

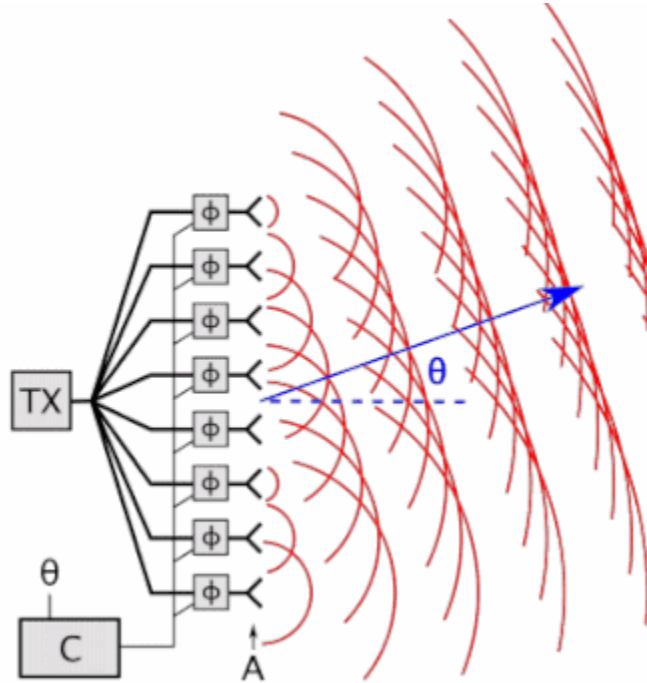
Operation Manual

Digital Beamsteering Phased Array

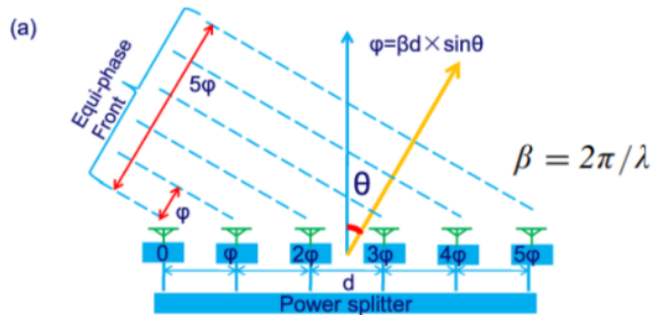
Team 311: Katheryn Potemken, Tiernen Pan, Christian Balos, William Snyder, and Andrew Cayson

I. Project Overview

The purpose of this project is to have an antenna array, where the main lobe of the radiation pattern (beam) can be pointed in a direction that the user chooses. This increases communication distance and has a stronger transmission signal than that of a regular single antenna system. Having a stronger transmitting signal means less errors for the receiver. By using this method to transmit we are also introducing weaker noise signals into the environment reducing interference with other systems. Below is a diagram of how the project will operate but only using four antennas



Each of the antennas in the array is offset by a factor of phi based off what direction we want to steer the beam.

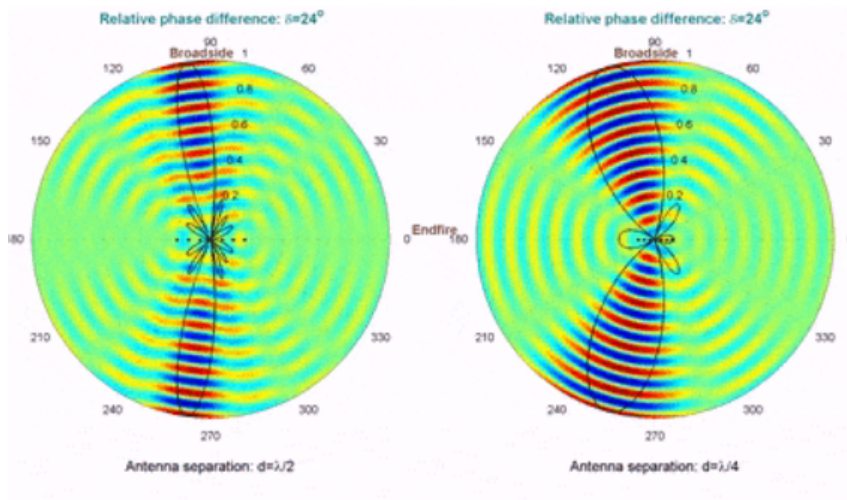


$$d = \frac{\lambda}{2}$$

$$\phi = \frac{2\pi}{\lambda} d \times \sin(\theta)$$

$$\phi = \frac{2\pi}{\lambda} \frac{\lambda}{2} \times \sin(\theta) \rightarrow \phi = \pi \times \sin(\theta)$$

The above shows a mathematical representation of what the phi values would need to be to steer the beam. Since we are using a distance of $\lambda / 2$ apart, our equation simplifies greatly.



Not only does using the $\lambda / 2$ distance make the equation simpler, but we also get a stronger signal with a smaller beamwidth (this is good). Above you can see two commonly used antenna separation distances: $\lambda / 2$ (left) and $\lambda / 4$ (right).

Knowing the above information our group came up with the following project.

