In-order vs. Out-of-order Execution

In-order instruction execution
• instructions are fetched, executed & completed in compiler-generated order
• one stalls, they all stall
• instructions are statically scheduled

Out-of-order instruction execution
• instructions are fetched in compiler-generated order
• instruction completion may be in-order (today) or out-of-order (older computers)
• in between they may be executed in some other order
• independent instructions behind a stalled instruction can pass it
• instructions are dynamically scheduled

Dynamic Scheduling

After instruction decode:

• check for structural hazards
  • an instruction can be issued when a functional unit is available
  • an instruction stalls if no appropriate functional unit

• check for data hazards
  • an instruction can execute when its operands have been calculated or loaded from memory
  • an instruction stalls if operands are not available

If both criteria satisfied, an instruction is said to be ready
Dynamic Scheduling

Out-of-order processors:

• don’t wait for previous instructions to execute if this instruction does not depend on them

• 2 situations in which ready instructions can execute before earlier instructions
  1. when go around a load instruction that is stalled for a cache miss:
     • use lockup-free caches that allow instruction issue to continue while a miss is being satisfied
     • the load-use instruction still stalls

Dynamic Scheduling

Out-of-order processors:

2. when go around a branch instruction:
   • the instructions that are issued from the predicted path are issued speculatively, called speculative execution
   • speculative instructions can execute (but not commit) before the branch is resolved
   • if the prediction was wrong, speculative instructions are flushed from the pipeline
   • if prediction is right, instructions are no longer speculative
**Speculative Execution**

Instruction *speculation*: executing an instruction before it is known that it should be executed

- all instructions that are fetched because of a prediction are speculative
- in-order pipeline:
  - branch is executed before the path
  - i.e., by the time the path is executed, it is no longer speculative
- today’s out-of-order pipeline:
  - path can be executed before the branch
  - but not committed!

In addition, executing speculative instructions:

- must be safe (no additional exceptions) or must handle the exceptions after the instruction is no longer speculative
- must generate the same results