NanoVNA User Guide

By: **Satoh, Hiroh** (cho45)

Updated Oct 2, 2019
Translated from Japanese to English by Google Translate
Reformatted for printing by L. Rothman

All Table of Contents links resolve to translated webpages, not this document.

1. Introduction
    - i. What is NanoVNA
    - ii. What you need to work
    - iii. NanoVNA basics
    - iv. NanoVNA oscillation frequency
2. To do first
3. Input method
4. How to read the screen
    - i. Main screen
        - a. 1. START 2. STOP frequency
        - b. 3. Marker
        - c. 4. Calibration status
        - d. 5. Reference position
        - e. 6. Marker status
        - f. 7. Trace status
        - g. 8. Battery status
    - ii. Main screen 2
        - a. 9. CENTER frequency 10. Span
        - b. Menu screen
        - c. 11. Menu
    - iii. Keypad screen
        - a. 12. Numeric keys
        - b. 13. Back key
        - c. 14. Unit key
        - d. 15. Input field
        - e. 16. Keyboard key
5. Start measurement
    - i. Basic measurement sequence
6. Calibration method
7. Function
    - i. Trace display
        - a. Trace format
        - b. Trace channel
    - ii. marker
    - iii. Time domain operation
        - a. Time domain bandpass

        b. Time domain lowpass impulse
        c. Time domain lowpass step
            a. Step response example
        d. Time domain window
        e. Setting the wavelength factor in the time domain
        f. Set frequency from marker
    iv. Setting the measurement range
        a. Setting the start frequency and stop frequency
        b. Setting the center frequency and span
        c. Zero span
        d. Temporarily stop measurement
    v. Recall calibration and settings
    vi. Device settings
        a. Touch panel calibration and testing
        b. Saving device settings
        c. Display version
        d. Firmware update
8. How to update the firmware
    i. How to obtain the firmware
        a. ttrftech version firmware
        b. hugen79 version firmware
        c. Build yourself
    ii. How to write firmware
        a. Writing with dfu-util (Ubuntu)
        b. Writing with dfu-util (macOS)
        c. Writing with dfu-util (Windows 10)
    iii. How to write firmware (Windows GUI)
        a. Convert file format with DFU File Manager.
        b. Write firmware with DfuSe Demo
9. Firmware development guide
    i. Build with Docker
    ii. On-chip debugging with Visual Studio Code
        a. tasks.json
        b. launch.json
        c. Start debugging
10. Example of use
    i. Adjusting the bandpass filter
    ii. Antenna adjustment
        a. Trace settings
    iii. Check the cable
    iv. Common mode filter measurement

1. Introduction

This document is an unofficial user guide for NanoVNA. The URL is https://cho45.github.io/NanoVNA-manual/.

It is managed in the github repository .

Please send a Pull-request if there is a correction, such as when there is a conflict with the latest firmware.

It is also available in PDF format on the GitHub Releases page.

- https://github.com/cho45/NanoVNA-manual/releases

What is NanoVNA

There are several types of NanoVNA hardware, and this document covers the following hardware:

- ttrftech version (original) ttrftech / NanoVNA
- hugen79 versionhugen79 / NanoVNA-H

These hardware components are almost the same on the circuit, and common firmware can be used.

What you need to work

The following are required at a minimum.

- NanoVNA body
- SMA LOAD 50Ω
- SMA SHORT
- SMA OPEN
- SMA Female to Female Through Connector
- SMA Male to Male cable x 2

NanoVNA basics

VNA (Vector Network Analyzer) measures the frequency characteristics of reflected power and passing power of a high frequency network (RF Network).

NanoVNA measures the following elements:

- Input voltage I / Q signal
- Reflected voltage I / Q signal
- Pass voltage I / Q signal

From here we calculate:

- Reflection coefficients S11
- Transmission coefficient S21

Some of the following items that can be calculated from these can be displayed:

- Reflection loss
- Passing loss
- Complex impedance
    - resistance
    - reactance
- SWR

Newer firmware versions may add additional measurement functions.

NanoVNA oscillation frequency

NanoVNA measures the reflection coefficient and transmission coefficient for 101 points in the frequency band to be measured.

The local frequency of NanoVNA is 50kHz to 300MHz. For higher frequencies, use harmonic mode. The fundamental wave is not attenuated even in harmonic mode. The usage modes for each frequency are as follows.

- Up to 300 MHz: fundamental wave
- 300MHz to 900MHz: 3rd harmonic
- 900MHz to 1500MHz: 5th harmonic

Note that there is always a fundamental wave input, especially when checking the amplifier gain.

In either case, the input is converted to an intermediate frequency of 5kHz. The signal is converted from analog to digital at 48kHz sampling. Digital data is signal processed by the MCU.

2. To do first

Before you can use it, you must first calibrate it. First, calibrate as follows.

- Make sure START is 50kHz
- Make sure STOP is 900MHz
- Calibrate according to the calibration method

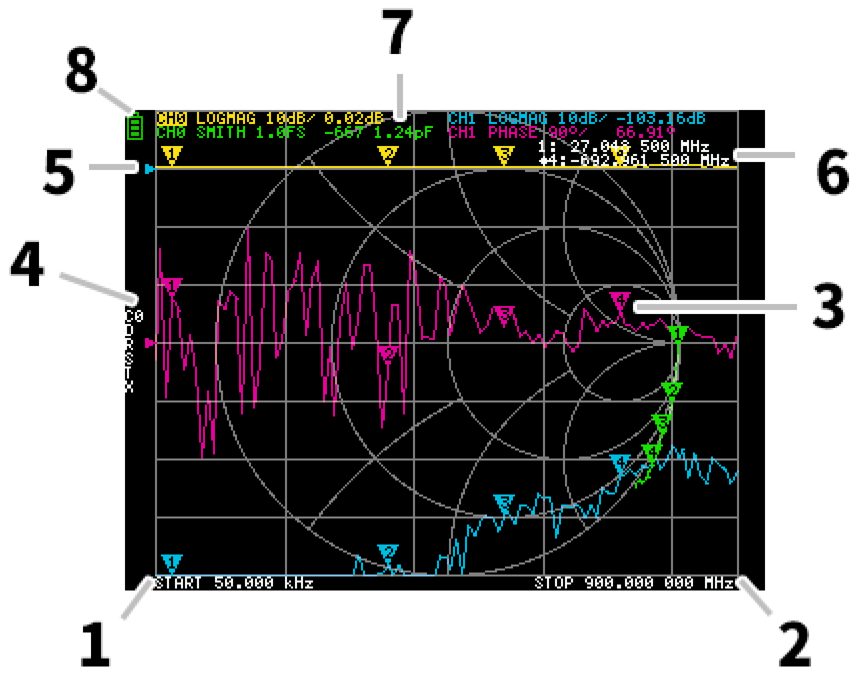
3. Input method

NanoVNA has the following inputs.

- Touch panel long tap
- Lever switch
  - L / L long press
  - R / R long press
  - Push / Push long
- Power slide switch

4. How to read the screen

Main screen



1. START frequency

2. STOP frequency

Each frequency when start / stop is specified is displayed.

3. Marker

The marker position for each trace is displayed. The selected marker can be moved in the following ways.

- Drag a marker on the touch panel
- Press and hold LR on the lever switch

4. Calibration status

Displays the data number of the calibration being read and the error correction applied.

- **`C0 C1 C2 C3 C4`** : Each indicates that the corresponding calibration data is loaded.
- **`c0 c1 c2 c3 c4`** : Each indicates that the corresponding number of calibration data is loaded, but the frequency range has been changed after loading, indicating that the error correction is using complement.
- **`D`** : Indicates that directivity error correction is applied
- **`R`** : reflection tracking Indicates that error correction is applied
- **`S`** : source match Indicates that error correction is applied
- **`T`** : transmission tracking Indicates that error correction is applied
- **`X`** : indicates that isolation (crosstalk) error correction is applied

5. Reference position

Indicates the reference position of the corresponding trace. You can change the position with:
`DISPLAY` → **`SCALE`** → **`REFERENCE POSITION`** .

6. Marker status

The active marker that is selected and one marker that was previously active are displayed.

7. Trace status

The status of each trace format and the value corresponding to the active marker are displayed.

For example, if the display is **`CH0 LOGMAG 10dB/ 0.02dB`** , read as follows.

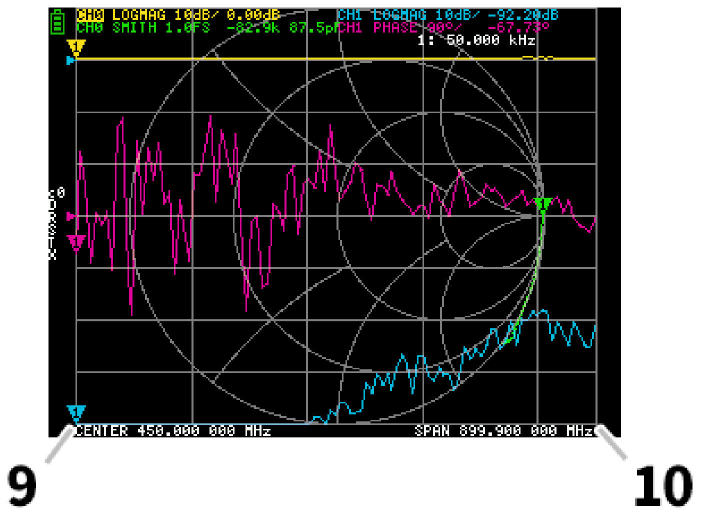
- Channel CH0 (reflection)
- Format **`LOGMAG`**
- Scale is 10dB
- Current value is 0.02dB

For active traces, the channel display is reversed.

8. Battery status

If a battery is installed and a **1N4148** surface mount diode is mounted on the PCB at location **D2**, an icon is displayed according to the battery voltage. If the diode is missing, the icon will show an empty battery.

Main screen (continued)

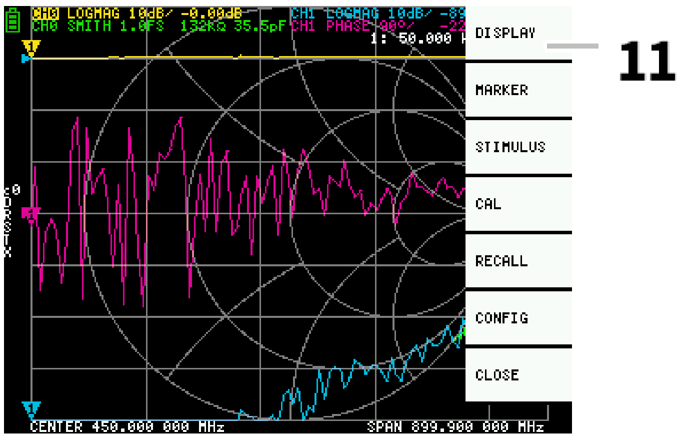


9. CENTER frequency

10. Span

Each frequency when the center frequency and span are specified is displayed.

