

LabLANreadme

Final Year Project, 2019
by Anthony Baudinette & Jarred Paola

Setup Summary	2
Client Setup	2
Server Setup	2
Definitions	2
External Dependencies and Required Pre-Installations	3
Server Dependencies	3
Client Dependencies	3
Network Settings and Static Routing (begin trouble-shooting here)	4
Server Static IP Settings	5
Client Static IP Settings	6
Switch IP Settings and Forwarding	6
LAN-side Address Configuration	6
WAN-side Address Configuration	7
Port-forwarding Settings	7
Forwarding VISA Devices	7
'config.txt' Format	8
Client-side 'config.txt'	8
Multi-server and 'changeConfigs.py' Usage	8
Server-side 'config.txt'	9
Client Direct Scripts Functions	9
Low Level Python Functions	10
MATLAB-Python Interface	10
High Level MATLAB Functions	11
Client MATLAB GUI	12
Setup via Installer	12
Modifying Source Code via MATLAB Built-in App Designer	12
TCP Keystings	13

Setup Summary

Client Setup

1. Copy simpleUploadAndCapure into desired folder, or run MATLAB GUI installer.
2. Ensure dependencies are installed; MATLAB 2019 and Python 3.7.
3. Set static IP address as shown in network settings section of this document.
4. Set config.txt with the IP address and port number to match that of the server used.

Server Setup

1. Copy 'server' folder into the desired location on the to-be server.
2. Install dependencies (see instructions on page 2).
3. Setup static IP addresses (as shown in network settings section of this document).
4. If using switch between client and server configure the port forwarding on the router.
5. **Allow python.exe through server firewall**, the location of python.exe changes depending on the options chosen during setup, it is usually in 'users/appdata/local/' or 'Program Files'.
6. Set config.txt, the IP and port should match and point the client to the server. The 'first hop' address should be used for the IP, e.g. if the switch is used, it will be the wan side switch IP(192.168.0.1 in network map on page 4), otherwise if the client and server are directly connected the local IP of the server is used. Also ensure the File_Path setting is an existing folder otherwise uploading files will cause an error.
7. Edit runAll.bat with the location of the soft front panel exe.
8. Create & name a shortcut to runAll.bat

Definitions

TCP: "Transmission Control Protocol" used by the server and client to transmit the command strings in plain text format.

Static IP: In most networks an end device e.g. laptop will automatically ask the router it is connected to for an address, robust networks for server hosting such as in labLAN override this automation and set static/unchanging IP address.

Switch / Router: Device which manages and directs the traffic. The traffic flow is done by port forwarding and is explained in the network settings section.

Port forwarding: since all clients on the LAN (server/awg/oscilloscope side) appear as the same IP on the WAN (client side) due to NAT on the switch, different tools on the LAN are contacted on different TCP ports. The port forwarding section of the switch settings defines how messages are transmitted to their intended recipient.

VISA: “Virtual Instrument Software Architecture” simplifies the various serial, ethernet, and on-board bus communication protocols into a uniform interface. VISA software is installed on the server and performs actions as tasked by the client.

External Dependencies and Required Pre-Installations

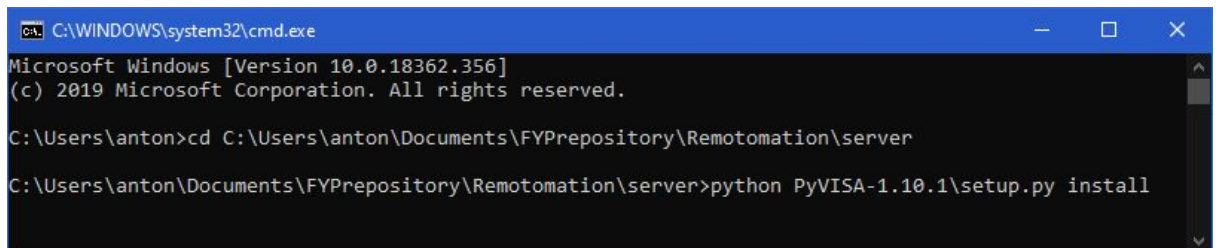
Server Dependencies

- pyVisa/VISA

This library provides the tools to consolidate the instrument interfaces, the two branches of the server code rely on different libraries the main recommended branch uses python 3 and pyvisa 1.9.1 however for compatibility with existing lab systems server_py2branch/ uses python 2 and pyvisa 1.5 (note that list devices functionality is not operational through the server_py2 branch).

