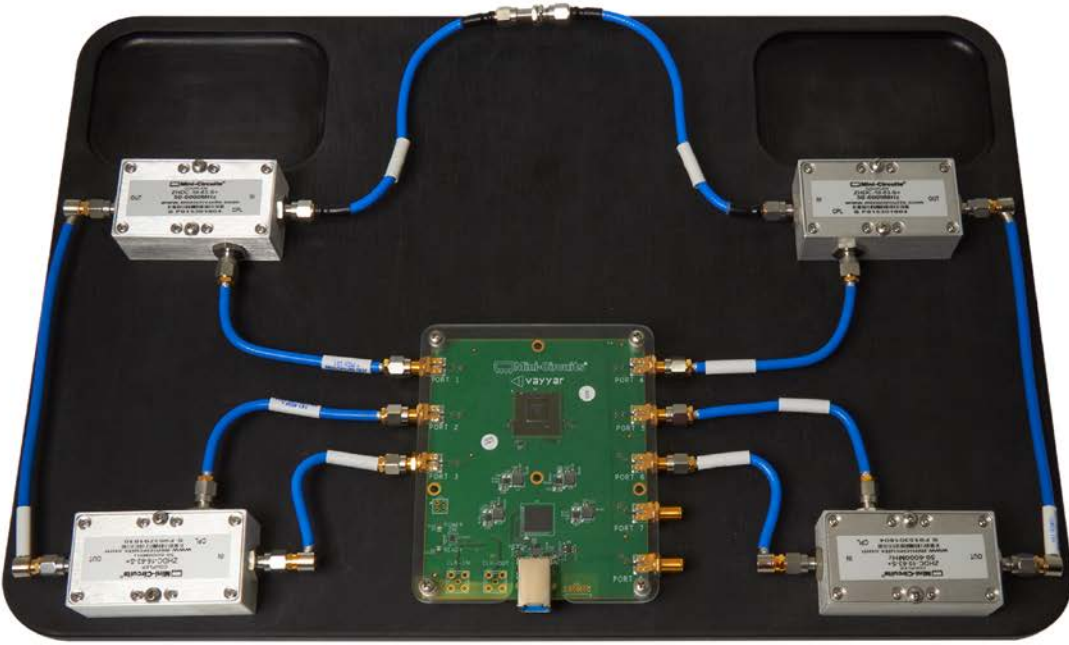


# MICROWAVE TRANSCEIVER KIT

## PROJECT NO. 1: UVNA-63 DIY VECTOR NETWORK ANALYZER



### Software and Programming Guide



## Important Notice

This guide is owned by Mini-Circuits and is protected by copyright, trademark and other intellectual property laws.

The information in this guide is provided by Mini-Circuits as an accommodation to our customers and may be used only to promote and accompany the purchase of Mini-Circuits' Parts. This guide may not be reproduced, modified, distributed, published, stored in an electronic database, or transmitted and the information contained herein may not be exploited in any form or by any means, electronic, mechanical recording or otherwise, without prior written permission from Mini-Circuits.

This guide is subject to change, qualifications, variations, adjustments or modifications without notice and may contain errors, omissions, inaccuracies, mistakes or deficiencies. Mini-Circuits assumes no responsibility for, and will have no liability on account of, any of the foregoing. Accordingly, this guide should be used as a guideline only.

## Trademarks

MATLAB is a registered trademark of The MathWorks, Inc. Python is a registered trademark of Python Software Foundation Corporation.

All other trademarks cited within this guide are the property of their respective owners. Neither Mini-Circuits nor the Mini-Circuits UVNA-63 are affiliated with or endorsed or sponsored by the owners of the above referenced trademarks.

Mini-Circuits and the Mini-Circuits logo are registered trademarks of Scientific Components Corporation.

## Mini-Circuits

13 Neptune Avenue

Brooklyn, NY 11235

Phone: 1-718-934-4500

Email: [apps@minicircuits.com](mailto:apps@minicircuits.com)

Web: [www.minicircuits.com](http://www.minicircuits.com)

# Contents

Introduction .....	4
GUI .....	4
Task Bar .....	5
Settings Interface .....	5
Settings the TX ports # .....	6
Software Buttons & Indicators.....	5
Python (vnaKit.py).....	8
Classes from vnaKit module .....	8
Functions for vnaKit module .....	9
MATLAB (VNAKit.CSharp.dll) .....	11
Classes, Structures, and Enums in VNAKit .NET object.....	11
Methods for class VNAKit.VNAKit.....	13

# 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](https://www.minicircuits.com/WebStore/uvna_63.html)

## GUI

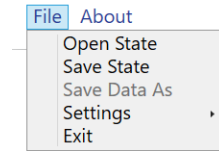
The screenshot shows the VNAKit 0.0.0 GUI software interface. Annotations with red arrows point to various components:

- Task Bar:** Points to the top-left corner of the application window.
- Settings:** Points to the settings panel on the left side of the window.
- Software Buttons & Indicators:** Points to the 'Init', 'Run', and 'Stop Execution' buttons at the bottom of the settings panel.

