3 Project Plan

3.1 Project Management/Tracking Procedures

For our project, we have elected to use the waterfall project management style. This is because we have certain tapeout deadlines that must be met, such as the November and April tapeouts. Due to the nature of the chipfoundry delivery dates, we will not know if our modules are silicon-proven until after the next tapeout. Thus, we feel that the common goals of agile are not as relevant to our project, and we need to have our design mostly to fully realized before we start creating the submodules.

We track our progress by keeping a list of tasks to do and tasks completed. This list is stored in a OneNote and is updated during our weekly meetings. Additionally, our project is stored in a Git repository through GitLab, so we create branches for individual modules and use merge requests to ensure quality HDL and code is submitted.

3.2 Task Decomposition

1. Install toolchain and do ChipForge tutorials
   (a) Complete the blinky, UART, and Wishbone Adder tutorials
   (b) Edit the Wishbone Adder to get used to the submodule organization
   (c) Create a custom user project to understand module design from scratch
2. Define project specifications
   (a) Decide what digital ASIC we will implement
   (b) Decide and setup a specific verification method
   (c) Declare what the inputs and outputs of the software are
   (d) Declare what the inputs and outputs of the hardware are
3. Define project design
   (a) Create block diagram to describe flow of data
   (b) Define what operations happen in software and what happens in hardware
   (c) Write ISA to run on the specified cores
   (d) Decide on what peripheral modules may be needed
4. Develop hardware to run μGPU[1]
   (a) Create PMOD PCB design for external memory
   (b) Test design with FPGA SPI controller and VGA controller

5. Develop software to run μGPU[1]
    (a) Learn how to parse through Wavefront object file (.obj)
    (b) Define where data will be stored in memory
    (c) Store textures, indices, and vertices in memory
6. Write and test Verilog modules[1]
    (a) Assign each module to a member, and the tests for that module to another member
    (b) Write the Verilog module & SVUnit tests in parallel
    (c) Ensure module passes quality tests
    (d) Assign a 3rd and 4th person to review the module before it is merged
7. Connect and test Verilog modules in the top level design[1]
    (a) Once a submodule is verified, implement within a larger module
    (b) Verify the larger module with SVUnit
    (c) Ensure top level design and software work as expected
8. Test synthesis options
    (a) Ensure designs pass timing
    (b) Calculate maximum possible clock speed of GPU
    (c) Decide what PnR (Place and Route) technique should be used
9. Complete pre-check and layout
    (a) Ensure the entire GPU design fits within die area
    (b) Ensure design passes DRC (Design Rule Check) and LVS (Layout Versus Schematic)
10. Submit for tapeout
    (a) Ensure design is valid and passes Skywater DRC
    (b) Submit to chipfoundry
11. Write user guide
    (a) Write procedures for validation and bring-up once the ASIC is shipped
    (b) Write a guide on how to use the μGPU

