides a TTL-level 50Ω OCXO 10 MHz output. I built this circuit and made the signal available on the front panel. Similarly, the circuit around U2A through U2E provides a TTL-level 50Ω GPS 10 kHz output which I did *not* build.

For monitoring, U5C buffers the output from the main phase comparator U5A and feeds it into a monitoring filter around R7/R11/C13, allowing to monitor the output from the phase comparator at much higher frequency (the time constant of the monitor filter is much lower than the main PLL loop filter). U7 acts as an instrumentation amplifier. Note that R7/R11 divide the output of the filter by two. This is needed because U7 cannot be driven to its positive rail.

Finally, main power is entered on P3. Jumper JP1 has to be shortened for the circuit to be powered at all. The jumper allows for current-consumption monitoring, but is really a remnant from earlier versions of the board in which it was powered, optionally, from the OCXO board.


10 MHz Schmitt-Trigger Frequency Divider (:1000)
74HC390
VCC_U1_U3
10 MHz TTL Driver (optional)
10MHZ_TTL
VCC_U6
VCC <= 50 mA
R12/R14: enable/disable 10MHz TTL out
R18/R19/R20/R21/P7: Optional
Power
PWR_FLAG
JP2: NormalClosed Current Sense Alternative PWR
EPWR_FEED
PLL_EXT_POWER
CONN_PLL_EXT_POWER
Optional Schmitt-Trigger AC Coupling and DC Bias Circuit
Front-Panel Manual Tuning
CONN_PLL_TUNE
10 MHz Gain and DC Bias
OPA690ID
10MHz_IN
CONN_LNA_IN
CONN_LNA_OUT
CONN_PLL_RF
R7/R11 compress the voltage range because the LT1006 cannot supply +5V (but it can sink close to GND).
Phase Difference LPF and Line Driver for (Lock) Monitoring
LT1006
LPF 30 Hz
CONN_PLL_TUNE_MON
PLL_REF_OCXO
PLL_12V_OCXO
CONN_PLL_OCXO
10 kHz Input Circuit
PLL_10KHZ_IN
CONN_PLL_10KHZ_GPS
10 kHz TTL Driver (optional)
10KHZ_TTL
CONN_PLL_10KHZ_TTL
R1/R2/R3/R4/P2: Optional
R5/R6: enable/disable 10kHz TTL out
Phase Comparators
74AC86
PLL Loop Filter
CONN_ALT_LPF_IN
CONN_ALT_LPF_OUT
SONGLE_SRD_xVDC_SL_C
CONN_PLL_TUNE_SEL
Sheet: /
File: pa3gyf-gpsdocxo-2016-pll-r2.sch
Title:
Size: A4
Date:
Rev:
KiCad E.D.A. kicad 4.0.7
Id: 1/1

5.11 Arduino [Micro] Interface Board Circuit and PCB Design

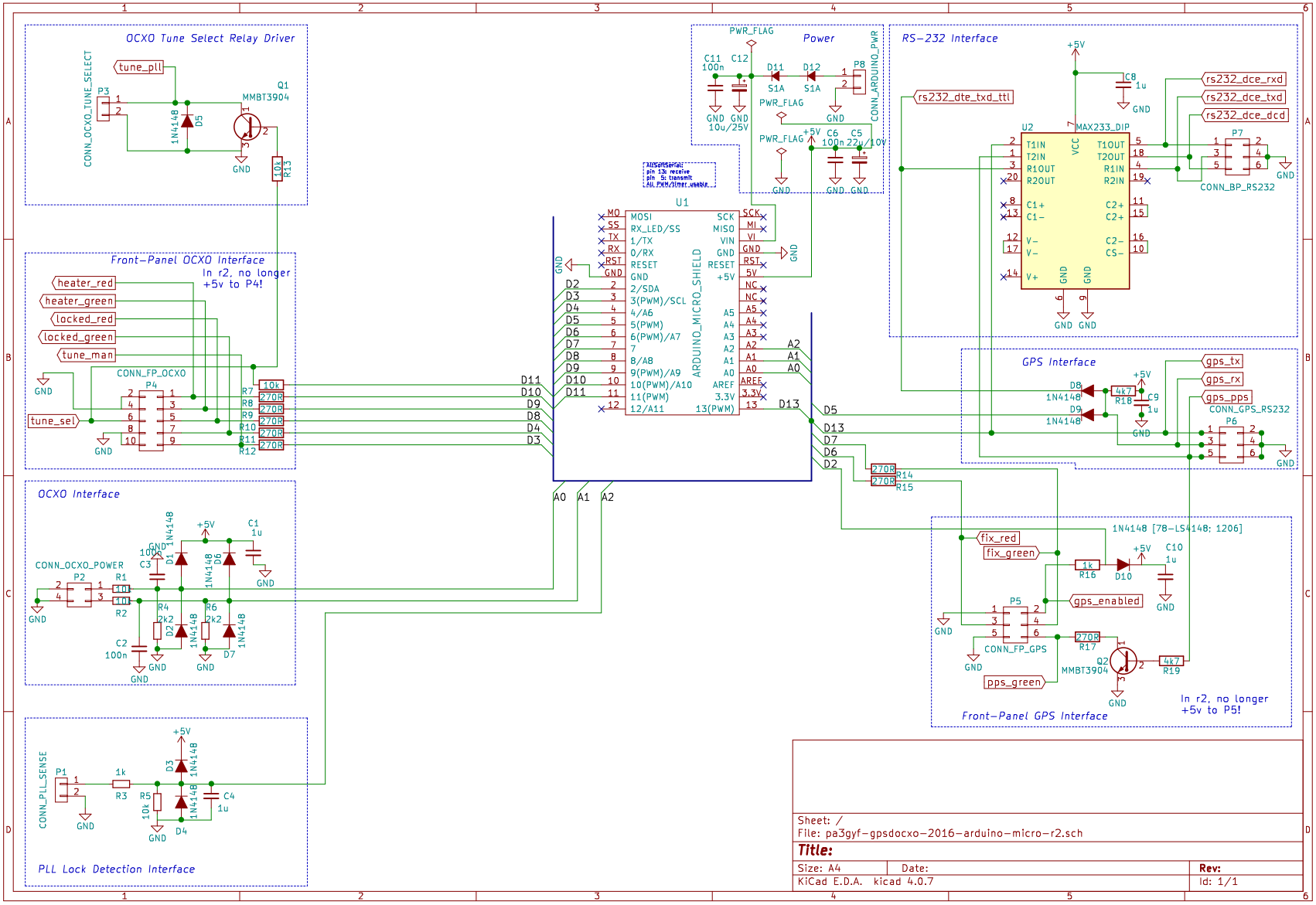
The unit can be constructed with either an Arduino Micro, as described in this section, or an Arduino Uno, described in the next section. The Arduino Micro build consists of an Arduino [Micro] Interface Board holding an Arduino Micro in a socket. The board provides interfaces to GPS serial, OCXO power sensing, PLL lock sensing, RS-232, front-panel switches and LEDs and the tuning-select relay on the PLL board. Its circuit and PCB in the V1 release have been designed in KiCAD as a stand-alone project named *pa3gyf-gpsdocxo-2016-arduino-micro-r2*. In the figure below (on a separate page) we show its circuit diagram.