II. Module Descriptions

1. AD9959 Direct Digital Synthesizer

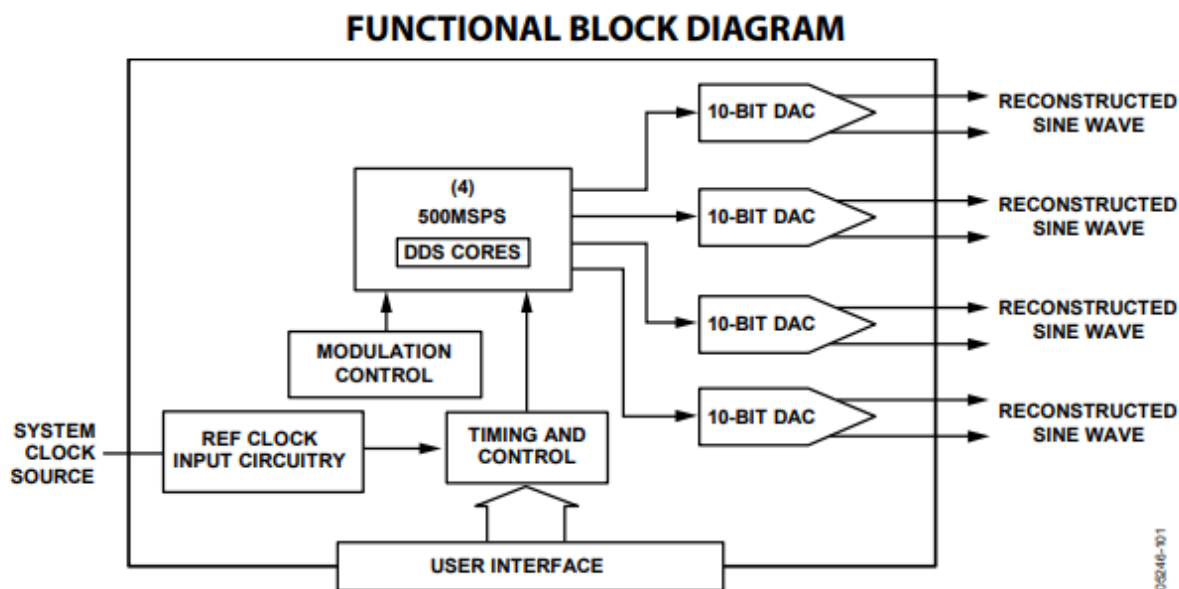


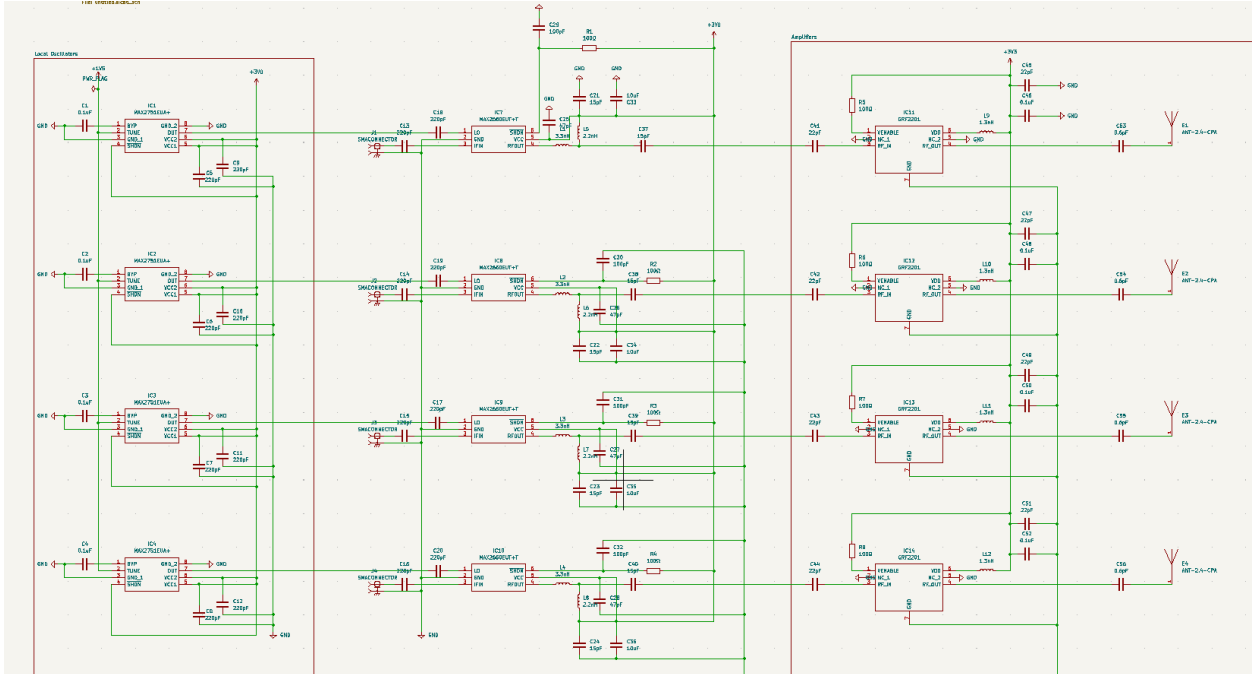
Figure 1.

The above diagram shows the basic block diagram for the AD9959. The input on the left is a system clock which will be supplied by a function generator. The user interface has the option for the user to change the settings for the DDS's basic functionality and choose options for the user to get their intended outputs. The timing and control block includes a PLL, which is a phase locked loop. The PLL increases the frequency of the system clock depending on what the user specifies in the user interface. The DDS cores are controlled with the timing and control block. This block contains the necessary instructions to shift the phase and perform other functions necessary to the signal to get the intended output. The DDS cores split into four channels and go to the 10-BIT DAC. The outputs are four analog signals.

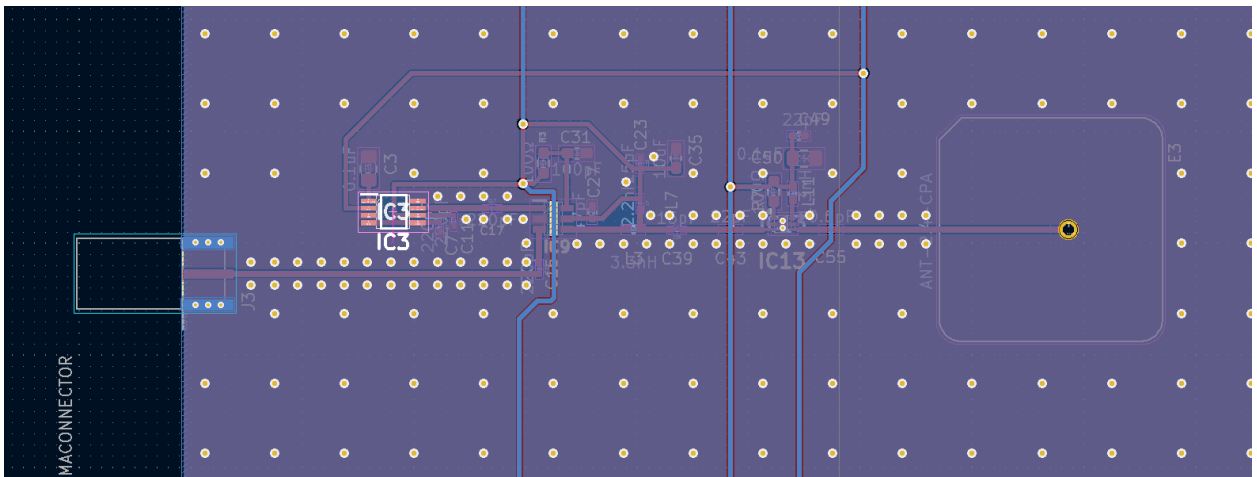
2. PCB Design

The PCB design consists of 3 major components. The MAX2751 which serves as the Voltage Controlled Oscillator. This component creates the carrier 2.2 GHz signal which is essential to upconverting our project. It takes in 3.0 V and the tuning voltage is 1.8V for it to output 2.2 GHz. You will know that this component is working if when you power the PCB up, you immediately receive a 2.2GHz signal. This is because, when the PCB is powered, the only necessary input for the MAX2751 to turn on is 1.8V from the power supply. Next, we have the MAX2660 which is the mixer for the project. This component takes in 3.0 V and serves to add and subtract the 2.2 GHz carrier signal and

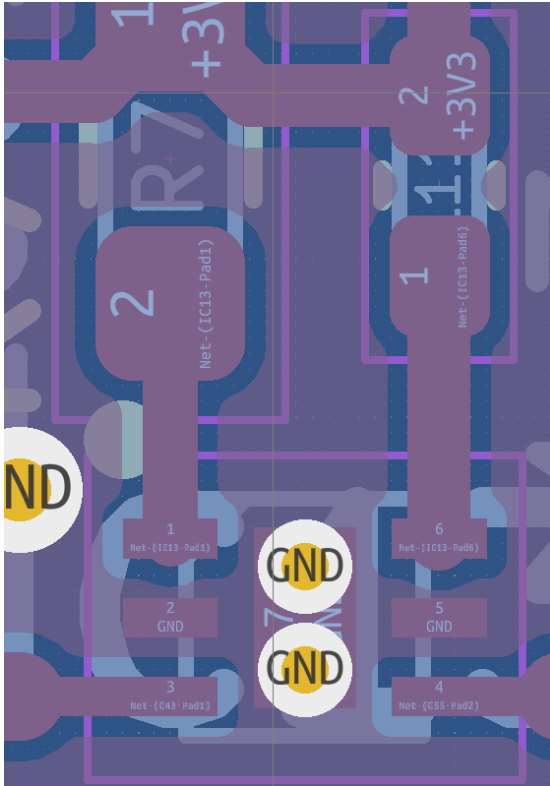
the 200MHz baseband signal from the DDS. Theoretically, in order to test if the component is working, when the DDS is properly set up, you should see both a 2 GHz signal and a 2.4 GHz signal on the microwave analyzer. However, since there is not an SMA connector output, you will not be able to see this happen as the RF signal will go straight to the amplifier and to the antenna. Just for some background knowledge, if the PCB is working correctly you will only see a 2.4GHz signal transmitted to the receiver as the Antenna only accepts frequencies form 2.4-2.5 GHz and will act as a band pass filter removing all other frequencies. Next we will talk about the GRF 2201 amplifier. This component takes in 3.3 V and will amplify our signal by 20 dB before being sent to the antenna and transmitted. Just some background information, The DDS 200 MHz signal outputs at -9 dBm approximately. When it exits the PCB and before it is transmitted it outputs 3dBm, therefore it can be assumed that our PCB board loses about 6dBm along the traces. Finally, it is sent to the antenna. According to the Friis equation, we calculated that at 1m we should expect a -38 dBm signal when testing 1 channel. Note: these values for the friis equation can be found on the spec sheets. That is a brief rundown of the components and the circuit design, the other components, such as the resistors, capacitors and inductors, and their values can be found on the MAX2751, MAX2660 and GRF2201 spec sheets. Next we will talk about via placement. The vias on the PCB allow for the RF signal to stay as strong as possible as it travels along the circuit. Therefore it is imperative that they are placed correctly. Every via should be placed $(\lambda/20)$ mm around the board where possible. However, around the RF trace, vias should be placed at $(\lambda/60)$ mm in order to reduce loss from the RF signal. Please note that lambda stands for wavelength. Should one of these vias fail, this is no cause for concern. There are about 1000 vias on the board and one failing should not be of concern. However, should multiple vias fail, especially those along the RF signal trace, a new board should be used for maximum performance and minimal signal loss.



