FPGA Project Grading Information

Spring 2023

Project requirements and grading will be different this semester than in recent semesters. We would like to be as fair as possible and to give you an experience that closely matches what you will find in industry. Therefore we will assign your project grade using a *figure of merit (FOM)* computed based on your design. The FOM in this case will be a combination of your processor's maximum clock frequency (F_{max}), average cycles per instruction (CPI), and cost:

$$FOM = \frac{F_{max}}{CPI \cdot cost}$$

F_{max} is the maximum clock frequency at which your processor correctly passes our test benchmarks. CPI is the average cycle per instruction for your processor running our benchmarks, as described below. Cost will be based on which and how many FPGA elements your design includes, as detailed in the post place synthesis report. It will be computed using a script that processes your report and forms a weighted sum of all the elements in your design. The weights correspond to the relative chip area that each element occupies. *We will not, however, include the cost of FPGA block RAM memory elements.*

We include *cost* in the FOM because we want to encourage simpler designs. Experience shows that debugging times increases very quickly with design complexity, and in the past many students have had trouble debugging complex designs by the end of the semester. You will have a much better experience and learn more by keeping your design simple. On the other hand, we would like you to explore options for improving performance. But while doing so you should be mindful of whether or not extra complexity to improve performance is worth the extra cost. Therefore if you are tempted to add complexity, make sure that it will actually improves the FOM.

The lab grading break down will be as follows:

50% Correctly functioning 3-stage (at least) processor, without regard for performance and cost.

35% FOM optimizations.

5% Checkpoints.

10% Final report.

Once you have a functionally correct processor, the next steps will be to modify it to achieve a higher FOM and thus achieve a higher grade. We will give you suggestions on how to maximize the FOM (and therefore your grade), but it will be up to you to make good decisions as to how to optimize your design.

For purposes of computing the FOM, we will rely on two counters that you are required to implement as described in the project specification document. One counts cycles and the other counts instructions. Our benchmarks automatically clear the counters and reports the counts after the run. In addition to using our benchmark programs you are welcome to write your own. We will provide a script that takes as input the two counter values, your F_{max}, and a report (.rpt) file, and outputs the CPI and the FOM values. For final checkoff we will use the F_{max} value from running your processor on the FPGA board and will use the "post_place_utilization.rpt" file to determine the cost. However, since place and route might take significant time as your design grows in complexity, to speed up your design space exploration, you might want to use the "post_synth_utilization.rpt" file for costs, estimate F_{max} based on your synthesis target and available slack from the timing report, and use simulation to get the cycle and instruction counts. However, becasuse the simulator is significantly slower than running on the actual FPGA, this approach will only work for small benchmark programs.

It should go without saying that to improve the FOM, you will need to increase F_{max}, decrease CPI, decrease cost, or some combination of these. You should be able to improve F_{max} without substantially changing the microarchitecture of your design by shortening the critical path. Of course, after you improve one path, you might then want to optimize the next longest. Improving a path could come down to how you write the Verilog, or you might need to rearrange the logic. Also, obviously, increasing the number of pipeline stages might also shorten the critical path, but such a change could have an adverse effect on CPI and cost; so proceed with caution. In this design, a small Dcache and/or Icache might also help improve F_{max}, if you can find a way to build a cache that is substantially faster than block RAM. To improve CPI, you might consider using branch prediction. Also, you might even want to consider mechanisms that would bring CPI below 1 — however, these mechanisms can get tricky to design. If you have other ideas and are unsure if it will help, feel free to talk with us and get our feedback before investing time in an idea that may or may not work out.

For a 3-stage unoptimized design, we expect typical values around: F_{max} = 60 MHz, CPI = 1.5, and cost = 1.5 M, and therefore a FOM of around 25.

At a later date we will post approximate FOM grade targets (what FOM values correspond to project point assignments). Also, we will try to find a way to occasionally anonymously post updates on FOM values that other groups have achieved.