# ARM Pipelined CPU

** Warning – this lab takes MUCH longer than all the previous ones. Start EARLY **

**Introduction**: For this project you are to design a 64-bit ARM CPU with Pipelining. The CPU instructions to be implemented are the same as project 3. Your pipelined CPU will have 1 delay slot after each load and branch instruction, as discussed in class. It is also responsible for covering data forwarding. Note that you should carefully consider how data forwarding and branches will interact. You will use your three previous projects (the file register, the ALU, and the single cycle CPU) so you will need to have these fully functional. As with project 3, we will provide you with sample instruction sequences with which to test your CPU, and you will have to display the modified registers at the end of the execution. Please remember the rules from previous labs as well.

Note that handling the forwarding logic in your design may be complex, and the book has some ambiguities on this topic. However, if you carefully think through the scenarios where forwarding may be required, it should be easy to figure out the necessary logic. Issues to consider:

1.) What happens if an instruction writes back to register 31, which is required to always be 0 regardless of what is written to that register?
2.) The ALU and the Memory can each provide values to be written to the register file, and thus may both be sources of forward information (though you may cleverly be able to combine some of this).
3.) The ALU, branch logic, and memory can all need values from the register files, and thus may need to have values forwarded to them.
4.) Not all instructions write registers, and different instructions have register IDs at different places in the instruction word. These should be carefully considered.

## TURN-IN

For this lab you will turn in the code electronically and demo the functionality of your CPU to the TAs. Note that all of the memories and programs from lab #3 work for this CPU as well (the programs were written to be mostly insensitive to pipelining, though some changes in output will occur).