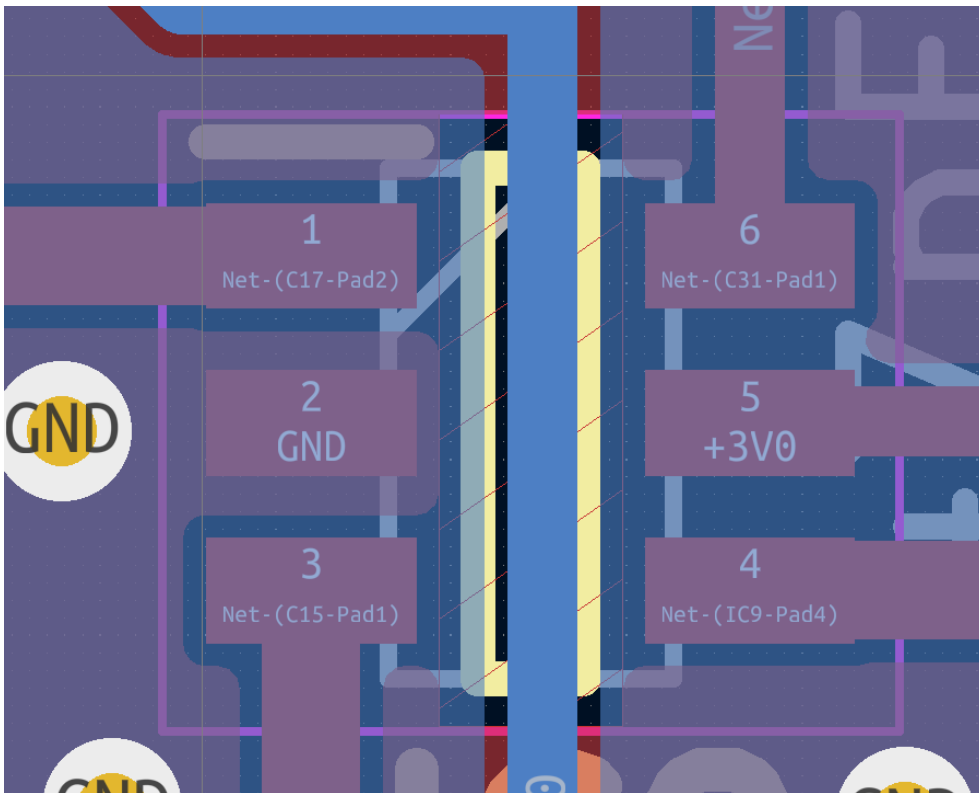
PCB Schematic



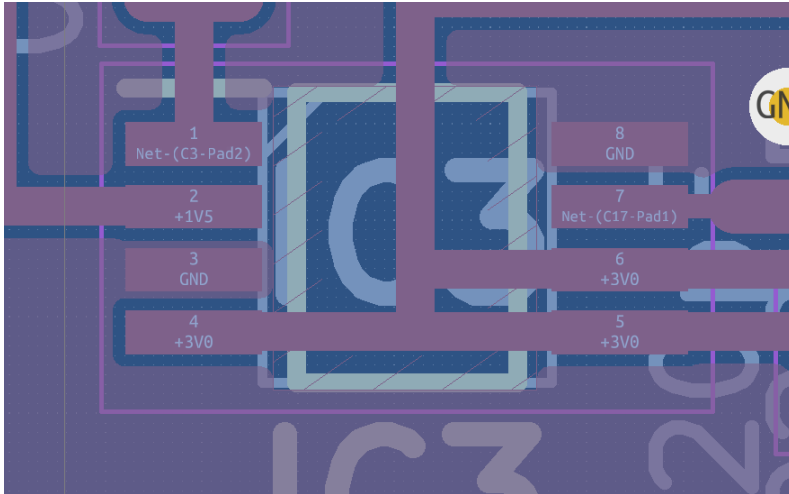
1 channel circuit of PCB



GRF2201 Schematic



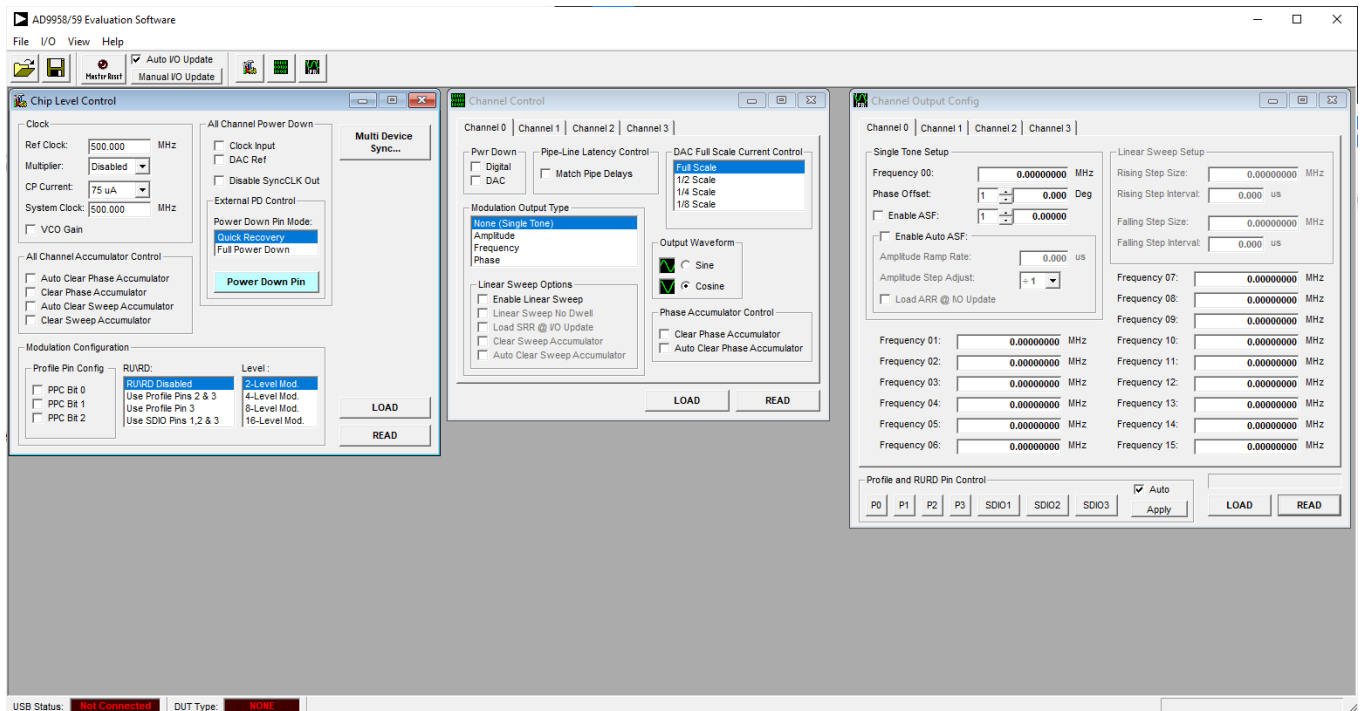
MAX2660 Schematic



MAX2751 Schematic

3. DDS GUI and Python GUI

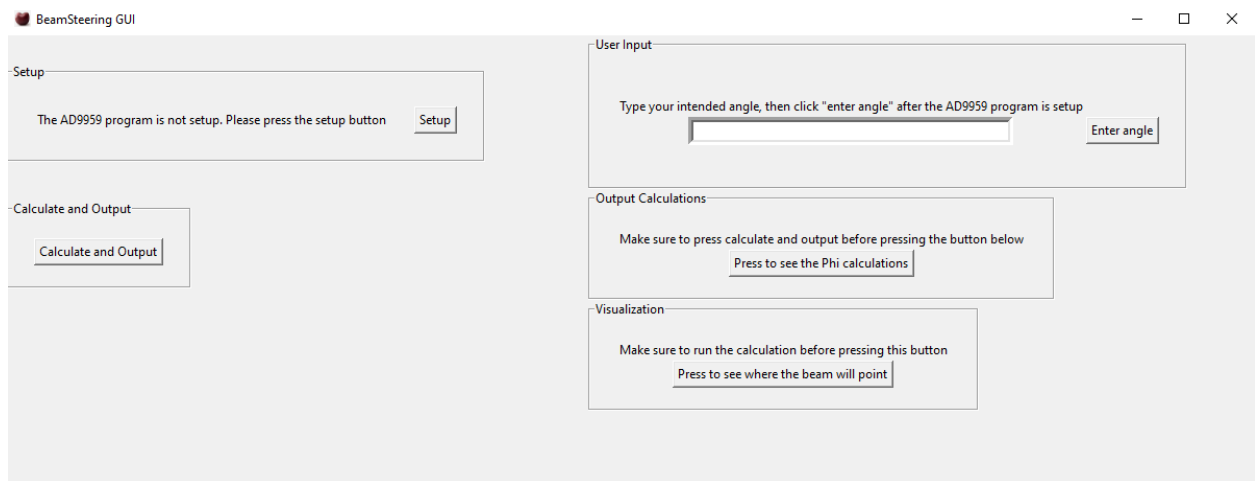
For our design we do not need to worry about the user interface seen in the block diagram for AD9959. This is because our design includes a python GUI that the user interacts with.



DDS user interface

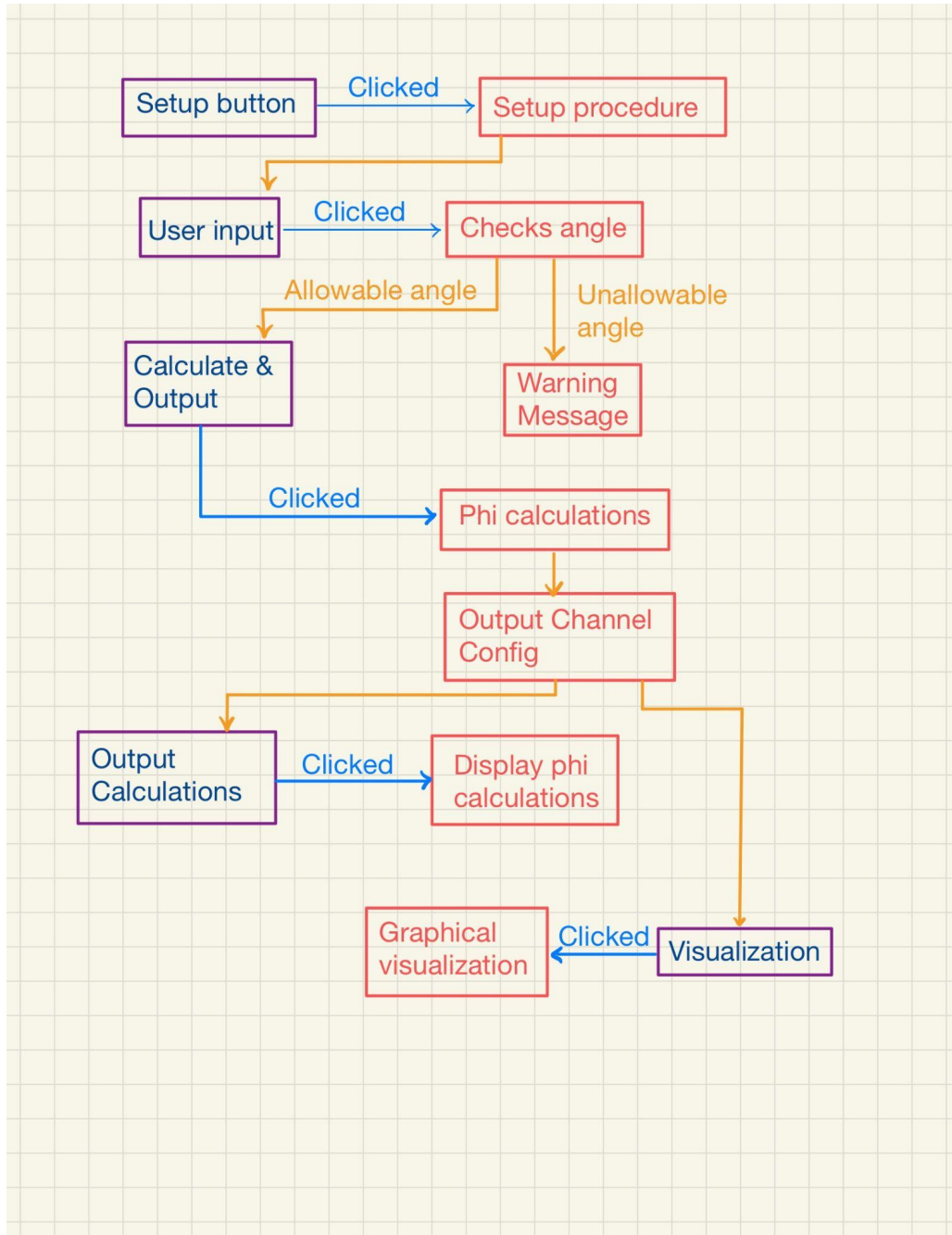
The DDS user interface contains a chip level control where you can change the reference clock setting and set the PLL to user specifications. It also contains a VCO gain which is necessary when using the PLL. For the intents of our project we will not worry about channel control since our project is not attempting to send any encoded data through the use of any modulation control types. The channel output config is where the user can change the frequency of the output signal and the phase of each individual channel's signal.

The purpose of the python GUI is so the user only has to type in their intended angle in order to steer the beam. The python GUI will set up the DDS's user interface.



