**Report**

* **design changes after phase 1**
* Divide logic of branching into 3 parts instead of one part in phase 1 in this phase we make unconditional jumps in decode stage .conditional jumps in execution stage and (RET, RTI) in the memory stage.
* Handle problem of 32 bit instruction from phase 1 and use 16 bit instruction
* Finite state machine to handle interrupt signal
* **Hazards**
* **Structural hazard**

Handle Structural hazard by duplication and use memory for instruction and memory for data (Harvard).

* **Data hazard**
* Using data forwarding unit to deal with data hazard to pass data from Pipeline to be used in the same clock cycle instead of waiting data to be finished after write back stage
* Using **H**azard **D**etection **U**nit to solve pop instruction case, in this case if the pop instruction in execution phase and followed by instruction in the decode stage deal with the popped data , **HDU** solves this case, we don’t need to use HDU in load use-case because we insert bubble after the instruction which complete load instruction (32 bit) we use 16 bit instructions.
* **Control hazard**

Using Static Branch Prediction always predict that branches will be untaken and flush instructions in pipeline if branch actually taken