Menu screen

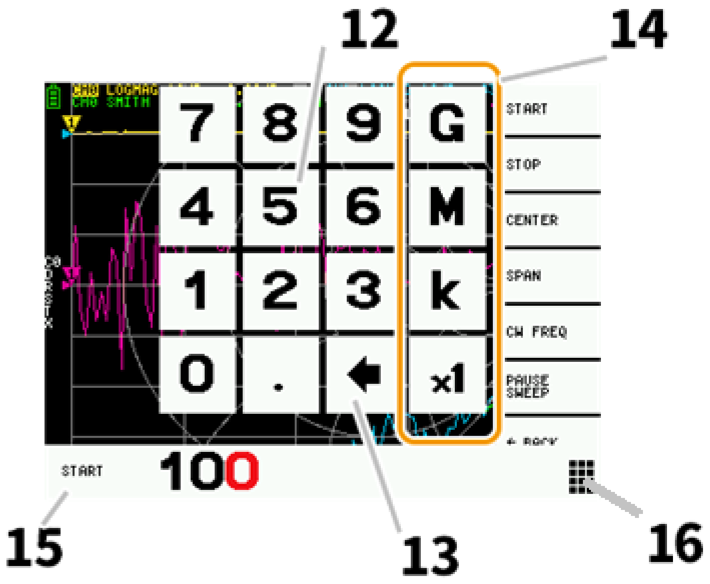


11. Menu List

The menu can be displayed by the following operations.

- When a location other than a marker on the touch panel is tapped
- Push the lever switch

Keypad screen



12. Numeric keys

Tap a number to enter one character.

13. Back key

Delete one character. If no character is entered, the entry is canceled and the previous state is restored.

14. Unit key

Multiplies the current input by the appropriate unit and terminates input immediately. In case of × 1, the entered value is set as it is.

15. Input field

The name of the item to be entered and the entered number are displayed.

16. Keypad icon

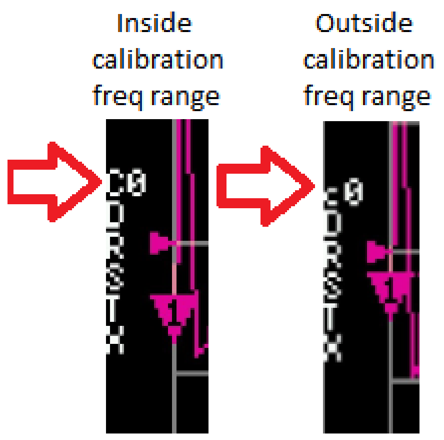
The large numeric entry keypad will appear on-screen any time the small keypad icon is pressed

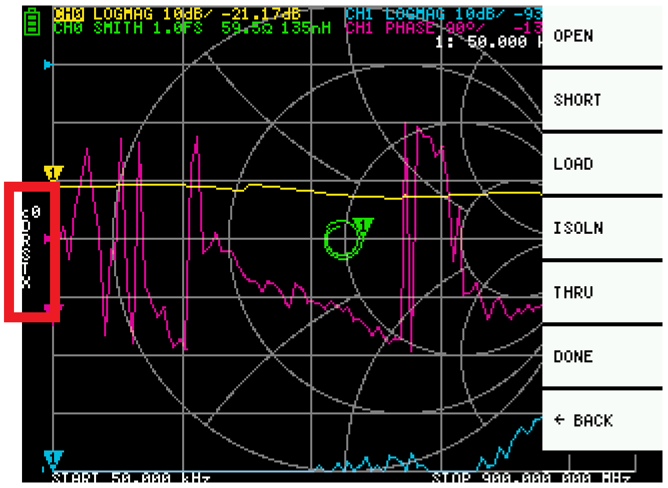
5. Start measurement

Basic measurement sequence

1. Set the frequency range to be measured
2. Perform calibration
3. Connect the Device Under Test (DUT) and measure

6. Calibration method





Calibration should basically be performed whenever the frequency range to be measured is changed. If the error has been corrected correctly, the calibration status display on the screen will be **`Cn DRSTX`** Where: **`n`** is the calibration dataset number being loaded.

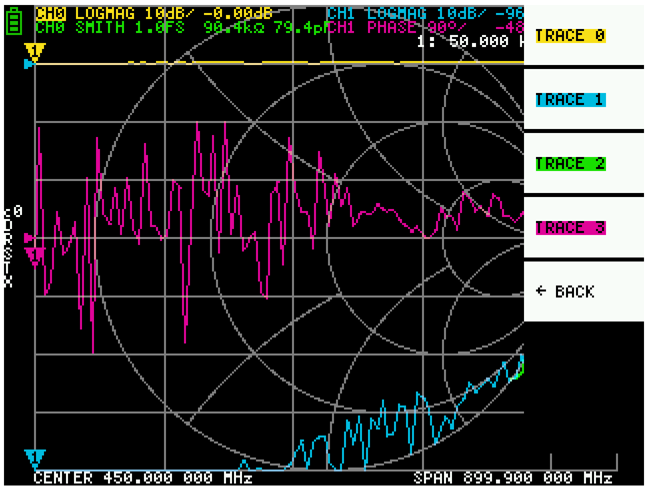
However, NanoVNA can complement the existing calibration information and display corrections to some extent. This will happen if the frequency range is changed after loading the calibration data. At this time, the calibration status display on the screen is **`cn DRSTX`** Where: **`n`** is the calibration dataset number being loaded. See image above.

1. Reset current calibration state. Select **CAL MENU →RESET** and then **→CALIBRATE**
2. Connect OPEN standard to CH0 port and execute **→OPEN** .
3. Connect SHORT standard to CH0 port and execute **→SHORT** .
4. Connect the LOAD standard to the CH0 port and execute **→LOAD** .
5. Connect the LOAD standard to CH0 and CH1 ports and execute **→ISOLN** .
   If there is only one load, the CH0 port can be left unconnected.
6. Connect a cable between the CH0 and CH1 ports, and execute **→THRU** .
7. Finish calibration and calculate error correction information **→DONE**
8. Specify the dataset number (0 to 4) and save. **`→SAVE 0 (0 is the power-on default)`**

* Each calibration step should be completed after the display is sufficiently stable.

7. Function

Trace display



Up to four traces can be displayed, one of which is the active trace.

Traces can display only what is needed. To switch the display, select `DISPLAY →TRACE →TRACE n` . The following methods can be used to switch the active trace.

- Tap the trace marker you want to activate
- Select `DISPLAY →TRACE →TRACE n` to display. (If already displayed, it will be temporarily hidden)

Trace format

Each trace can have its own format. To change the format of the active trace, select the format you want to change to `DISPLAY →FORMAT` .

The display of each format is as follows.

`LOGMAG` :	Logarithm of absolute value of measured value
`PHASE` :	Phase in the range of -180 ° to + 180 °
`DELAY` :	Delay
`SMITH` :	Smith Chart
`SWR` :	Standing Wave Ratio
`POLAR` :	Polar coordinate format
`LINEAR` :	Absolute value of the measured value
`REAL` :	Real number of measured value
`IMAG` :	Imaginary number of measured value
`RESISTANCE` :	Resistance component of the measured impedance
`REACTANCE` :	The reactance component of the measured impedance

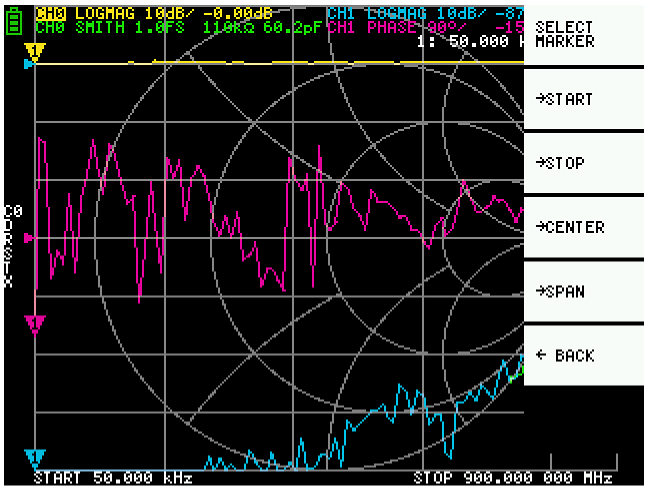
Trace channel

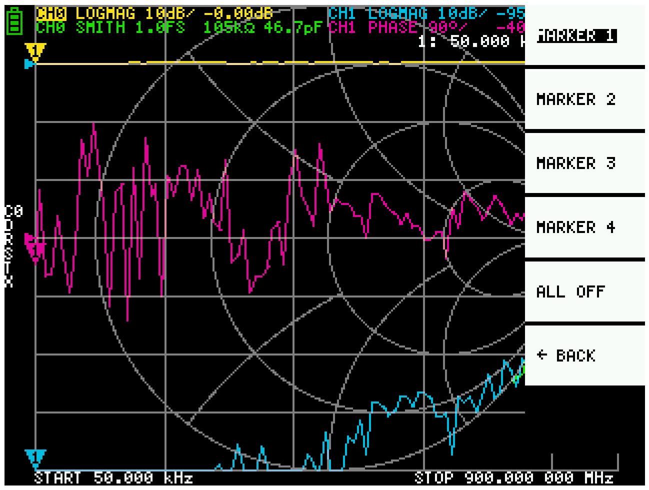
NanoVNA has two ports, **`CH0`** and **`CH1`** . The following S parameters can be measured at each port.

- `CH0` S11 (reflection loss)
- `CH1` S21 (insertion loss)

To change the trace channel, select
`DISPLAY →CHANNEL →CH0 REFLECT` or **`DISPLAY →CHANNEL →CH1 THROUGH`** .

Markers





Up to 4 markers can be displayed.
Markers are displayed from **`MARKER →SELECT MARKER →MARKER n`** .
When you display a marker, the active marker is set to the displayed marker '**n**'.

Time domain operation

NanoVNA can simulate time domain measurements by signal processing frequency domain data.

Select **`DISPLAY →TRANSOFRM →TRANSFORM ON`** to convert measurement data to the time domain. **`TRANSFORM ON`** is enabled, the measurement data is immediately converted to the time domain and displayed.
The relationship between the time domain and the frequency domain is as follows.