Beamsteering Python GUI

The setup area has one button. By clicking that button, the AD9959 should set itself up. The user input section allows the user to input the angle of the beam direction. The calculate and output section allows the user to output the required phi necessary to change the direction of the beam. Output calculations section lets the user see the calculated phis that the python program calculated. The validation section shows the direction of the beam. The next page shows a diagram of how the code works

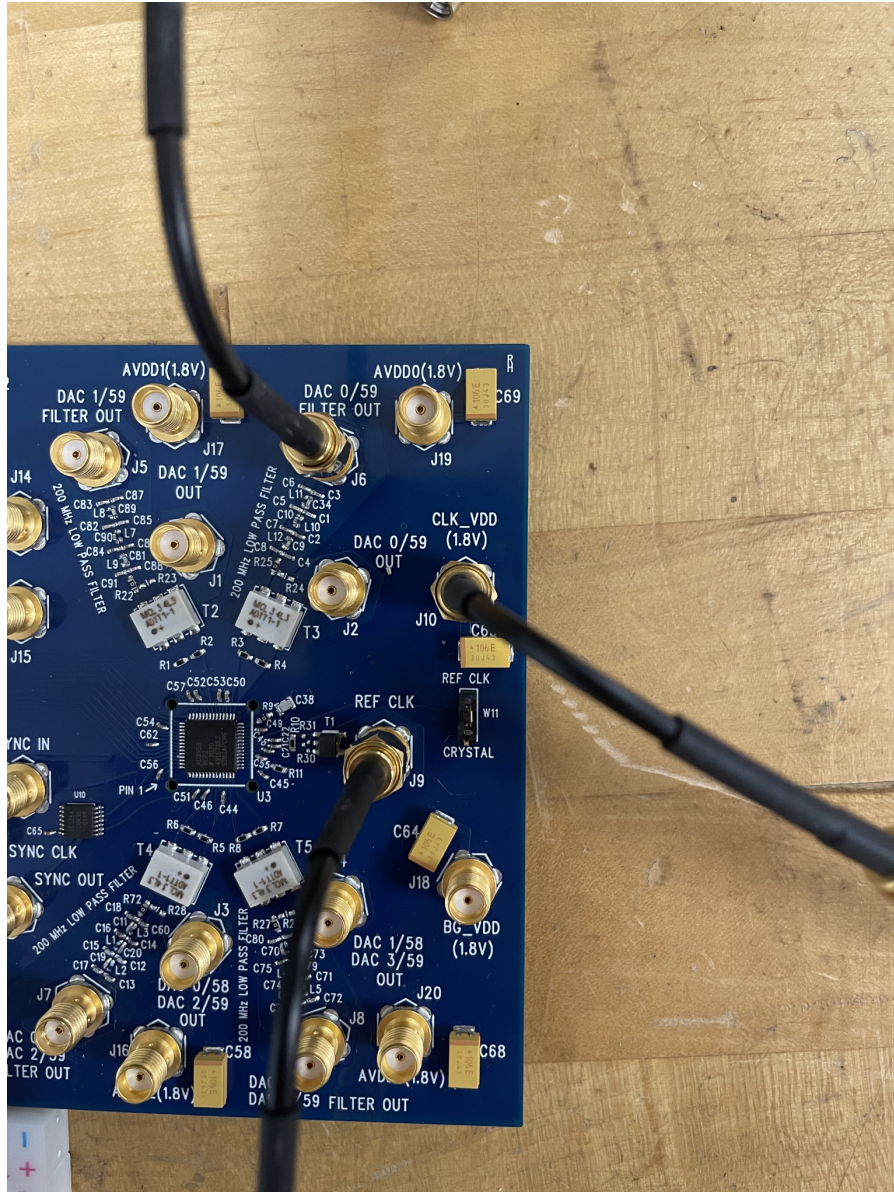


Code diagram and Github: <https://github.com/NaniMoh/FSU-Team-311-BeamSteering>

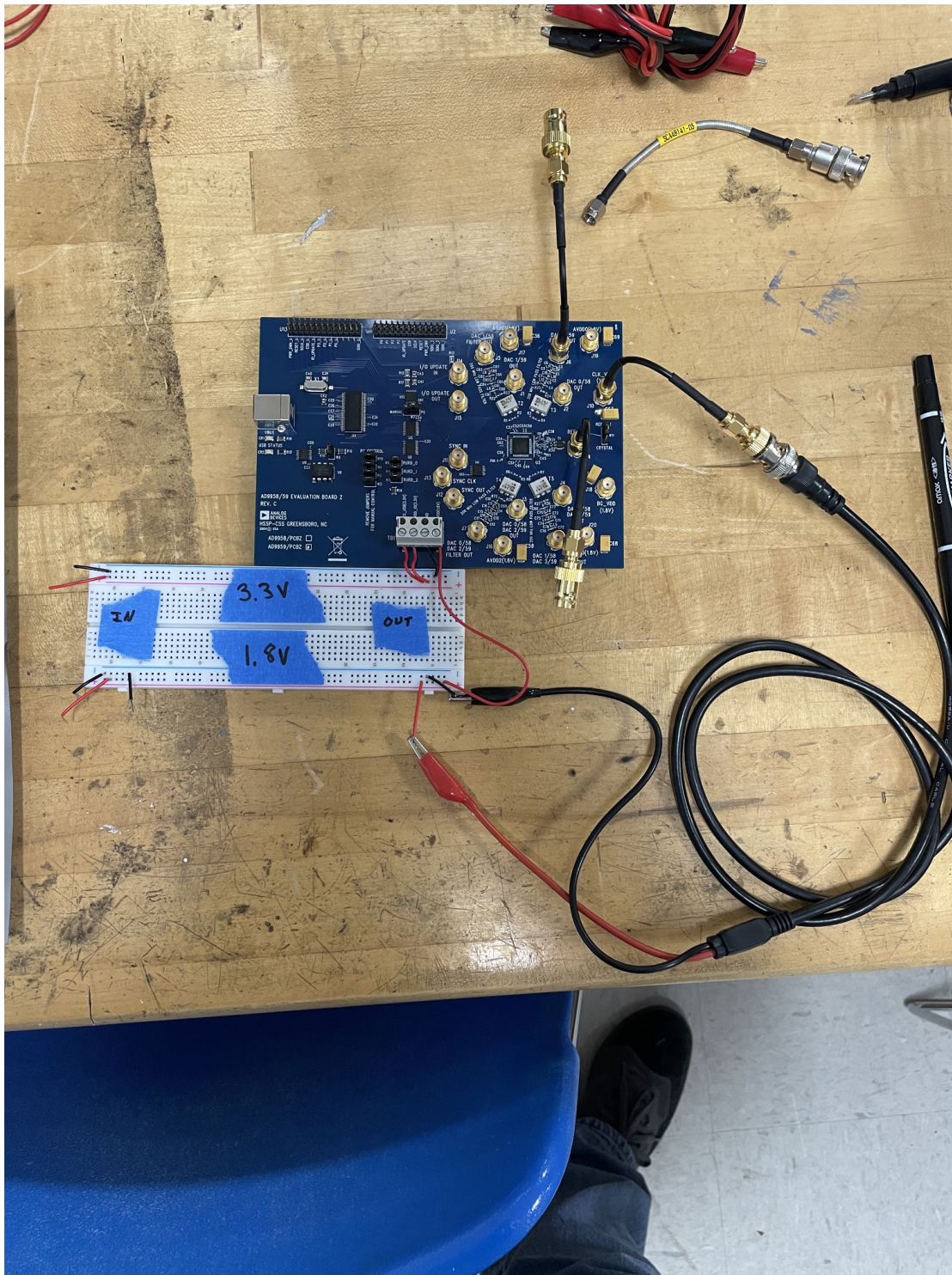
III. Integration & Operation

1. AD9959 Direct Digital Synthesizer

For the AD9959 to operate we need to supply a reference clock signal, and power the clock. The below picture shows how to power and provide a reference clock. DAC 0/59 is the 0th channel

