Goal: implement 2x2 SVD in Verilog.

Current Deliverables:

- Verilog CPU with RISC-like ability (should be bug free)
- Python compiler that converts MIPS-style assembly into verilog-executable code (not all instructions fully implemented)
- 2x2 SVD in MATLAB (works for limited set of matrices)

Hours left to throw at project:

- Chaz: 10 hours
- Jimmy: 5 hours
- Thus, responsibility for green features has about a 2:1 ratio

<u>Green features:</u>

Implement 2x2 SVD in MIPS assembly code:

--Implement subfunctions:

- 2x2 matrix multiplcation || Responsibility: Chaz
- 2x2 matrix transpose || Responsibility: Chaz
- Inverse tangent || Responsibility: Jimmy
- Sine, Cosine || Responsibility: Jimmy

--Ensure 2x2 SVD in MIPS assembly works || Responsibility: Chaz

Create final project report:

--Write following subsections:

- Formatting: Title, Author Names, Date || Responsibility: Chaz
- Introduction: 1 paragraph summary of what we did || Responsibility: Chaz
- Background:
    - A (brief) explanation of SVD || Responsibility: Jimmy if has time, Chaz otherwise
    - Discuss and professionalize “We used the CPU from Lab 3, and made a compiler to convert our MIPS-style assembly code to machine code” || Responsibility: Jimmy
- Design:
    - An top-down explanation of 2x2 CORDIC SVD design || Responsibility: Chaz
    - How we implemented CORDIC || Responsibility: Jimmy
    - How we implemented matrix math || Responsibility: Chaz
- Correctness:
    - MIPS Testbench results for Matrix Math || Responsibility: Chaz
    - MIPS Testbench results for CORDIC || Responsibility: Jimmy
    - MIPS Testbench results for 2x2 CORDIC SVD || Responsibility: Chaz
- Time Analysis:

    - Clock cycle count on each subfunctions (aka count the number of instructions) || Responsibility: Subfunction writer adds their subfunctions
    - Clock cycle count of total program || Responsibility: Chaz
- References:
    - ala Wikipedia, || Responsibility: Whoever has sources to cite

Yellow features:

-Get Python compiler completely implemented and working. || Responsibility: Jimmy

Red features:

-Systolic Arrays. Ain’t nobody got time for that.