**Parts of the design:**

For this checkpoint, we worked toward implementing advanced features and fixing bugs in our baseline checkpoint.

Some things we implemented were:

1. Branch Prediction
2. Simple BTB
3. Karatsuba Multiplier
4. Basic Divider
5. Instruction prefetching
6. L2 Cache

**Major component divisions:**

1. Branch Prediction: Worked toward creating a local branch history table. It stored the past 6 commits of the branch, and created a dynamic 2-bit prediction scheme.   
   Done by Geitanksha
2. Branch Target Buffer: Used a DM cache to store 30 bits of tag comparing PC to the incoming PC in the fetch stage, and then if valid and is not being waited upon by another instruction, sends back the target address that the PC must jump to. Done by Geitanksha
3. Karatsuba multiplier: The base case is a 4x4 Wallace tree to do easily parameterized multiplication. Done by Quinn
4. Basic divider scheme. Done by Quinn

Arbiter to choose whether the multiplier or divider should run at a given time, and to send a stall signal to the main CPU pipeline modified pipeline to allow for the CPU pipeline to grab data from the multiplier when necessary.

1. L2, 64 set cache: Modifications were made to the provided cache to make a L2 cache 64 set cache, and connected it up to the L1 Cache. Done by Johan
2. Verification and debugging: We faced a lot of small issues in our code, several typos, and logical misconceptions. We all worked on it to fix those bugs.

**Functionality:**  
  
We implemented all the design choices we wanted successfully.

**Testing:**

For testing, we started out by making sure that the base processor worked perfectly on all the test files given by the TAS and used Verdi to understand all the signals, check if the registers stored the correct value at the end, which covered most instructions – this time we did not fail the advanced codes for the competition. Then, we slowly started adding each of our implementations one at a time to determine whether they all worked with the given code, and each of us made testbenches to test the functionality of our advanced feature separately.

We also wrote simple assembly to test every instruction that we had implemented, logical, bitwise and arithmetic. We put predefined values in with no NOPs and checked if the result we expected in the registers was achieved. We also tested every version of the Branch scheme.