**Overview**

During the Spring 2023 Semester, I began my senior design project at Iowa State University. Senior Design is a two-semester course, with a focus on research and planning in the first semester, and then design and implementation in the second semester. The project that I chose to join with three other group members was Digital ASIC Fabrication, which involves designing and fabricating a digital ASIC. Our client and advisor are Dr. Duwe, a professor at Iowa State who researches and teaches about computer architecture and cyber security topics. Dr. Duwe has had two previous senior design groups follow the same process and design cycle as we are, each staggered one semester apart. The first senior design team was able to successfully get their chip approved and fabricated, and we are currently waiting to receive it. For our project, our requirements are to design a digital ASIC chip using Verilog and open-source tools, alongside developing firmware and a bring-up plan to assist a future team in verifying the functional requirements of our design.

We are able to create our digital ASIC by working through the company eFabless, which is sponsored by Google. The goal of eFabless is to use open-source tools that are freely available to silicon prove both digital and analog ASICs for hobbyists, researchers, and students, who do not have a large pool of resources to take advantage of. Being a senior design team, this was an excellent opportunity to learn more about writing Verilog, using new open-source tools, and designing a chip for fabrication that would come back on an SOC development board. All of the submitted designs are required to be open-source, with the benefit of receiving 5 ASICs that are placed on a development board. eFabless produces their silicon wafers through the U.S. based foundry SkyWater, which takes advantage of 130 nm technology and a standard cell library to fabricate our project. Due to this, we are limited to this fabrication process and the standard cell library that is provided by SkyWater in their 130 nm product development kit. Each project submission is produced on a wafer that houses open-source designs from multiple different projects, enabling eFabless to produce small amounts of ASICs for each team at a reduced cost. While working with open-source tools and a limited production cycle has been difficult through eFabless, they are in a very unique and rare position by offering free digital ASIC fabrication, that is returned on an integrated development SOC board for us to test.

As of December 2023, the project has been completed, and has passed the precheck so that it is ready for submission in the next Multi Project Wafer submission at the Skywater foundry with the 130nm process and open-source PDK. I learned a lot about this process both from utilizing rapidly evolving open-source tools, and working on varying parts of the ASIC design process, such as functional design, verification, and place and route all at different times! This project was extremely valuable and was a great capstone for my senior year of school at Iowa State University.

Useful Links:

* Senior Design Website: <http://sddec23-06.sd.ece.iastate.edu/>
* Design Document: <http://sddec2306.sd.ece.iastate.edu/reports/Design_Doc_Rev1_1.pdf>
* Design Presentation: <http://sddec2306.sd.ece.iastate.edu/reports/Final_Presentation.pdf>
* eFabless: <https://efabless.com/>
* Other projects on eFabless: <https://platform.efabless.com/projects/project_definition/1>
* Toolflow Documentation: <https://caravel-user-project.readthedocs.io/en/latest/#quickstart>

**Semester One**

During the first semester of senior design, our team focused on ironing out our digital ASIC design, learning how to use all of the required open-source tools, and beginning implementation and testing on submodules of our design. Throughout this process, we were able to learn more about eFabless and what requirements and constraints we had to follow for our design, which helped motivate some of our design decisions.

We decided to implement a modular design which included the following modules all in the user area of our digital ASIC:

* Clock Gating
* Standard Cell Test
* Custom Cell Test
* Wishbone Test
* DSP Road Noise Audio Filter
* Backdoor SPI Interface

The clock gating, standard cell, custom cell, and wishbone tests were included in our design so that we could verify the fabrication of our chip with more concrete and basic tests. For example, if the rest of the fabrication failed, we should ideally be able to still probe the Wishbone communication bus with the Wishbone Test module to ensure that something is working in our chip. We also can use the Standard Cell Test to verify that one basic SkyWater 130 nm standard cell from the provided product development kit library is fabricated correctly. This cell will be a basic combinational circuit that will be easy to verify functionality of. Other tests, such as the Backdoor SPI Interface, are used in case the Wishbone bus fails. In case it does, we have included a separate SPI interface so we can use our own communication bus that we designed to program the configuration of each module, alongside reading data that is output from each addressable module. Of these modules, I am responsible for leading the Backdoor SPI module, alongside assisting with documentation and system level integration throughout the project. I was very excited to work on the Backdoor SPI module, since it meant that I would have to work with interfacing between each module to configure and receive data from them with a standardized serial communication protocol.