Install instructions:

For latest update of pyvisa use pip install or for offline install the latest version at release of labLAN is included in server folder through command line using the instructions as shown below (note: installation requires python installed first)



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.18362.356]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\anton>cd C:\Users\anton\Documents\FYPrepository\Remotomation\server
C:\Users\anton\Documents\FYPrepository\Remotomation\server>python PyVISA-1.10.1\setup.py install
```

- Python

As with above the python dependency varies with branch the main branch runs on python 3.7.

The latest version of python is available from <https://www.python.org/downloads/>

- SFP

The keysight AWG soft front panel GUI is required to be running and in the connected state to communicate with the AWG. note that communication to other devices does not require soft front panel to run however in this case the program should be started through __run__.py instead of run.bat.

Client dependencies

- MATLAB

The low level processes are independent of MATLAB, and therefore the low level functions can be utilised without MATLAB installed but for ease of use the high level

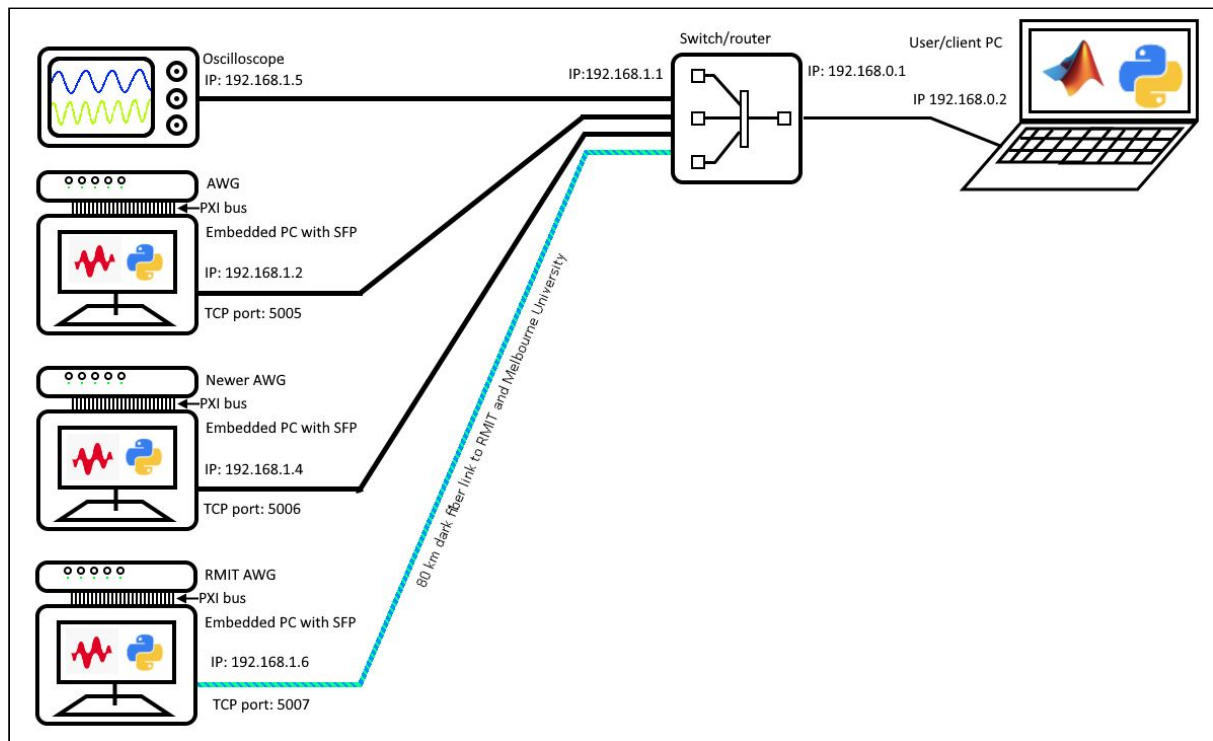
functions are run in MATLAB.

- Python

The fundamental network interface of the program is formed by inbuilt libraries and has been written to be compatible with all versions of python 2 and python 3 with no extra libraries.

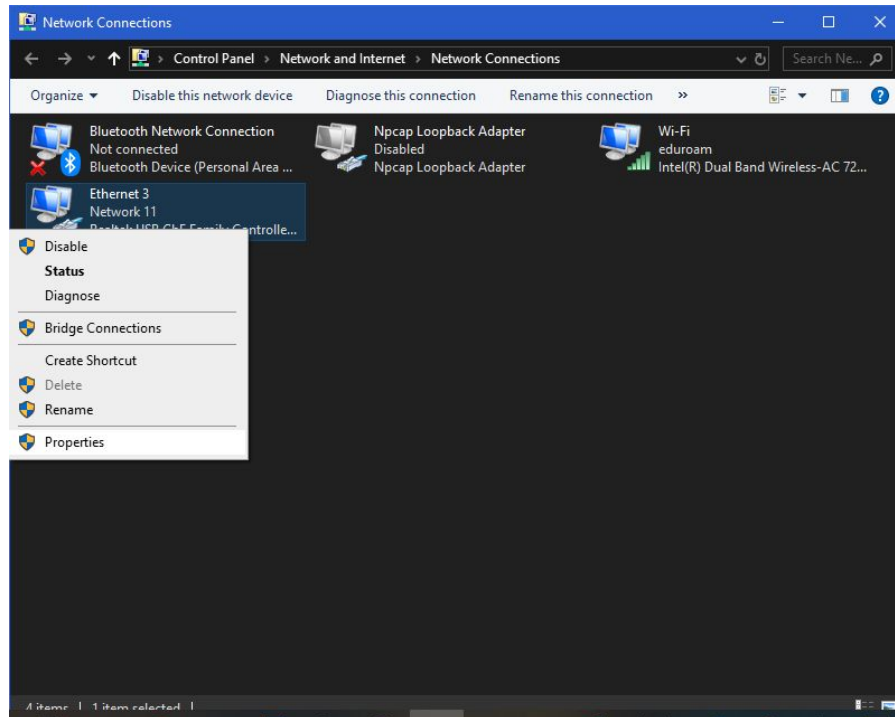
The latest version of python is available from <https://www.python.org/downloads/>

Network Settings and Static Routing (begin trouble-shooting here)



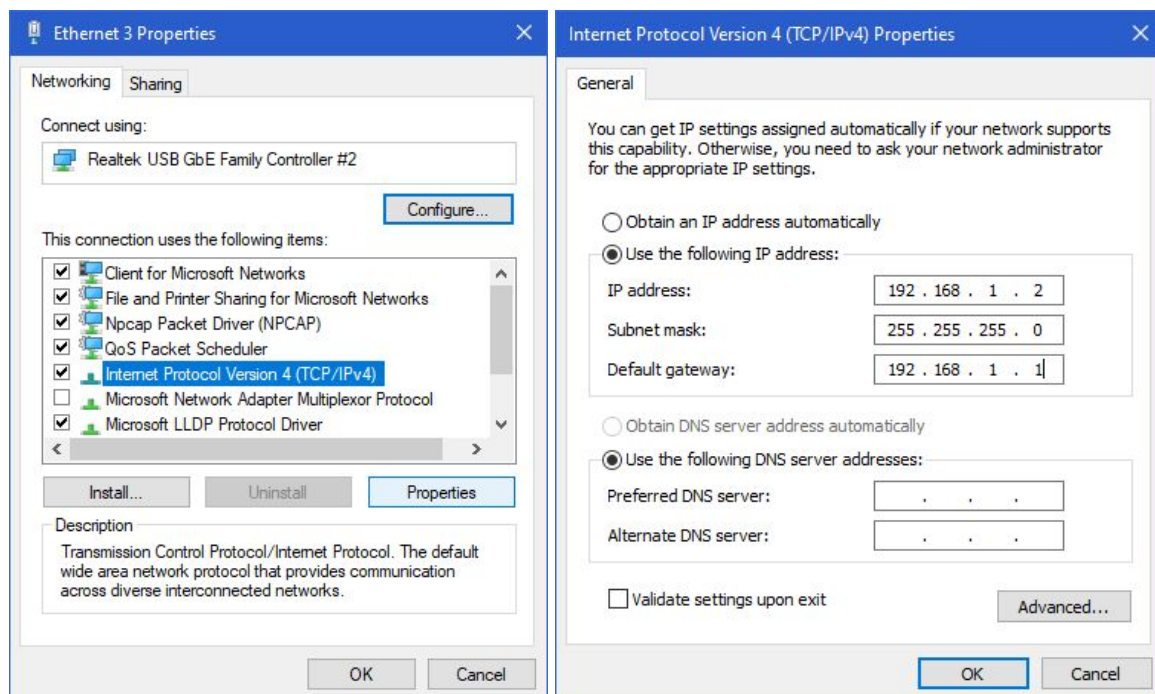
Server Static IP Settings

