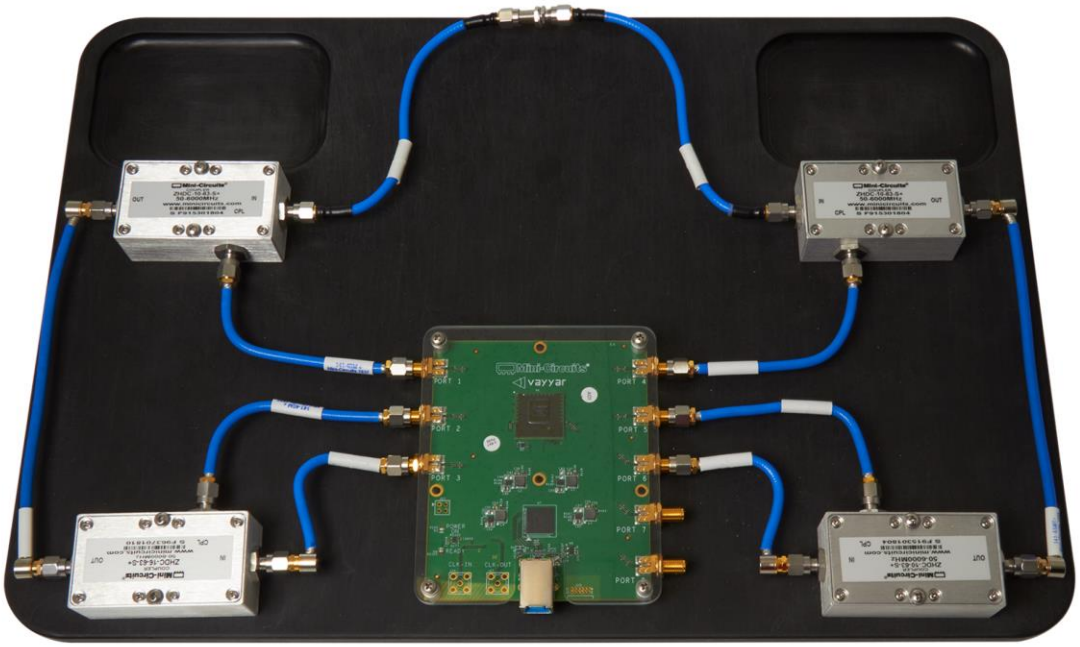
MICROWAVE TRANSCEIVER KIT

PROJECT NO. 1: UVNA-63

DIY VECTOR NETWORK ANALYZER



Software and Programming Guide

Mini-Circuits®

Important Notice

Trademarks

Mini-Circuits
13 Neptune Avenue
Brooklyn, NY 11235
Phone: 1-718-934-4500
Email: apps@minicircuits.com
Web: www.minicircuits.com