- Increasing the maximum frequency increases the time resolution
- The shorter the measurement frequency interval (ie, the lower the maximum frequency), the longer the maximum time length

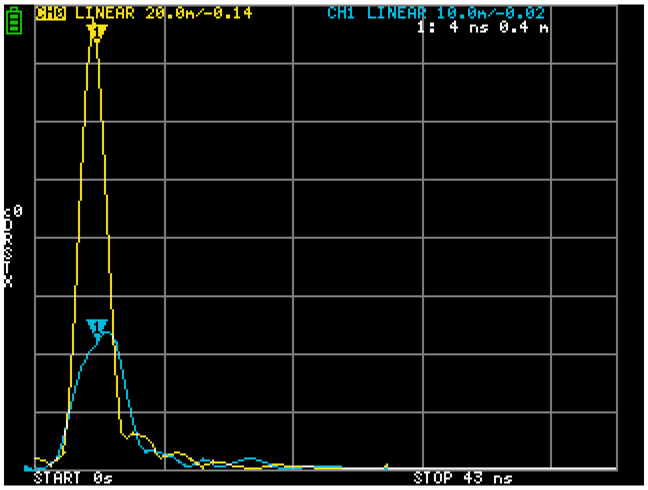
For this reason, the maximum time length and time resolution are in a trade-off relationship. In other words, the *time length is the distance*.

- If you want to increase the maximum measurement distance, you need to lower the maximum frequency.
- If you want to specify the distance accurately, you need to increase the maximum frequency.

Time domain bandpass

In bandpass mode, you can simulate the DUT response to an impulse signal.

The trace format can be set to **`LINEAR, LOGMAG or SWR`** .
The following is an example of the impulse response of a bandpass filter.

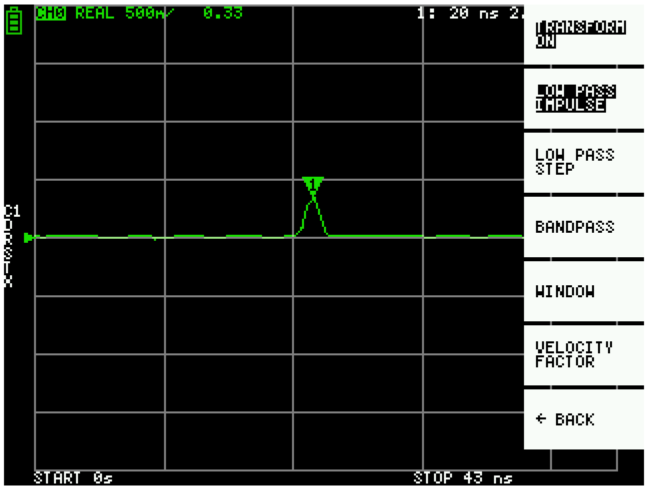


Time domain low pass impulse

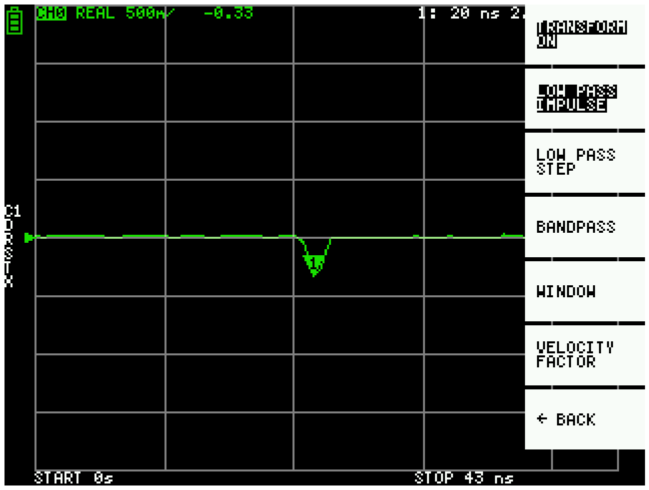
In low-pass mode, you can simulate TDR. In low-pass mode, the start frequency must be set to 50 kHz, and the stop frequency must be set according to the distance to be measured.

The trace format can be set to **`REAL`** .
Examples of Impulse response in open state and impulse response in short state are shown below.

Open:



Short:



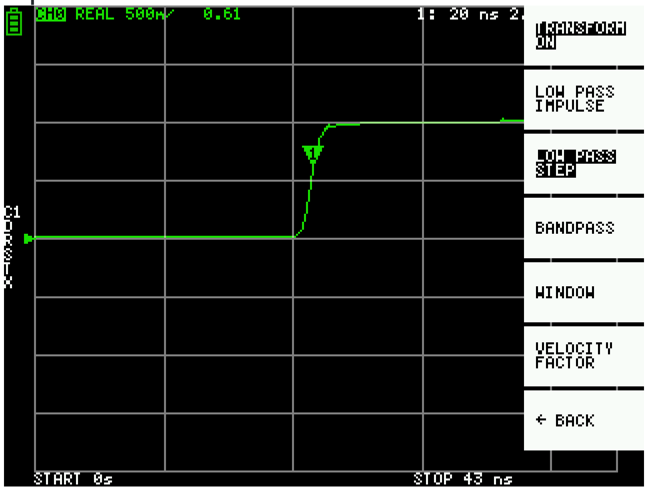
Time domain low pass step

In low-pass mode, you can simulate TDR. In low-pass mode, the start frequency must be set to 50 kHz, and the stop frequency must be set according to the distance to be measured.

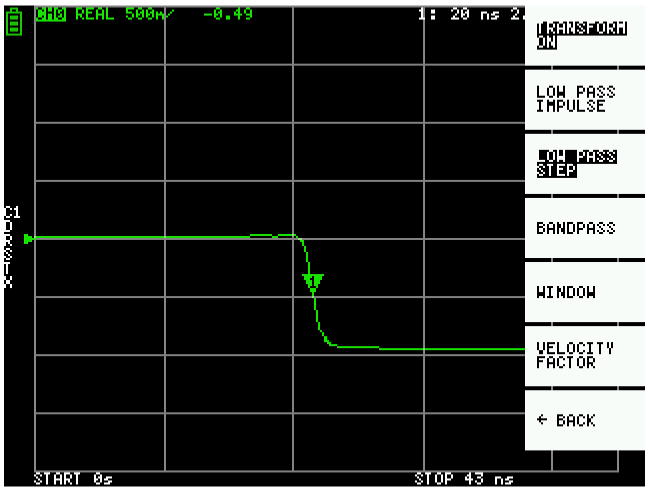
The trace format can be set to **`REAL`** .

Examples of Step response in open state and Step response in short state are shown below.

Open:

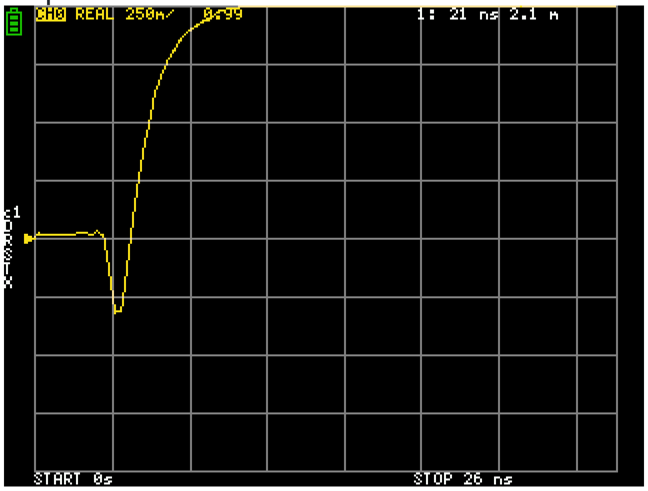


Short:

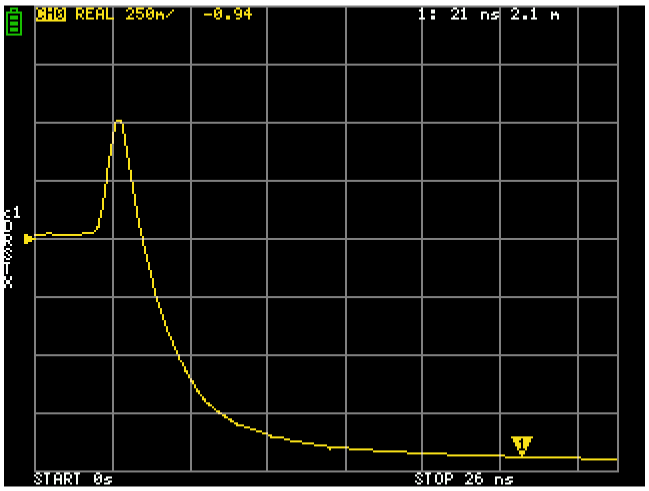


Step response examples

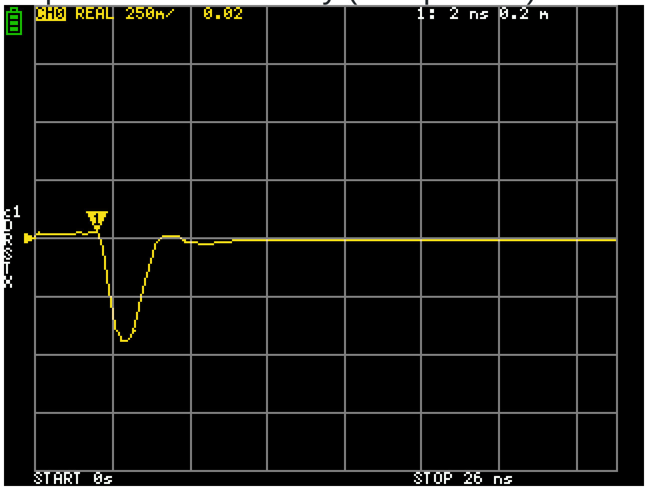
Capacitive short:



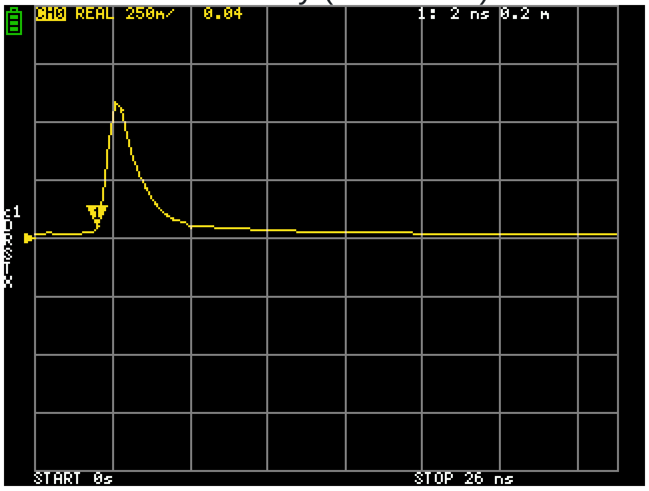
Inductive short:



Capacitive discontinuity (C in parallel):



Inductive discontinuity (L in series):



Time domain window

The range that can be measured is a finite number, and there is a minimum frequency and a maximum frequency. A window can be used to smooth out this discontinuous measurement data and reduce ringing.