The Arduino Micro can be safely fed from a 12V supply, but we reduced the supply voltage somewhat with D11 and D12. I do not recall why this was deemed necessary, honestly, though I vaguely remember having thermal trouble with the Arduino in relation to LED driving.

Most of the rest of the circuit is quite straightforward. U2 provides the level conversion to RS232-C, available on P7. The serial data from the GPS, connected through P6, is fed both into D13 on the Arduino Micro and U2. The serial data *to* the GPS, however, is the diode-OR-ed TTL signal from D5 on the Arduino Micro (through diode D9) and TXD signal (seen from the DCE) from the RS232-C interface through P7 on the rear panel (through diode D8).

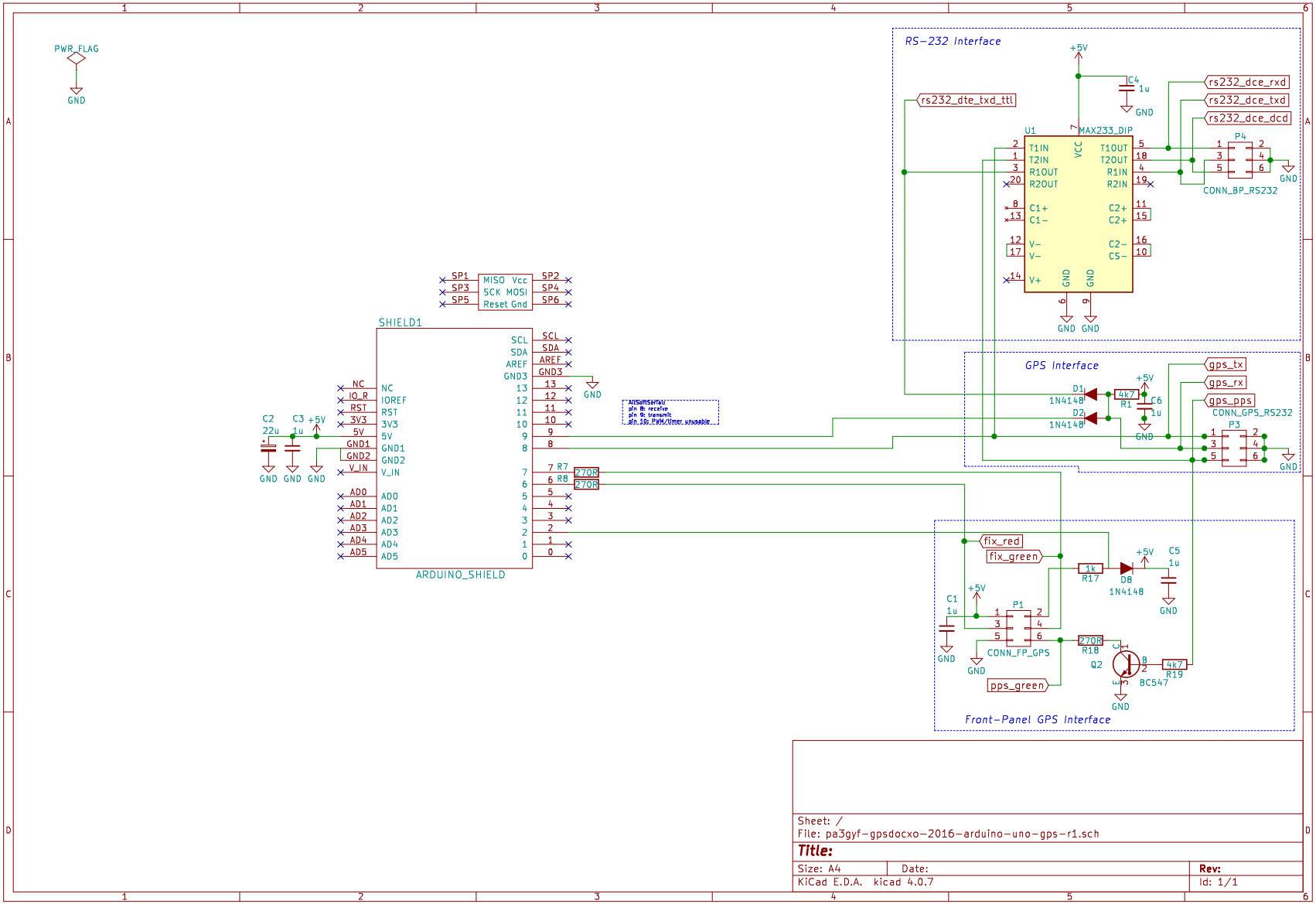
All LEDs obtain +5V DC directly from the power-supply unit; the Arduino Micro “only” needs to sink a LED’s current in order to activate it.

