Landon Bates

April 10th, 2024

Artificial Reverberation Project Usage Guide

1. Follow the Pynq-Z2 setup guide at: <https://pynq.readthedocs.io/en/v2.3/getting_started/pynq_z2_setup.html>. This guide assumes that you are able to communicate with the PYNQ board through its ethernet and serial ports.
2. Clone the PYNQ files from github at <https://github.com/Xilinx/PYNQ.git>. This directory will be used to build .bit files for use on the board.
3. Clone my new code from https://github.com/LandonBates/Artificial\_Reverb.git. This repo contains the code that I wrote for the project.
4. Make sure you have the 2022.1 version of Vivado installed. This is the version that the default FPGA overlay for the PYNQ board was built in, and using a different version of Vivado will not work unless you want to upgrade the entire project to a newer addition of Vivado (very hard)
5. Find where you cloned the PYNQ repository to and navigate to “PYNQ/boards/ip”. Replace the audio\_codec\_ctrl\_v1.0 directory with the new audio\_codec\_ctrl\_v1.0 directory located in the git repository cloned in step 3.
6. Open Vivado and using the tcl console navigate to where you cloned the PYNQ repository from github. Then navigate to “PYNQ/boards/Pynq-Z2/base/” and run the command “source build\_ip.tcl” inside of the tcl console. After that command is finished executing, run the command “source base.tcl”. This will create a new Vivado project containing the block diagram for the PYNQ board’s default FPGA image named “base”.
7. With the project open, add directories “ip” and “new” contained in the repo from step 3 to the newly created project.
8. Open the block diagram (base.bd) and navigate to the bottom-left most corner of the diagram. You should see the audio\_codec module. Click and drag the file named full\_convolve.vhd from the project file list to the block diagram. This will create a new module named full\_convolve on the block diagram.
9. Connect the module as shown below:

A computer screen shot of a computer

Description automatically generated

1. If you want to change the filter implemented in the full\_convolve block, follow steps 11 and 12. If not, skip to step 13.
2. Open the tap\_gen.py file from the repo cloned in step 3. To generate an RIR filter, replace the file inside of the wav.read on line 10 to your own RIR recording .wav file. Set the numtaps variable to the number of filters sections you want to implement times 2048. Then, run the tap\_gen.py file. This should generate a separate .coe file for each filter section.
3. Now you will need to load the generated .coe files into the FIR compiler blocks inside of Vivado. To do this, double click on the IP core (start with F1), and change the .coe file in the IP settings.

A screenshot of a computer

Description automatically generated

Then, navigate to the Implementation tab of the IP settings and ensure that the output has 40 fractional bits. It should say this at the bottom of the tab. If it does not say 40 fractional bits, adjust the “Coefficient Fractional Bits” setting so that it is 40.

1. Right click on the base.bd file in Vivado and select “Create HDL Wrapper”. Wait for the wrapper to be generated. You should have a new file named “base\_wrapper.v” in Vivado once the process is finished running. Set this new file to the top file.
2. Now you are set to run synthesis, implementation, and generate bitstream. This will create a .bit file that is used to load the FPGA system to the PYNQ board.
3. After you have ran the synthesis, implementation, and generate bitstream commands inside of Vivado, navigate to the PYNQ board through your internet browser.
4. Upload the .bit file generated in step 13 and the Start\_Audio.ipynb file located in the directory cloned in step 3 to the PYNQ board using the PYNQ board site in your browser. The .bit file will be located in PYNQ/boards/Pynq-Z2/base/base/base.runs/impl\_1 and should be called base\_wrapper.bit
5. Open a serial connection to the PYNQ board. Instructions for how to do this are in the startup guide from step 1. Once you are connected to the board through serial, navigate to pynq/overlays/base and make a new copy of the base.bit. Make sure the copy is named something different. We are going to overwrite the base.bit file, so it is good to have a copy of the original file in case things break. Now, navigate to ../../../jupyter\_notebooks. Copy the base\_wrapper.bit file to ../pynq/overlays/base/base.bit. This will overwrite the base overlay file and allow us to use our own custom overlay without extra work.

These are the commands you should run in the serial terminal:

xilinx@pynq:~/pynq/overlays/base$ cp base.bit base.bit\_old

xilinx@pynq:~/pynq/overlays/base$ cd ../../../jupyter\_notebooks/

xilinx@pynq:~/jupyter\_notebooks$ cp base\_wrapper.bit ../pynq/overlays/base/base. Bit

1. Now, back on jupyter notebook website for the PYNQ board, open the Start\_Audio file and run all three code blocks. Now there should be audio with reverberation streaming through the board.