0000-Project_Glen

Project Number: 0000

Project Name: Project Glen

This project is on the request of a buddy.

What he wants

Software needs to be able to read and write VISA inquires to a signal generator to output at various frequencies along a spectrum. At each of these frequencies the software should read and record the peak to peak voltage of both the input and output signals (Channel 1 and Channel 2 of the oscilloscope). Once the sweep is completed, a bode plot should be generated showing the gain of the circuit across a frequency sweep.
Two devices are connected to the software over the VISA protocol.
Device 1 is a UNI-T UTG962 Signal Generator and Device 2 is a Rigol DS1054Z Oscilloscope.

Configurable Arguments:

- Frequency Start and Stop
- Frequency Incrementation Value
- Signal to generate (sine, square, ramp, etc)
- Output location
- Plot Output Type

What he has given me: (Example)###

- Sig Gen:
  - Channel 1:
    - Type: Sine Wave
      - Frequency: Sweep from 1 kHz to 100 kHz
      - Amplitude: 200 mVpp
      - DC Offset: 2.5V
  - Channel 2:
    - Type: Square
      - Amplitude: 2.5V
      - DC Offset: 1.25V

- Oscilloscope:
  - Channel 1:
    - Volts: 2 V
    - Horizontal Time Divisions: ????
    - Measure: Vpp, Phase
  - Channel 2:
    - Volts: 200 mV
    - Horizontal Time Divisions: ????
    - Measure: Vpp, Phase
  - Horizontal Divisions: 5 ms
  - Channel 1 Volts: 2V
  - Channel 2 Volts: 200 mV

Flowchart

0. Detect Test Equipment
1. Connect to Test Equipment
2. Configure Channels on Signal Generator
3. Define Channel 2 as Output Value on Signal Generator
4. Collect user input:
    1. Ask for Starting Frequency
    2. Ask for Starting Frequency Unit of Measure
    3. Ask for Ending Frequency
    4. Ask for Ending Frequency Unit of Measure
    5. Ask for Frequency Incrementation Value
    6. Ask for Vpp Value
    7. Ask for Offset Value
    8. Ask for Phase Value
    9. Ask for Oscilloscope Channel 1 V/div
    10. Ask for Oscilloscope Channel 2 V/div
5. Run src.app.calculate_frequency_list() function
    1. f_list = empty list
    2. determine how many zeros to add to f_max, f_min, f_incr_val
    3. f_num_of_runs = (f_max - f_min) / f_incr_val
    4. for i in range(0, f_num_of_runs):
        1. f_list.append(f_min + f_incr_val)
    5. return f_list

6. Send SPCI Commands to machines
    1. Call for s_usb.dev_command.frequency_set() function
    2. Call for s_usb.dev_command.Vpp_set() function
    3. Call for s_usb.dev_command.DcOffset_set() function
    4. Call for s_usb.dev_command.phase_set() function
    5. Call for s_usb.dev_command.V_div_set() function
7. Call for src.app.run_data_gather() function
    1. data_list = empty list
    2. for i in range(0, f_num_of_runs):
        1. data_point = [0, 0]
        2. write signal generator to send f_list[i]
        3. data_point[0] = read oscilloscope channel 1 to recv data
        4. data_point[1] = read oscilloscope channel 2 to recv data
        5. data_list.append(data_point)
    3. return data_list ###
8. Call for src.app.calc_data_points() function
    1. take in data_list
    2. plot_data = empty list
    3. for i in range(len(data_list)):
        1. plot_data.append(data_list[i][0]/data_list[i][1])
    4. return plot_data
9. Call for src.app.cvt_log_scale() function
    1.
10. Call for src.app.plot_data_points() function
    1.
11. Call export plot_graph() function
    1.