The OCXO current-sense signals enter on P2. Both are RF-filtered (R1/C3 and R2/C2), scaled down by, roughly, a factor 5.5 (R1/R4 and R2/R6), and clamped for additional Arduino input protection, to the positive rail and GND (D1/D2/D6/D7). Note that both input voltages are from the unregulated 15V (2x7.5V) supply, and may therefore top at 21.2V.


OCXO Tune Select Relay Driver
CONN_OCXO_TUNE_SELECT
tune_pll
Q1
MMBT3904
1N4148
D5
10k
R13
Front-Panel OCXO Interface
In r2, no longer
+5v to P4!
heater_red
heater_green
locked_red
locked_green
tune_man
tune_sel
CONN_FP_OCXO
P4
10k
270R
OCXO Interface
CONN_OCXO_POWER
P2
1N4148
PLL Lock Detection Interface
CONN_PLL_SENSE
P1
1k
10k
Power
PWR_FLAG
CONN_ARDUINO_PWR
P8
S1A
U1
ARDUINO_MICRO_SHIELD
MOSI
SCK
MISO
RX_LED/SS
1/TX
0/RX
RESET
GND
+5V
2/SDA
3(PWM)/SCL
4/A6
5(PWM)
6(PWM)/A7
7
8/A8
9(PWM)/A9
10(PWM)/A10
11(PWM)
12/A11
13(PWM)
AREF
3.3V
A0
A1
A2
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D13
RS-232 Interface
rs232_dte_txd_ttl
rs232_dce_rxd
rs232_dce_txd
rs232_dce_dcd
U2
MAX233_DIP
T1IN
T2IN
R1OUT
R2OUT
T1OUT
T2OUT
R1IN
R2IN
C1+
C1-
C2+
C2-
V-
V+
VCC
CS-
P7
CONN_BP_RS232
GPS Interface
gps_tx
gps_rx
gps_pps
CONN_GPS_RS232
P6
4k7
R18
1N4148 [78-LS4148; 1206]
fix_red
fix_green
gps_enabled
pps_green
CONN_FP_GPS
P5
1k
R16
270R
R17
Q2
MMBT3904
4k7
R19
In r2, no longer
+5v to P5!
Front-Panel GPS Interface
Sheet: /
File: pa3gyf-gpsdocxo-2016-arduino-micro-r2.sch
Title:
Size: A4
Date:
Rev:
KiCad E.D.A. kicad 4.0.7
Id: 1/1

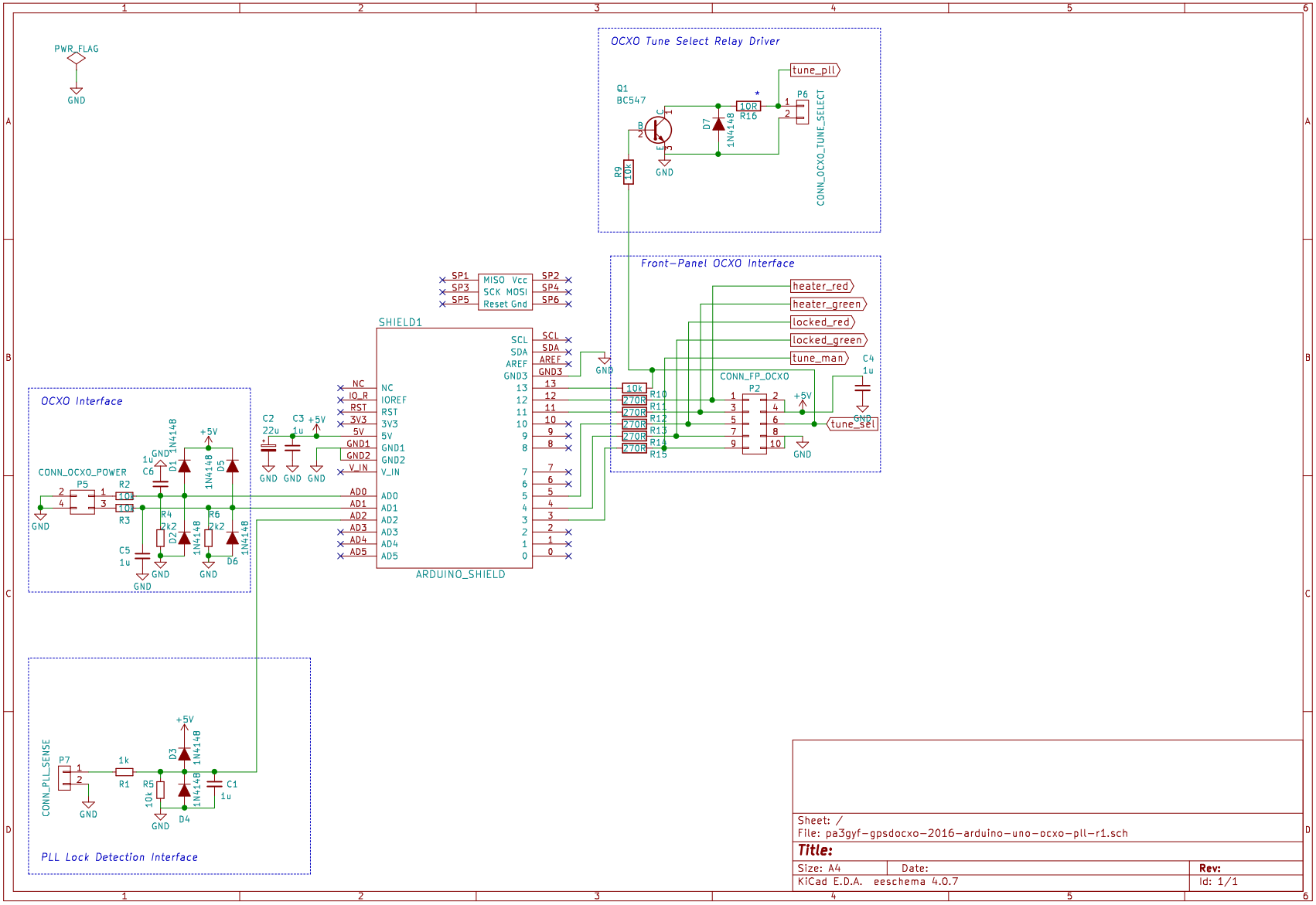
5.12 Arduino [Uno] GPS Interface Board Circuit and PCB Design