Contents

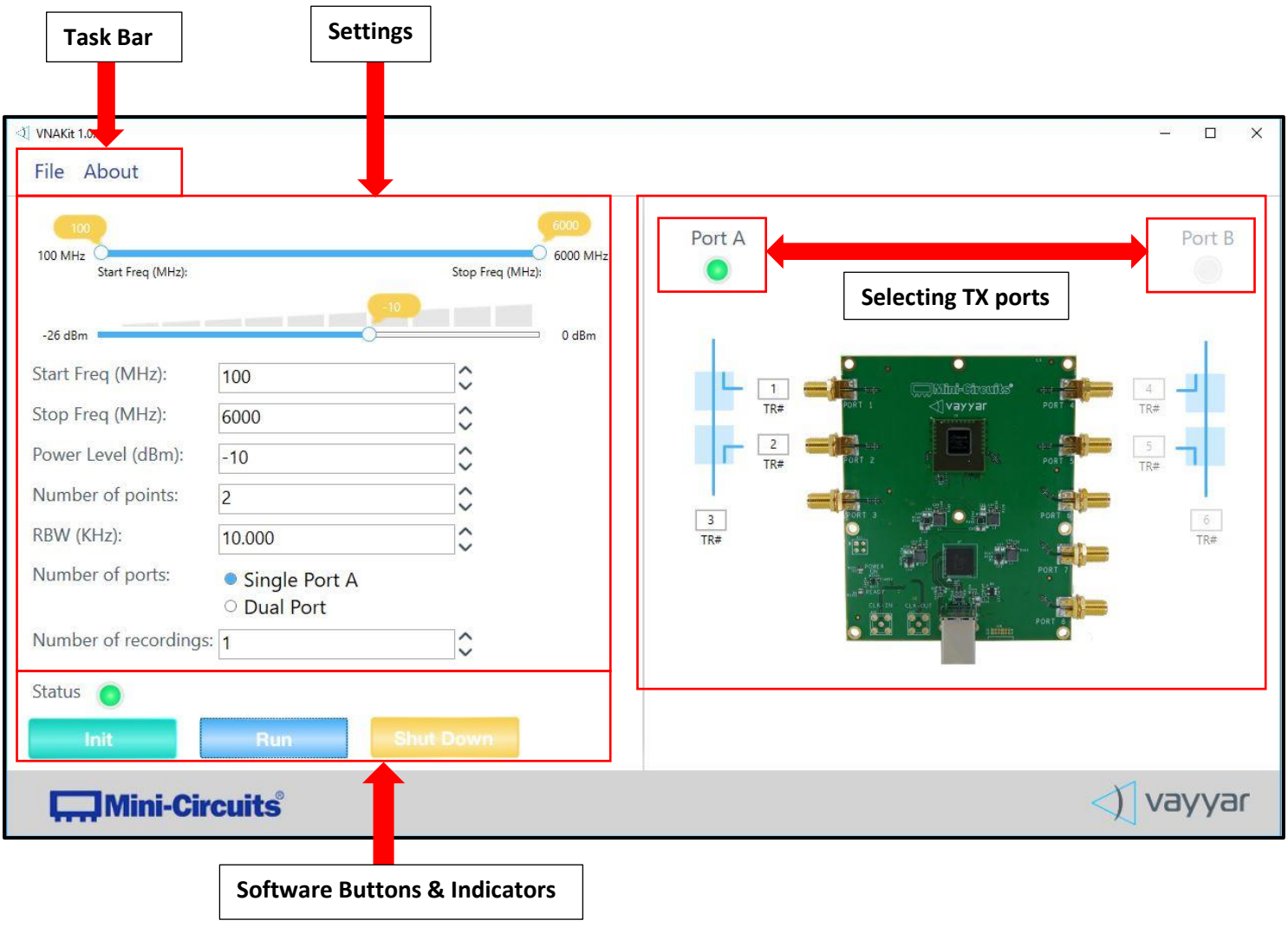
Introduction

The Mini-Circuits UVNA-63 is supplied with easy-to-install, user-friendly GUI software to control the transceiver and manage your measurements with the kit. If you prefer a more hands-on experience, a full set of .py and .dll files are provided to program your own S-parameter algorithms and other functions in either Python® or MATLAB®

This software is available for free download at: https://www.minicircuits.com/WebStore/uvna_63.html

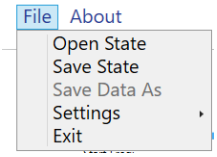
Note: While running the .exe file for installer. Right-click on the installer file and "Run as administrator".

Graphical User Interface (GUI)



Task Bar

File:

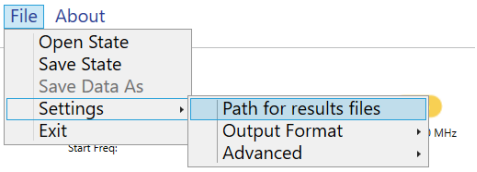


- **Saving the current settings:** File → Save State

- **Recalling previously saved settings:** File → Open State

- **Setting path for result files:** File → Settings → Path for results files → "Select a path/folder"

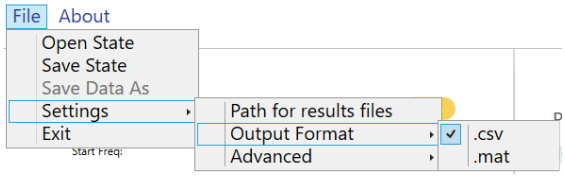
Set a path to save the raw recorded data from the Vayyar board (Re-presented as I and Q components of a phasor).