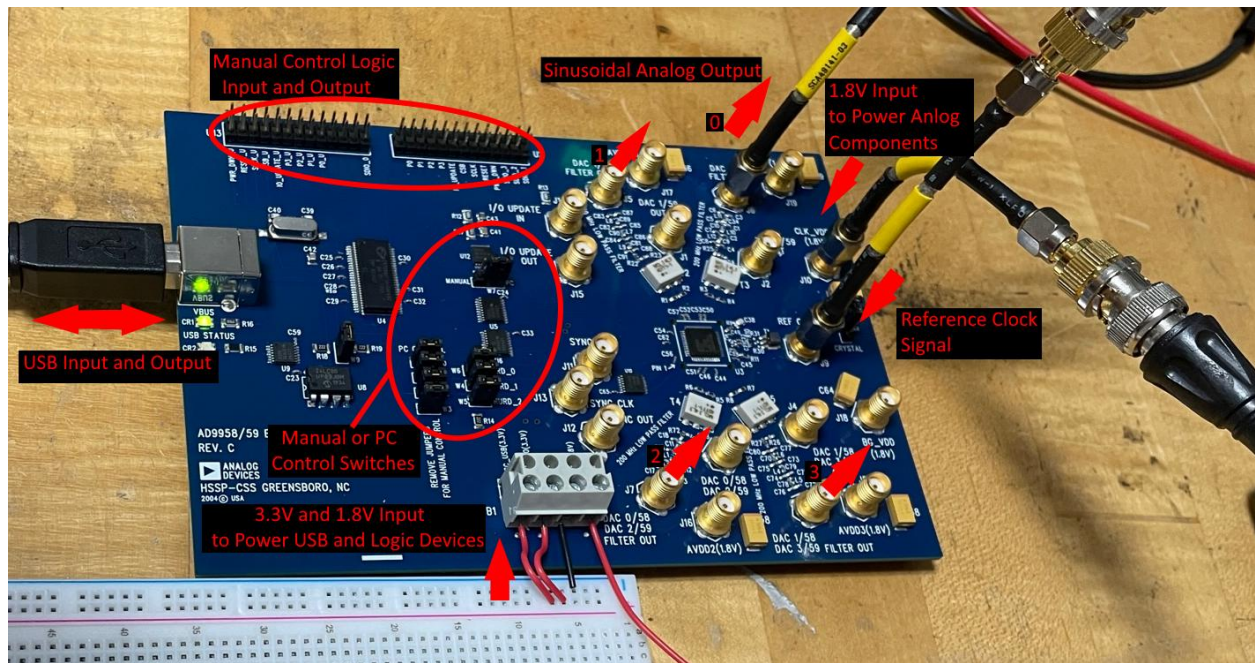


Below is the picture of a zoomed out version of the setup. 1.8 V goes into clk_vdd. And the reference clock should be supplied by the function generator. Make sure the function generator is operating at 25 MHz.

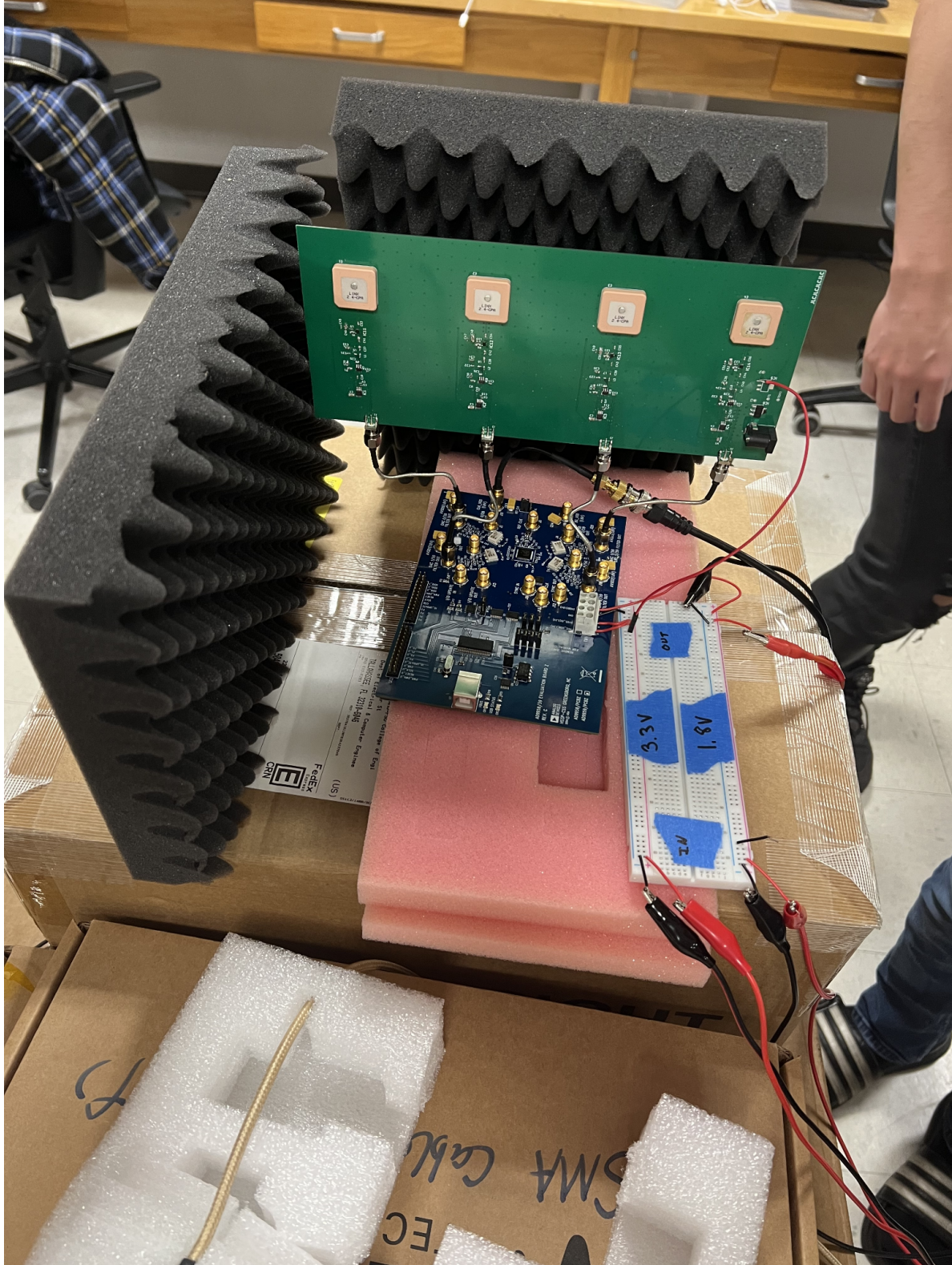


Note the above picture. On the left side of the breadboard where it says in for input, supply the power using power sources. The upper part of the breadboard has 3.3 V supplied to it and the bottom part of the breadboard has 1.8 V.

Carefully follow the diagram below



When using all channels for the beamsteering the project can look like the below picture. The inputs stay the same but you are getting outputs from each of the channels.



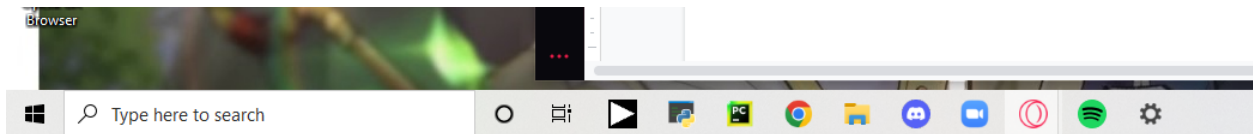
The DDS and the PCB are connected through SMA connectors. Make sure that the SMA connectors are connected so that channel 0 is connected to the 1st antenna, channel 1 is

connected to the 2nd antenna, channel 2 is connected to the 3rd antenna, and channel 3 is connected to the 4th antenna. Reading from left to right.

