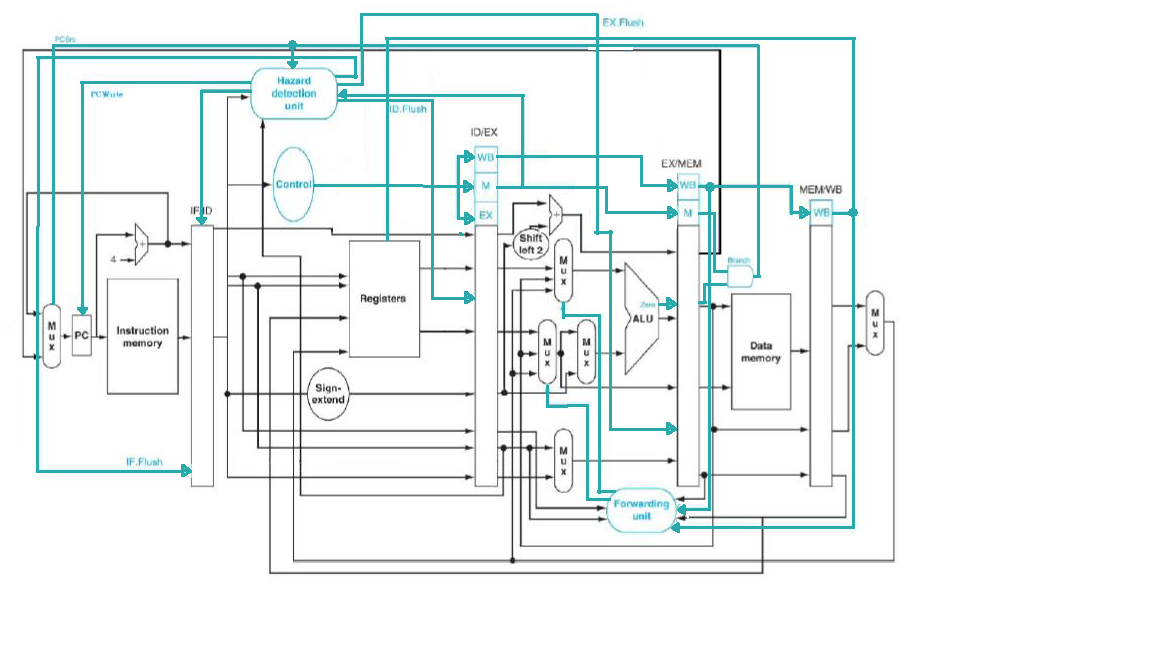
**Computer Organization**

**HDL simulator you used: ModelSim**

**Finished part:**

There are 27 modules used to design this model. The newly added or modified modules:  
- Pipeline\_CPU.v  
- Forwarding\_Unit.v  
- Hazard\_Detection\_Unit.v  
- IF\_ID\_register.v  
- ID\_EX\_register.v  
- EX\_MEM\_register.v  
- MEM\_WB\_register.v

**Architecture diagrams:**

**Hardware module analysis:**

The newly added or modified modules:  
- Pipeline\_CPU.v: The modules that wrap all finished module to gain the wanted result.  
- Forwarding\_Unit.v: Used to detect come condition of data hazard and resolve it using forwarding.  
- Hazard\_Detection\_Unit.v: Hazard detection used to determine whether the current instruction in every pipelined area need to be flushed, maintained, or kept.  
- IF\_ID\_register.v: Used as register between IF and ID stage.  
- ID\_EX\_register.v: Used as register between ID and EX stage.  
- EX\_MEM\_register.v: Used as register between EX and MEM stage.  
- MEM\_WB\_register.v: Used as register between MEM and WB stage.

**Problems you met and solutions:**

* Problem of differentiating IF\_Flush and IF\_Write at the early stage of designing the module due to the similarity of the step needed to be done to accomplish some different wanted result.  
  Solution: By making the register IF\_ID\_register to receive two different signals that each indicate flush and write respectively, the confusion can be evaded.

**Summary:**

Just as every time revising this CPU module design. The biggest thing need to be considered is to be as detailed as possible. By doing the cable step by step and in the way that they are related, we can minimize the error possibility as low as possible. Modelsim will not let out error message if the wire used in the module is not declared. It will be interpret as 1 bit wire declaration, which will be disastrous if the bit number needed is not equal with the one truly needed.