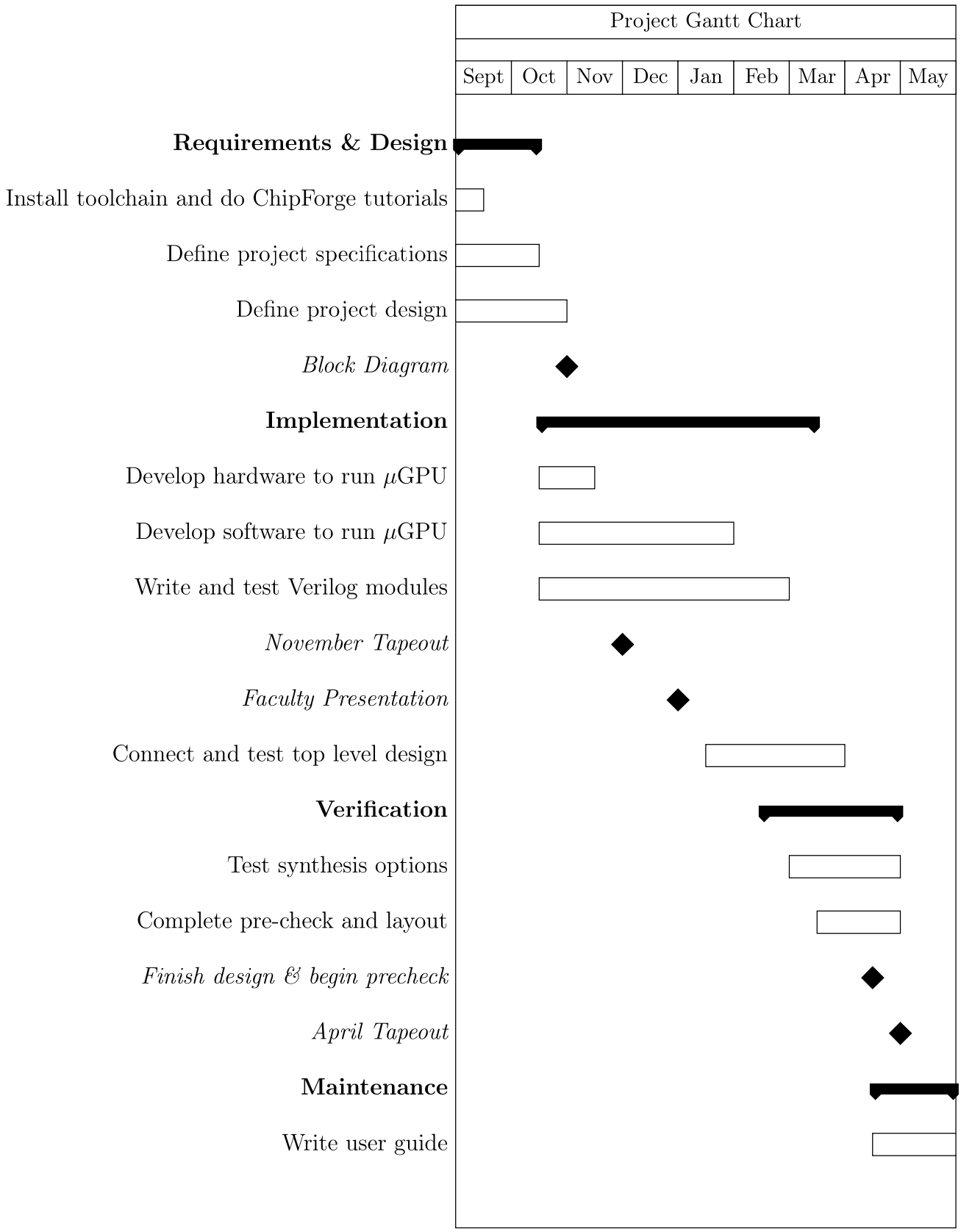
[1]: Tasks done in parallel

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

- Block Diagram
  - Complete GPU block diagram with all required modules
  - Get Client and Advisor approval of design
- November Tapeout
  - Finish verification of rasterizer and bus components
  - Add all verified designs in layout and pass DRC/LVS
- Faculty Presentation
  - Finalize GPU design for presentation
- Finish Design and Verification. Begin Final Precheck
  - Get entire GPU optimized to fit within die area
- April Tapeout
  - Complete precheck, layout, and pass DRC/LVS
- Industry Review Panel
  - Finalize Validation plan
  - Complete software implementation for a visual presentation

3.4 Project Timeline/Schedule

The following is a Gantt chart of the tasks listed above, with the bolded categories being waterfall groups, and diamonds being key milestones


Project Gantt Chart
Sept
Oct
Nov
Dec
Jan
Feb
Mar
Apr
May
Requirements & Design
Install toolchain and do ChipForge tutorials
Define project specifications
Define project design
Block Diagram
Implementation
Develop hardware to run μGPU
Develop software to run μGPU
Write and test Verilog modules
November Tapeout
Faculty Presentation
Connect and test top level design
Verification
Test synthesis options
Complete pre-check and layout
Finish design & begin precheck
April Tapeout
Maintenance
Write user guide

3.5 Risks and Risk Management/Mitigation

Most of our major risk comes from our fabrication requirements. This includes fitting the design within the die area, meeting our functional requirements with the permitted frequency, and submitting by the tapeout deadline. Another major risk is possible errors during fabrication. There is a non-zero chance that the chip that is returned has issues that cannot be found unless exhaustive validation is done. In this case, this risk is completely out of our control, but having a validation plan can prevent further issues.

In addition to fabrication, the risks fall mainly on design and implementation. Most of these risks are simply the case that the module is not completed or a failed verification. These risks would have great consequences on the project, since every component needs to be present for the GPU to work as expected. Although the big impact has been observed, the probability that a design is not finished is much lower than other risks due to DRC, LVS, or area constraints.

Although our end goal is fabrication of the μGPU, the impact of fabrication risks do not make this project meaningless. The design itself will still benefit Chipforge members, allowing them to learn and build off GPU architecture. In this case, most risks are that a functional component does not work as intended. We plan on reducing these risks significantly by making sure each module is verified by a different member through SVUnit.

Risk	**Probability**	**Project Impact**
Chipfoundry closes or doesn't allow tapeout	5%	High
Chipfoundry deadline not met	10%	High
Cannot optimize design to fit the die area	20%	High
Cannot optimize design for 15Hz framerate	30%	Medium

Table 1: Project Risks

3.6 Personnel Effort Requirements

These are the predictions of our time requirements for the μGPU, which is based on our initial weeks working on senior design and past ChipForge experience. This is also based on the work of previous senior design teams with projects in ASIC design.

Task	**Time Estimate (hours)**
Install toolchain and do ChipForge tutorials	35
Define project specifications	15
Define project design	40
Develop hardware to run μGPU	20
Develop software to run μGPU	50
Write and test Verilog modules	100
Connect and test Verilog modules in the top level design	25
Test synthesis options	80
Complete pre-check and layout	40
Submit for tapeout	10
Write user guide	90

Table 2: Personnel Effort Requirements

3.7 Other Resource Requirements

Our extra resource requirements will consist mostly of the equipment that can be found in the senior design lab and Chipforge lab. This includes osciliscopes, logic analyzers, FPGAs, and computers. All of these resources are freely available to use, so will not be a problem to attain.