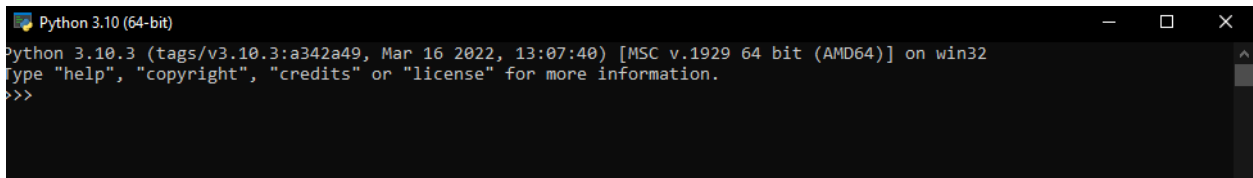
2. Python Program & GUI

Make sure that the USB status light is blinking or the design will 100% fail.

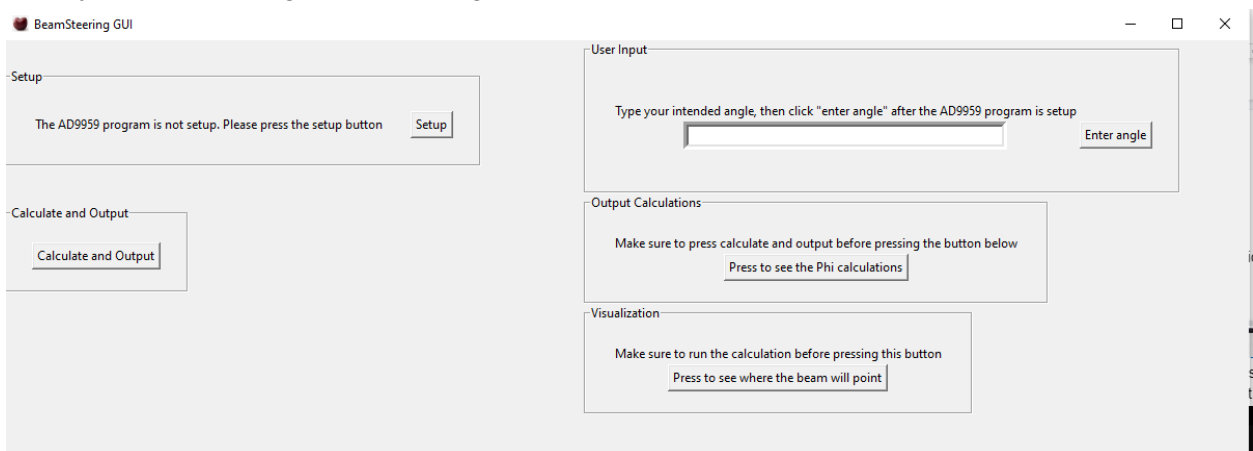
Make sure that your laptop's resolution is set to 1600 x 900 or you may have some difficulties. You must also align your apps on the bottom taskbar like the following:



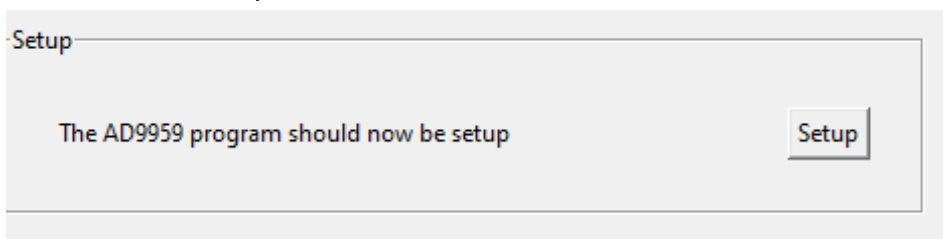
Where the AD9959 program is first and then your python 3.10 (64-bit). A later version is fine, but an earlier version of python may not. Any version of python before 3.0 will definitely not be fine.



Run the python code in a virtual environment. I recommend pycharm. The python GUI should be displayed after running the main program.



Notice that under the setup area, it says “The AD9959 program is not setup. Please press the setup button”. Click the setup button. The setup section should now say: “The AD9959 program should now be setup”.



Now you can type in your intended angle.

User Input


Type your intended angle, then click "enter angle" after the AD9959 program is setup

If the angle you select is between the range of 0 - 180 degrees then you will see this pop-up.


User Input

Type your intended angle, then click "enter angle" after the AD9959 program is setup