The GUI features a menu bar with 'File' and 'About' options. The settings panel includes a 'Remote Control' checkbox, frequency sliders (100 MHz to 6000 MHz), a power level slider (-26 dBm to 0 dBm), and input fields for Start Freq (MHz), Stop Freq (MHz), Power Level (dBm), Number of points, RBW (KHz), Number of ports (Single Port A or Dual Port), and Number of recordings. The main display area shows a central image of the UVNA-63 hardware with six ports labeled 1 through 6. A 'Selecting Tx' box is overlaid on the hardware image, with a red double-headed arrow indicating the selection process between Port A and Port B.

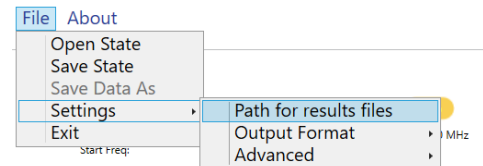
# Task Bar

## File:

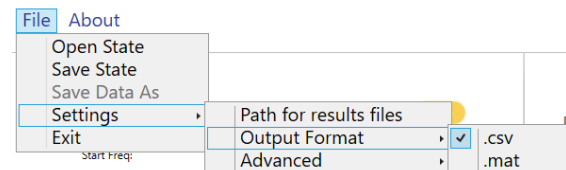


- **Saving the settings:** File → Save State
- **Recalling previously saved settings:** File → Open State
- **Setting path for result files:** File → Settings → Path for results files → “Select a path/folder”

Set a path to save the raw recorded data from the Vayyar board



- **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



## Advanced:

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

## About:

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

## Settings Interface

- **Remote Control:** Check this box if you want to control the Vayyar board with some other software (or code in Matlab or Python while the GUI is open). If you do not check this box when using some other software, you will not be able to communicate with the board.
- **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)
- **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 you set it up only for one port measurements, use the transmitters and receivers at port A only.
- **Number of recordings:** Select the number of recordings that you want to take from 1 to 100.

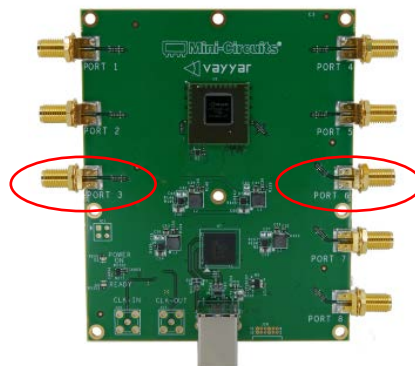
Min #Points	Max #Points	RBW Min (kHz)
2	51	0.001
52	100	0.02
101	400	0.08
401	800	0.17
801	1001	2

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



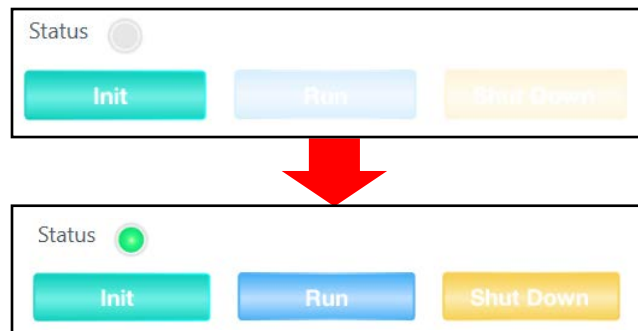
## Software Buttons & Indicators

**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”) “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**.



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

Example Error:

Init Error: VNAKIT\_INSTRUMENT\_NOT\_FOUND

# Python (vnaKit.py)

To use the python modules, first you must run the following script with your python installation:  
%ProgramFiles%/Vayyar/VNAKit/python/install\_vnaKit.py

Once you've run that once, all the scripts in %ProgramFiles%/Vayyar/VNAKit/python/demos should run correctly.

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

```
import vnaKit
vnaKit.Init()
```

Then you are connected and ready to use any python API functions in the vnaKit module. API is compatible with python versions: 2.7, 3.5, 3.6, & 3.7

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 (vnaKit\_ex) please refer to the vnaKit\_ex python manual 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 (vnaKit\_ex), please refer to the vnaKit\_ex Matlab manual 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:

© Mini-Circuits 2018. All rights reserved.

- 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

© 2017 Mini-Circuits

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](http://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.