To set a static IP address, access the network adapters by run > ncpa.cpl or through the control panel to display the window shown below.



From here, select properties and then IPv4 properties to set the local IP address with the default gateway as the address of the switch. Shown here is the setting of the AWG address.

If the client is connected to the server without the switch sitting between, the default gateway would be the client address.



Client Static IP Settings

Follow the same sequence of run > ncpa.cpl > ethernet properties > IPv4 > properties, as shown in the previous section. Once again, set IP address to the local address and default gateway to the switch address (192.168.0.1). The window below shows the client set to be connected to the newer AWG (from network map above) without the switch between them.

Internet Protocol Version 4 (TCP/IPv4) Properties

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

☐ Obtain an IP address automatically

☒ Use the following IP address:

IP address: 192 . 168 . 1 . 1

Subnet mask: 255 . 255 . 255 . 0

Default gateway: 192 . 168 . 1 . 4

☐ Obtain DNS server address automatically

☒ Use the following DNS server addresses:

Preferred DNS server:

Alternate DNS server:

☐ Validate settings upon exit

Advanced...

OK Cancel

Note: setting the ethernet driver to a static IP will prevent it from connecting to another network e.g. UNI internet through the same adapter. Windows troubleshoot wizard will reset this setting to automatic, so this will need to be re-configured before use again.

Switch IP Settings and Forwarding

As the method of applying these settings differs between router manufacturers, this section will focus on the settings applied rather than the method of using the router.

The configuration panel of the router can be accessed by typing the default gateway into a browser window.