The angle you have selected is: 35

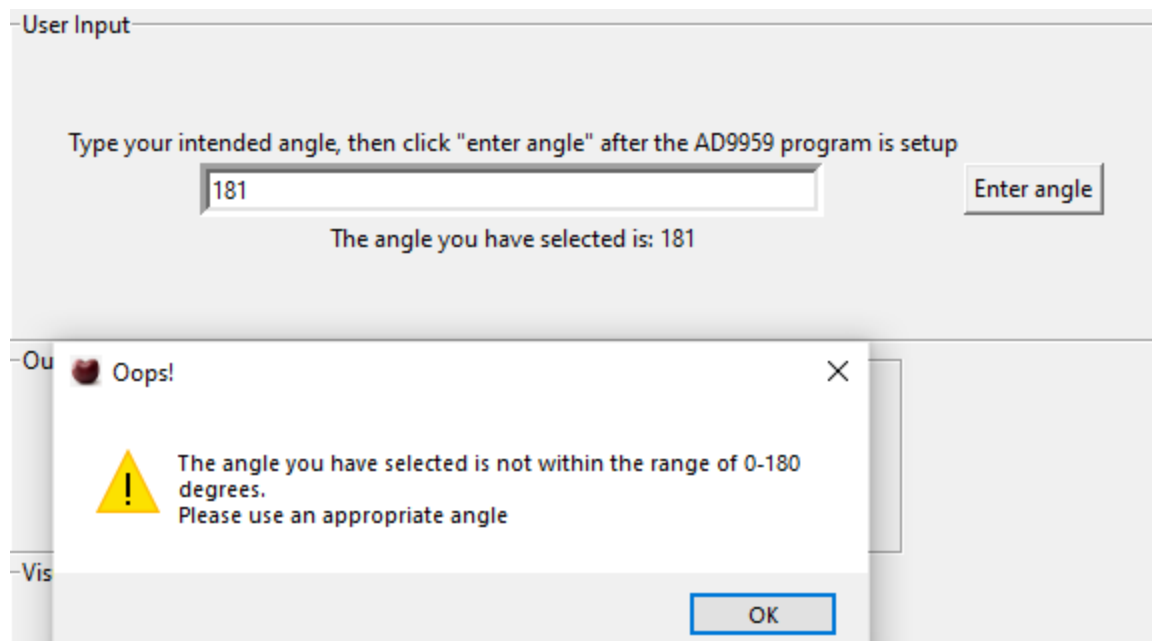
Output C  Good Job ×

Make elow

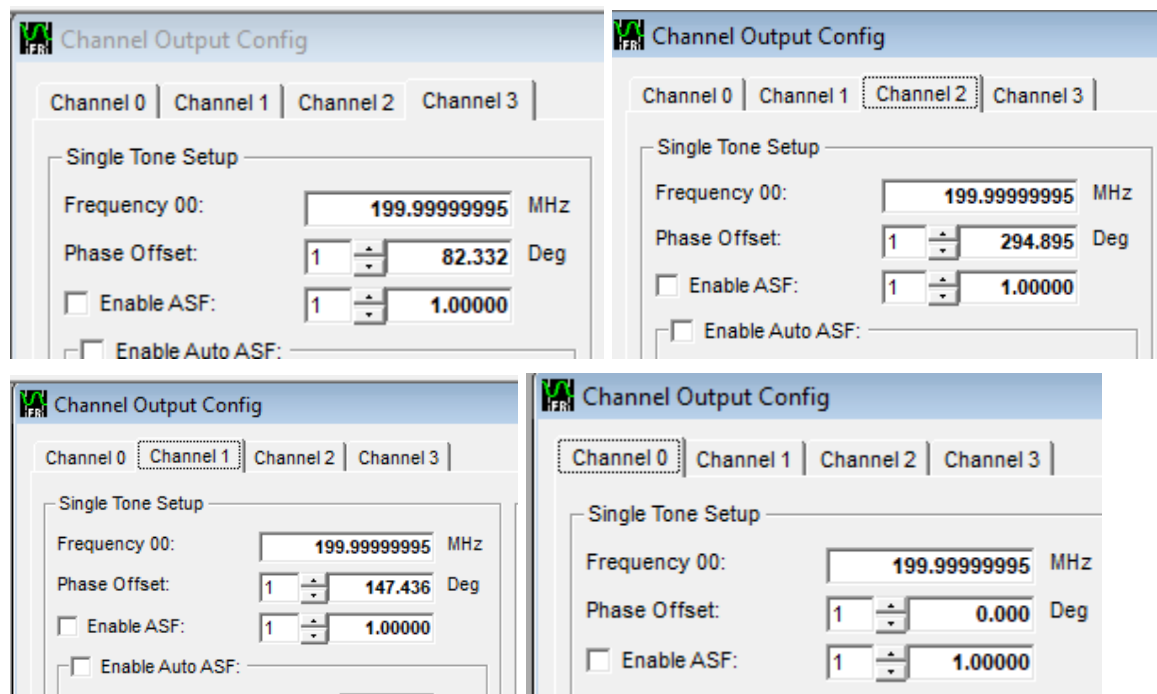
 Good, the angle you have selected is acceptable

Visualizat

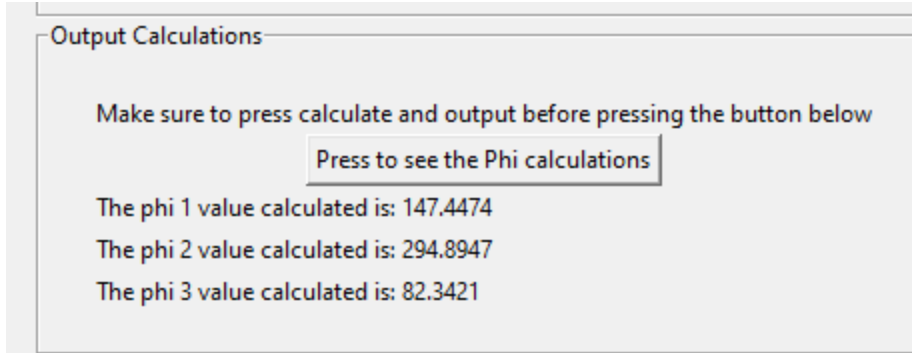
If the angle you selected is not between the range of 0 - 180 degrees you will see this pop-up



Once you have selected an appropriate angle, click calculate and output. The program will then calculate the phase offsets required for each of the channels and type them into the DDS GUI.



If you would like to see the calculated this please select "Press to see the phi calculations."



The visualization does not work properly because it is not centered properly.

Notes: Make sure to do the setup before anything else, enter a proper angle before pressing calculate and output, and press calculate and output before seeing the calculations.

III. Troubleshooting

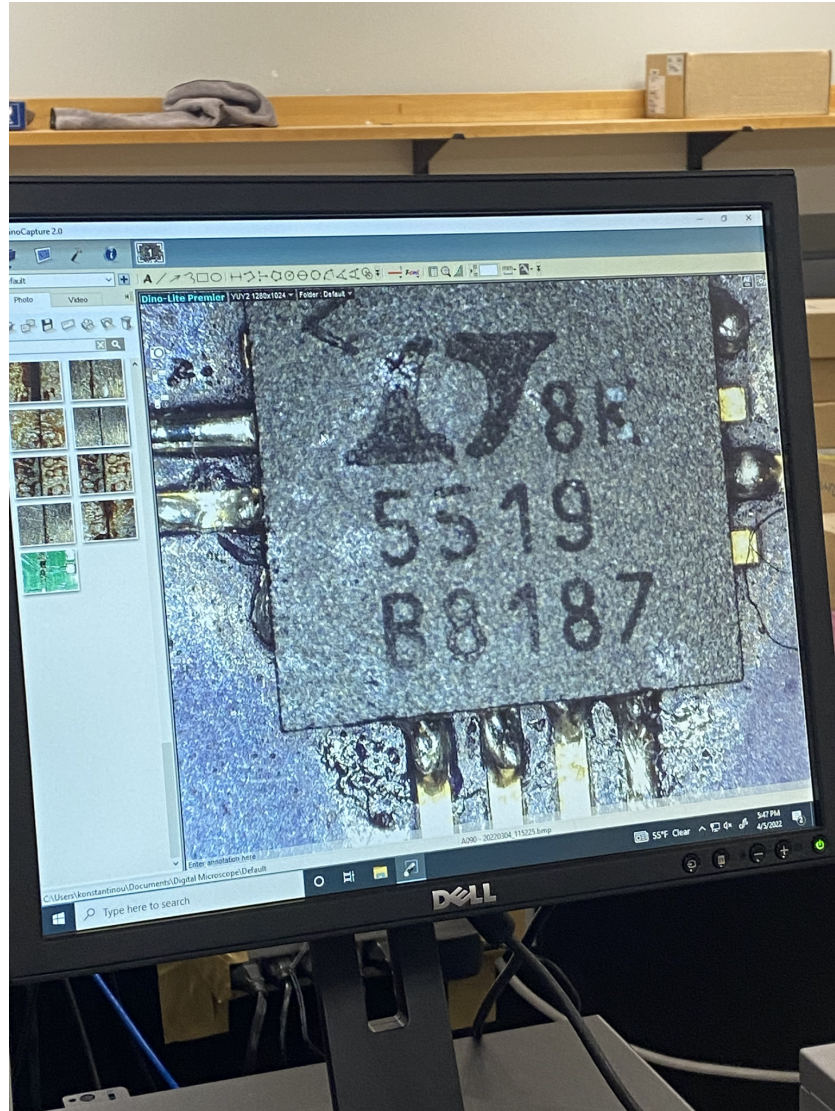
1. AD9959 direct digital synthesizer common issues

There are two modes for the DDS manual and computer. We are looking to use the computer mode. To do this you will have to flash the DDS. You may also have an issue where the USB status light is not blinking. This could be because you are not in computer mode or your computer may not recognize the device. Both issues can be solved using the documentation provided by the AD9959 website:

<https://www.analog.com/en/products/ad9959.html#product-overview>

2. PCB troubleshooting

We have done extensive testing on the PCB. The circuit is correct and performs to expected values. All of our issues have been because the amplifiers have not been working properly because of soldering issues. The components are very small, so very hard to solder. If you are having an issue with your signal it is most likely because the amplifier is not soldered properly. Please reference page 4-5, section titled "PCB design" for a more in depth explanation for what values should be expected.



This is an example of a well soldered amplifier. It is not from this project but it is a good example.

3. Python GUI

There are many things that can go wrong with the python program. One, you can have an issue where your resolution is different from the one that the team uses (1600 x 900) this is bad because we use a package named pyautogui, which controls your laptop interface (moves your mouse, clicks, and types things). When selecting a location on the screen, you feed in an x value and y value, which corresponds to the pixels that you are specifying. If you cannot make your resolution the same as the team's. You can edit where the mouse is going to move by changing the x and y values of the pyautogui functions within the code. To get your x and y

locations you can use the mouse locator code that I have supplied in the following Github:
<https://github.com/NaniMoh/FSU-Team-311-BeamSteering>