There are three levels of windows.

- MINIMUM (no window, ie same as rectangular window)
- NORMAL (equivalent to Kaiser window β = 6)
- MAXIMUM (equivalent to Kaiser window β = 13)

MINIMUM provides the highest resolution and MAXIMUM provides the highest dynamic range. NORMAL is in the middle.

Setting the wavelength (Velocity) factor in the time domain

The transmission speed of electromagnetic waves in the cable varies depending on the material. The ratio to the transmission speed of electromagnetic waves in vacuum is called the wavelength factor (Velocity Factor, Velocity of propagation). This is always stated in the cable specifications.

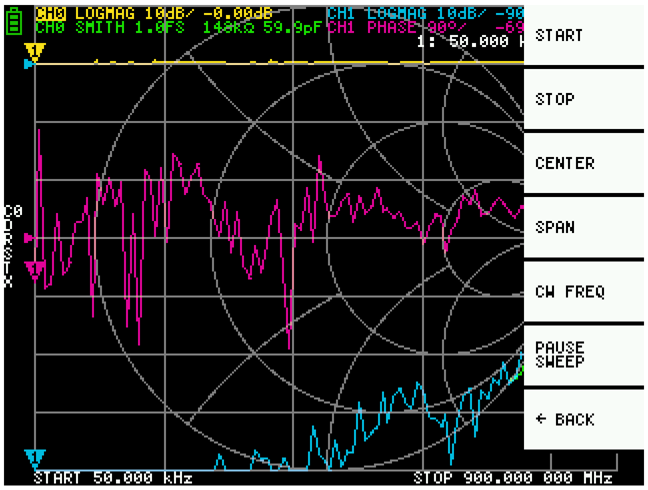
In the time domain, the displayed time can be converted into distance. The wavelength shortening ratio used for distance display can be set with **`DISPLAY →TRANSFORM →VELOCITY FACTOR`** .
For example, if you measure the TDR of a cable with a wavelength reduction rate of 67%, specify **`67`** for the **`VELOCITY FACTOR`** .

Set frequency from marker

You can set the frequency range from the marker as follows:

• **`MARKER →START`**	Sets the active marker frequency to the start frequency.
• **`MARKER →STOP`**	Sets the active marker frequency to the stop frequency.
• **`MARKER →CENTER`**	Sets the frequency of the active marker to the center frequency. The span is adjusted to maintain the current range as much as possible.
• **`MARKER →SPAN`**	Sets the two displayed markers including the active marker to the span. If only one marker is displayed, nothing happens.

Setting the measurement range



There are three types of measurement range settings.

- Setting the start frequency and stop frequency
- Setting the center frequency and span
- Zero span

Setting the start frequency and stop frequency

Select and set **`STIMULUS →START`** and **`STIMULUS →STOP`** , respectively.

Setting the center frequency and span

Select and set **`STIMULUS →CENTER`** and **`STIMULUS →SPAN`** respectively.

Zero span

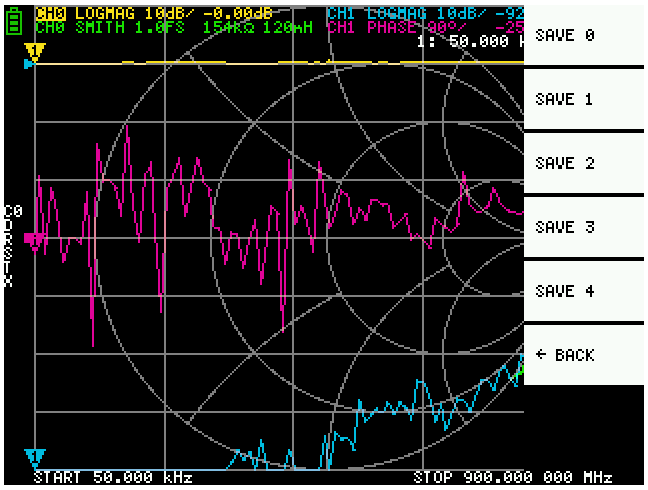
Zero span is a mode in which one frequency is sent continuously without frequency sweep.

Select and set **`STIMULUS →CW FREQ`** .

Temporarily stop measurement

`STIMULUS →PAUSE SWEEP` is selected, measurement is temporarily stopped.

Recall calibration and settings

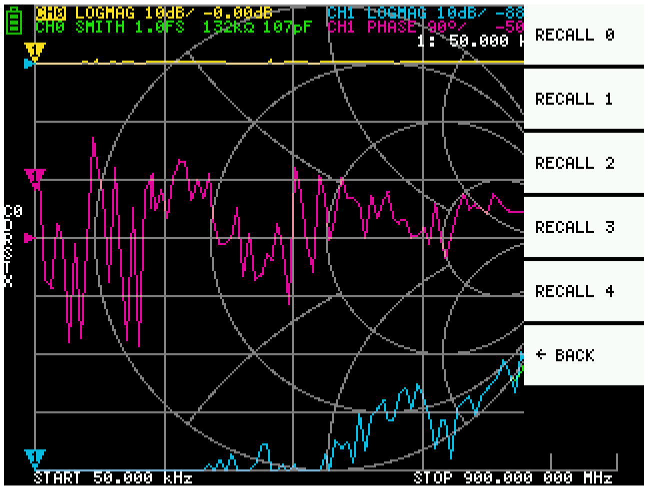


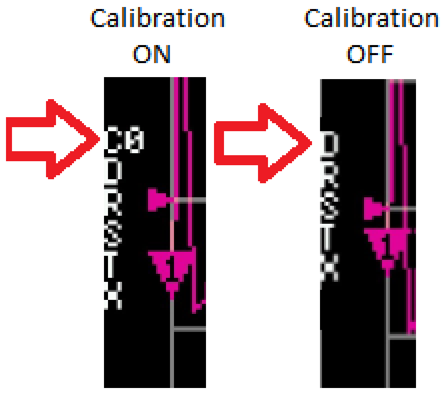
Up to 5 calibration datasets can be saved. NanoVNA loads number 0 data immediately after startup.

Calibration data is data that includes the following information:

- Frequency setting range
- Error correction at each measurement point
- Trace setting status
- Marker setting status
- Domain mode settings
- Setting the wavelength shortening rate
- electrical delay

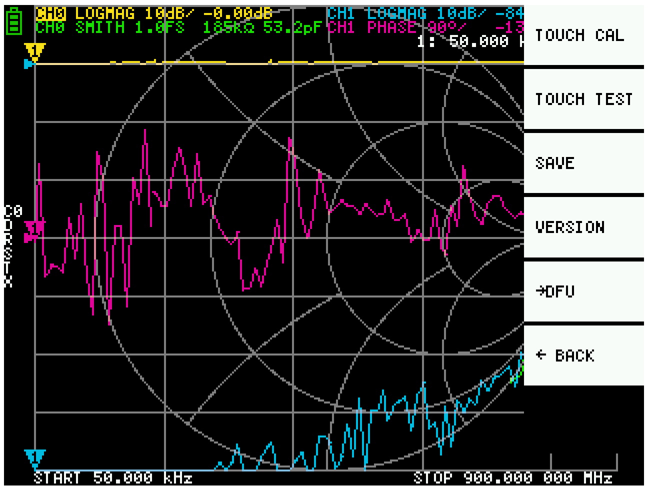
You can save the current settings by selecting `CAL MENU →SAVE →SAVE n`.
Current calibration data can be reset by selecting `CAL MENU →RESET`.
NOTE: If you want to recalibrate, you ***need*** to reset.
`CAL MENU →CORRECTION` indicates whether error correction is currently being performed. You can select this to temporarily stop error correction. (Inverted **button** text=**ON**, Normal button text=**OFF**)





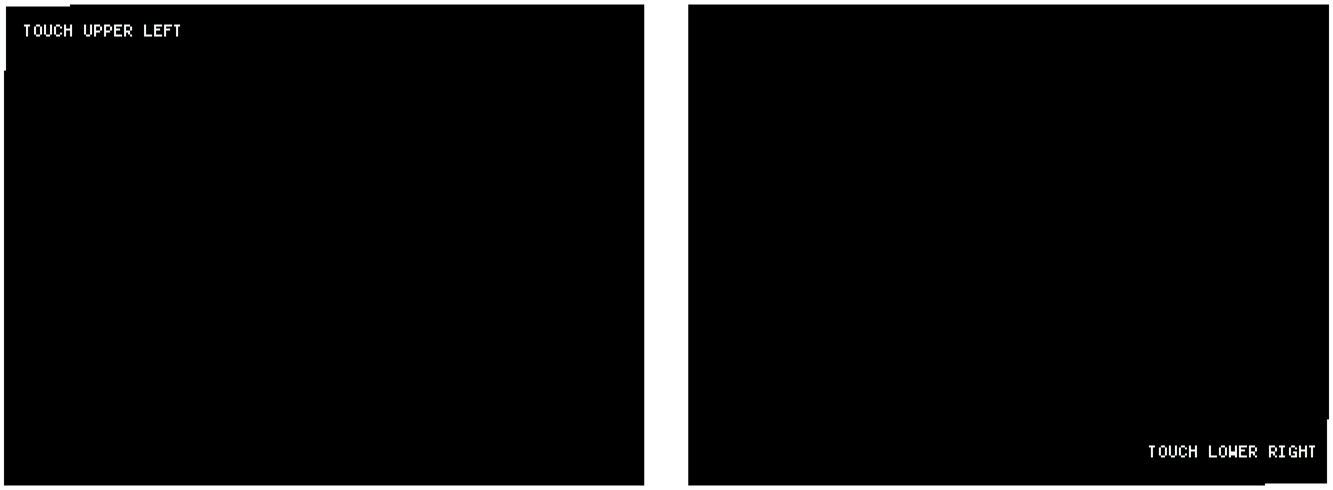
`RECALL` the saved settings by selecting `CAL MENU →RECALL →RECALL n`

Device settings

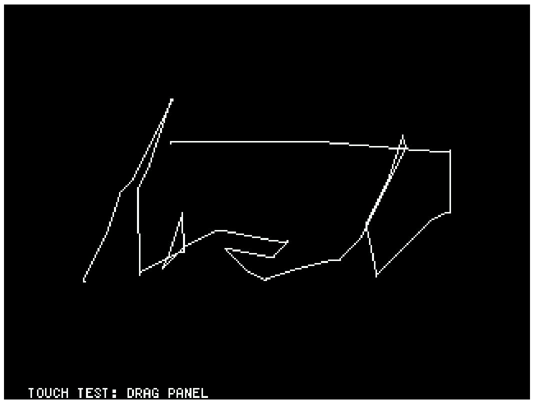


The **`CONFIG`** menu allows you to make general settings for the device.