LAN-side Address Configuration

Network Setup	
Router IP	Local IP Address: 192 . 168 . 1 . 1 Subnet Mask: 255 . 255 . 255 . 0
Network Address Server Settings (DHCP)	DHCP Server: <input checked="" type="radio"/> Enable <input type="radio"/> Disable Starting IP Address: 192.168.1.100

The DHCP controls the automatic IP setting that the previous section bypasses, the importance of this setting is just to prevent interference with the static IPs manually set.

WAN-side Address Configuration

Internet Setup	
Internet Connection Type	Static IP
Internet IP Address:	192 . 168 . 0 . 1
Subnet Mask:	255 . 255 . 255 . 0
Gateway:	192 . 168 . 0 . 2

This setting is the opposite of the client static setting (i.e. swapped IP and gateway addresses).

Port-forwarding Settings

Firstly one rule needs to be set for each server needs to have its port (as set by server/config.txt) forwarded to its static IP.

External Port	Internal Port	Protocol	To IP Address	Enable
---	---	---	192.168.1.0	<input type="checkbox"/>
---	---	---	192.168.1.0	<input type="checkbox"/>
---	---	---	192.168.1.0	<input type="checkbox"/>
---	---	---	192.168.1.0	<input type="checkbox"/>
---	---	---	192.168.1.0	<input type="checkbox"/>
5005	5005	Both ▼	192.168.1.2	<input checked="" type="checkbox"/>
5006	5006	Both ▼	192.168.1.4	<input checked="" type="checkbox"/>

Forwarding VISA Devices

While a Raspberry Pi virtual solution is recommended for the future, the immediate solution to forwarding VISA devices through the switch is to forward the range of ports that are not used in servers to the VISA device's IP. The main limitation of this solution is that it is only able to forward one VISA device onto the WAN connection at a time.

This is achieved either by setting the device as DMZ host;

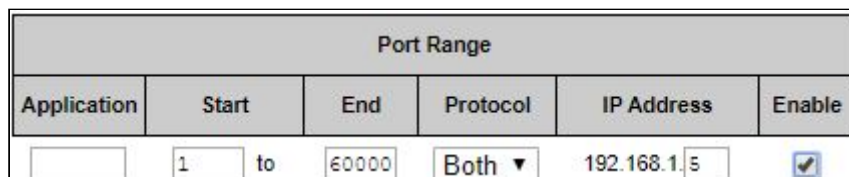


DMZ