Default path is "C:\Temp" (if path is not set manually).

- **Setting the output format:** File → Settings → Output Format → .csv / .mat

Set the format in which you want your raw data to be saved. Data can be saved in .csv and .mat files.



Default output format is ".csv"

Advanced:

- **Enable logger –** For internal De-bugging only

About:

General Information about version of software and also on Mini-circuits and Vayyar.

Settings Interface

- **Frequency Range:** Set the frequency range by the moving the yellow sliders. You can also type the frequency in MHz in the boxes below. Range is 500-6000MHz
- **Power level:** Set the output power of the transmitters by moving the slider from Min to Max. It can also be specified in steps of 2dB from -26dBm to 0dBm
- **Number of points:** Set the number of measurement points by specifying an integer between 1 and 1001. Max number of points is limited by RBW setting. (See Table 1 below)
- **RBW:** Changing the Resolution Bandwidth (RBW) helps increase/decrease the "accuracy vs speed" of your measurement. A smaller RBW will result on a longer but more accurate measurement. Range is 1Hz-140kHz. Minimum RBW setting is limited by number of points setting. (See Table 1)
- **Number of ports:** Decide whether you want to make measurements at one port or at two ports.
  - If "Single Port A" is selected, Port 3 columns for I and Q components in the generated .csv file will be zero, which represents that Port 3 is transmitting and Port 1 and Port 2 are receiving. All other ports will also be zero as they are inactive.
  - If "Dual Port" is selected, the columns for port which is transmitting will be a zero for I and Q components in the generated .csv file, which represents that this particular port is transmitting and all other ports are receiving.
- **Number of recordings:** Select the number of recordings that you want to take from 1 to 100.

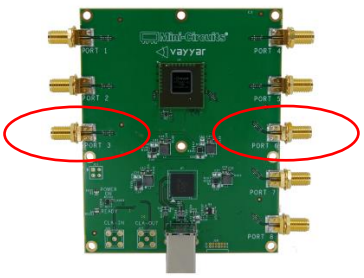
Min # Points	Max # Points	RBW Min (kHz)
2	51	0.001
52	100	0.02
101	400	0.08
401	800	0.17
801	1001	2

Table 1: Number of Points and RBW Allowable Settings

Settings the TX ports

The port selection decides whether the signal will be transmitted from port A or B

The **Red** circled ports Port 3 and Port 6 are the designated transmitters for Network Analyzer Port A and Port B respectively. To change the transmitter port to an arbitrary port, please refer to the Python/Matlab API in Software and Programming Guide



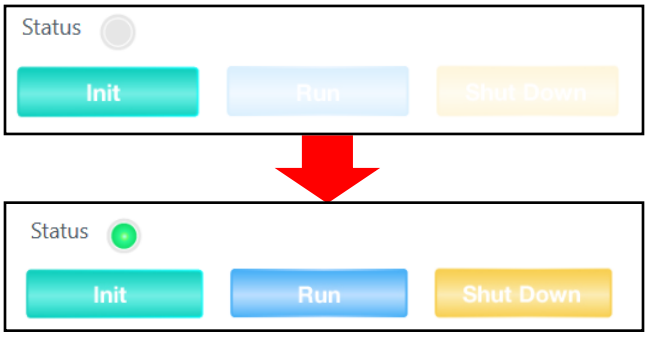
Software Buttons & Indicators

Init: Once you have filled out all the above fields, make sure that the VNA Kit is connected to your computer. Then tap “Init” to initialize the board with the settings.

Run: Records data from the respective initialized ports and saves them to the specified path (Refer “setting the path for result files”). The measurement phasor data taken using the GUI are represented in terms of I &Q components at each port. “Run” is greyed out till the board is properly initialized without any error.