Touch panel calibration and testing



`CONFIG` →**`TOUCH CAL`** is selected, the touch panel can be calibrated. If there is a large difference between the actual tap position and the recognized tap position, this can be solved by executing this. After performing **`TOUCH CAL`** , perform **`TOUCH TEST`** to confirm that the settings are correct. NOTE: Be sure to save the settings with **`SAVE`** .



You can test the touch panel by selecting `CONFIG →TOUCH TEST`. A line is drawn while dragging the stylus along the touch panel. When released from the touch panel, it returns to its original state.

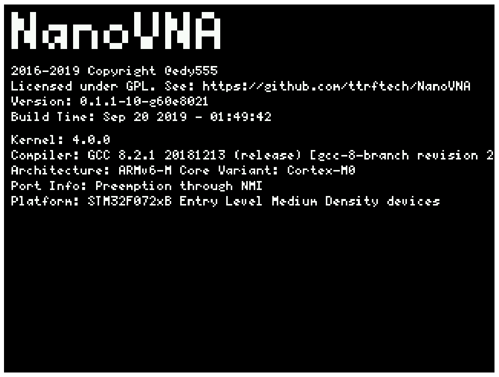
Saving device settings

Select `CONFIG →SAVE` to save general instrument settings. General device settings are data that includes the following information:

- Touch panel calibration information
- Grid color
- Trace color
- Calibration data number loaded by default

There is currently no way to set other than touch panel calibration information.

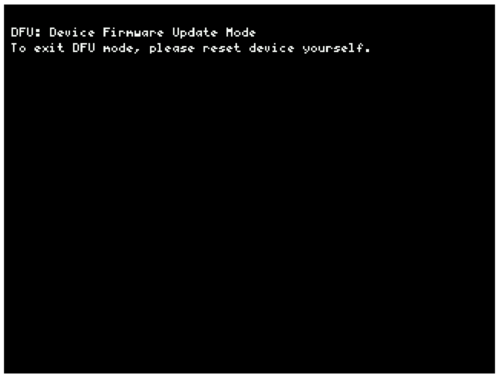
Display version



```
NanoVNA

2016-2019 Copyright @edy555
Licensed under GPL. See: https://github.com/ttrftech/NanoVNA
Version: 0.1.1-10-g60e8021
Build Time: Sep 20 2019 - 01:49:42

Kernel: 4.0.0
Compiler: GCC 8.2.1 20181213 (release) [gcc-8-branch revision 2
Architecture: ARMv6-M Core Variant: Cortex-M0
Port Info: Preemption through NMI
Platform: STM32F072xB Entry Level Medium Density devices
```

Select `CONFIG →VERSION` to display device version information.

Firmware update



```
DFU: Device Firmware Update Mode
To exit DFU mode, please reset device yourself.
```

`CONFIG →DFU RESET` and `ENTER DFU` mode . Select `RESET AND ENTER DFU` to reset the device and enter DFU (Device Firmware Update) mode. In this mode, firmware can be updated via USB.

8. How to update the firmware

How to obtain the firmware

ttrftech version firmware

Original firmware. It is versioned and frequently developed.

- GitHub releases
- CircleCI build

GitHub releases has some stable release firmware.

CircleCI has all the firmware with every commit. Use this if you want to try the latest features or check for problems.

hugen79 version firmware

- Google Drive

Google Drive has the latest firmware.

Build yourself

You can easily clone the github repository and build it yourself.

How to write firmware

There are various ways to write, but here we will explain using dfu-util . dfu-util is a cross-platform tool, and binaries are also provided on Windows.

Writing with dfu-util (Ubuntu)

There is dfu-util in the standard package repository.

```
sudo apt-get install dfu-util dfu-util --version
```

Start the device in DFU mode. Use one of the following methods to enter DFU mode.

- Turn on the power while jumpering the BOOT0 pin on the PCB. (Remove the jumper after turning on the power.) The screen turns white, but it is normal.
- `CONFIG` →`DFU` `RESET` AND `ENTER DFU` Select `RESET AND ENTER DFU`

Run the following command: build / ch.bin describes the path to the downloaded firmware file .bin.

```
dfu-util -d 0483:df11 -a 0 -s 0x08000000:leave -D build/ch.bin
```

Writing with dfu-util (macOS)

It is recommended to install using homebrew .

Install the brew command.

```
ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

Install dfu-util command.

```
brew install dfu-util
```

Confirm that the dfu-util command can be started normally.

```
dfu-util --version
```

Start the device in DFU mode. Use one of the following methods to enter DFU mode.

- Turn on the power while jumpering the BOOT0 pin on the PCB. (Remove the jumper after turning on the power.) The screen turns white, but it is normal.
- `CONFIG` →`DFU` `RESET AND ENTER DFU` Select `RESET AND ENTER DFU`

Run the following command: build / ch.bin describes the path to the downloaded firmware file .bin.

```
dfu-util -d 0483:df11 -a 0 -s 0x08000000:leave -D build/ch.bin
```

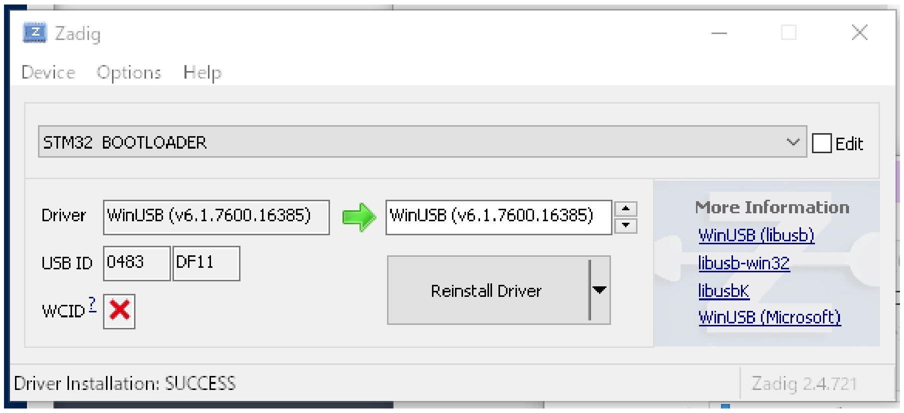
Writing with dfu-util (Windows 10)

For Windows, the device driver is automatically installed when NanoVNA in DFU mode is connected, but dfu-util cannot be used with this device driver. Here, Zadig is used to replace the driver.

Start the device in DFU mode. Use one of the following methods to enter DFU mode.

- Turn on the power while jumpering the BOOT0 pin on the PCB. (Remove the jumper after turning on the power.) The screen turns white, but it is normal.
- `CONFIG` →`DFU` `RESET AND ENTER DFU` Select `RESET AND ENTER DFU`

Start Zadig with NanoVNA in DFU mode connected and use WinUSB as a driver for STM32 BOOTLOADER as follows.



STM32 BOOTLOADER

Driver WinUSB (v6.1.7600.16385) → WinUSB (v6.1.7600.16385)

USB ID 0483 DF11

WCID ? ✗

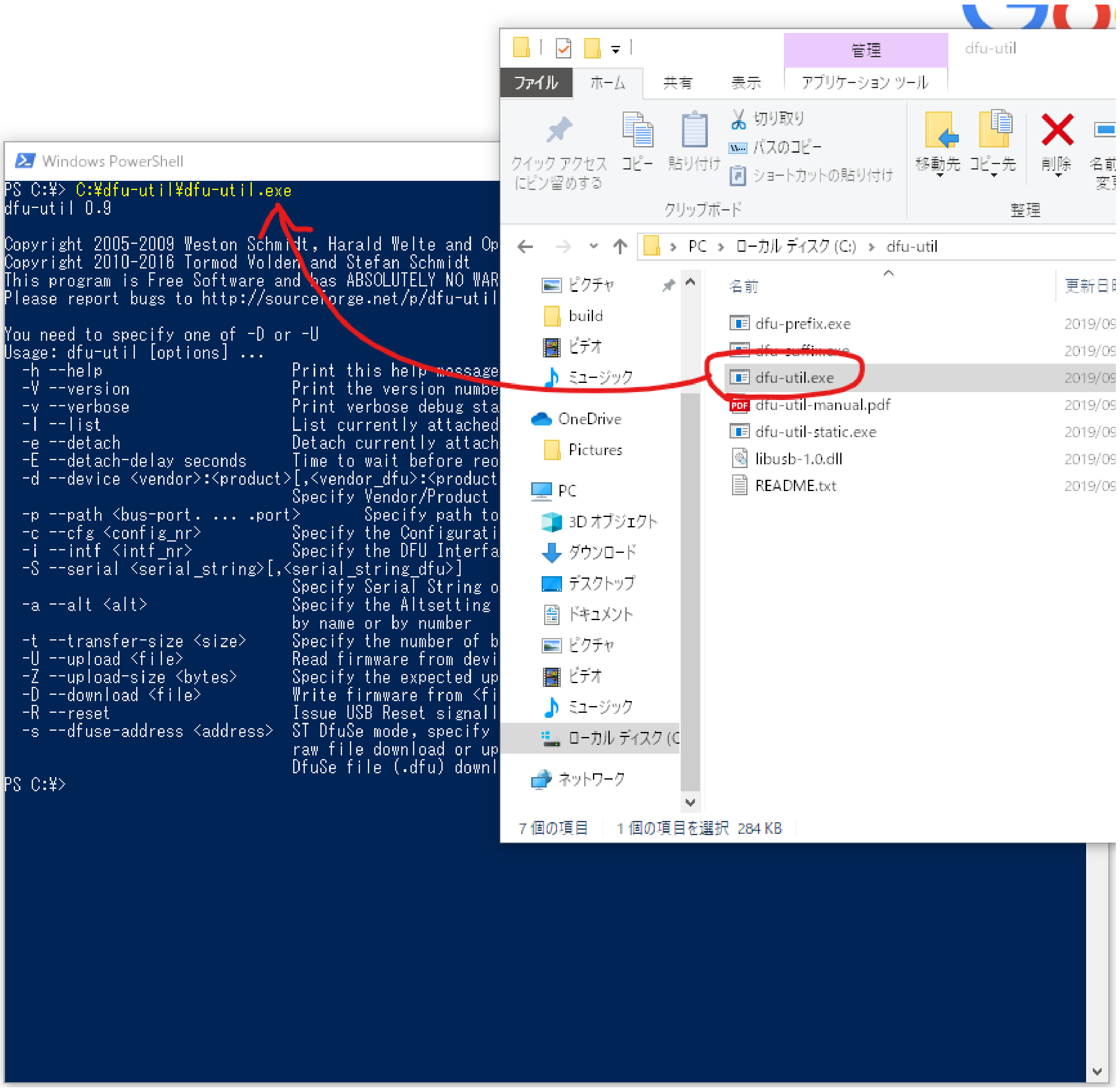
Driver Installation: SUCCESS

Zadig 2.4.721

* If you want to restore the driver, find the corresponding device from “Universal Serial Bus Controller” in “Device Manager” and execute “Uninstall Device”. The driver is automatically installed when the USB connector is disconnected and reinserted.

Next, place dfu-util. Download dfu-util-0.9-win64.zip from releases and extract it. Here, as an example, it is assumed that it is expanded to C: \ dfu-util (it does not matter anywhere).

Right-click on the Start menu and select Windows PowerShell. A shell screen opens.

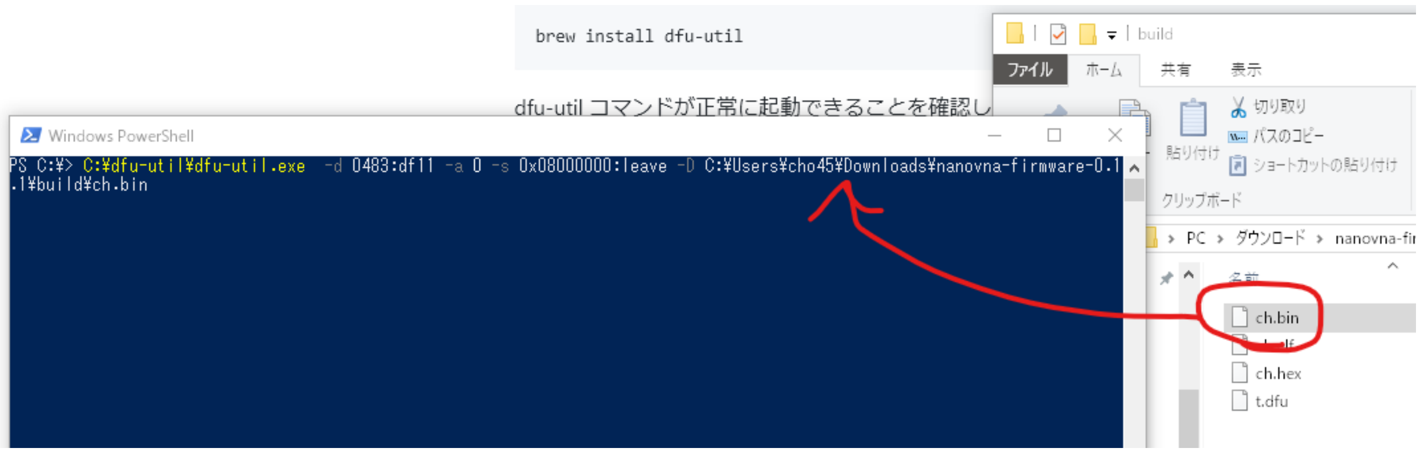


