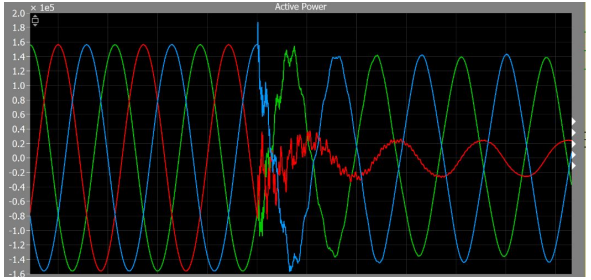
Lightning Talk: Testing-Design

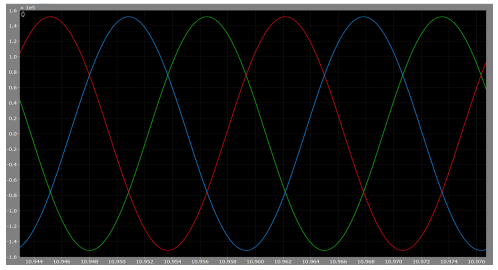
Group 9
Chye Stecher, Josh Vrenick, Keegan Kraft , Matthew Dobrzynski, Taylor Semple, Anthony Ruffalo

Units and Interface

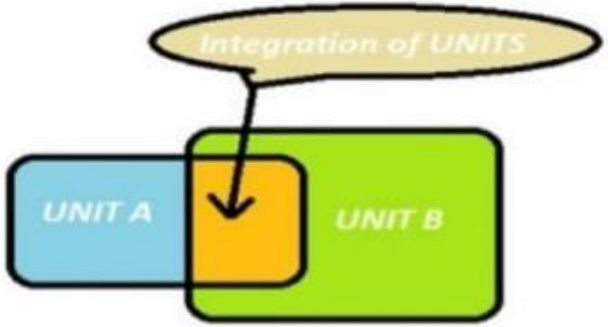


- The units we are testing/observing are voltage and current.
  - Their output is in the form of waveforms.
  - In our testing we are observing how they change when various types of faults are simulated.

- Our interface is a PLECS Model. This model is essentially a giant block diagram of a miniature power grid.
  - Within this model we can add a “fault block” that simulates a fault along our transmission lines.
  - This block can be set up to simulate all of the different types of faults.

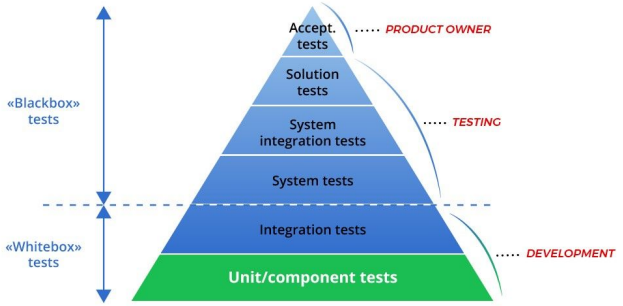


Integration



- Our integration paths are going to be our RT Box sending a signal to our PLECS model which then feeds data to a csv file that our python code will be able to use to determine if there is a fault in the signal.
- To test that the RT Box is properly sending a signal to PLECS we can make a basic model that shows us the signal being used in PLECS and if it matches what the RT Box is sending.
- Then, to make sure the PLECS model is sending the correct information to the csv file we will just have to use analysis tools in PLECS to make sure the information in PLECS matches what is on the csv file.
- Finally, to test the python code, we would run the whole project together and if it can accurately detect faults then we know that our python code is running properly and our project can detect faults.

System Testing



Unit Tests:

- Testing of the specific faults, their detection, and the circuit breaker switch.

Interface Testing:

- Testing between PLECS, python, C, excel, and our real-time simulator (RT box).

Integration Testing:

- Include the final runs of our testing. This will include all of our softwares coordination of the C code into the real-time simulator with the data that we previously collected from PLECS.

Regression

- We’ll be using GitLab as our Version Control software, which will help save and share our project at different points to ensure that if needed, we can revert our work.
- We’ll also use CI/CD testing to make sure new features don’t break old functionality.
- Branching out our work means that we can keep a stable and working version of our system while working on experimental features.

Acceptance and Results

- Demonstrate optimal performance through timing and satisfactory operation; operates within 2 Hz, is able to accurately detect distance of fault, and type of fault which occured.
- Does not have “false positive” readings during normal operation of the system.
- Shall provide statistics of the system as well as show our client how the system performs.
- Results will come in the form of opening a circuit breaker what a fault is detected by our system and accurately relaying the location and type of fault. All correct actions performed by neural network.