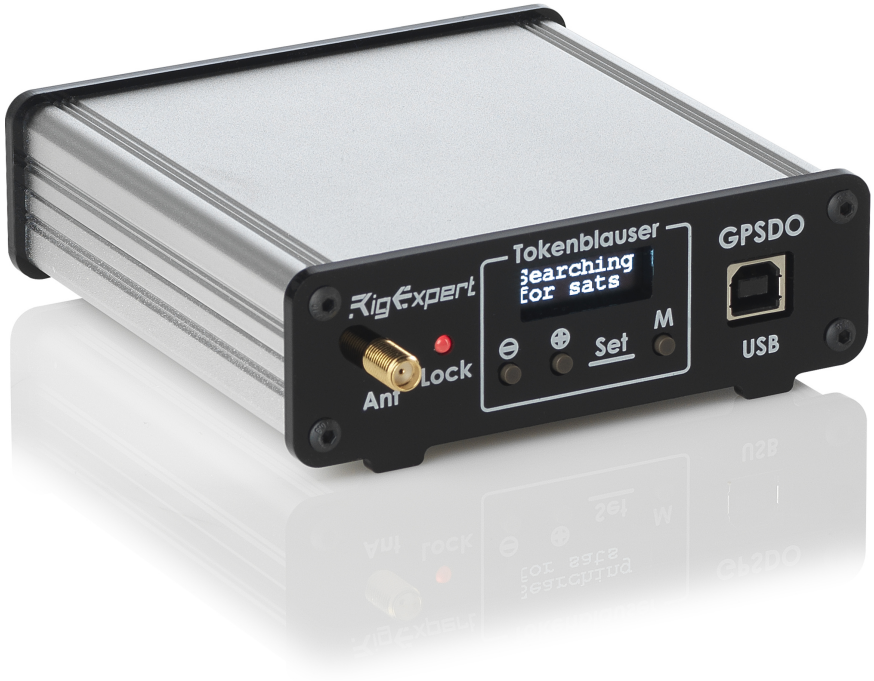
# Tokenblauser

Low-noise GPSDO,
quad clock source –
0.16 to 200 MHz

RigExpert®



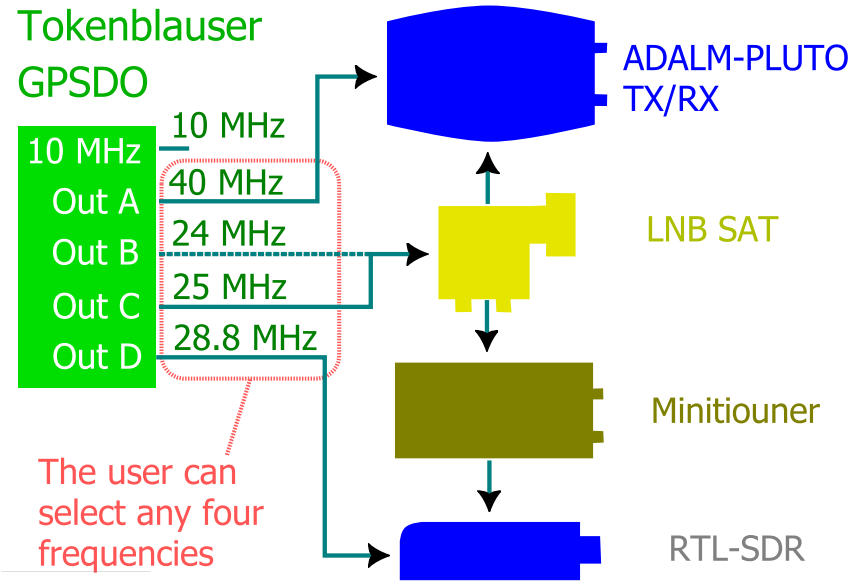

User's manual

# Table of contents

# Tokenblauser – a versatile GPSDO platform

The Tokenblauser is a laboratory-grade GPS disciplined clock source with very low noise output, suitable for QO-100 and SHF applications, including operating FT8 and other narrow band digital modes.