As mentioned in the previous section, alternatively, the unit can be constructed with an Arduino Uno. The Arduino Uno build, described in this and the next sections, consists of an Arduino Uno with two shields, one for the GPS and one for the PCXO/PLL. The GPS interface circuit and PCB for the Arduino in the V1 release have been designed in KiCAD as a stand-alone project named *pa3gyf-gpsdocxo-2016-arduino-uno-gps-r1*. In the figure below (on a separate page) we show its circuit diagram.


PWR_FLAG
GND
RS-232 Interface
+5V
C4
1u
GND
rs232_dte_txd_ttl
rs232_dce_rxd
rs232_dce_txd
rs232_dce_dcd
U1
MAX233_DIP
T1IN
T2IN
R1OUT
R2OUT
C1+
C1-
V-
V-
V+
VCC
GND
GND
T1OUT
T2OUT
R1IN
R2IN
C2+
C2+
C2-
C5-
P4
CONN_BP_RS232
SP1
SP3
SP5
MISO Vcc
SCK MOSI
Reset Gnd
SP2
SP4
SP6
SHIELD1
SCL
SDA
AREF
GND3
NC
IO_R
RST
3V3
5V
GND1
GND2
V_IN
AD0
AD1
AD2
AD3
AD4
AD5
ARDUINO_SHIELD
C2
22u
C3
1u
R7
R8
270R
270R
GPS Interface
gps_tx
gps_rx
gps_pps
D1
D2
1N4148
1N4148
4k7
R1
C6
1u
CONN_GPS_RS232
P3
Front-Panel GPS Interface
fix_red
fix_green
pps_green
1k
R17
D8
1N4148
C5
1u
C1
1u
P1
CONN_FP_GPS
270R
R18
Q2
BC547
4k7
R19
Sheet: /
File: pa3gyf-gpsdocxo-2016-arduino-uno-gps-r1.sch
Title:
Size: A4
Date:
Rev:
KiCad E.D.A. kicad 4.0.7
Id: 1/1

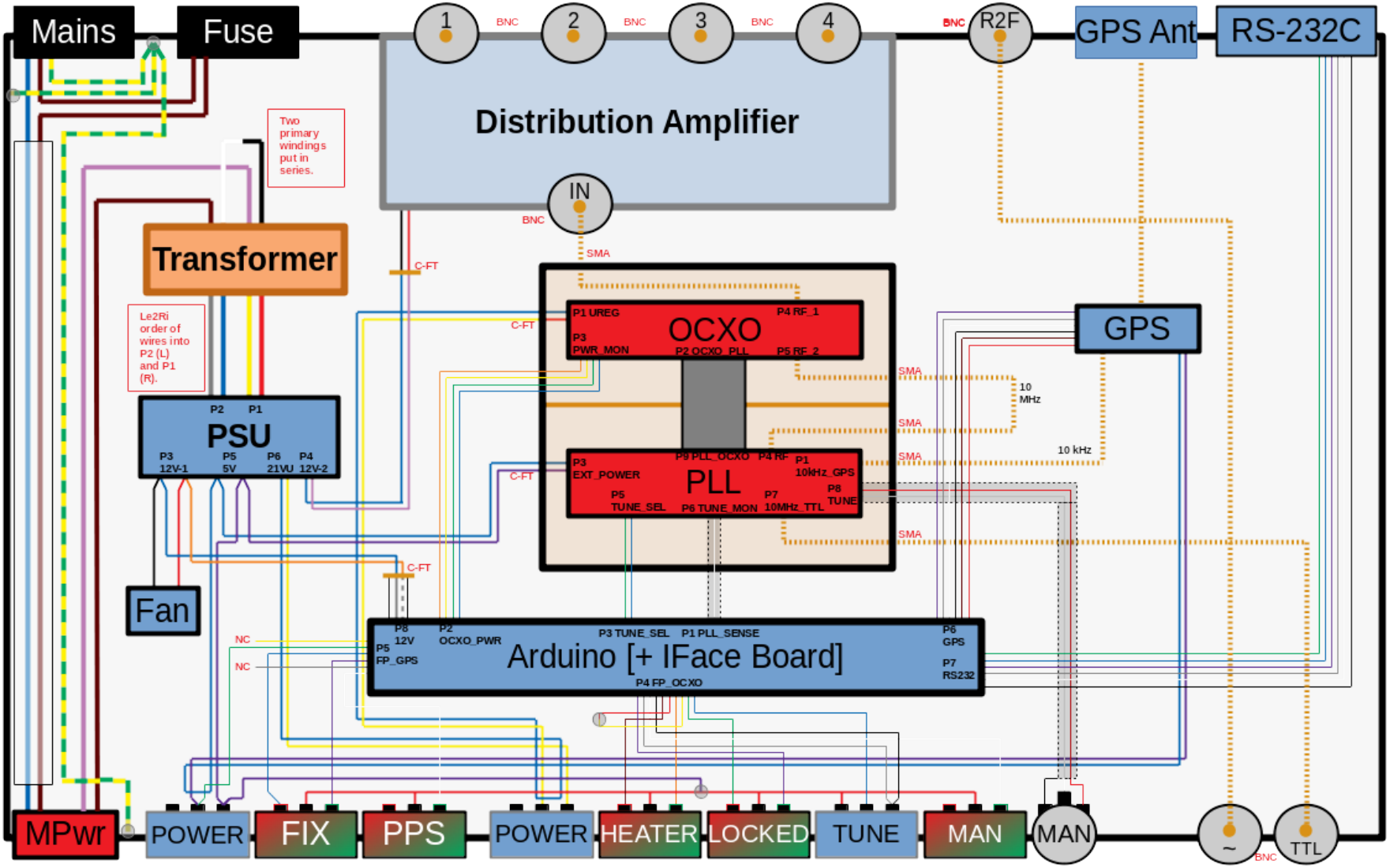
5.13 Arduino [Uno] OCXO/PLL Interface Board Circuit and PCB Design