Shut Down: Shuts down the board. “Shut Down” is greyed out till the board is properly initialized without any error.

Status: After tapping “Init”, if everything is O.K., then the status light will turn **Green** . “Run” and “Shut Down” buttons will now become active.



Otherwise an error message in **Red** will appear on the right-hand side below the Vayyar board.

Example Error:

Python (vnakit.py)

To use the python modules to communicate with the transceiver board, first you must run the following script with your python installation: %ProgramFiles%/Vayyar/VNAKit/python/install_vnakit.py

API is compatible with python versions: 2.7, 3.5, 3.6, & 3.7

Once you've run that, it should install two python packages: “vnakit” (to use API commands) and “vnakit_ex “(to run and use Demos provided). Check if both these packages were installed in your python package list.

Now you can run the Demos which are located at “%ProgramFiles%/Vayyar/VNAKit/python/demos”.

To connect to the UVNA-63 transceiver board, append the following code to the beginning of your own python script:

```
import vnakit
vnakit.Init()
```

Then you are connected and ready to use any python API functions in the vnakit module.

The VnaKit API includes all of the functions necessary for controlling the board settings, recording, and bringing data into the python environment. For documentation of the post-processing functions please refer to the “Python Documentation” on the UVNA-63 landing page.

This documentation includes two python directories as follow:
hidden: This includes hidden-source helper functions for UVNA-63
utils: This includes helper functions to accompany the Vayyar VNAkit API.

The source code is hidden to the users but it describes the function, input and output parameters to assist users to code these functions themselves.

For further assistance refer to our “Applications Notes” on the UVNA-63 landing page.

Classes from vnakit module

FrequencyRange(f_start,f_stop,num_f_pts)

Frequency range Object. Attributes are:

- freqStartMHz
- freqStopMHz
- numFreqPoints

FrequencyLimits

Frequency Range Limits Object. Attributes are:

- min_MHz
- max_MHz

- step_MHz
- nPointsMin
- nPointsMax

PowerLimits

Power Limits Object. Attributes are:

- min_dbm
- max_dbm
- step_dbm

RbwLimits

Resolution Bandwidth Limits Object. Attributes are:

- min_KHz
- max_KHz
- step_KHz

RecordingSettings(VNAKit.FrequencyRange freq_range, double rbw, double pwr, int tx_num, int mode)

Recording Settings Object. Attributes are:

- freqRange
- rbw_khz
- outputPower_dbm
- txTr
- mode

VNAKitError

VNAKit specific exception object

VNAKitSettingsError

Exception thrown by ValidateSettings if settings are invalid

Functions for vnakit module

ApplySettings(vnakit.RecordingSettings settings)

Apply stimulus settings.

float[] GetFreqVector_MHz()

Gets the actual measurement frequencies for the requested frequency range

{int:complex[]} GetRecordingResult()

Returns recording results vector

Init()

Board Initialization. Must run this function before ApplySettings.

Record()

Execute single recording of the signals

WriteRecording(str output_dir, enum output_format)

Write captured results to file (csv or mat)

EnterStandbyMode()

Enters a low-power state until the next call to Record() or ApplySettings(). Can only call function after call to Record() or ApplySettings().

float GetActualRBW_KHz()

Returns the actual RBW setting

vnakit.FrequencyLimits GetFrequencyLimits()

Returns limits to the Frequency Settings

float GetMinRbwKhz_ForNPoints(int nFreqPoints)

Returns limits to the RBW for the given Number of Points

float GetNMaxPoints_ForRbw(float rbw_KHz)

Returns limits to the Number of Points for the given RBW

VNAKit.PowerLimits GetPowerLimits()

Returns limits to the Power Settings

VNAKit.RbwLimits GetRbwLimits()

Returns absolute limits to the RBW settings

ValidateSettings(vnakit.RecordingSettings settings)

Throws an exception if settings are invalid

string FormatError(int error_code)

Convert a win32 error code into a string. If the error code is not given, the return value of a call to GetLastError() is used.

