# DAC Pulse Generator

Leonardo Aviles, *tue51778@temple.edu*

## i. INTRODUCTION

IN project the objective was to create a DAC or digital to analog converter and be able to change the frequencies, duty cycle, and amplitude. An external SPI bus 12 bit DAC Pmod is used in order to do the conversion needed.

The specifications of the project are as follows.

- The duty cycle or pulse on to pulse period will range from 5% to 95% which will increment by 5%
- The frequency will range from 1Hz to 10kHz in steps of 1 Hz
- Amplitude will range from 50mV to 3000mV in steps of 50 mV

To complete the project, the modules rcvr.v and txmit.v, are reused. The following new module was implemented.

- "GenDAC" in this module the analog signal will be created from the digital values.
- "GenFreq" was used in order to check what the user inputs and change the frequency based on the input. The generated frequency would then be placed in the clock.v module in order to make the necessary clock for that specific frequency
- "GenVolt" in this module different value are placed in DacData in order to create the different Vouts or amplitudes.

## ii. DISCUSSION

### *A. Top Module: "Lab4"*

This module acted as the controller module sending the output and input data to their respective modules such as the RXD and TXD values. Another example is using the JE4, JE5, JE6, and JE7, which will be the sync and the output of the signal.

### *B. Generate Dac: "genDac.v"*

In this module the dacValue is placed bit by bit into dacdata, which will then produce the analog output signal.

### *C. Generate the specified frequency "GenFreq.v"*

GenFreq is used to generate the frequency that the user will enter. Depending on the input, which will be TDIN, the frequency will change depending the ASCII value.

### *D. Generate amplitude/Vout "GenVolt":*

This module will use the TDIN input value and change the voltage based on the ASCII value. Using the equation Vout=(D[11:0] *Vref)/2^12 the different voltages can be found.

## iii. CONCLUSION

The project was not implemented correct when trying to program the Zybo board. The original project stated that the value would need to be incremented by different steps by using ASCII character but instead different ASCII values were assigned to either different voltages or frequencies. When trying to test the code the Tera Term program was used and whichever ASCII value was typed in would be shown on the terminal. This area did not work properly and would not show the user which key was pressed.

The next step to check if the program/code worked correctly was to hard code the different frequencies and amplitudes of the signal. For the code the frequency was set to 100Hz, which would cause the clock scale to be 625,000. The amplitude or Vout would be set to 500mV, which, would be made by placing the value 621 in to dacData. When the signal is viewed through a portable oscilloscope on the waveform program a small wave was found that looked look similar to a triangle wave. This wave could have either been produced by some interference or by the code. In this project a square pulse wave was expected but was not received.

## iv. Appendix

*Figure 1: Shows the controller and data path and the different signals and data received*