As mentioned in the previous section, alternatively, the unit can be constructed with an Arduino Uno instead of an Arduino Micro. The Arduino Uno build, described in this and the previous sections, consists of an Arduino Uno with two shields, one for the GPS and one for the OCXO/PLL. The OCXO/PLL interface circuit and PCB for the Arduino in the V1 release have been designed in KiCAD as a stand-alone project named *pa3gyf-gpsdocxo-2016-arduino-uno-ocxo-pll-r1*. In the figure below (on a separate page) we show its circuit diagram.


PWR_FLAG
GND
OCXO Tune Select Relay Driver
Q1
BC547
tune_pll
10R
R16
P6
D7
1N4148
R9
10K
CONN_OCXO_TUNE_SELECT
Front-Panel OCXO Interface
heater_red
heater_green
locked_red
locked_green
tune_man
tune_sel
C4
1u
CONN_FP_OCXO
P2
10k
270R
R10
R11
R12
R13
R14
R15
+5V
SP1
SP3
SP5
MISO Vcc
SCK MOSI
Reset Gnd
SP2
SP4
SP6
SHIELD1
NC
IO_R
RST
3V3
5V
GND1
GND2
V_IN
AD0
AD1
AD2
AD3
AD4
AD5
SCL
SDA
AREF
GND3
ARDUINO_SHIELD
C2
22u
C3
1u
OCXO Interface
CONN_OCXO_POWER
P5
R2
R3
10k
R4
R6
2k2
C5
C6
D1
D2
D5
D6
1N4148
CONN_PLL_SENSE
P7
1k
R1
R5
10k
C1
D3
D4
PLL Lock Detection Interface
Sheet: /
File: pa3gyf-gpsdocxo-2016-arduino-uno-ocxo-pll-r1.sch
Title:
Size: A4
Date:
Rev:
KiCad E.D.A. eeschema 4.0.7
Id: 1/1

5.14 Interconnection Diagram [Arduino Micro]

The interconnection diagram for the Aduino Micro build is shown in the figure below (on a separate page).


Mains
Fuse
1
BNC
2
BNC
3
BNC
4
BNC
R2F
GPS Ant
RS-232C
Distribution Amplifier
IN
BNC
SMA
Two primary windings put in series.
Transformer
Le2Ri order of wires into P2 (L) and P1 (R).
C-FT
P2 P1
PSU
P3 12V-1
P5 5V
P6 21VU
P4 12V-2
P1 UREG
P4 RF_1
P3 PWR_MON
OCXO
P2 OCXO_PLL
P5 RF_2
GPS
SMA
10 MHz
10 kHz
P3 EXT_POWER
P9 PLL_OCXO
P4 RF
P1 10kHz_GPS
PLL
P5 TUNE_SEL
P6 TUNE_MON
P7 10MHz_TTL
P8 TUNE
Fan
P8 12V
P2 OCXO_PWR
P3 TUNE_SEL
P1 PLL_SENSE
P6 GPS
P5 FP_GPS
Arduino [+ IFace Board]
P7 RS232
P4 FP_OCXO
NC
NC
MPwr
POWER
FIX
PPS
POWER
HEATER
LOCKED
TUNE
MAN
MAN
~
BNC
TTL

6 Miscellaneous Notes

The core of the project consists of a Morion MV89A OCXO and a suitable Rockwell / Conexant / Navman Jupiter GPS receiver (I'm using a Conexant, marked TU30-D140), and is more or less equal to the design of James Miller G3RUH, see http://http://www.jrmiller.demon.co.uk/projects/ministd/frqstd0.htm and a distribution amplifier after G4HUP, see http://www.g4hup.com/DA/DA1_4.htm. The OCXO is a 10 MHz version and has a +/- 2.5 Hz pulling range and internal +5V reference voltage. The nice thing about the GPS receiver used is that it has a highly stable 10 kHz TTL output, in addition to the classic (1 Hz) PPS signal.

Although I bought and assembled James' PCB, I felt the need to change a couple of things. Relative to James' version, I added/modified the following (the applicable PCB is shown between square brackets):

- [pa3gyf-gpsdocxo-arduino-gps] An interface ('shield') to an Arduino (Uno/R3) for monitoring the NMEA strings sent out by the GPS Receiver on its TXD pin (TTL). The Arduino is also capable of sending command strings to the GPS receiver. The RS-232 interface (including 1PPS on the DCD pin) to a host system is still present (e.g., for use with NTP). The shield also interfaces to the GPS-section on the front-panel.
- [pa3gyf-gpsdocxo-ocxo] A dedicated 78S12-based +12V power supply for the OCXO with a series resistor for sensing the OCXO current (and thus its heater status). The OCXO itself is also on this board. The 10 MHz output is split in two using an ADP-2-1W splitter/combiner in order to isolate the PLL spurious signals from the (sine-wave) signal path to the DA.
- [pa3gyf-gpsdocxo-arduino-ocxo] An interface ('shield') to an Arduino (Uno/R3) for monitoring the OCXO current/heater status, and for checking the PLL lock status (using one of the Arduino's ADC's). The shield also interfaces with the OCXO-section on the front-panel, and with the PLL board (e.g., for switching between manual mode and PLL mode).
- [pa3gyf-gpsdocxo-psu] A power supply for the 12V (Arduino/DA) and 5V (GPS/GPS Antenna/PLL/FP LEDs) circuits. Optionally, the PLL can also be fed from the OCXO board directly through an on-board 7805.
- [pa3gyf-gpsdocxo-pll] The PLL board with (optional) relay-based switching between manual and PLL mode (for the OCXO), an (optional) on-board 7805 power-supply drawing from the OCXO board, and some serious filtering. The OCXO and PLL boards are “sandwiched” (see photos). The PLL board also provides (optionally) TTL versions of the 10MHz OCXO output and the 10kHz reference signal from the GPS.
- [pa3gyf-gpsdocxo-lna] A Low-Noise Amplifier (and DC-bias provisioning) for the 10 MHz from the OCXO to the PLL (ST) input. For this board, no PCB design is foreseen. Perhaps, in the future, the circuit will be integrated onto the OCXO or PLL boards.