MATLAB (VNAKit.CSharp.dll)

To use the Matlab API you must load the dll and import the .Net Assembly, set the config file, and initialize the board. This is accomplished by running the following lines of Matlab or appending the same code to the beginning of the script:

```
modulePath = 'C:/Program Files/Vayyar/VNAKit/bin/VNAKit.CSharp.dll';
NET.addAssembly(modulePath);
import VNAKit.*;
VNAKit.VNAKit.SetConfigFile('C:/Program Files/Vayyar/VNAKit/bin/.config');
VNAKit.VNAKit.Init();
```

If the board is properly initialized, then all of the functions that control the board are called as member functions of the VNAKit.VNAKit class.
The VnaKit API includes all of the functions necessary for controlling the board settings, recording, and bringing data into the Matlab environment. For documentation of the post-processing functions please refer to the "MatLAB Documentation" on the UVNA-63 landing page.

This documentation includes two MatLAB directories as follow:
hidden: This includes hidden-source helper functions for UVNA-63
utils: This includes helper functions to accompany the Vayyar VNAkit API.

The source code is hidden to the users but it describes the function, input and output parameters to assist users to code these functions themselves.

For further assistance refer to our "Applications Notes" on the UVNA-63 landing page.

Classes, Structures, and Enums in VNAKit .NET object

VNAKit

VNAKit Class containing all MATLAB API functions

FrequencyRange(f_start,f_stop,num_f_pts)

Frequency range Object. Attributes are:

- freqStartMHz
- freqStopMHz
- numFreqPoints

FrequencyLimits

Frequency Range Limits Object. Attributes are:

- min_MHz
- max_MHz

- step_MHz
- nPointsMin
- nPointsMax

PowerLimits

Power Limits Object. Attributes are:

- min_dbm
- max_dbm
- step_dbm

RbwLimits

Resolution Bandwidth Limits Object. Attributes are:

- min_KHz
- max_KHz
- step_KHz

RecordingSettings(VNAKit.FrequencyRange freq_range,rbw,pwr,tx_num,VNAKit.Mode mode)

Recording Settings Object. Attributes are:

- freqRange
- rbw_khz
- outputPower_dbm
- txTr
- mode

Complex

Complex Number Object. Attributes are:

- real
- imag

VNAKIT_RESULT

VNAKit ErrorCode Enum

- VNAKIT_SUCCESS
- VNAKIT_USERERR__NOT_INITIALIZED
- VNAKIT_USERERR__NO_ACTIVE_APPLICATION
- VNAKIT_USERERR__NO_SETTINGS_APPLIED
- VNAKIT_USERERR__NO_RECORDING
- VNAKIT_INPUTERR__OUT_OF_RANGE

- VNAKIT_INPUTERR__INVALID_SETTINGS
- VNAKIT_INPUTERR__BAD_RESULT_SIZE
- VNAKIT_INSTRUMENT_NOT_FOUND
- VNAKIT_DEVICE_ERROR
- VNAKIT_INIT_FAILED
- VNAKIT_BAD_CONFIG
- VNAKIT_GENERAL_ERROR

VNAKIT_SETTINGS_STATUS

RecordingSettings ErrorCode Enum

- VNAKIT_SETTINGS__VALID
- VNAKIT_SETTINGS__FREQ_RANGE_MALFORMED
- VNAKIT_SETTINGS__FREQ_OUT_OF_RANGE
- VNAKIT_SETTINGS__FREQ_BAD_STEP
- VNAKIT_SETTINGS__NPOINTS_OUT_OF_RANGE
- VNAKIT_SETTINGS__RBW_OUT_OF_RANGE
- VNAKIT_SETTINGS__RBW_BAD_STEP
- VNAKIT_SETTINGS__POWER_OUT_OF_RANGE
- VNAKIT_SETTINGS__POWER_BAD_STEP
- VNAKIT_SETTINGS__NPOINTS_EXCEED_RBW

Mode

Enum for one or two port recording selection. Type is used to define mode in the MATLAB RecordingSettings object.

