EE/CprE/SE 4910 WEEKLY REPORT 5

3/7/2026 – 3/20/2026

Group number: SD26-10

Project title: Open-sourced Radio Microcontroller for Fabrication

Client &/Advisor: Henry Duwe duwe@iastate.edu

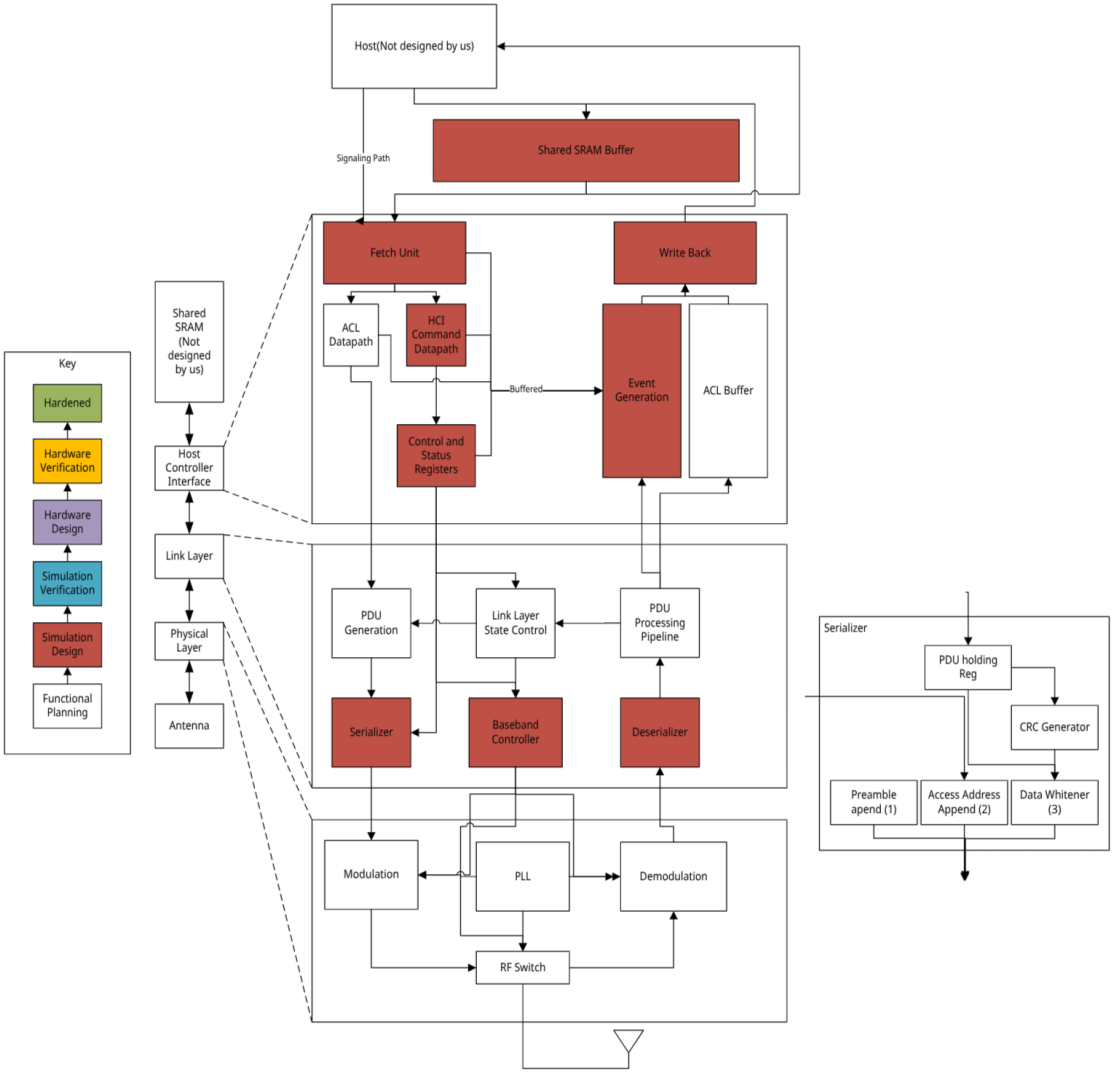
Client/Company/Organization: ISU; ECPE

Team Members:

- ***Zane Salti***
- ***Jackson Doyle***
- ***Daniel Pytel***
- ***Tyler Bibus***
- ***Aiden Han-Lindemyer***
- ***Parker Pederson***
- ***Akash Gojuru***
- ***Seth Klaassen***