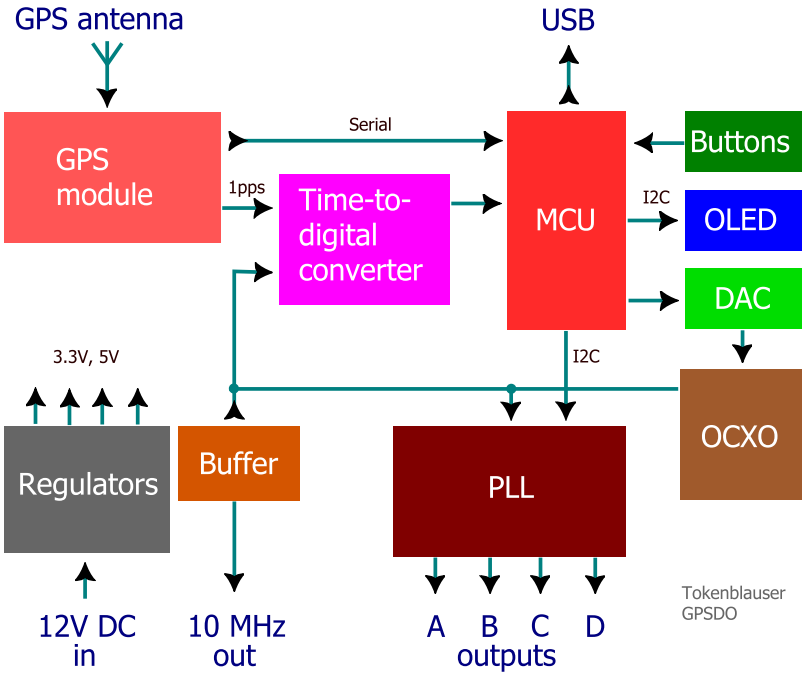
Unlike many other GPSDOs with fixed 10 MHz output, it additionally generates up to four arbitrary frequencies simultaneously. This functionality is provided "out of the box" and does now require advanced skills from the user. Just apply power to the device and connect a GPS antenna, then enter or change output frequencies.




*Example of use for operating the QO-100 satellite*

However, we designed the Tokenblauser to be also a versatile platform for experimenters. The device is Arduino-compatible, and the firmware is open-source. With no or minimal changes in the code or in the hardware, an advanced user may:

- Connect the GPSDO to a computer for debug purposes or data processing (such as plotting ADEV/MDEV graphs);
- Modify the firmware to experiment with different PLL and FLL algorithms;
- Use own GPS modules and GPS antennas;
- Use different types of OCXOs and Rubidium oscillators.



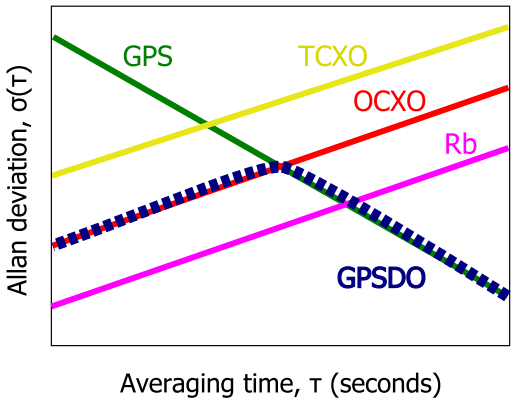

*Structure diagram of the Tokenblauser GPSDO*

The Tokenblauser is based on ideas of Brooks Shera, Lars Walenius and many other enthusiasts.

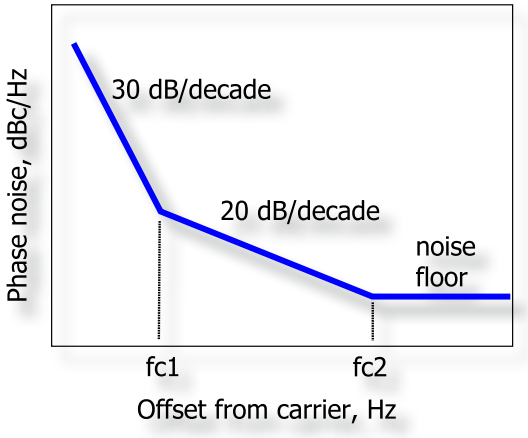
# High stability and low noise

Just look at a simplified **Allan deviation chart**, which compares the stability of different types of generators over time. The GPS receiver itself has a bad short-term stability, where the OCXO is good. In opposite, during long periods, the GPS is a winner.