```
PS C:¥> C:¥dfu-util¥dfu-util.exe
dfu-util 0.9

Copyright 2005-2009 Weston Schmidt, Harald Welte and Op
Copyright 2010-2016 Tormod Volden and Stefan Schmidt
This program is Free Software and has ABSOLUTELY NO WAR
Please report bugs to http://sourceforge.net/p/dfu-util

You need to specify one of -D or -U
Usage: dfu-util [options] ...
  -h --help                     Print this help message
  -V --version                  Print the version numbe
  -v --verbose                  Print verbose debug sta
  -l --list                     List currently attached
  -e --detach                   Detach currently attach
  -E --detach-delay seconds     Time to wait before reo
  -d --device <vendor>:<product>[,<vendor_dfu>:<product
                                Specify Vendor/Product
  -p --path <bus-port. ... .port>       Specify path to
  -c --cfg <config_nr>          Specify the Configurati
  -i --intf <intf_nr>           Specify the DFU Interfa
  -S --serial <serial_string>[,<serial_string_dfu>]
                                Specify Serial String o
  -a --alt <alt>                Specify the Altsetting
                                by name or by number
  -t --transfer-size <size>     Specify the number of b
  -U --upload <file>            Read firmware from devi
  -Z --upload-size <bytes>      Specify the expected up
  -D --download <file>          Write firmware from <fi
  -R --reset                    Issue USB Reset signall
  -s --dfuse-address <address>  ST DfuSe mode, specify
                                raw file download or up
                                DfuSe file (.dfu) downl
PS C:¥>
```

名前	更新日
dfu-prefix.exe	2019/09
dfu-suffix.exe	2019/09
dfu-util.exe	2019/09
dfu-util-manual.pdf	2019/09
dfu-util-static.exe	2019/09
libusb-1.0.dll	2019/09
README.txt	2019/09

When dfu-util.exe is dragged and dropped from Windows Explorer to PowerShell, the path is automatically inserted. Dfu-util version can be displayed by starting with `--version` as follows.

```
C:\dfu-util\dfu-util.exe --version
```



```
brew install dfu-util
```

dfu-util コマンドが正常に起動できることを確認し

```
PS C:¥> C:¥dfu-util¥dfu-util.exe -d 0483:df11 -a 0 -s 0x08000000:leave -D C:¥Users¥cho45¥Downloads¥nanovna-firmware-0.1
.1¥build¥ch.bin
```

Similarly, the firmware file can also be entered by dragging and dropping it from Explorer to PowerShell.

Run the following command: build / ch.bin describes the path to the downloaded firmware file .bin.