- For 1-port mode, use VNAKit.Mode.VNAKIT_MODE_ONE_PORT
- For 2-port mode, use VNAKit.Mode.VNAKIT_MODE_TWO_PORTS

WriteFormat

Enum for specifying output format when writing a recording. *.csv and *.mat files are supported. Members are:

- VNAKIT_MAT_FORMAT
- VNAKIT_CSV_FORMAT

Methods for class VNAKit.VNAKit

VNAKit.VNAKIT_RESULT ApplySettings(VNAKit.RecordingSettings settings)

Apply stimulus settings.

System.Double[] GetActualFreqVector()

Gets the actual measurement frequencies for the requested frequency range

VNAKit.Complex[,] GetRecordingResult()

Returns recording results vector

VNAKit.VNAKIT_RESULT Init()

Board Initialization. Must run this function before ApplySettings.

VNAKit.VNAKIT_RESULT Record()

Execute single recording of the signals

VNAKit.VNAKIT_RESULT WriteRecording(System.String outDir, VNAKit.WriteFormat format)

Write captured results to file (csv or mat)

VNAKit.VNAKIT_RESULT Set(double f_start, double f_stop, int32 num_f_pts, double rbw, double tx_pwr, int32 tx_port, VNAKit.Mode mode)

Sets VNA settings. Equivalent to defining a RecordingSettings and calling ApplySettings

VNAKit.VNAKIT_RESULT EnterStandbyMode()

Enters a low-power state until the next call to Record() or ApplySettings(). Can only call function after call to Record() or ApplySettings().

double GetActualRBW_KHz()

Returns the actual RBW setting

VNAKit.FrequencyLimits GetFrequencyLimits()

Returns limits to the Frequency Settings

double GetMinRbwKhz_ForNPoints(int32 nFreqPoints)

Returns limits to the RBW for the given Number of Points

double GetNMaxPoints_ForRbw(double rbw_KHz)

Returns limits to the Number of Points for the given RBW

VNAKit.PowerLimits GetPowerLimits()

Returns limits to the Power Settings

VNAKit.RbwLimits GetRbwLimits()

Returns absolute limits to the RBW settings

logical ReferenceEquals(System.Object objA, System.Object objB)

Internal Usage Only

VNAKit.VNAKIT_SETTINGS_STATUS ValidateSettings(VNAKit.RecordingSettings settings)

Returns an error code if settings are invalid

IMPORTANT NOTICE

This document is provided as an accommodation to Mini-Circuits customers in connection with Mini-Circuits parts only. In that regard, this document is for informational and guideline purposes only. Mini-Circuits assumes no responsibility for errors or omissions in this document or for any information contained herein.

Mini-Circuits may change this document or the Mini-Circuits parts referenced herein (collectively, the "Materials") from time to time, without notice. Mini-Circuits makes no commitment to update or correct any of the Materials, and Mini-Circuits shall have no responsibility whatsoever on account of any updates or corrections to the Materials or Mini-Circuits' failure to do so.

Mini-Circuits customers are solely responsible for the products, systems, and applications in which Mini-Circuits parts are incorporated or used. In that regard, customers are responsible for consulting with their own engineers and other appropriate professionals who are familiar with the specific products and systems into which Mini-Circuits' parts are to be incorporated or used so that the proper selection, installation/integration, use and safeguards are made. Accordingly, Mini-Circuits assumes no liability therefore.

In addition, your use of this document and the information contained herein is subject to Mini-Circuits' standard terms of use, which are available at Mini-Circuits' website at www.minicircuits.com/homepage/terms_of_use.html.

Mini-Circuits and the Mini-Circuits logo are registered trademarks of Scientific Components Corporation d/b/a Mini-Circuits. All other third-party trademarks are the property of their respective owners. A reference to any third-party trademark does not constitute or imply any endorsement, affiliation, sponsorship, or recommendation: (i) by Mini-Circuits of such third-party's products, services, processes, or other information; or (ii) by any such third-party of Mini-Circuits or its products, services, processes, or other information.