By locking the frequency of a local oscillator to the reference GPS receiver, **GPSDOs are capable to have good stability in both short-term and long-term areas.** The position of the intersection point depends on the type and the quality of oscillator used; for OCXOs, it is usually 100 to 1000 seconds. Thus, OCXO-based GPSDOs are able to provide good frequency stability for laboratory and hobby use.




The next chart is **one of standard models of an oscillator phase noise.** Obviously, for a better GPSDO, the corner frequencies fc1 and fc2 need to be located as close to the carrier frequency as possible.



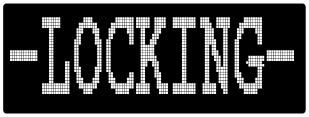

If the GPSDO output is multiplied by the PLL-based oscillator, we usually think of 20 dB/decade as a noise increment. However, **this is not an axiom**: much more noise is added for frequencies to the left of **fc1**. By using a professional-grade OCXO, as well as carefully designing the RF circuits, very good noise parameters of the GPSDO can be achieved.

# The first use

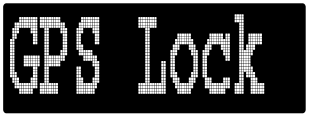
Connect the GPS antenna and locate it in such a way, that at least ½ of the sky is visible. It is always better to locate the antenna outdoors, whenever possible.

Connect it to **9-15V** power supply. Once the power is turned on, the red LED on the front panel starts flashing: the GPS module is now searching for the satellites to synchronize time.


Once the satellites are discovered, the GPSDO starts the locking process. The green LED is now flashing. Be patient, this may take several minutes.

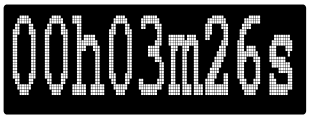



Once the output frequency is stable for the last 100 seconds, the GPSDO indicates a locked status. The green LED on the front panel now stops flashing.




In addition to the locked status, the OLED shows the time since the last lock event, and the average signal level of the received GNSS satellites. Adjust the position of your antenna for the highest signal level:

**< 20 dBHz = poor, 20..25 dBHz = good, > 25 dBHz = excellent.**





**Buttons**

Press ⊖ or ⊕ button shortly to browse through the menu. Hold these buttons for approximately two seconds to change current profile number, set output frequencies or to enter **Preferences**. Press the **M** (Menu) button to exit back to the **Status** screen.

# Menu description

### Setting output frequencies

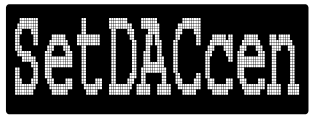

To change profile number, press ⊖ or ⊕ button to select the Profile menu item, then hold one of these buttons for approximately two seconds to change the profile number.


For each of four profiles, you may change frequencies for each of A...D outputs. With ⊖ or ⊕ button, select the desired output, and then hold one these buttons to edit its frequency.


An editable digit starts flashing. Short press ⊖ or ⊕ to select the digit to edit, or hold one of these buttons to increment or decrement the selected digit. Please make sure that the desired frequency is inside the specifications of this GPSDO.

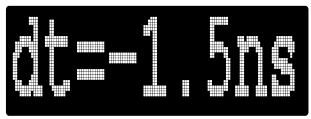
Press **Menu** to exit the editor mode.

### Preferences




Select **Prefs** from the top menu, then hold ⊖ or ⊕ to enter the **Preferences** menu. Experienced users may tweak several parameters of the GPSDO here.

### Checking the OCXO center frequency



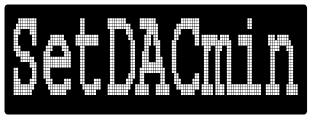

First, use the **SetDACcen** menu to set the DAC to its center value (32768).

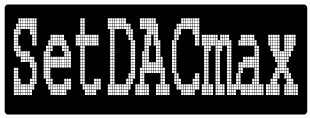
Notice the value of the dt while in **Hold** mode. If the value if out of the ±100 ns range, adjust the potentiometer (if installed) on the printed circuit board to set is as close to zero as possible.

