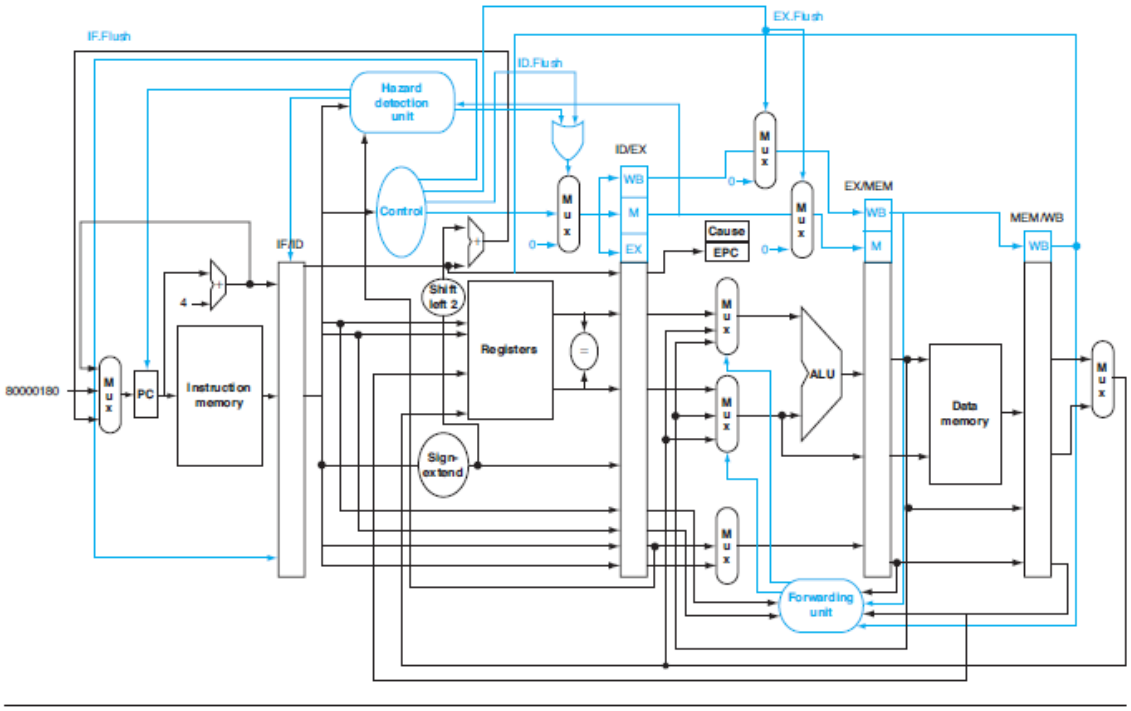
**Computer Architecture Final Project**

**Second Semester, 1396-97**

**Pipeline Processor:**

The purpose of this project is to design a Pipeline Processor, Each step is designed to take one clock cycle. It will be able to handle different instructions, R-type, I-type, and J type and hazards




The Pipeline Processor breaks simple instructions down into a series of steps. These steps typically are the:

1. Instruction fetch step:

   During the instruction fetch step the Pipeline Processor fetches instructions from the memory and computes the address of the next instruction, by incrementing the program counter (PC).

2. Instruction decode and Register fetch step:

   During the second step, the Instruction decode and register fetch step, we decode the instruction to figure out what type it is: memory access, R-type, I-type, branch. The format of these individual instructions is the same as what you learned in class.

3. Execution, memory address computation, or branch completion step:

   The third step, the Execution, memory address computation, or branch completion step functions in different ways depending on what type of instruction the processor is executing. For a memory access instruction the ALU computes the memory address

4. Memory access (or R-type instruction completion step) :

   In load word, store word, R-type, and I-type instructions this is the step where the result from the ALU computation is stored back into the destination register.

5. Memory read completion step:

   This is the memory read completion step. In a load instruction the value of the memory data register is stored back into the register file.

These different steps are all controlled by controller. The controller is a finite state machine that works with the Opcode to walk the rest of the components through all the different steps, or states. The controller controls when each register is allowed to write and controls which operation the ALU is performing.

**Implementation Concerns:**

1-The program should be able to handle following instructions

There are five R-type instructions: add, subtract, and, or, and set less than, **Nor and xor**

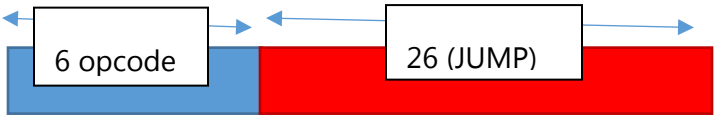
There are three I-Type instructions: load word, store word, and branch

The jump instruction is also supported

Note: All instructions are 32 bit format

For Example:

JUMP INSTRUCTION




Note:

2- Two separate memories are used for the data and the instructions

3- Concern hazard detection

3- **Document** the project and the design process

**Implement the project with c or c++**

**Any duplication from others or the Internet, will have consequences on your score.**