```
C:\dfu-util\dfu-util.exe -d 0483:df11 -a 0 -s 0x08000000:leave -D build\ch.bin
```

How to write firmware (Windows GUI)

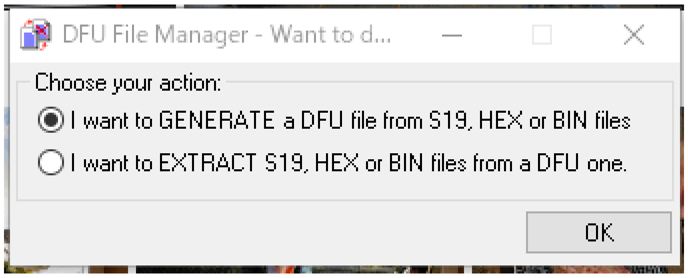
For those who are unfamiliar with CUI, a little troublesome procedure is required, but the method of writing using the DfuSE Demo tool provided by ST is also introduced for reference.

Download STSW-STM32080 from the ST site .

- DFU File Manager: a tool to create .dfu files from .bin or .hex
- DfuSe Demo: A tool for writing .dfu files to devices is included.

Convert file format with DFU File Manager.

First, start DFU File Manager.



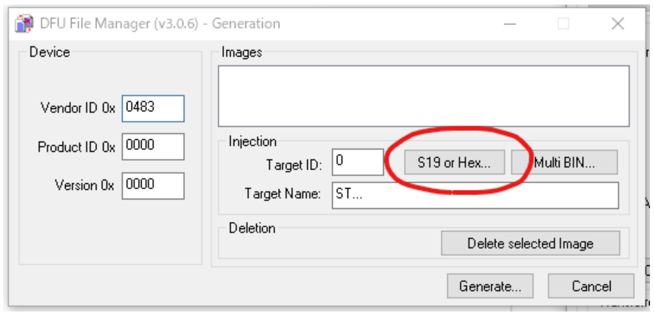
DFU File Manager - Want to d...

Choose your action:

[✓ I want to GENERATE a DFU file from S19, HEX or BIN files]

I want to EXTRACT S19, HEX or BIN files from a DFU one.

Select: **`I want to GENERATE a DFU file from S19, HEX or BIN files`** .



Vendor ID 0x 0483
Product ID 0x 0000
Version 0x 0000
Target ID: 0
Target Name: ST...

Click the **`S19 or Hex...`** button. **`ch.hex`** firmware **`ch.hex`** file such as **`ch.hex .`**

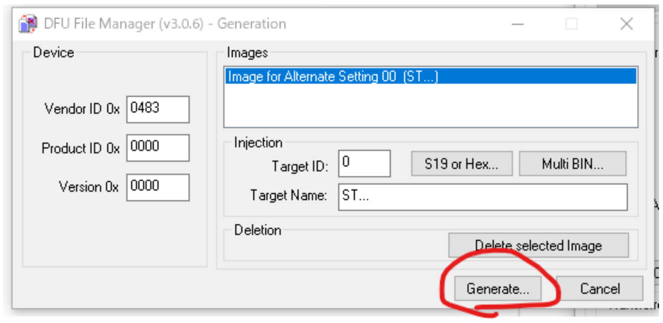


Image for Alternate Setting 00 (ST...)
Vendor ID 0x 0483
Product ID 0x 0000
Version 0x 0000
Target ID: 0
Target Name: ST...

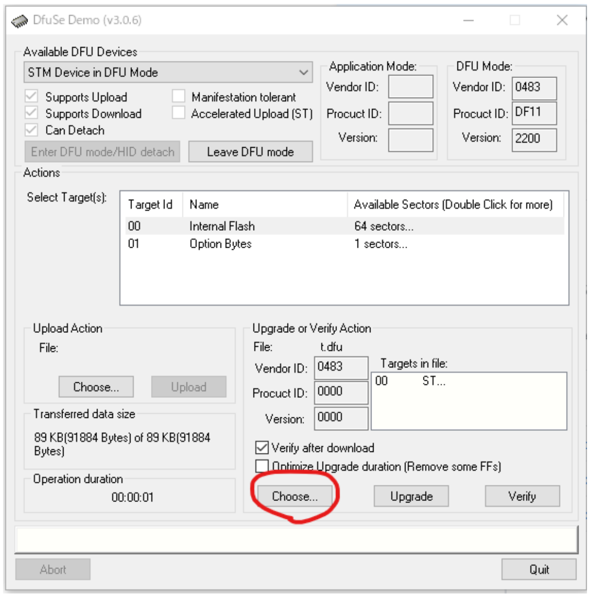
Click the **`Generate...`** button and create a .dfu file with a suitable name.

Write firmware with DfuSe Demo

First start the device in DFU mode. Use one of the following methods to enter DFU mode.

- Turn on the power while jumpering the BOOT0 pin on the PCB. (Remove the jumper after turning on the power.) The screen turns white, but it is normal.
- **`CONFIG →DFU RESET`** `AND` **`ENTER DFU`** Select **`RESET AND ENTER DFU`**

Start DfuSe Demo. Make sure that Available DFU Devices has ***`STM Device in DFU Mode`*** and click **`Choose...`**



STM Device in DFU Mode
☑ Supports Upload ☐ Manifestation tolerant
☑ Supports Download ☐ Accelerated Upload (ST)
☑ Can Detach

Application Mode:		DFU Mode:	
Vendor ID:		Vendor ID:	0483
Procuct ID:		Procuct ID:	DF11
Version:		Version:	2200

Select Target(s):

Target Id	Name	Available Sectors (Double Click for more)
00	Internal Flash	64 sectors...
01	Option Bytes	1 sectors...

File:

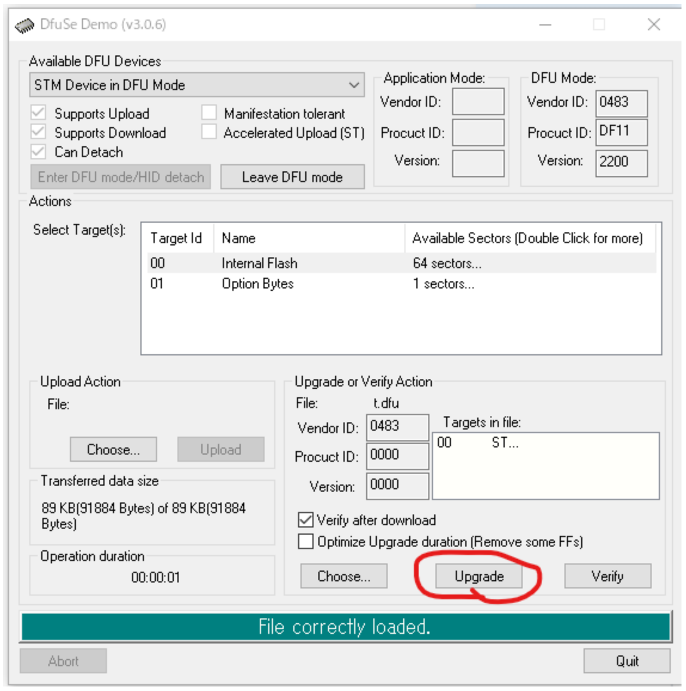
Transferred data size
89 KB(91884 Bytes) of 89 KB(91884 Bytes)

Operation duration
00:00:01

File: t.dfu
Vendor ID: 0483
Procuct ID: 0000
Version: 0000
Targets in file:
00 ST...

☑ Verify after download
☐ Optimize Upgrade duration (Remove some FFs)

Select the .dfu file you saved earlier.


DfuSe Demo (v3.0.6)

Available DFU Devices: STM Device in DFU Mode

☑ Supports Upload ☐ Manifestation tolerant
☑ Supports Download ☐ Accelerated Upload (ST)
☑ Can Detach

Application Mode: Vendor ID: Procuct ID: Version:

DFU Mode: Vendor ID: 0483 Procuct ID: DF11 Version: 2200

Actions

Select Target(s):

Target Id	Name	Available Sectors (Double Click for more)
00	Internal Flash	64 sectors...
01	Option Bytes	1 sectors...

Upload Action
File:

Transferred data size
89 KB(91884 Bytes) of 89 KB(91884 Bytes)

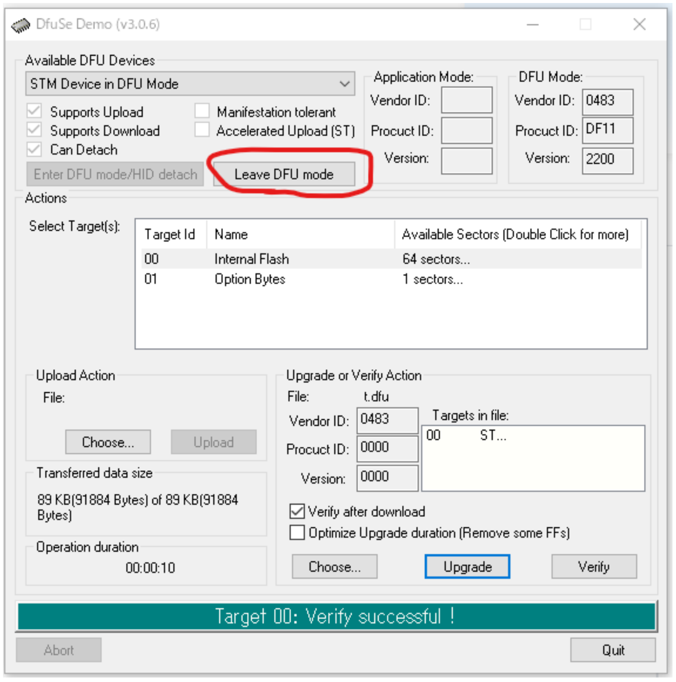
Operation duration
00:00:01

Upgrade or Verify Action
File: t.dfu
Vendor ID: 0483
Procuct ID: 0000
Version: 0000
Targets in file:
00 ST...

☑ Verify after download
☐ Optimize Upgrade duration (Remove some FFs)

File correctly loaded.

Click the **`Upgrade`** button.


DfuSe Demo (v3.0.6)

Available DFU Devices: STM Device in DFU Mode

☑ Supports Upload ☐ Manifestation tolerant
☑ Supports Download ☐ Accelerated Upload (ST)
☑ Can Detach

Application Mode: Vendor ID: Procuct ID: Version:

DFU Mode: Vendor ID: 0483 Procuct ID: DF11 Version: 2200

Actions

Select Target(s):

Target Id	Name	Available Sectors (Double Click for more)
00	Internal Flash	64 sectors...
01	Option Bytes	1 sectors...

Upload Action
File:

Transferred data size
89 KB(91884 Bytes) of 89 KB(91884 Bytes)

Operation duration
00:00:10

Upgrade or Verify Action
File: t.dfu
Vendor ID: 0483
Procuct ID: 0000
Version: 0000
Targets in file:
00 ST...

☑ Verify after download
☐ Optimize Upgrade duration (Remove some FFs)

Target 00: Verify successful !

This screen will be displayed when writing is complete. Click the **`Leave DFU mode`** button to exit DFU mode. The device will reset and boot with the new firmware.

9. Firmware development guide

NanoVNA firmware development needs are as follows.

- Git
- gcc-arm-none-eabi
- make

If you already have these, you can build the firmware with **`make`** .

```
 git clone git@github.com:ttrftech/NanoVNA.git cd NanoVNA git submodule update --init --
recursive make
```

Build with Docker

Use docker to build without bothering you. docker is a free, cross-platform container utility. It can be used to quickly reproduce a specific environment (in this case, the build environment).

Just install docker and run the following command:

```
docker run -it --rm -v $(PWD):/work edy555/arm-embedded:8.2 make
```

On-chip debugging with Visual Studio Code

Visual Studio Code (hereinafter VSCode) is a multi-platform code editor provided free of charge by Microsoft. By installing Cortex-Debug Extension, on-chip debugging can be done with GUI.

The platform-dependent part is omitted, but in addition to the above, the following are required.

- openocd
- VSCode
- Cortex-Debug

Cortex-Debug is searched from Extensions of VSCode and installed.

tasks.json

First, define a “task” to make the entire NanoVNA on VSCode.

```
{
    "tasks" :   [
        {
            "type" :   "shell" ,
            "label" :   "build" ,
            "command" :   "make" ,
            "args" :   [
            ],
            "options" :   {
                "cwd" :   "${workspaceRoot}"
            }
        }
    ],
    "version" :   "2.0.0"
}
```

Now you can make it as a task on VSCode.

launch.json

Next, define how to start during Debug. Set as described in Cortex-Debug.

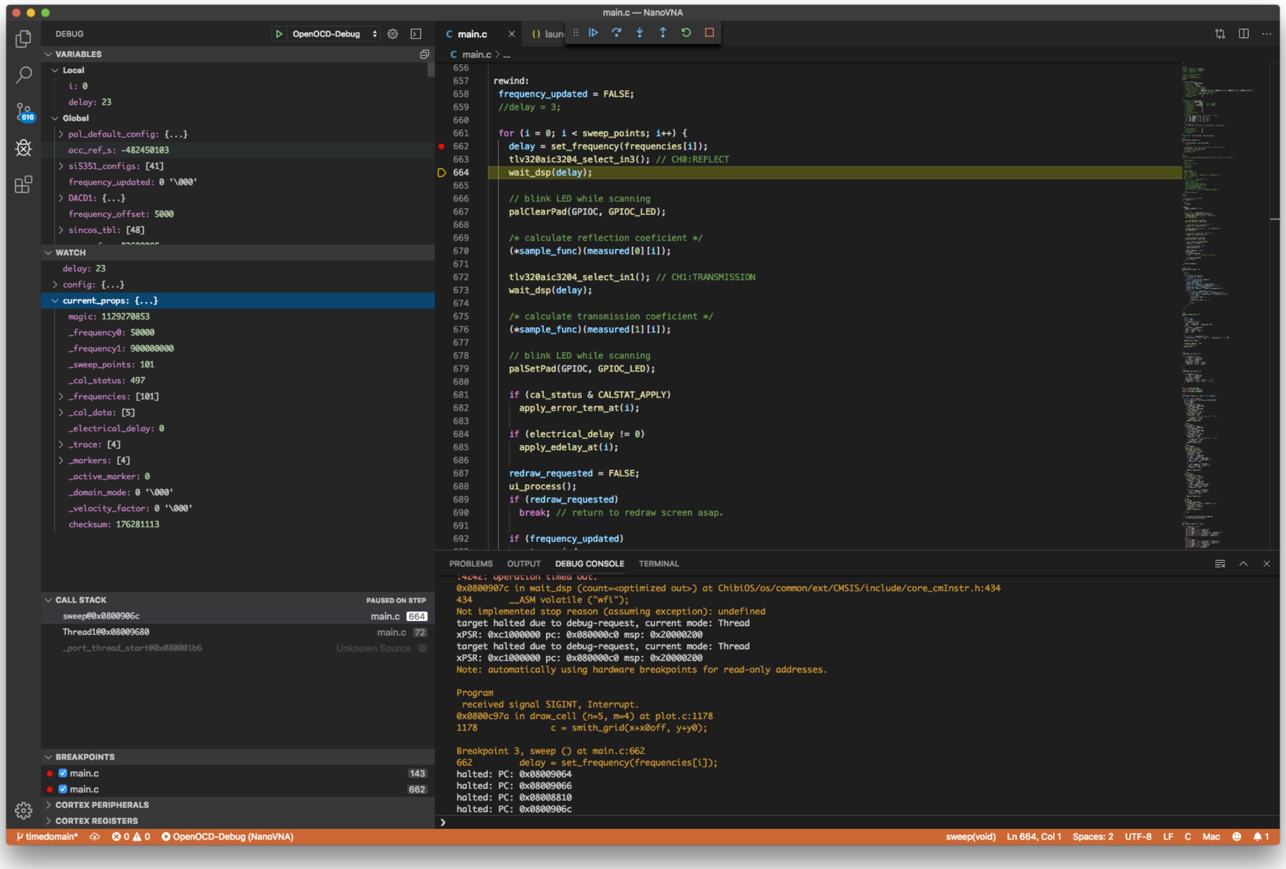
The following settings are for ST-Link. If you use J-Link,
replace **`interface/stlink.cfg`** with **`interface/jlink.cfg`** .