☒ Enable ☐ Disable

DMZ Host IP Address : 192.168.1.5

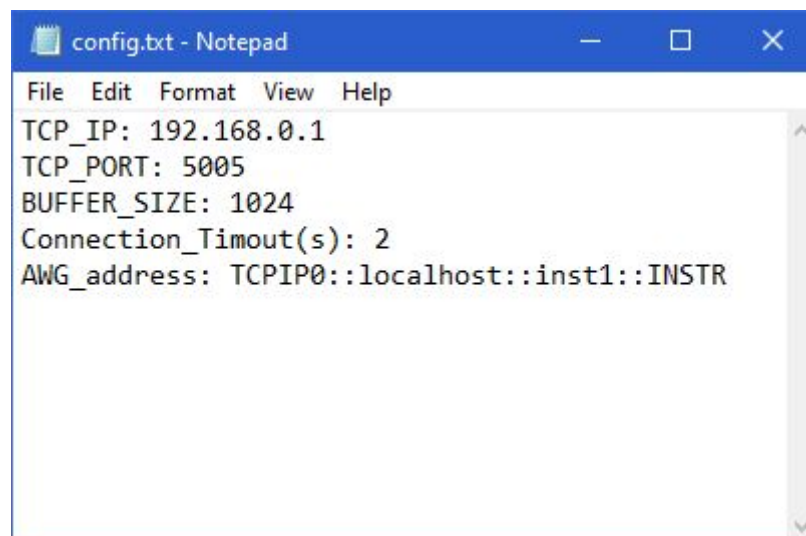
or forwarding the device IP address to the entire port range;



Port Range					
Application	Start	End	Protocol	IP Address	Enable
	1	to 60000	Both ▼	192.168.1.5	<input checked="" type="checkbox"/>

‘config.txt’ Format

Client-side ‘config.txt’



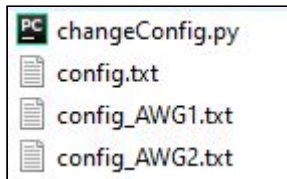
```
config.txt - Notepad
File Edit Format View Help
TCP_IP: 192.168.0.1
TCP_PORT: 5005
BUFFER_SIZE: 1024
Connection_Timeout(s): 2
AWG_address: TCPIP0::localhost::inst1::INSTR
```

The IP address and port number set in client/config.txt point the Python functions to the server. Here the IP points to the switch's address, then the switch sends the connection onto the desired server. BUFFER_SIZE and ConnectionTimeout(s) are used by the TCP library but shouldn't need changing. AWG_address specifies the VISA address to be used when the python functions are called with the argument of 'AWG' in place of VISA address.

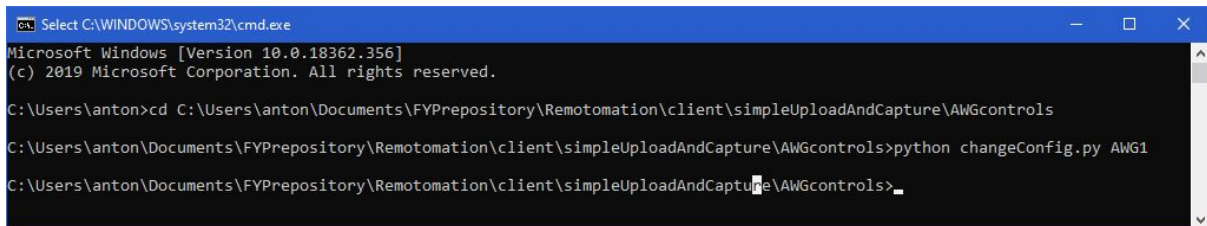
Note: the config.txt files on both the client and server side should each point toward the same IP address (the server/AWG) when connecting the server and client directly.

Multi-Server and 'changeConfigs.py' Usage

To support multiple servers the client can change between servers by changing config.txt. As shown below there is a config_*****.txt for each server.



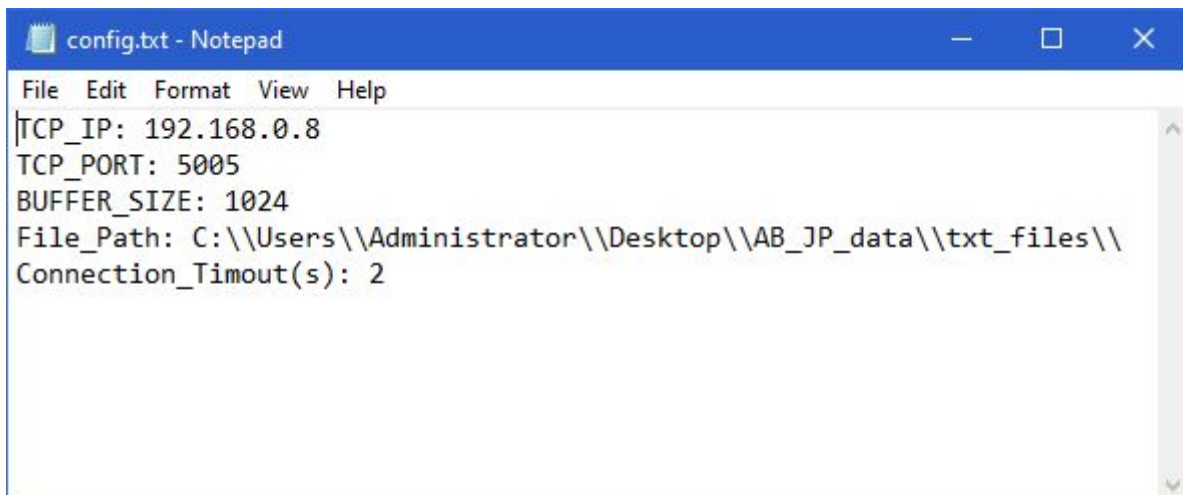
To change between servers, changeConfig.py overwrites the existing config.txt with the contents of other .txt files. The window below shows config.txt being overwritten with config_AWG1.txt using the command *python changeConfig.py AWG1*.

A Windows command prompt window titled "Select C:\WINDOWS\system32\cmd.exe". The text inside shows the following commands and output:

```
Microsoft Windows [Version 10.0.18362.356]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\anton>cd C:\Users\anton\Documents\FYPrepository\Remotomation\client\simpleUploadAndCapture\AWGcontrols
C:\Users\anton\Documents\FYPrepository\Remotomation\client\simpleUploadAndCapture\AWGcontrols>python changeConfig.py AWG1
C:\Users\anton\Documents\FYPrepository\Remotomation\client\simpleUploadAndCapture\AWGcontrols>
```

Server-side 'config.txt'



The IP address and port number specifies where the server listens to wait for a connection, and therefore must be the same as the server's static IP setting. Removing or commenting TCP_IP will default the IP address to that of the PC, which may cause issues if multiple network adapters are enabled, as client BUFFER_SIZE and Connection_Timout(s) are used by the TCP library. File_Path must point to an existing folder and is used for storing files to be uploaded to the AWG, the "\\" characters are replaced with "\\\" due to the decoding of strings in the python code, e.g. "\\n" becomes a newline character, and "\\\" becomes \.

Client Direct Scripts Functions

Low Level Python Functions

The connection between the server and client is run on a set of Python scripts.

These scripts can either be imported as Python functions;

```
1 import write
2 import query
3
4 write.write("AWG", ":INIT:IMM")
```

Or run manually from a CMD window.

(This formatting is also used in the MATLAB-Python interface, shown below)