**OCXO tuning range settings**

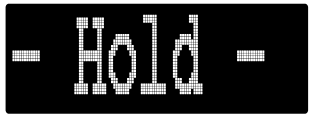
To set the maximum tuning range of the OCXO, select **Range** and hold ⊖ or ⊕ to change. The tuning range is set in ppt (parts per trillion). It is necessary to set the tuning range properly for the algorithms to work correctly.


The OCXO tuning range is defined as a relative frequency change of the OCXO when the DAC changes its value to from its minimum to its maximum. Use SetDACmin and SetDACmax menus to control the DAC.

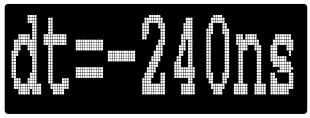



Hold ⊖ or ⊕ to set the 16-bit DAC in its minimum (0) and then maximum (65535) value. After applying this setting, the GPSDO will enter the **Hold** mode, returning to the **Status** screen.




Notice the value of the **dt** while in **Hold** mode.


The value shows how much the output the 1PPS output of the GPS is ahead or behind of the pulses of the internal OCXO. Positive values mean that the frequency of the OCXO is lower than 10 MHz, and negative values mean higher OCXO frequencies.

To find out the exact tuning range of the OCXO installed in your GPSDO, follow this algorithm:

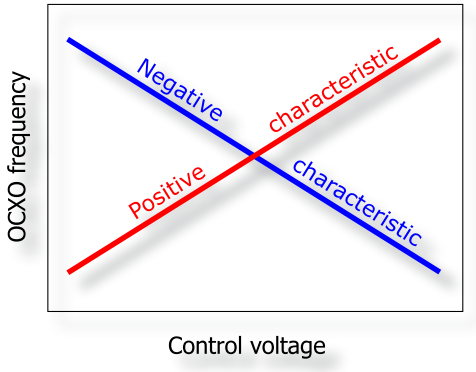
1) Set the DAC to its minimum value by using the **SetDACmin** menu. Write down the **dt** value, such as 241ns.

2) By using the **SetDACmax** menu, choose maximum value. Again, write down the **dt** value. Example: -240 ns.

With the above example, the tuning range of the OCXO (limited by resistor network inside the GPSDO) is

*241 + 240 = 481 ppb (parts per billion) -or- 481000 ppt (parts per trillion)*

Enter the value 481000 ppt in the **Range** menu, for the PLL algorithm to work properly. For OCXOs with negative control voltage characteristic, the **Range** value will be negative.

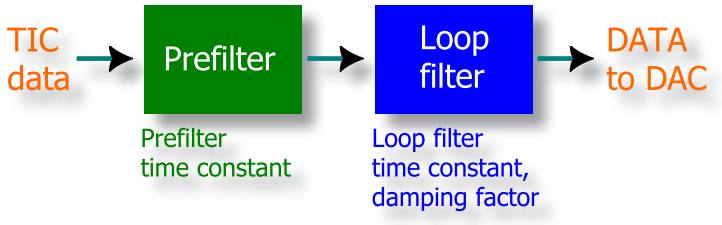



*Positive and negative control voltage characteristics*

**PLL settings**

By using the **Prefilter** time constant, the **Loop filter** time constant and the **Damping factor** menus, parameters of the phase locked loop may be tweaked, as illustrated by the picture below.

Example loop filter settings: 100 sec – faster locking, less stable output ($10^{-9}$). 500 sec – slower locking, more stable output ($10^{-10}$).




*Prefilter and PI-loop filter*

**Hold and Run modes**

By activating the **Hold** menu, the phase locked loop algorithm is deactivated and the DAC holds voltage to control the OCXO.

The **Run** menu gets the phase locked loop back to normal operation.

**GPS settings**

**The Init GPS** menu is responsible for re-initializing the GPS module every time the GPSDO is turned on (default: **YES**). For various experiments, you may choose **NO**.

The **Communication mode** defines the behaviour of the virtual COM port. The **Command line** is a normal operation mode of the GPSDO. The **GPS mode** creates a transparent connection between the GPS module and the virtual COM port. Use this mode to be able to control your GPS module directly by specialized software, such as u-center or Lady Heather.

