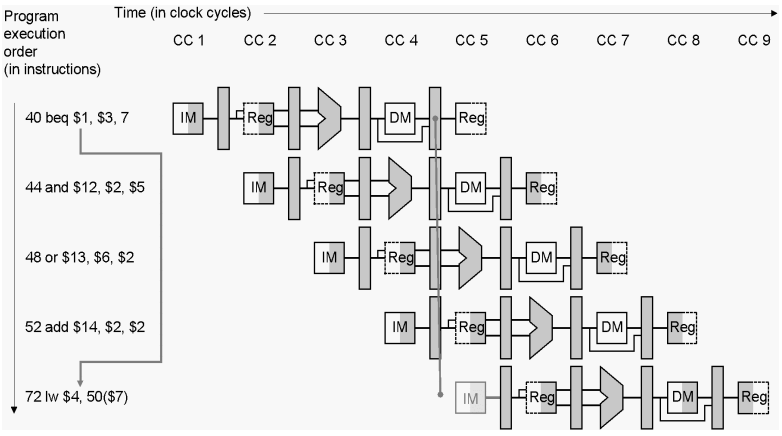
Branching and Jumping in a Pipeline

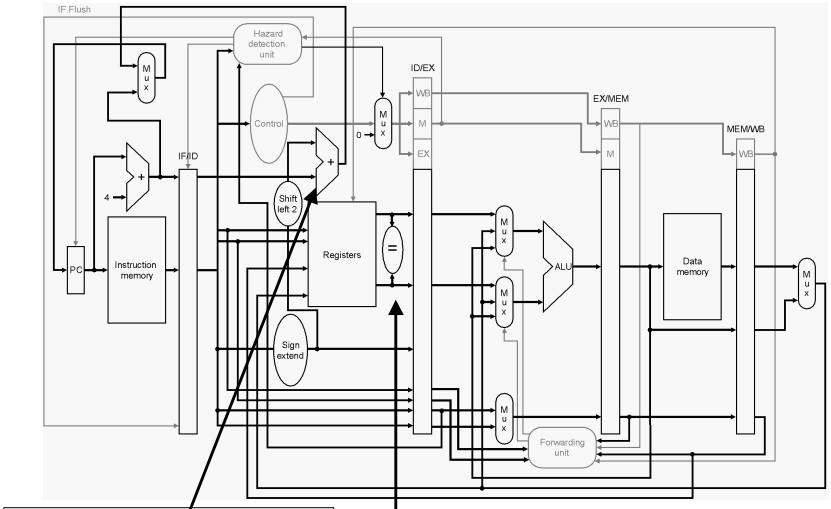
The decision to change the next instruction (branching and jumping) can possibly cause launched instructions to be flushed

Branching Hazard

- A normal pipeline assumes “branch not taken”
- But when there is a branch, instructions must be killed



Branch Hazard Detection Logic



Reducing the Delay of Branches

- A fast equality test is used to compare the register values involved in the branch condition rather than a general comparator
  - It does not slow down the ID stage much, if at all, which maintains the cycle time
  - The fact that it is an equality test is reflected in the ISA
- This early “execution” of the branch means that only the instruction following the branch is in the pipeline
  - We only need to flush one part of the pipeline
- The branch delay is only 1 cycle! (c.f. 4 cycles on slide 2)

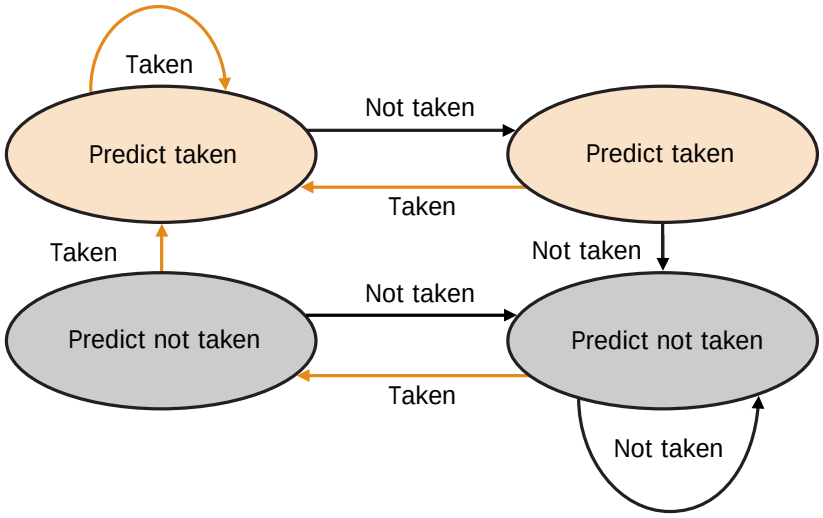
Dynamic Branch Prediction

- Assuming that a branch is not taken is a crude form of prediction
  - If 50% of branches are taken, we will be right 50% of the time
- To do better than this, we can examine past behavior of the branch to hint what will happen this time
- We maintain a small *branch prediction buffer* or *branch history table*
  - The table is indexed by the low order bits of branch instruction addresses (why not the high order bits?)
  - Each entry is a single bit which tells us whether the branch was taken
- Improves accuracy to 80-90%

Going Loopy

- Loops cause problems with the previous scheme
- The first and last iteration of a loop will be mispredicted
- If the loop has been executed earlier, then the first time we encounter the branch instruction, we will predict that it will not be taken
- On the final iteration, we will predict that the branch will be taken
- To handle this case, we use more than 1 bit of state in our branch prediction buffer

2-bit Prediction Scheme



Delayed Branches

- Make the control hazard an architectural feature
- The instruction following a branch is always executed
- The compiler or assembler has to find an instruction to fill this slot
  - If none can be found, a NOP has to be inserted
- The instructions scheduled into the delay slot must
  - EITHER always be executed whether the branch is taken or not
  - OR have no side-effects
- Less popular now since longer pipelines and multiple instruction issue mean the single delay slot does not help as much
- Dynamic predictors have increased in popularity as transistor density has increased

Playing the Slots