```
C:\Users\anton\Documents\FYPPrepository\Remotomation\client\simpleUploadAndCapture\AWGcontrols>python write.py AWG :INIT:IMM
```

If the client receives a bad response or the server indicates an error, the Python functions return a string "err:" followed by a description of the error.

MATLAB-Python Interface

MATLAB and Python communicate by passing strings through the `system()` function. In the example shown below, "AWG" and ":INIT:IMM" are Python input arguments, and the server will respond with a success message by returning `status=0` and `cmdOut=""`.

Calling scripts from MATLAB with `system()`:

```
16
17 - [status,cmdOut] = system("python write.py AWG :INIT:IMM");
18 - if status~=0
19 -     warning("system error: "+cmdOut)
20 - end
```

High Level MATLAB Functions

To further abstract the VISA interface and specific instructions, the user can call a set of MATLAB functions with the command strings pre-programmed. e.g. calling *AWGrun()* sends the command to run the outputs as shown below:

```
1  function AWGrun()
2  % starts signal generation on enabled outputs
3  % Usage:
4  %     AWGrun();
5  % Inputs:
6  %     none
7  % Outputs:
8  %     none
9
10
11  AWGadd = "AWG";
12  Command = ":INIT:IMM";
13
14  % in the form of ">python (python_command) (device) (device_command)"
15  cmdStr = "python write.py " + AWGadd + " " + Command;
16
17  [status,cmdOut] = system(cmdStr);
18  if status~=0
19      warning("system error: "+cmdOut)
20  end
21  end
```

Client MATLAB GUI

Setup via Installer

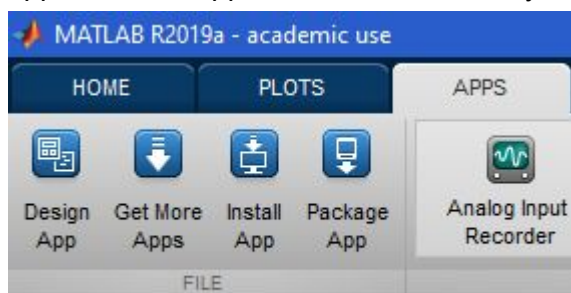
The GUI can be installed into MATLAB by running '*client/GUI/labLAN.mlappinstall*'.

Note: the app by default runs the scripts from the MATLAB installed apps folder. If the app is run without the python low level function files in the same directory, the app will need to be pointed to them.



Modifying Source Code in MATLAB's Inbuilt App Designer

The MATLAB app designer can be started by `>>appdesigner` or selecting apps>design App, note: the app cannot be modified by some earlier versions of MATLAB.



TCP Keystings

The strings received by the server and client are parsed as an array of arguments separated by a “,”. The valid keystings are shown below for some responses such as “visa, write, instID” Extra array elements beyond the number expected are combined into one argument.

Client-to-Server Commands

arg[0]	arg[1]	arg[2]	arg[3]	arg[4]	description
ping					
file	txRequest	Destination FileName			Upon receiving “file, rxReady” client is to send the .txt contents to be saved onto the server
file	upload	FileName	ChannelNumber	InstID	Server instructs the AWG to pull the .txt file sent to the server using the above function
visa	write	instID	message		
visa	query	instID	message		
visa	read	instID			
visa	listResources				

Server-to-Client Responses

arg[0]	arg[1]	arg[2]	arg[3]	description
pong				
err	Unrecognised_cmd / command_error			
file	rxReady			
file	writeResult	errorFlag	errorDescription (optional)	
file	uploadResult	errorFlag	errorDescription (optional)	
visa	writeResult	errorFlag	errorDescription (optional)	
visa	readResult	errorFlag	errorDescription / instrument response	