**Other settings**

**Display** – select either **Bright** (default) or **Dark** mode.

**Run boot loader** – enter the boot loader mode for firmware update (see below).

**Factory reset** – revert to default settings.

**SI5338 menu** – display the device grade and maximum output frequency.

FW ver. – show current firmware version.

# Serial port interface

The virtual serial port interface provides various functions, such as control or monitoring of the GPSDO. Open a virtual serial port (for example, **COM22**) with terminal software, such as **Putty** or **Arduino IDE**.

Type "**?**" or "**help**" to get a list of available commands, as well as current settings of the device. The serial interface is self-documenting. Most commands are available through the front-panel menu of the GPSDO.

```
hold - stop PLL
run - run PLL
dac<N> - set DAC output to value N=0...65535 in hold mode
savedac - save current DAC value
readdac - read saved DAC
trans - enter GPS transparent mode, +++ to exit
pre<N> - set prefilter time constant to N seconds, or 0 to
   disable prefilter
loop<N> - set loop time constant to N seconds
damping<N> - set loop damping to N (0.5 to 10)
range<N> - set OCXO tuning range (in ppt) to N
saw<N> - set sawtooth correction to 1 (on) or 0 (off)
initgps<N> - init GPS at startup: 1 (on) or 0 (off), initgps to
   re-init
offset<N> - set frequency offset (in ppt)
ss - take a screenshot, ss<char> to use <char> instead of bricks
verb<N> - set verbose level to 0 (min) or 1 (max)
legend or <CR> - print legend
restart - restart the device
boot - jump to bootloader
```

When the 1PPS signal is available from the GPS module, a row of numbers is output every second:

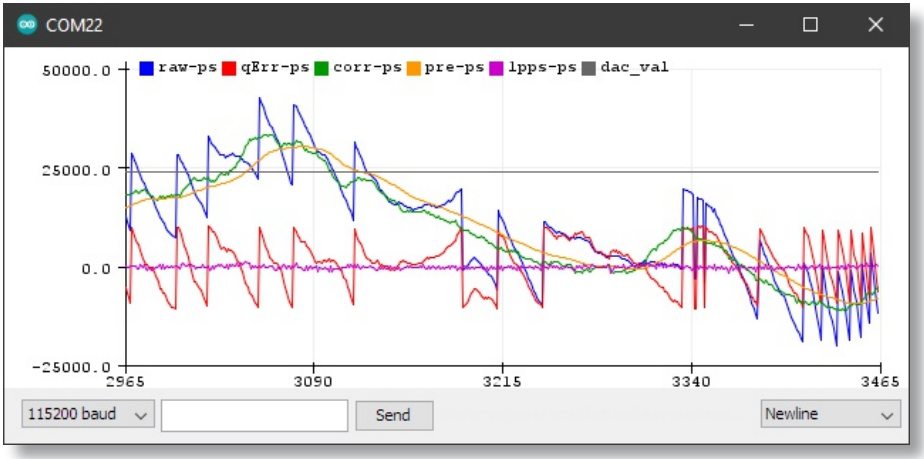
```
raw-ps      qErr-ps corr-ps pre-ps  1pps-ps dac_val
6142        5995    147     2279    347     23915
4723        3698    1025    2237    878     23915
2951        1392    1559    2215    534     23915
```

**raw-ps** – raw output of the time interval counter (TIC); **qErr-ps** – quantization error value read from the GPS module; **corr-ps** – corrected TIC value; **pre-ps** – prefilter output; **1pps-ps** – phase difference between 1PPS pulses; **dac_val** – current DAC value.

### Arduino serial plotter

By using the **Serial plotter** function of the **Arduino IDE**, you may visualize the behavior of the phase locked loop.

Open Tools – Serial plotter (or press Ctrl-Shift-L), then type **legend** in the input window and click **Send** (or just click **Send** without typing anything).

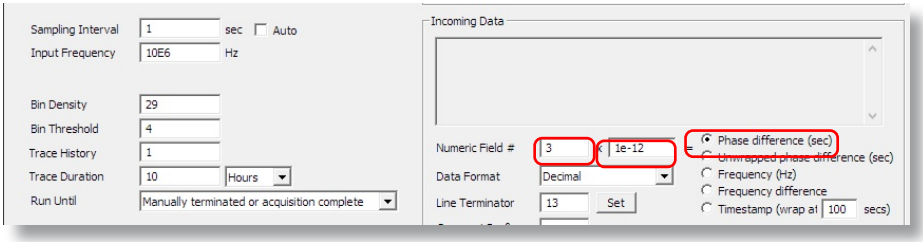



To be able to tweak PLL parameters on the fly, first enter the **verb0** command.

