User Guide: Universal Power Amplifier Test Controller

Prepared For: Collins Aerospace

Prepared By: Team M.A.D. Power

Alabi Oluwademilade, Avery Neff, Michael McGruther

Table of Contents

Installation

This software was created for and tested on a Raspberry Pi 2 Model B ver. 1.2 running Raspbian Buster Operating System. It runs using Python 3.7.3.

The following packages are required to run the test controller:

- pyvisa https://pyvisa.readthedocs.io/en/latest/
- pyvisa-py https://pyvisa-py.readthedocs.io/en/latest/
- PyQt5 https://pypi.org/project/PyQt5/

Currently, these packages can be installed with the following commands from the Raspberry Pi's Terminal App:

```
$ sudo apt-get update
$ sudo apt-get install pyvisa
$ sudo pip3 install pyvisa-py
$ sudo apt-get install python3-pyqt5
$ sudo apt-get install qt5-default pyqt5-dev pyqt5-dev-tools
```

Additional packages were used for unit-testing and simulation during development. To run these testing scripts, the following packages must also be installed:

- pyvisa-sim https://pyvisa-sim.readthedocs.io/en/latest/

These can be installed using:

```
$ sudo pip3 install pyvisa-sim
```

The source code is available at https://github.com/mmcgruther/CollinsTestController and can be cloned to a device with the following command:

```
$ git clone https://github.com/mmcgruther/CollinsTestController.git
```

Equipment Setup

Use the following steps to set up automated testing hardware:

1. Connect the Raspberry Pi's to a LAN.
2. Connect electronic test instruments to the LAN.
3. Configure each device to a unique static IP address within a common subnet.
4. Determine instrument identification strings. Some manufacturers display this string in the Utility or Setup screens. If not, this information can be obtained while running the software.

Equipment Configuration

Before executing tests on the test controller software, the equipment configuration JSON files must be updated for connected devices. By default, this file is equipment.json.

Each connected instrument needs a root element with the following structure:

Element Description	JSON File
Equipment name	"Rigol Spectrum Analyzer": {
Equipment address	"address": "TCPIP0::192.168.1.21::INSTR",
Equipment IDN	"idn": "Rigol Technologies,DG4162,DG4C151300243,00.01.04",
Equipment timeout	"timeout": 1000,
IDN command	"idn_cmd": "*IDN?",
Function name	"set_peak_table_state": {
Function SCPI string	"cmd": ":TRACe:MATH:PEAK:TABLe:STATe {0}",
Function mode	"type": "w",
	},
Function name	"set_peak_sort": {
Function SCPI string	"cmd": ":TRACe:MATH:PEAK:SORT {0}",
Function mode	"type": "w",
	},
Function name	"get_trace": {
Function SCPI string	"cmd": ":TRACe? TRACE1",
Function mode (w/q)	"type": "q",
	}
	},

Equipment name: Any unique name for equipment.
Equipment address: The address pyvisa will use to communicate with a device. This address specifies interface, address, and driver used. Theoretically, a USB or GPIB address could be specified. In our testing, we only used TCPIP interface and the INSTR driver.
Equipment IDN: The identification string returned by a device when given the standard IDN query. If unknown, a dummy string can be entered. Then, run the software and attempt to connect to this device. Connection will fail without the correct IDN to match, but an error printed in the command console will indicate the actual IDN string that was received by the query.

Function name: A new element for each unique SCPI command used with the instrument. The given name can be any descriptive string. Each new element must have members as shown.
Function SCPI string: The constant component of the SCPI command string with shown formatting syntax for dynamic command parameters.
Function mode: Whether the command is a write command (“w”) or query request (“q”)

Once the equipment is correctly configured, the test controller software should successfully detect and connect to the configured instruments.

Test Configuration

Similarly to equipment, tests are also configured by JSON file prior to running the test controller. By default, this file is tests.json.

Each test needs a root element with the following structure:

Element Description	JSON File
Test name	"Simple FuncGen/Oscope": {
Test equipment	"Function Generator": {
Equipment “config” phase	"config": {
Command display name	"Disable Output": {
Function name	"name": "set_output",
Function arguments	"args": [
List of arguments	"OFF"
	]
	},
Command display name	"Set Volts": {
Function name	"name": "set_volts",
Function arguments	"args": [
List of arguments	"5"
	]
	}
Equipment “run” phase	"run": {},
Equipment “reset” phase	"reset": {
Command display name	"Reset Device": {
Function name	"name": "reset_device",
Function arguments	"args": []
	}
	}
	},
Test equipment	"Oscilloscope": {

Test name: Displayed name for the test. Used for test selection in the program.
Test equipment: Equipment name as configured in equipment JSON file.
Equipment phases: Controller uses three phases for test execution: config, run, and reset. All three phase elements must be included for each equipment, but can be left empty as shown.
Command display name: Displayed name for a command instance.
Function name: Equipment function name as entered in equipment's configuration JSON object.
Function arguments: List of string elements inserted into SCPI commands where formatted.