Before we could begin on designing the Verilog and testbenches for each module, we had to first install and configure our open-source tool flow. eFabless provides a GitHub repository that can be cloned, which includes make files and the 130 nm SkyWater standard cell library. Alongside this, the repository also includes an MPW shuttle precheck pass, which will be used next semester for our final acceptance testing, so that we will have a verified submission for the project’s fabrication. The main open-source tools that we are using in our project include OpenROAD0, KLayout, GTKWave, and Magic. OpenROAD is used for our RTL synthesis, and takes our Verilog designs and the SkyWater Product Development Kit to develop GDSII layout files of our design. With this layout, we are able to view our project with KLayout, verify our hardened gate level designs, and submit our project to eFabless by instantiating our project in a user wrapper module in the provided repository. GTKWave is used to verify our waveforms from Verilog testbenches and C firmware simulations at both the register transfer level and gate level. We can instantiate our modules either on their own or through the caravel wrapper, depending on the stage of verification and how robust we need at the time. The open-source tool Magic will be used for our Custom Cell module, as we can use it to layout the traces for the combinational cell we will design. After this cell is laid out in Magic, we can instantiate it within the user wrapper module. These tools have been difficult to setup and use initially, but it has been a great learning experience to try a new tool flow and new constraints, all for an open-source project.

After getting situated with our tools, I began to design the Backdoor SPI module with the help of my teammates. We decided to use shift registers to serially shift in data from an external microcontroller and shift out data from our user area modules. Alongside this, since we were using a clock provided from an external microcontroller, and the clock in the user area were different, we used 2 D Flip Flops to act as a buffer to ensure that there would be no metastability errors in our data flow. During this semester, I began by designing the shift registers in Verilog, and derived testbenches for them similar to my previous FPGA projects and MIPS processor. We used error flags and an overall test status bit to indicate when errors would occur, to help automate testing and assist us in our future bring-up plan. During this time, I also worked on developing a 10-page documentation sheet, including block diagrams on clock synchronization, module implementation, submodule design, and waveform results from my shift register tests. This documentation will act as a baseline for the Verilog design I will work on over the Summer for the top-level Backdoor SPI module.

Near the end of the semester, we finished the first iteration of our design document, which encompasses our design plan, testing plan, and much more context to our project. During the week before finals, we presented our project to a faculty panel of three professors at Iowa State University. This was an excellent opportunity to summarize our project in a concise 15-minute presentation, and both of these documents can be accessed on the Overview tab of my website, or our senior design website, which is also linked.

**Semester Two**

In the second semester of senior design, we were ready to implement our design using open-source tools after we took a semester to design our system and test plan. For me, this involved designing the external SPI interface and interfacing with two other modules in our system, the Wishbone Test Module and the DSP Accelerator module. I was required to design and verify functional Verilog modules and testbenches to guarantee that these two modules would interface with the SPI communication bus as expected. Multiple waveform results can be seen in the Appendix of our design document with functional verification waveform results. When this was complete, we integrated all of the other module designs and the SPI interface into the top-level wrapper design, which included custom layout cells using Magic, functional Verilog designs, and pre-hardened open source SRAM modules that would work functionally with the 130nm Skywater fabrication process for this open-source, sponsored project.

The largest roadblock in the second semester of senior design was the synthesis and place and route steps that were ran while utilizing OpenROAD, the open-source tool that would combine Verilog synthesis, macro placement, place and route, and STA all in one tool flow. We had multiple issues synthesizing Verilog and getting power pins to connect to the premade macros such as the prehardened SRAM or custom 2 input NAND cell that we created utilizing Magic. This was critical to solve because without fixing LVS issues which flagged that our power pins were not hooked up, we would not be able to power parts of our design for validation post-fabrication. We solved most of these issues by looking on the public Slack forum for eFabless and online at other projects, which was available to us due to the open-source nature of the project.

Eventually we were able to get all of our individual design modules hardened in the top-level wrapper, in which we were able to verify and view each of these design modules together using another open-source tool, KLayout, which can be utilized to view GDSII files after we completed place and route with the OpenROAD toolchain. It was very gratifying to see all of the hours and work laid out in front of us, which will eventually be fabricated into a real chip for free! After our top level design had hardened and passed DRC and LVS checks, we had to run the eFabless submission precheck tool, which was a separate git repository which would cross check multiple aspects of our design to verify that it was ready to submit for the next open-source MPW Shuttle submission. This precheck included multiple analysis points such as DRC and LVS checks to verify our layout and netlists matched, alongside other more basic tests such as verifying that our README had been updated and our source Verilog code contained a unique design that was not just a copy of the example code. After passing precheck, we had satisfied the first deliverable requirement which was to submit a functional design to our client/advisor which would pass precheck for a submission ready design.

After our design passed precheck, our focus switched mainly to documenting all of our work in the form of a user guide, which overviewed our design process utilizing all of the open-source tools throughout the year we spent on our project. We also created a bringup plan, which a future team can utilize to validate our design after our chip is fabricated, packaged, and delivered after another MPW shuttle submission is opened, expected as of now April 2024. Finally, we created a poster, presentation, and updated design document for a industry panel presentation, where we overviewed our implementation and testing procedures as outlined above.