6.1 PCB Overview

The table below summarizes the PCBs/kits in the project:

Code	Version	Function
[pa3gyf-gpsdocxo-psu]	3	Power supply (+5V, +12V) for all circuits; unregulated output (+21V) for the dedicated OCXO PSU.
Arduino Uno R3 with pin headers	R3	Arduino controller.
[pa3gyf-gpsdocxo-arduino-gps]	1	Interface to GPS receiver for FIX status monitoring and to FP GPS section.
[pa3gyf-gpsdocxo-arduino-ocxo]	1	Interface to OCXO and PLL and to FP OCXO section.
Rockwell/Connexant/Navman Jupiter TU30-D140-xxx Jupiter TU30-D410-xxx	*	GPS Receiver with 10 kHz reference output (AND proper version of the firmware, see the G3RUH reference).
[pa3gyf-gpsdocxo-ocxo]	1	OCXO board for Morion MV89A (with some extra drilling, I *assume* a STP2145A could be made to fit as well, but unfortunately I did not add this feature in Version 1 of the PCB).
[pa3gyf-gpsdocxo-pll]	1	The PLL board. There are some known issues with the board, but they appear to be fixable.
Transformer		Toroid, 2x7.5V/2A (yes, this is a bit high in voltage, and a bit tight in power, 30W; still to be tested…).

6.2 R1 Finalization and Evaluation

Just before Christmas 2016, I completed the entire R1 circuitry, and started testing it; doing Arduino firmare development on the fly. I ran into the following problems:

- The PSU heatsink reached 65 °C with the lid open, too high to my taste. I decided to change the internal construction somewhat in order to facilitate a small (4 mm, I think) fan, blowing air from below. The heatsink temparature now settles at 30-35 °C, a significant relief for my thermal worries. I learned a lot from the following statement (in my own words, to be confirmed) I found on Internet: “A heatsink is not a magical black hole that will swallow all excessive heat”. No, you need a plan the get the excessive heat to the outside world.
- One of the bridge rectifiers (the one taking the full voltage, i.e., the one rectifying the series circuit on the secondary of the transformer) still gets a bit too hot to my taste (> 65 °C), even in presence of the fan. I drilled some extra holes nearby, but it did not seem to improve things a lot… But it seems to hold for now, and I therefore decided to defer a solution for this.
- The 10 MHz from the OCXO (after the 3 dB splitter/combiner loss) is not strong enough to drive the Schmitt-Triggers on the PLL board. Bummer… Also, I had forgotten to add bias voltage (2.5V) to the input of the ST. Because I really wanted to first test the LOCK status (and its detection) on the output of the XOR gate, and start designing the LOCK-detection software, I assembled a 50Ω OPAMP-based single-rail variable-gain amplifier using

Through-Hole technology. A bit to my surprise, this turned out harder than I expected. First, I never realized that x5 (voltage) amplification at 10 MHz with a single rail is quite challenging for “old-school” OPAMPS like the 741, CA3140, and the like. I also started to worry about phase-noise performance, but decided to ignore it somewhat for now. Also, intuitively, phase noise in this section of the PLL circuit does not play such a large role due to the excessive low-pass filtering following the phase comparator. (This statement is yet to be confirmed by a more thorough theoretical analysis; please contact me if you have a strong opinion about this…) In the end, though, the OPAMP amplifier worked, and the Schmitt-Triggers could be activated.

- The Arduino UNO with the two shields mouted on top of it turned out to not fit into the enclosure. Bummer, but this was a risk I had taken.
- The available SRAM on the Arduino Uno is critically low, 2 kB. I had to resort to memory-saving algorithms quite early in the firmware design.
- There seems to be a 50 Hz ripple on the 10 MHz output during OCXO heating (still to be confirmed). An indication that somewhere, the PSU design is not adequate. Once the OCXO is heated, the ripple goes away.
- Spectral purity still to be determined.
- Phase noise still to be determined.
- Power Measurements: <TBW>

In the end, through, everything worked! I managed to detect a GPS FIX (using AltSoftSerial and TinyGPS 3rd-party libraries) quite rapidly, and finalized the firmware for these functions (about an evening of effort). The LOCK detection took a bit more effort, due to the SRAM limitation mentioned earlier, but it worked quite well in the end.

6.3 R2 Rationale and Design

R1 Limitations and Errors:

- The controller part, viz., the Arduino UNO with its two shields is too high to fit into my enclosure. Also, the construction does not appear to be very RF-friendly, and there is a clash with the shields and the USB and POWER connectors on the Arduino UNO.
- Front-panel LEDS should *not* be powered from the Arduino regulated +5V.
- The OCXO output needs to be amplified/swept up.

- The PLL 10 MHz Schmitt-Trigger circuitry needs +2.5V DC bias.
- The use of the regulated 12V from the OCXO circuit on the PLL is really a bad idea; it interferes with the current-detection circuits and software, and is really not needed since we have +5V available from the PSU.
- Feeding the 10MHz OCXO output (after the splitter/combiner) through the SIL header is also a bad idea. Better use an appropriate 50Ω coaxial connection.

6.4 R2 Changes

Compared to R1, the (System-Level) R2 has changes to the Arduino and its shields, and the PLL.