Weekly Summary:

Top Level Status:

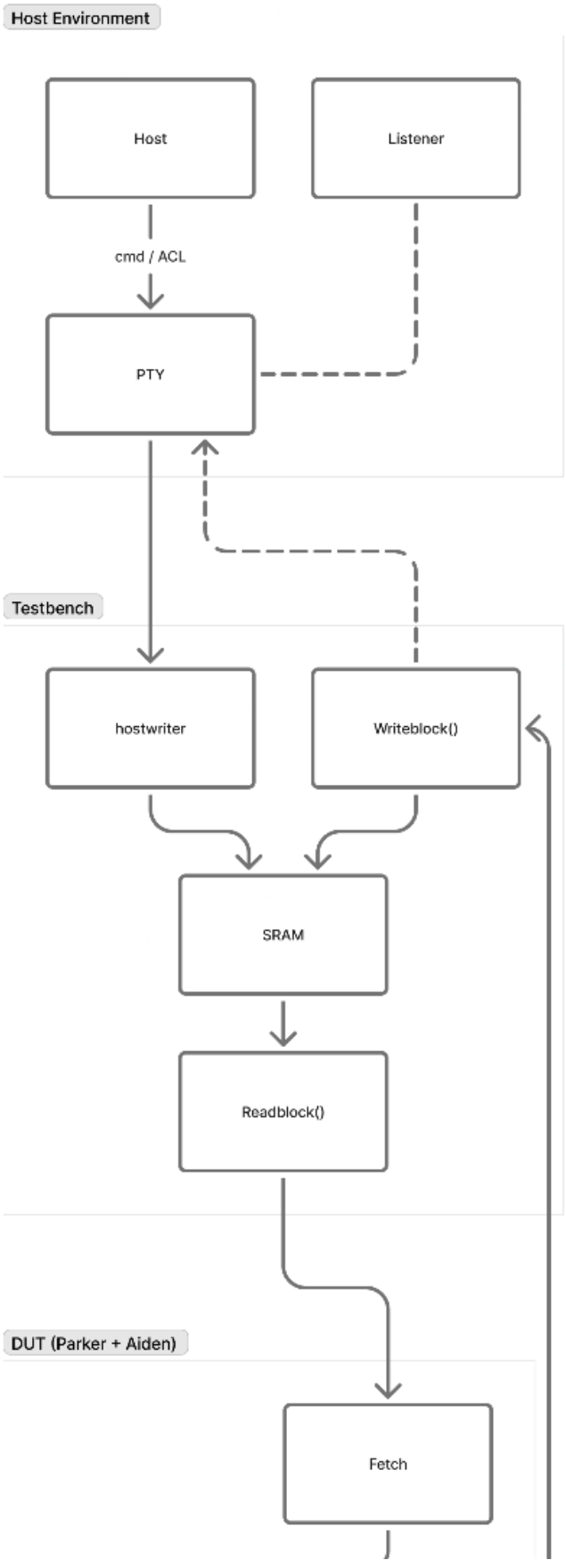


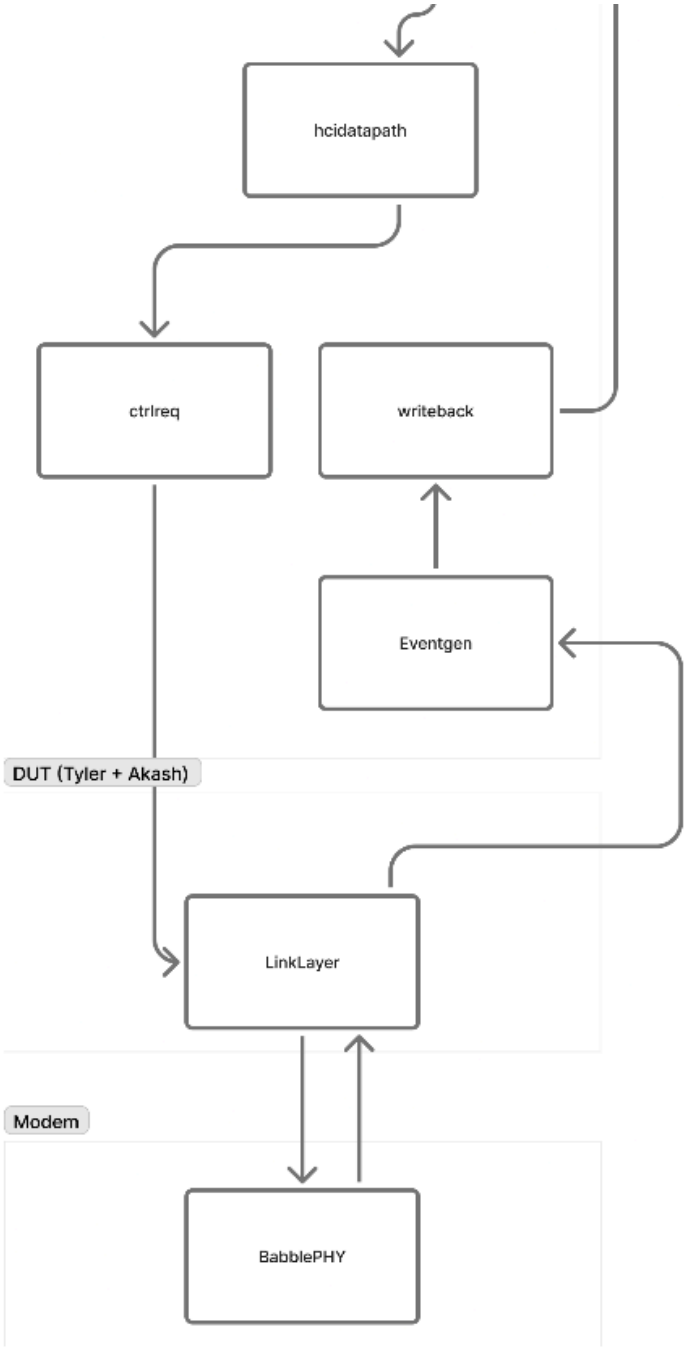
Currently most of the digital segments of the project are being simulated–which are the blocks in red.

Link Layer + Host Controller Interface Team:

During the team meeting the digital team expanded the simulation plan to cover development. Given there are four members and two sides of the simulation the plan is to split the work into two teams, one working from the host-side down and the other team from the physical layer up while meeting in the middle. Currently there is solid development of the host interface with a reasonable amount of the simulated design–which is being simulated to implement and confirm logic–while the interface between the link layer and physical simulation is still obfuscated and being

developed. This simulation should provide a solid logical foundation for the transition into digital logic and HDL as well as be a useful tool for further testing and modification. The figure below is the current architecture of the simulation framework, large blocks like the link layer need to be decomposed into smaller pieces for useful simulation.


Host Environment
Host
Listener
cmd / ACL
PTY
Testbench
hostwriter
Writeblock()
SRAM
Readblock()
DUT (Parker + Aiden)
Fetch