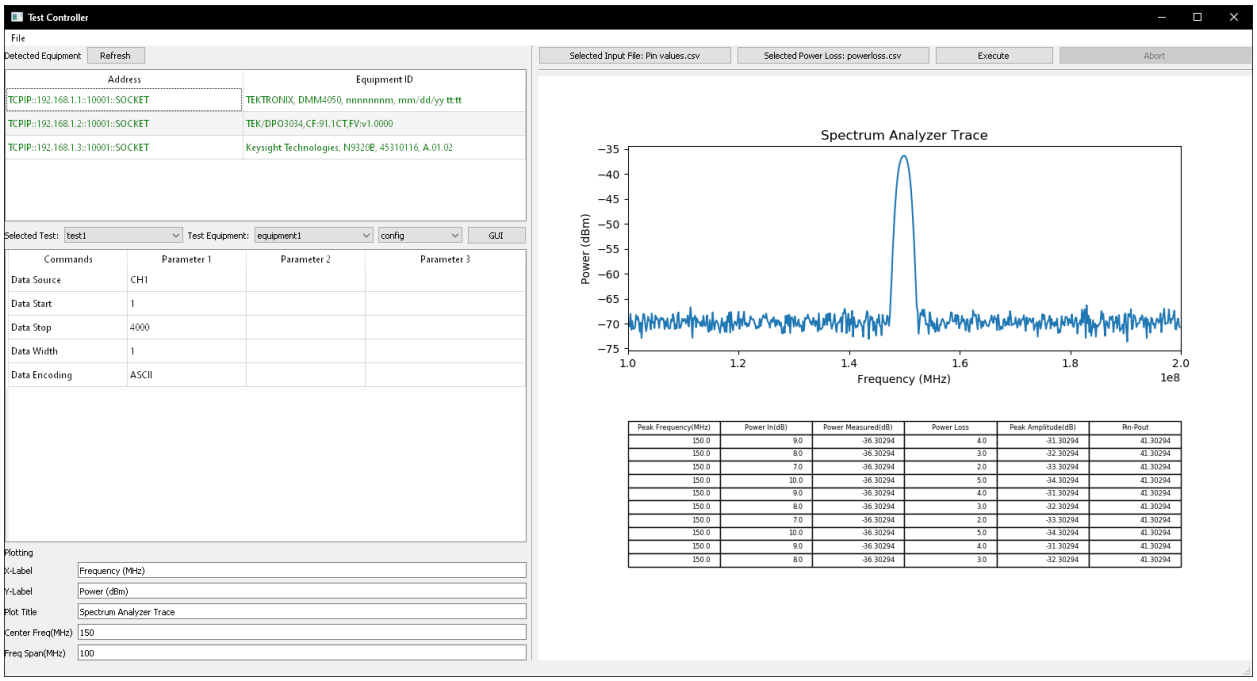
Test configuration in the test controller program is also available. Use the provided toolbar commands to test new tests, test equipment, and commands. Arguments can be modified in the displayed table. Changes made in-program are currently not saved on application exit.

Running the Test Controller

Once installed, the application can be run with the following command in the source code's directory:

$ sudo python3 main.py

The test controller opens with the following window:



To execute a configured test, use the following steps:

1. Click "Refresh" above the Detected Equipment window. Properly configured equipment should be connected, and their address lines become green. If red, no configured device was detected. Refer to terminal outputs for connection error information.
2. Choose the desired "Selected Test". The test's "Test Equipment" and phase can be viewed by selection in drop-down boxes. Command parameters can be changed by selecting the table elements beside a command.
3. Enter plotting labels and axis ranges in lines beneath "Plotting".
4. Select Power Input and Power Loss .csv files using buttons along the top of the right column. See demo files for formatting.

5. Click “Execute” to begin the automated test. Real-time data will be displayed in the plots on the right half of the screen. Initial errors preventing test execution are displayed in the status bar at the bottom of the screen.
6. Final output data is logged to file power_out.csv

Verbose program status updates are printed to the terminal throughout program operation. These can be used to troubleshoot unexpected behavior while connecting to devices or performing automated testing.