6.5 Arduino and Shields

6.6 PLL

Number	Change (wrt R1)	Remark
Circuit		
C.1	Removed connection to pin 3 (labeled PLL_12V_OCXO) on P9 (CONN_PLL_OCXO).	Do not use +12V power supply from OCXO; always use an external +5V power supply. Leave in a jumper (re-designated JP2) for current measurements.
C.2	Removed JP1 (I/E POWER).	
C.3	Moved JP2 to connect PLL_EXT_POWER to VCC, and renamed JP2 as EPWR_FEED.	
C.4	Removed U4 (7805), C8, C9, and labels 7805_IN and 7805_OUT.	
C.5	Removed connection to pin 5 (labeled PLL_RF_OCXO) on P9 (CONN_PLL_OCXO).	Removed the option to feed the 10 Mhz OCXO signal through P9.
C.6	Removed R9.	
C.7	Removed R10, replaced with trace connection.	
C.8	Renamed P4 to CONN_PLL_RF.	
C.9	Deleted R15.	Removed the option to bypass the tune-select relay.
C.10	Deleted R16 and replaced with trace connection.	
C.11	Deleted R17 and replaced with trace connection.	
C.12	Deleted P10 and P11 and replaced with trace connection.	Removed intra-PCB wire connection.

Number	Change (wrt R1)	Remark
PCB		
P.1	Smaller footprint for C4 and C14 (P28 → P18).	

7 Errata for V1

This section contains errata and improvement suggestions for and further documentation on pa3gyf-gpsdocxo-v1. All known and verified errata are listed first. Subsequent sections describe test, measurement and fix sessions. Or just ideas, observations, or brain dumps.

- **arduino-micro-r2:**
  - **R13 must be replaced by a 2k2 resistor (instead of 10k).** This solves the problem of the PLL/MAN relay failing to activate because it does not receive sufficient coil power (see further in this document).
  - **The footprint of Q1 (MMBT3904) is wrong on the PCB [arduino-gpsdocxo-arduino-r2].** The transistor has to be rotated in order for it to work properly. Looking from above at the transistor, with the single pin in north position, the single N pin is C (collector), the LEFT pin in south position is B (base), whereas the right pin is E (emitter).

8 Test Sessions for V1

8.1 20190921: The R13 and Gravity Problem [solved]

- **Background:** Despite GPS FIX and proper OCXO heating, the gpsdocxo-v1 often fails to lock in PLL mode. Then again, often, it does lock perfectly often. My initial thoughts were that the OCXO simply could not always be 'locked', perhaps it had to be pulled outside of its tuning range. Or I made have made an error in the Arduino code.
- **What happened next?** As it "sometimes works just perfectly", I did not pay too much attention to the problem with the gpsdocxo. Getting the OCXO locked to the GPS (following my criteria of "locked) may take like 15 minutes. However, after the purchase of several beautiful new instruments from an auction at work, I wanted to test in some more detail their frequency stability, so I *had to* have the gpsdocxo operational and locked. Since it did not work well, I decided to spend time on trying to fix it.
- **Observation 1:** While playing with the device, switching between PLL and MAN mode, I accidentally turned the TUNE pot while the device was in PLL mode, and discovered that I could fine-tune the output frequency. This, however, is not supposed to happen… The tune pot should have *no effect whatsoever* while in PLL mode.
- **Observation 2:** After opening the device and starting to measure some key voltages, I realized that the device was working just perfectly. One way or another, I found out that GRAVITY plays a key role in this problem: While the unit was placed upside-down on my desk, it would work just fine, whereas turning the unit in its normal position would often make the unit fail.
- **Red Flag Conclusion:** The PLL/MAN relay on the pll PCB does *not always activate if it should.*
- **Confirmation of Relay Activation Problem:** Luckily, this was quite easy to confirm. For one, while in PLL mode, I measured the voltage across the pll/p5 connector (effectively the CE voltage of the driving transistor supposed to be saturated while activating relay K1 in PLL mode) and measured **1.4V**. This is *way too high* for a transistor supposed to in saturation. The current measured was **48 mA**. (The spec says 80 mA is required.) Connecting a DC amp meter (my newly acquired hp3478a) across pll/p5 shows **63 mA**. Hence, we hypothesize that arduino/Q1 needs more base current in order to saturate and properly activate the relay pll/K1.
- **Solution:** After consulting the MMBT3904 and the Songle SRD-5VDC-SLC data sheets, I realized that the transistor was not supplied with sufficient base current for it to saturate and active the relay. Surprisingly, the DC current amplification @100 mA collector current is specified to be at least (ONLY) 30. A bit of calculation revealed that the base resistor R13 on the arduino-r2 had too high resistance for the job.
- **Fix and Test 20190927:** Replaced arduino-r2/r13 with a 2k2 version. Measured current with the arduino/Q1 switched to PLL is now 65 mA (it varies somewhat between 60 and 65 mA).

This is much better and close to the current measured by simply short-circuiting pll/p5. (In other words, were doing quite OK with arduino/Q1 as a switch.)

- **Thermal Check of arduino/Q1 with R13=2k2 20190927:** I left the relay activated (i.e., PLL mde) for over two hours to see if the increased base and collector current do not cause thermal problems with pll/Q1. With the measured values, there should not be a problem as the CE voltage is (now) measured at 194 mV (MUCH better than the 1.4V with R13=10k!) and the current at 65 mA, leading to (CE) dissipation of approximately 13 mW, much lower than the MAX spec of 350 mW. (The specified MAX continuous collector current is 200 mA for the transistor.) After over two hours, pll/Q1 was only mildly warm; you could feel it was a bit warmer now (30 degrees or so?), but all looked OK.
- **Final Check:** Put everything back in place, and closed the lid. Performed several checks on PLL locking, and also did acoustic (:-)) checks on the relay while switching between PLL and MAN. All tests passed.
- **Conclusion:** Problem solved; requires lowering arduino/R13 to 2k2.