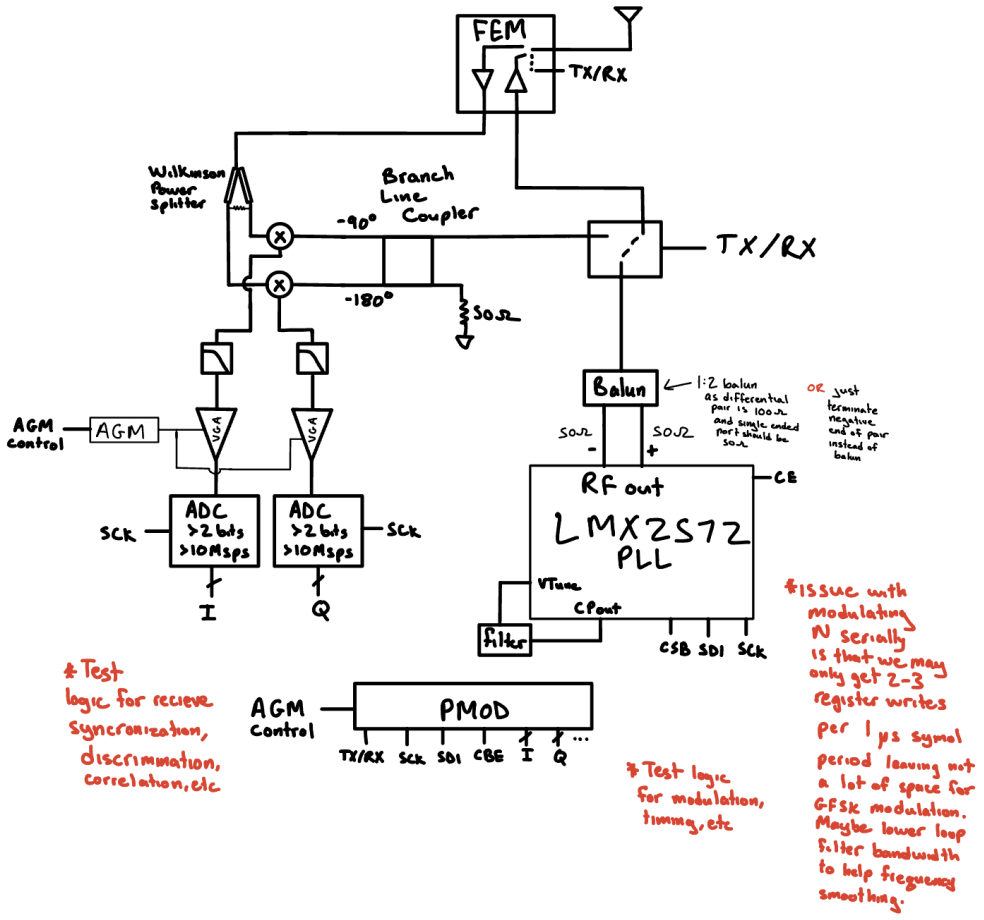
Past week accomplishments

- Zane Salti:

  Explored different potential methods for hardware testing of the Link Layer with a physical radio. The different methods have different levels of how accurately they replicate the target IC PHY design interface with the digital side or Link Layer.

One way of testing is to use the nRF52840 development board (that has been previously used) and have it be interfaced with an FPGA containing the Link Layer and Host management core. The nRF52840 can be programmed such that it just takes in and gives out bitstreams for transmitting and receiving and that it only interacts with the radio peripheral and doesn't use any higher level abstractions or libraries. The device would then basically be used just for its radio peripheral. The testing prospects of this would be that packet formation (barring the preamble –which the peripheral always includes, to some degree), whitening, CRC, the Link Layer algorithm, and the timing of the Link Layer can all be tested for.

A more intimate level of testing would be to have the FPGA interface with a fractional-N PLL (as part of a PCB transceiver) to more closely mimic what the digital side or Link Layer would have to interact with with the actual target IC PHY to be designed.



This could also allow for the testing of digital demodulation (which could include synchronization, correlation, etc). The earlier option doesn't expose raw IQ samples such that it allows us to test real time digital demodulation logic. The LMX2572 PLL chip was selected as it has an integrated VCO and a FSK mode which allows you to change frequency on demand. An issue with this option, though, is that all frequency changes have to happen through its SPI interface, which has a maximum of 75 MHz on the clock. You use the SPI interface to write to the frequency deviation register, but the whole transaction takes many cycles (even if you lock the address with block programming) such that within a single symbol transmission period, you may only have time to do a few new frequency deviations (R/W bit + 7 bit address + 16 bit data = 24 bits and for block programming once the address is locked it's 16 bits. 75 cycles or 75 bits in a 1 us symbol time so 4 new frequency

deviations with block programming in a 1 us symbol time, at 75 MHz). We want to be able to do gaussian frequency shift keying (GFSK) where the frequencies shift in a smooth gaussian way from the negative deviation to the positive deviation (and vice versa) instead of just hard or coarse frequency changes. Having only four frequency deviations in a single symbol time may give us too coarse of a frequency shifting than desired for GFSK.