### TimeLab interfacing

With the famous **Timelab** software from Miles Design LLC., plotting Allan deviation of the internal TIC is made easy. The real-time mode is activated by clicking the Acquire – Acquire from counter in Talk-Only mode.

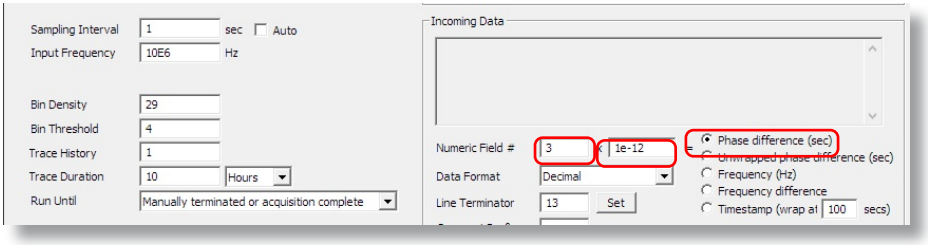
**To plot the phase data**, select Phase difference, then enter **Numeric Field #3** (corrected TIC value) and set the phase multiplier to **1e-12**.



### TimeLab interfacing

With the famous **Timelab** software from Miles Design LLC., plotting Allan deviation of the internal TIC is made easy. The real-time mode is activated by clicking the Acquire – Acquire from counter in Talk-Only mode.

**To plot the phase data**, select Phase difference, then enter **Numeric Field #3** (corrected TIC value) and set the phase multiplier to **1e-12**.



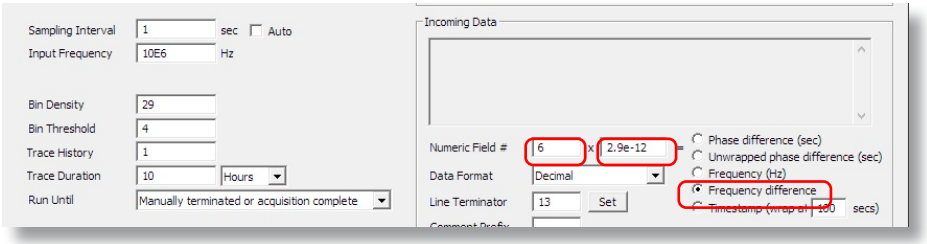

Click Start Measurement.


*A sample of Allan deviation plot for the phase data*

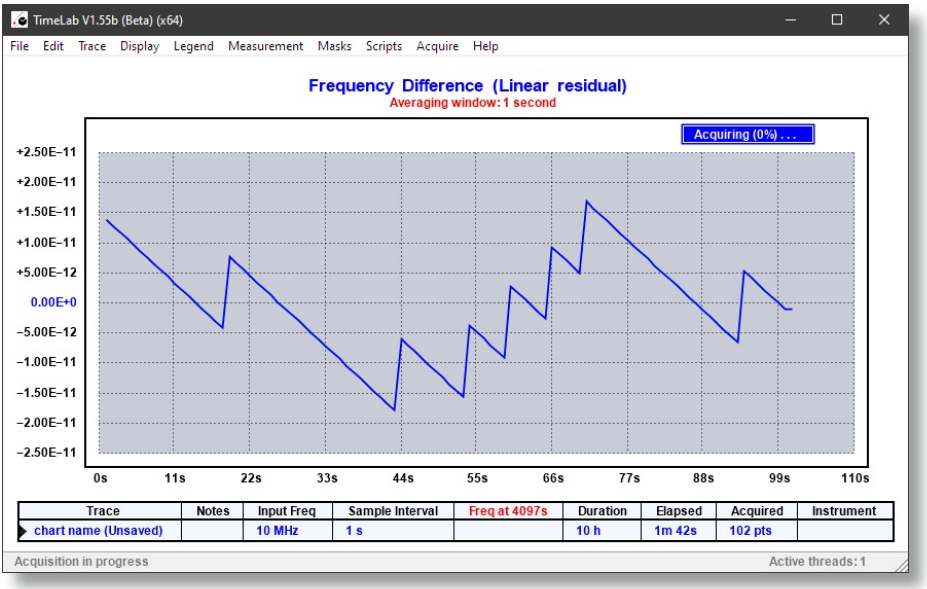
**To plot the frequency data**, select Frequency difference, then enter **Numeric Field # 6** (DAC value) and set the frequency multiplier. The multiplier as calculated as:

