Assignment 6: Cordic Sin & Cos

In this assignment we're building a streaming sin & cos generator using the Cordic algorithm, and introducing quantization to convert the floating point model to fixed-point hardware implementation. You will be comparing the precision of the implementation against the software C library to gauge the quantization error.

Implementation:

- Build a quantized cordic algorithm that generates the Sin & Cos values
- Simulate in software for theta in range -360 to 360 degrees, and generate quantized outputs for sin and cos
- Implement a 16-stage hardware pipelined cordic in SystemVerilog, which outputs a new value every 16-cycles. You should instantiate 16 hardware-pipelined cordic_stage components using the GENERATE-FOR statement.
- The cordic module should read in theta via FIFO, and output SIN and COS values to separate output FIFOs.
- Implement the UVM model to validate the RTL model and compare fixed point results to the software implementation, and determine the precision of quantization error.
- Determine the throughput in terms of number of samples per sec given a 100Mhz clock (10ns period).

Verification:

- In your testbench, generate 16-bit quantized fixed-point values for theta between -360 to 360 degrees, feeding them into the input FIFO of the cordic module.
- Configure your testbench with the following generic parameters:
    - 32-bit wide data for radians
    - 16-element FIFO
    - Frequency of 100MHZ (10ns clock period)
- Create a cordic_sim.do file that compiles the SystemVerilog and UVM models, sets up the wave form, and runs the simulation.
- Run the simulation in QuestaSim and verify the design is bit-true accurate.
- Synthesize the design in Synplify Premier to get high-level resource results and technology schematic. You don't need to go through Place and Route.

Compare Results:

- Compare the output of SIN and COS to the software values to check for bit-true accuracy.
- Report on quantization precision (e.g. to what degree of accuracy compared with the floating point software algorithm).

Reporting:

- Submit a PDF file with simulation and synthesis results with the following:
  - Simulation results:
    - Clock cycle count
    - Errors reported (if any)
  - Synthesis results (some might not apply):
    - Maximum frequency
    - Registers / LUTs / Logic Elements
    - Memory utilization
    - Multipliers (DSPs)
    - Worst path (timing analysis)
    - Schematic architecture (RTL)
    - Performance / Speedup
    - Throughput (samples-per-second)
    - Include pictures and a brief description of your architecture from Synplify Premier.

Turn in your designs with the report. Your file should be zipped, and should include the SystemVerilog files, synthesis, simulation, and input/output files. **PLEASE MAKE SURE TO REMOVE ALL THE INTERMEDIATE WORK DIRECTORIES.**