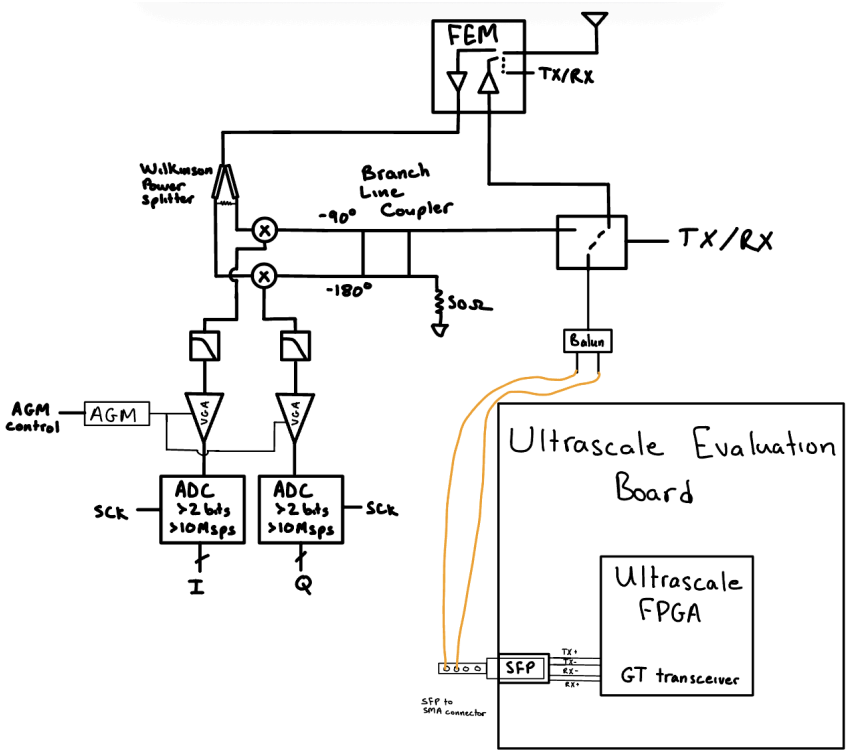
The next hardware testing option, and what seems to be the closest “model” of testing, may sound strange. Some FPGAs are known to have high speed transceivers for off chip communication. For example, some AMD FPGAs have GT (GTX, GTH, GTY, etc.) transceivers. While these transceivers are supposed to be used to send high speed serial data (up to 12 Gbps in some cases), we can try to have it instead just send alternating ones and zeros, a clock signal pattern. We have the frequency of the transceiver (which can be changed at runtime) set to the center of a BLE channel and then from there, we can do frequency deviations by taking advantage of the fractional-n PLLs found within the GT transceivers of certain AMD UltraScale FPGAs. The SDM0DATA and SDM1DATA bus/port is exposed allowing us to change the feedback divider at runtime.

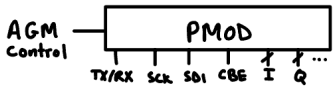
Table 2-14: **QPLL0/1 Ports *(Cont’d)***

Port	Direction	Clock Domain	Description
PMARSVD0[7:0]	In	Async	Reserved.
PMARSVD1[7:0]	In	Async	Reserved.
UltraScale+ FPGAs Only			
SDM0RESET/SDM1RESET	In	Async	This active-High port resets the sigma delta fractional divider inside the QPLL0/1.
SDM0DATA[24:0]/ SDM1DATA[24:0]	In	Async	Input to set the numerator of the fractional part of the feedback divider. Bit [24] is unused.
SDM0WIDTH[1:0]/ SDM1WIDTH[1:0]	In	Async	Input to set the denominator of the fractional part of the feedback divider. 00: 24 01: 20 10: 16 11: Reserved
SDM0FINALOUT[3:0]/ SDM1FINALOUT[3:0]	Out	Async	Reserved.

This manipulation of the feedback divider is more close to modeling how we intend to use the IC PHY that we want to design than any other option discussed so far. We can then try to have our Link Layer + Host run on the very same FPGA and have it be able to directly modulate on the FPGA without being limited by bandwidth and external communication interfaces that would prevent us from doing enough frequency changes in a symbol’s time to get a correct gaussian frequency shift. There also are currently UltraScale boards within ISU (but don’t know if they are particularly available for us as of now).

Considering that the transceiver outputs a digital signal, we would need to use a bandpass or lowpass filter to remove the higher order harmonics.





A differential pair is outputted by the transceiver. We can either ignore the negative side (by shunt terminating it), or we could use a balun. Considering that the signal has to go through the evaluation board and through an SFP connector, it doesn't seem like a bad idea to actually take advantage of the differential pair and balun it.

Made progress on the serializer hdl design.

Had a subgroup meeting with Dr. Neihart. For the PLL, he recommended using a quadrature oscillator or extending the ring oscillator VCO stages and tapping into a stage that is 90 degree shifted from output. He mentioned that deriving a quadrature signal from the PLL via using a by 2 divider (and running the PLL at twice the frequency) isn't a good idea for performance. He emphasized having a holistic understanding and modeling of the system.