```
{
    "version" :   "0.2.0" ,
    "configurations" :   [
        {
            "type" :   "cortex-debug" ,
            "servertype" :   "openocd" ,
            "request" :   "launch" ,
            "name" :   "OpenOCD-Debug" ,
            "executable" :   "build/ch.elf" ,
            "configFiles" :   [
                "interface/stlink.cfg" ,
                "target/stm32f0x.cfg"
            ],
            "svdFile" :   "./STM32F0x8.svd" ,
            "cwd" :   "${workspaceRoot}" ,
            "preLaunchTask" :   "build" ,
        }
    ]
}
```

`svdFile` file specified in **`svdFile`** can be downloaded from the ST site . **`svdFile`** is not specified, there is no problem in operation.

Start debugging

When Start Debugging (**`F5`**) is performed, OpenOCD starts automatically after the build by make and the firmware is transferred. When the transfer is complete, the reset handler breaks.



main.c — NanoVNA

DEBUG ▷ OpenOCD-Debug

```
VARIABLES
  Local
    i: 0
    delay: 23
  Global
    > pal_default_config: {...}
      acc_ref_s: -482450103
    > si5351_configs: [41]
      frequency_updated: 0 '\000'
    > DACD1: {...}
      frequency_offset: 5000
    > sincos_tbl: [48]
WATCH
    delay: 23
  > config: {...}
  current_props: {...}
      magic: 1129270853
      _frequency0: 50000
      _frequency1: 900000000
      _sweep_points: 101
      _cal_status: 497
    > _frequencies: [101]
    > _cal_data: [5]
      _electrical_delay: 0
    > _trace: [4]
    > _markers: [4]
      _active_marker: 0
      _domain_mode: 0 '\000'
      _velocity_factor: 0 '\000'
      checksum: 176281113
CALL STACK                                    PAUSED ON STEP
    sweep@0x0800906c                          main.c 664
    Thread1@0x08009680                        main.c 72
    _port_thread_start@0x080001b6             Unknown Source 0
BREAKPOINTS
  ● ✓ main.c                                  143
  ● ✓ main.c                                  662
> CORTEX PERIPHERALS
> CORTEX REGISTERS
```

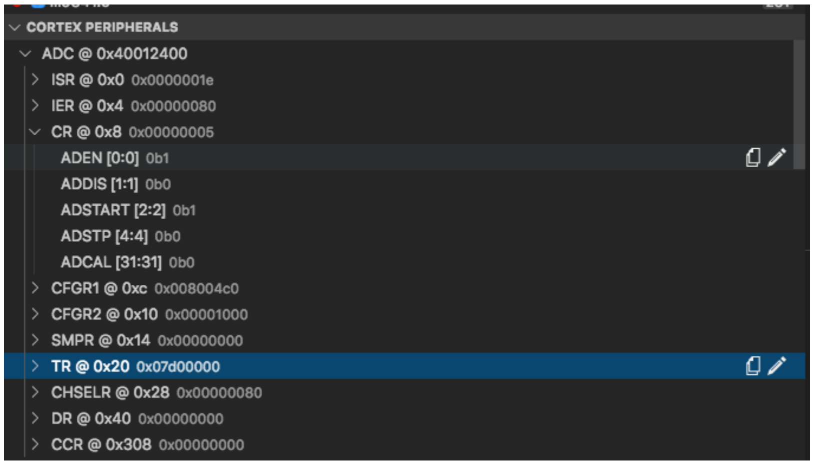
```c
  rewind:
    frequency_updated = FALSE;
    //delay = 3;

    for (i = 0; i < sweep_points; i++) {
      delay = set_frequency(frequencies[i]);
      tlv320aic3204_select_in3(); // CH0:REFLECT
      wait_dsp(delay);

      // blink LED while scanning
      palClearPad(GPIOC, GPIOC_LED);

      /* calculate reflection coeficient */
      (*sample_func)(measured[0][i]);

      tlv320aic3204_select_in1(); // CH1:TRANSMISSION
      wait_dsp(delay);

      /* calculate transmission coeficient */
      (*sample_func)(measured[1][i]);

      // blink LED while scanning
      palSetPad(GPIOC, GPIOC_LED);

      if (cal_status & CALSTAT_APPLY)
        apply_error_term_at(i);

      if (electrical_delay != 0)
        apply_edelay_at(i);

      redraw_requested = FALSE;
      ui_process();
      if (redraw_requested)
        break; // return to redraw screen asap.

      if (frequency_updated)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
:4242: Operation timed out.
0x0800907c in wait_dsp (count=<optimized out>) at ChibiOS/os/common/ext/CMSIS/include/core_cmInstr.h:434
434       __ASM volatile ("wfi");
Not implemented stop reason (assuming exception): undefined
target halted due to debug-request, current mode: Thread
xPSR: 0xc1000000 pc: 0x080000c0 msp: 0x20000200
target halted due to debug-request, current mode: Thread
xPSR: 0xc1000000 pc: 0x080000c0 msp: 0x20000200
Note: automatically using hardware breakpoints for read-only addresses.

Program
 received signal SIGINT, Interrupt.
0x0800c97a in draw_cell (n=5, m=4) at plot.c:1178
1178              c = smith_grid(x+x0off, y+y0);

Breakpoint 3, sweep () at main.c:662
662         delay = set_frequency(frequencies[i]);
halted: PC: 0x08009064
halted: PC: 0x08009066
halted: PC: 0x08008810
halted: PC: 0x0800906c
```

timedomain* ⊗ 0 ⚠ 0 OpenOCD-Debug (NanoVNA) sweep(void) Ln 664, Col 1 Spaces: 2 UTF-8 LF C Mac 1

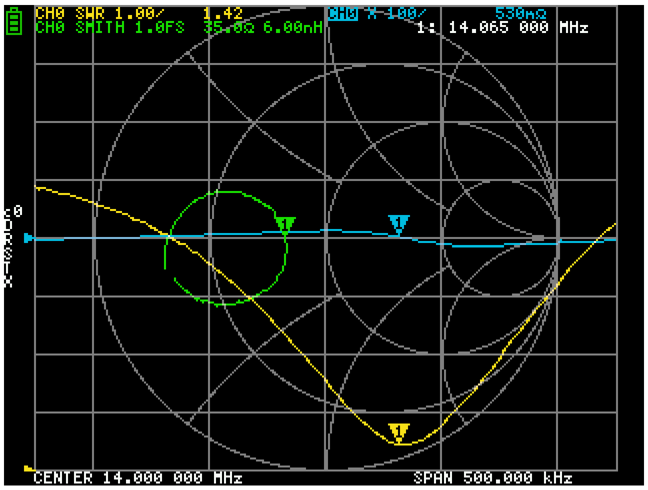
`svdFile` is specified, the defined MCU registers are displayed on the debug screen.



```
CORTEX PERIPHERALS
  ADC @ 0x40012400
    > ISR @ 0x0 0x0000001e
    > IER @ 0x4 0x00000080
      CR @ 0x8 0x00000005
        ADEN [0:0] 0b1
        ADDIS [1:1] 0b0
        ADSTART [2:2] 0b1
        ADSTP [4:4] 0b0
        ADCAL [31:31] 0b0
    > CFGR1 @ 0xc 0x008004c0
    > CFGR2 @ 0x10 0x00001000
    > SMPR @ 0x14 0x00000000
    > TR @ 0x20 0x07d00000
    > CHSELR @ 0x28 0x00000080
    > DR @ 0x40 0x00000000
    > CCR @ 0x308 0x00000000
```

10. Example of use

Antenna adjustment



An example of using NanoVNA as an antenna analyzer is shown below.

There are two important points in antenna adjustment:

- Whether the antenna is tuned and resonant (i.e., reactance is close to 0 at the desired frequency)
- Is the antenna's SWR low?

Trace settings

Since only CH0 is used for antenna adjustment, calibration is performed for all items except `THRU` and `ISOLN` .
Set the trace as follows.

- Trace 0: CH0 SWR
- Trace 1: CH0 REACTANCE
- Trace 2: CH0 SMITH
- Trace 3: OFF

Set the frequency you want to tune the antenna to `CENTER` and set `SPAN` appropriately.
Look for frequencies where trace 1 displaying reactance is close to zero. Since the frequency is the tuning point, adjust the antenna if it is deviated so that the tuning point comes to the target frequency.

If the tuning point is at the desired frequency, check that trace 0 displaying the SWR is displaying a sufficiently low (close to 1) SWR. If the SWR is not enough (less than 2), the Smith chart is used for matching. In this case, matching may be performed using an antenna tuner directly under the antenna.

If the SWR drops, the antenna is tuned at the desired frequency and the adjustment of the antenna with a low SWR is complete.

Adjusting the bandpass filter

TODO

Check the cable

You can simulate TDR using the time domain low pass mode. By using TDR, you can find faults in the transmission path.

TODO

Common mode filter measurement

TODO