*(OCXO tuning range in ppt)/65536*10-12*

For instance, for the tuning range of 847000 ppt, enter **12.9e-12**.



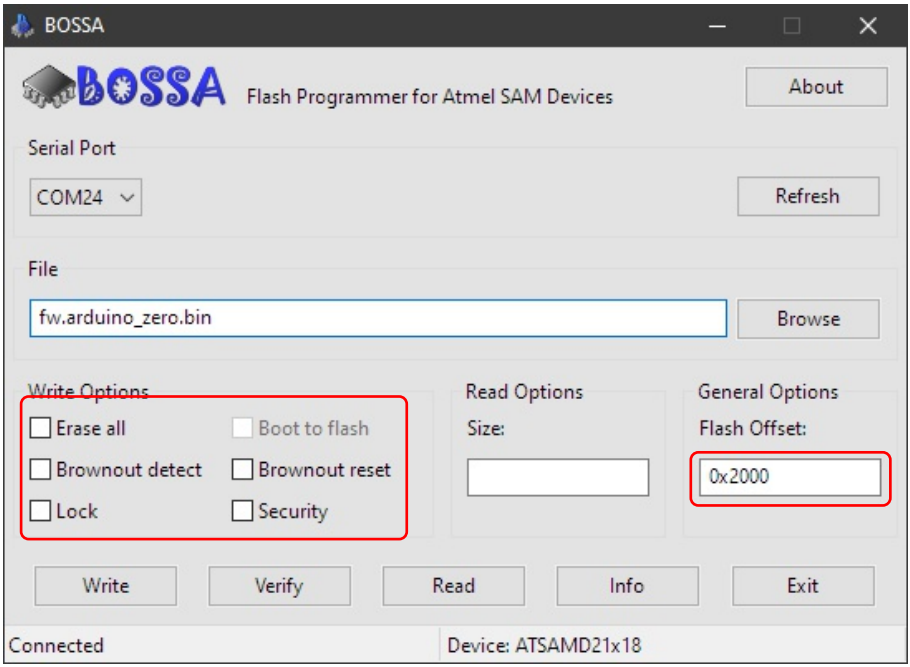

Click Start Measurement.




*A sample plot of the Frequency Difference*

### Updating the firmware

To update the firmware by using binary file, install the BOSSA programmer from **https://github.com/shumatech/BOSSA/releases** . Run the program.




Important: enter **0x2000** in the Flash Offset field. **Please do not touch any check boxes; else, your GPSDO may be bricked!**

Browse for the new firmware file.

Put the GPSDO into the boot loader mode: either use the **Run boot loader** menu from **Preferences**, send the **boot** command via the virtual serial port, or double-press the RESET button (if installed) at the PCB.

In the BOSSA dialog, click Refresh and choose an available Serial port number.

Click Write to start programming. Reset or re-plug the GPSDO after finishing the programming process.

Binary firmware files: **https://github.com/rigexpert/Tokenblauser/releases/**

**Compiling the firmware from source code**

To be able to modify and re-compile the firmware, make sure to:

1) Install the latest version of the Arduino environment from **www.arduino.cc** .

2) Open the Boards Manager from Tools – Board – Boards Manager menu. Install the Arduino SAMD Boards (32-bits ARM Cortex-M0+).

3) Open Library Manager from Tools – Manage Libraries menu, and then install the following libraries:

**SAMD_TimerInterrupt**

**FlashStorage_SAMD**

**Adafruit_SSD1306**

**Adafruit_GFX_Library**

**TDC7200 library: https://github.com/Yveaux/TDC7200**

4) Select **Arduino Zero (Native USB Port)** from Tools – Board - Arduino SAMD Boards menu.

