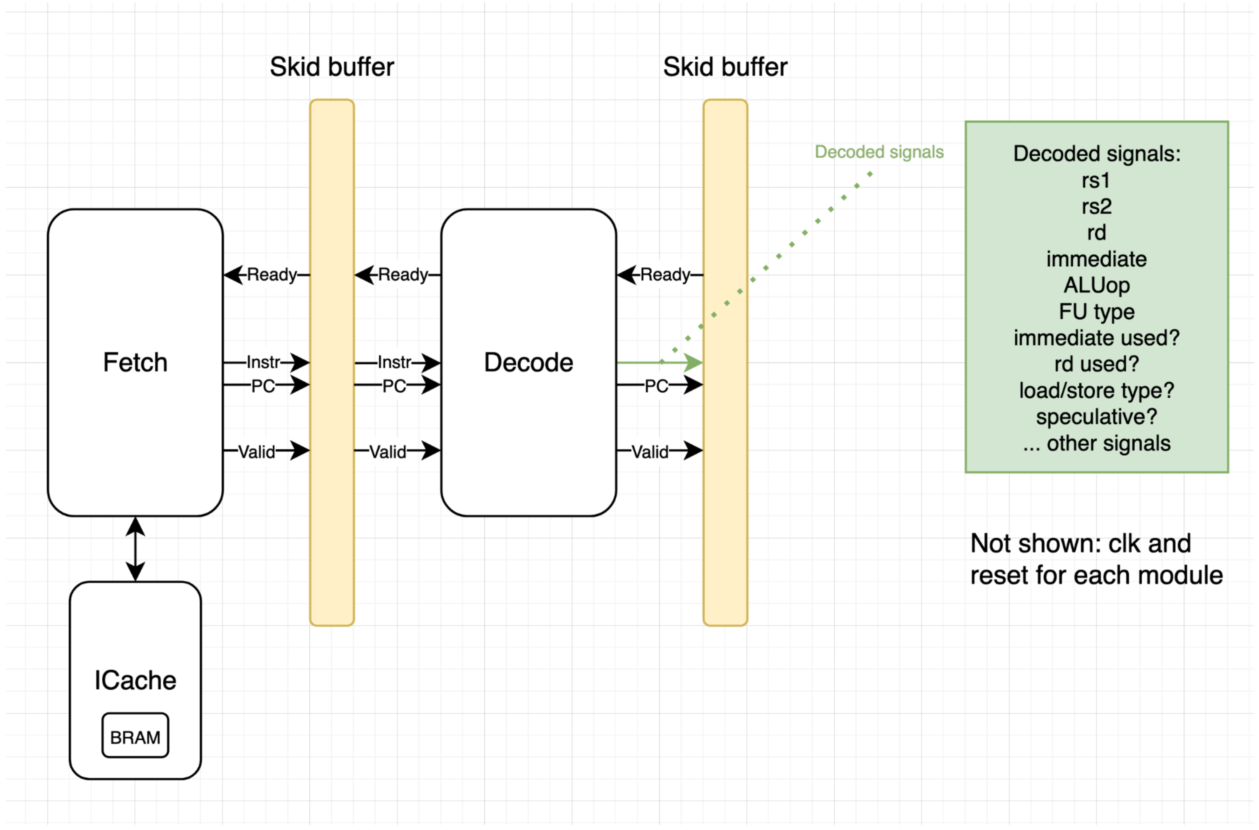
Phase 1- ICache/Fetch/Decode Deadline: Wed., Nov. 6, 11:59 PM

Fall 2025 (Upload it to Gradescope.)

Notes

- You will be implementing these modules and reusing them throughout the period of the project
- You are allowed to use AI tools for writing code as long as you understand what you're implementing and properly indicate what has been written by the tool.
- This phase (and all future phases) can be done in groups. Your group should have been listed under this document: Honors Group

Steps:



Block diagram

- The i-Cache Module:
  a. The i-Cache Module should be implemented as an FPGA BRAM. The purpose is to store the program code that'll be read out from (acting as a Read Only Memory (ROM)).
  b. You can store one instruction (32-bit) per row (cache block size). The total size can be any number (e.g., 2KB).

- Fetch Module:
  a. Similar to the In-Order Pipeline. Fetch one instruction at a time. Be careful on the offset and PC+4.
- Decode Module:
  a. The Decode Module should be completely combinational (apart from pipeline registers).
  b. Decode should describe which functional unit an instruction should be executed in (ex. ALU, load/store, branch unit). More details about the functional units will be provided in the next phase.
  c. Consider the same list of instructions as CA1 (i.e., you only need to support those).
  d. Tips:
     - Don't have too many enumerated cases when decoding
     - ex. ADD and ADDI should be the same

Note: for now, we are not adding the branch prediction logic/BTB. This will be included in Phase 4 if there is time.

What to submit

1. You need to upload all your code to a GitHub repository and share the link on Gradescope (assuming you have already done that). Share the commit number in your report.
2. A short report (a PDF file) that explains how each person in the group contributed to this phase.