# ECE 497: Special Project
# Weekly Report
## Week 01

Alexander Lukens  Karl Hallsby

Illinois Institute of Technology

January 28, 2021

# Software

- Set up a git repository, remotely hosted on GitHub.
- Wrote basic `shell.nix` file to work with lorri and direnv to ensure environment consistency between machines.
  - This `shell.nix` file can be converted so that we can build our project too, if we so wish.
- Developed Beamer-derived LaTeX document class for weekly slides.
- Read through several sections of ChipYard's documentation.
- Found and read ChipYard's IEEE paper.

# CORE-V CVA6

- Depends on verilator ($\geq$ 4.002, but not 4.106 or 4.108)
- Can use `spike` to resolve instructions to mnemonics
- Can run userland applications
- Supports emulation of the Genesys 2
  - Xilinx FPGA
  - Part Number XC7K325T-2FFG900C
- Can flash to FPGA as well

Overall, this is a "complete package" for a *single* RISC-V CPU. It has components and peripherals already designed and ready to be simulated/flashed to an FPGA.

# ChipYard Amid et al. 2020

- A platform for designing, simulating, and implementing RISC-V hardware systems using open-source libraries
- Supports three main RISC-V core designs:
  1. Rocket Core
  2. BOOM (Berkeley Out-of-Order Machine)
  3. CVA6 Core
- It seems that the Rocket Core is the most uniformly supported (SiFive-blocks library provides several system components intended to be used with the Rocket Core)
- `fpga-shells` are used to reduce the number of wrappers for physical devices.
- FPGA prototyping is supported
  - The specific FPGA board Alex already owns is explicitly supported (Xilinx Arty 35T).

# Toolchain

1. `riscv-tools`
   - Includes compiler, assember, ISA simulator (`spike`), Berkeley Boot Loader, and a proxy kernel.
   - Many of these have been upstreamed to Linux, GNU binutils, GNU coreutils, etc.
   - Included in ChipYard for consistency and usability.
2. `esp-tools`
   - Fork of `riscv-tools`
   - Intended to work with Hwacha non-standard RISC-V vector extension.
   - Can demonstrate how to add additional RoCC accelerators to spike and higher-level toolchains.

# Simulation

Supports several simulation platforms:

**Verilator** An Open-Source platform for simulating RTL logic, maintained by Veripool.

**FireSim** Intended for advanced simulation and is to be used on AWS cloud instances. Provides comprehensive I/O simulation, including timing-accurate DRAM, Ethernet, etc. simulations. Funding would be required if we intend to use this simulator.

- We anticipate that the Xilinx Vivado suite should provide sufficient FPGA simulation for our needs.
- Provides instructions for complete simulation using Verilator, including waveform generation. ECE Department Saturn Server provides tools that can be used to analyze waveforms (CosmoScope)

# RTL Generators

- All generators are written in Chisel.
- Given input parameters, Chisel outputs RTL-Verilog, which is then simulated.
- Some (if not all) generators can be mixed-and-matched, allowing for complete designs using separated components.

# Documentation

- The documentation for ChipYard is very thorough.
- Gets into enough detail for it to walk us through some steps.

# Our Goals

1. Build a RISC-V chip using ChipYard.
2. Implement prototype on Arty A7 FPGA.
3. Run meaningful program on prototype.
4. Analyze performance of design using various tests.
5. Alter Core design to determine how component functionality impacts performance.
6. Attempt to boot a minimal Linux on our programmed FPGA.

# References

Alon Amid et al. "Chipyard: Integrated Design, Simulation, and Implementation Framework for Custom SoCs." In: *IEEE Micro* 40.4 (2020), pp. 10–21. ISSN: 1937-4143. DOI: 10.1109/MM.2020.2996616.