5) Select the corresponding COM port number.

Download the source code from:

**https://github.com/rigexpert/Tokenblauser/**

**SI5338 configuration presets**

For making initial C code header configuration files for the SI5338 frequency synthersizer chip, use **Skyworks ClockBuilder Pro** software, see **https://www.skyworksinc.com/en/application-pages/clockbuilder-pro-software** .

Export the register file to Si5338-RevB-Registers-**X**.h , where **X** = 1..4 (profile number).

# Annex 1
# Specifications

**Power supply:** 9-15V, 1A max.

**USB interface:** USB 2.0, type B connector (dvice is not powered from USB); Virtual serial port compatible with Windows, Mac OS and Linux.

**GPS input:** 3V active antenna with SMA connector; hardware modifiable for 5V antennas; reception of GPS and Galileo constellations.

**Frequency stability:** $10^{-9}$...$10^{-10}$, depending on GPSDO settings, antenna location and satellite signal levels.

**Reference output:** One fixed 10 MHz output (BNC); 3.3V CMOS levels, hardware modifiable for sinewave output or 5V CMOS levels.

**Arbitrary frequency clocks:** Four PLL-synthesized outputs - A,B,C,D (BNC); independent frequency entry, 0.16 to 200 MHz in 1 Hz steps (very fine tuning is available with 1 ppt steps); 3.3V CMOS levels; hardware/software modifiable for LVPECL/LVDS/HCSL/CMOS/SSTL/HSTL levels and to 350/710 MHz (depending on the PLL chip grade).

**User interface:** 0.91″ monochrome OLED; three front panel buttons; four frequency profiles for fast switching of output frequencies.

**Software compatibility:** Arduino SAMD21 compatible; serial plotter supported in Arduino environment; debug output which can be analyzer by TimeLab and other software; transparent mode for direct access to the GPS module.

**Dimensions:** Enclosure: 105x105x35 mm; PCB: 100x100x25 mm.

**Experimenter's corner:** Tweak parameters of the GPSDO from the front panel; modify the firmware and upload to the device by using Arduino environment; use GPSDOs PCB as a module for your own device; replace built-in GPS module with external one; replace built-in OCXO with own OCXOs (rectangular or sinewave output) or Rubidium oscillator.

**Recycled/refurbished materials:** OCXO.

Specifications are subject to change without notice.

# Annex 2
# Glossary

**1PPS** – One Pulse Per Second

**ADEV** – Allan Deviation

**Arduino** – popular hardware and software platform, including development environment

**DAC** – Digital-to-Analog Converter (for a 16-bit DAC, the min value is 0 and the max value is 65535)

**FW** – Firmware

**GPS** – Global Positioning System (other navigation systems, such as Galileo and BeiDou, are also often called GPS)

**GPS module** – hardware module containing specialized processor

**GPSDO** – GPS Discipline Oscillator

**HOLD mode** – mode of ther GPSDO, when the PLL is not allowed to control the DAC

**Lars GPSDO** – original Arduino-based GPSDO design by Lars Walenius

**MDEV** – Modified Allan Deviation

**OCXO** – Oven Controlled Xtal (Crystal) Oscillator (usually, voltage controlled)

**OCXO Tuning Range** – typically, +-1...3 ppm in reaction to a full swing of control voltage

**OCXO Tuning Range Limiting** – a technique to limit the tuning range by resistor network

**PLL** – Phase Locked Loop

**PPB** – Part Per Billion (1e-9, example: 0.01 Hz at 10 MHz)

**PPM** – Part Per Million (1e-6, example: 10 Hz at 10 MHz)

**PPT** – Part Per Trillion (1e-12, example: 0.00001 Hz at 10 MHz)

**qErr** – quantization error of 1PPS; the data to correct this error, sent from a GPS module via UBX

**RUN mode** – the PLL is allowed to control the DAC

**Sawtooth correction** – software algorithm to combine TIC result with data read from a GPS module

**TIC** – Time Interval Counter

**TimeLab** – software for precision time and frequency measurement by Miles Design LLC.

**UBX** – proprietary data exchange protocol of u-blox

http://www.rigexpert.com



Designed in Ukraine

Doc.date: 10-Jan-2023