8.2 20190927: Output power varies 2 dB over the frequency range

- Noted this while working on the previous issue. Output varies roughly 2 dB while fine-tuning the OCXO. I also noted this with one of the other OCXOs in stock.
- **Done.**

8.3 20190927: The tuning range of the OCXO is ~7 Hz

- Through repeated measurements with the counters and the hp70000 system, we can safely conclude that the OCXO tuning range for the unit currently in V1 is 7 Hz.
- **Done.**

8.4 20190927: The OCXO's 10 MHz sweet spot is right in the middle of its tuning range

- **20190921 and 20190927:** The OCXO's 10 MHz mark is nicely centered within its range. This also follows from the rock-stable tuning voltage after at least two hours of "full lock": 2.231 V [ref voltage was measured to be 4.754 V]. (See below; the tuning voltage was measured on Test Point pa3gyf-gpsdocxo-pll-r2/PLLTUNE.)
- **Done.**

8.5 20190927: PLL Lock Time Measurement

- **20190927:** It takes roughly between 5 and 15 minutes to lock the OCXO with the GPS signal. The required time was measured in a setup with GPS fixed (for at least one hour), OCXO heated (for at least one hour). The OCXO was repeatedly taken out of lock by switching to manual tuning, and tuning to either on of the extreme ends of the tuning range,

or to its center. Typically, the LOCKED indication “went green” after 10 to 12 minutes, with outliers to 6 ½ minutes. Note: Lock Time refers to the time interval before the green LOCKED LED is starting to stabilize towards continuous green (i.e., no more or just occasional blinking with the red LED).

- **Conclusion:** PLL Lock Time (following V1 criteria) is between 5 and 15 minutes.

8.6 20190929: Lock Monitoring; Circuit and Performance

- While working on other issues on 20190921, I noticed the convergence of the tuning voltage of the OCXO with my hp3478a, a 5½-digit DMM (not calibrated and not fully tested yet). Basically, by measuring the voltage on Test Point pa3gyf-gpsdocxo-pll-r2/PLLTUNE. Interestingly, I could easily measure convergence of the tuning voltage PLLTUNE, but also discovered that upon entering LOCKED status, the PLLTUNE voltage was already close to its (assumed) limit, but certainly not yet there.
- I decided to take a closer look, and on 20190929, after at least one hour of warming up of all relevant equipment, I locked the gpsdocxo, and left it in lock for at least an hour, and then started to monitor PLLTUNE. On the hp3478a, the tuning voltage read 2.234V, decreasing to 2.232V two hours later. Given the fact that we assume (approximately) a 7 Hz tuning range over 5 V of OCXO tuning voltage, a more or less linear (enough) relation between tuning voltage and frequency, this amounts to at most 1 mV tuning voltage deviation over a 1 hour period.
- Under various reasonable assumptions, this amounts to 1 mV * 7 Hz / 5 V of frequency deviation of the OCXO over a one-hour period, viz., 0.0014 Hz/hour frequency deviation, equivalent to 0.14 Hz on 1 GHz and 1.4 Hz on 10 GHz, i.e., 0.14 ppb (0.14*10E-9) over a one-hour period. We *conservatively* estimate the long-term stability after at least 2 (several) hours in LOCKED state, in this particular case, at 0.2 pbb, or 0.0002 ppm over a one-hour interval. We may even want to guarantee 1 ppb / hour…
- Just a quick check: How realistic is this; are we within the proper order of magnitude? Well, 7 Hz on 10 MHz is roughly better than 1 ppm. If we stay within 1 mV on the 5 V tuning range, then that is a 5000 times increase of our accuracy, in other words, 0.2 ppb. Hence the calculation seems correct: 0.2 Hz on 1 GHz and 2 Hz on 10 GHz. **Wow!**
- We now turn our attention to the transient behaviour, in particular the relation between the LOCKED LED and the OCXO tuning voltage. From a more or less worst case phase difference at the PLL, we switched from MAN to PLL tuning, and obtained system-determined LOCKED status **16 minutes** later.
- **Intermediate Conclusion 2:** The worst-case system lock time (to the LOCKED status as shown by the LED), from a constant GPS FIX and HEATED state is estimated to be 20 minutes.
- The system entered LOCKED status with tuning voltage 2.276 V, quite before the (assumed) limit value of 2.232 V. After that, the LOCKED status remained active, apart from some occasional (2 or 3) cases of *not* being LOCKed. The tuning-voltage difference from the limit

value is less than 50 mV, the equivalence of (50 mV / 5V) * 7 Hz = 0.07 Hz (on 10 MHz), from the converged value. Since 0.1 Hz amounts to 1.0E-8 on 10 MHz, we carefully conclude that:

- **Intermediate Conclusion 2:** When entering the LOCKED state in the pa3gyf-gpsdocxo-v1 system, the output frequency is within 1.0E-8 (short-term, say 10s of seconds to minutes). This is 0.1 Hz on 10 MHz, 10 Hz on 1 GHz and 100 Hz on 10 GHz.

8.7 20190929: PCB pa3gyf-gpsdocxo-pll-r2 Test Points

- The PCB pa3gyf-gpsdocxo-pll-r2 has several undocumented (i.e., not in the circuit diagram) Test Points (list is not exhaustive):
  - **Near U5 [Phase Comparators]**
    - **PhI1:** First input of the phase comparator; U5p1: 10 kHz input from GPS.
    - **PhI2:** Second input of the phase comparator; U5p2; 10 kHz signal from OCXO.
    - **PhO:** Output of the phase comparator; U5/p8; buffered output from the phase comparator (U5C).
  - **Near K1 [Tune Select Relay] and P6 [PLL_MON_OUT Connector]**
    - **PLLMON:** PLL Tune Monitoring signal as it appears on P6; it is the output of U7. The signal is a low-passed version of the phase-comparator output with a different LPF than the PLL loop filter. This signal is (to be) fed into one of the Arduino DAC inputs through P6.
    - **PLLTUNE:** PLL Tune Signal as it appears on the middle switch contact of Tune Select Relay K1. This is (after some more filtering) the actual tune signal sent to the OCXO.

9 License

10 Resources

https://github.com/jandejongh/pa3gyf-gpsdocxo