- Jackson Doyle: Continued training with tool flow and met with Dr. Neihart. Current ideas for design and architecture of the phy layer are insufficient.
- Daniel Pytel: Continued toolflow training, gained access to more resources. Missed out on Neihart meeting but got caught up on info.
- Tyler Bibus: Added functional requirements and constraints to design documents. Created simulation architecture and plan seen in report with Parker.
- Parker Pederson: Worked on the hci simulation starting at shared sram buffer and moving down into command datapath and event generation. This mostly consists of creating

infrastructure so that adding future hci commands to the list of responses is simple to complete. The simulation currently responds to reset and list acceptable commands.
- Aiden Han-Lindemyer: Outlined a structured register map and API-based access approach to improve modularity and scalability.
- Akash Gojuru: Completed the setup of BabbleSim and verified that it is working correctly by successfully running the Bluetooth examples from the Zephyr project. I began implementing PDU generation in C as the next step.
- Seth Klaassen: had a meeting with professor Neiheart where it came to our attention that we needed to go to the bluetooth datasheet and create a system level model of what specifications our various components needed to meet, with this we can make decisions on what designs we need to meet. He also gave great advice on different filtering, mixing, and VCO design approaches that we could look into once we finish our system level model.

Pending issues

- Zane Salti: Need to see if UltraScales are available for us. Need to finish serializer.
- Jackson Doyle: Need to address design requirements of Bluetooth LE, and the process to implement it at a system level. “Take a step back” and re-evaluate the scope of our project and the system level approach we choose to take. Continue training in the tool flow via chipforge meetings to become better at it for when it becomes necessary to use it.
- Daniel Pytel: Neihart meeting called into question current PHY scope. Toolflow still has many issues.
- Tyler Bibus: In BabbleSim where the phy layer ends and where the link layer starts is difficult to discern. Need to stay on top of the design document and reports to ensure full team contribution.
- Aiden Han-Lindemyer: The global variables shared across files using extern make the simulation harder to maintain and extend.
- Akash Gojuru: Still working on understanding BabbleSim particularly how the physical layer operates and how tests are conducted within the simulator.
- Parker Pederson: The common status registers are currently implemented as global variables that get externed around the simulation. Should probably come up with a better plan than that.
- Seth Klaassen: the phy side

Time Tracking

NAME	Hours this Week + Break	HOURS cumulative
Zane Salti	8	37
Jackson Doyle	5	20
Daniel Pytel	4	17

Tyler Bibus	4	20
Aiden Han-Lindemyer	4	20
Parker Pederson	5	23
Akash Gojuru	5	19
Seth Klaassen	4	19

Comments and extended discussion

Plans for the upcoming week

- Zane Salti: Verify advertisement data can be broadcasted using strictly the nrf52840's radio peripheral, see if ultrascales are available for us to use, work on serializer, and get started with component selection and/or design for PCB transceiver (the second one I detailed earlier, assuming ultrascales are available for us, and maybe just the transmit side of it for now).
- Jackson Doyle: continue training with tool flow. Continue high level system parameter analysis and proceed onto allocation and architecture development from those parameter values.
- Daniel Pytel: Start implementing information from Neihart, modify PLL structure
- Tyler Bibus: Get interface defined between BabbleSim and current simulation structure. Decompose Link Layer simulator into components that can be more easily converted to digital components. Update design document based on advisor feedback. Begin early documentation for the simulator with usage and interfaces.
- Aiden Han-Lindemyer: Begin a centralized register structure with read/write access functions. Refactor existing HCI command handling code to interface and verify correct behavior
- Parker Pederson: Add support for more HCI commands and work on the underlying simulation architecture.
- Akash Gojuru: I will focus on implementing the current top-level design Link Layer in C and integrating it with the physical layer using BabbleSim. This will involve enabling communication between the Link Layer and the Physical layer.
- Seth Klaassen: work on system level model

Summary of weekly advisor meeting

The team reviewed the top-level block diagram, design document, and PHY status. For the top-level diagram, the group agreed to rotate it horizontally for readability, use paler colors in the key, remove the serializer decomposition (moving each module to its own page), and have the PHY team update their block to match the current design. The design document discussion focused on restructuring functional requirements to lead with core aspects rather than documentation, making requirements quantitative and exhaustive, listing specific Bluetooth sub-standards in the requirements/constraints, adding the Chipforge simulation environment as a formal deliverable with testability specified, and ensuring requirements and deliverables are independently verifiable by someone outside the team. Engineering standards should list only general standards, with sub-standards called out in the requirements section. Finally, the PHY team needs to coordinate with Ibrahim regarding missing symbols for PLL simulation.