

## In-order vs. Out-of-order Execution

### **In-order instruction execution**

- instructions are fetched, executed & committed in compiler-generated order
  - if one instruction stalls, all instructions behind it stall
- instructions are **statically scheduled** by the hardware
  - scheduled in compiler-generated order
  - how many of the next  $n$  instructions can be issued, where  $n$  is the superscalar issue width
    - superscalars can have structural & data hazards within the  $n$  instructions
- advantage of in-order instruction scheduling: simpler implementation
  - faster clock cycle
  - fewer transistors
  - lower power
  - faster design/development/debug time

## In-order vs. Out-of-order Execution

### **Out-of-order instruction execution**

- instructions are fetched in compiler-generated order
- instruction commit is in-order (today) or out-of-order (older computers)
- in between they may be executed in some other order
- instructions are **dynamically scheduled** by the hardware
  - hardware decides in what order instructions can be executed
  - instructions behind a stalled instruction can pass it if not dependent upon it
- advantages: higher performance
  - better at hiding latencies, less processor stalling
  - higher utilization of functional units

## In-order instruction issue

(at least) 2 styles of static instruction scheduling

- dispatch buffer & instruction slotting (Alpha 21164)
- shift register model (UltraSPARC-1)

## In-order instruction issue: Alpha 21164

### **Instruction slotting**

- can issue up to 4 instructions
  - completely empty the instruction buffer before filling it again
  - compiler can pad with `nops` so a conflicting instruction is issued with the following instructions, not alone

## 21164 Instruction Unit Pipeline

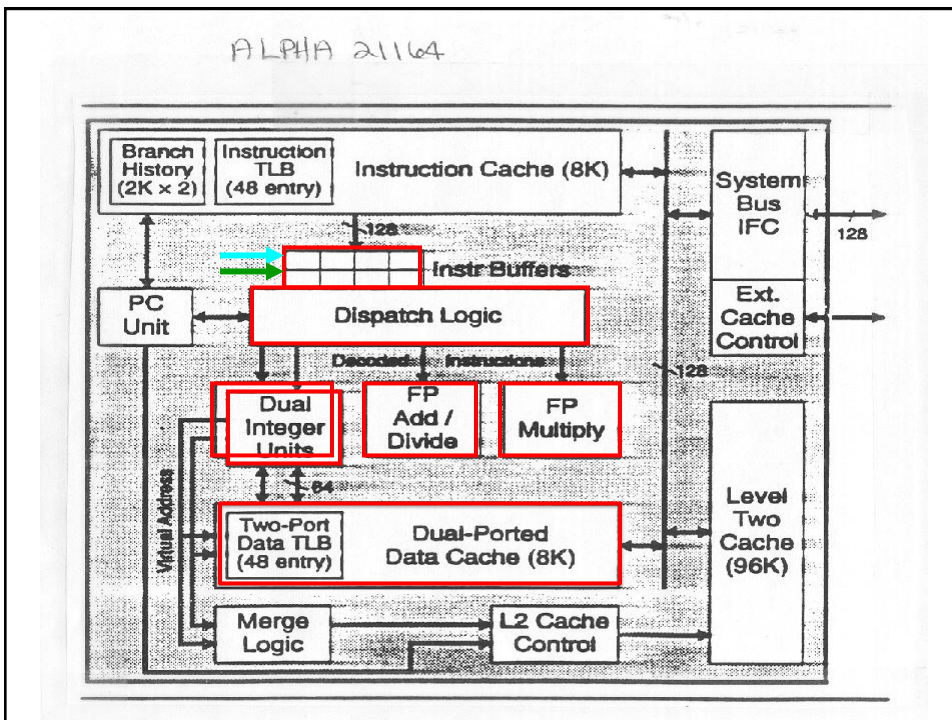
### Fetch & issue

- S0:** instruction fetch  
branch prediction bits read
- S1:** opcode decode  
target address calculation  
if predict taken, redirect the fetch
- S2: instruction slotting:** decide which of the next 4 instructions can be issued
  - intra-cycle structural hazard check
  - intra-cycle data hazard check
- S3: instruction dispatch**
  - inter-cycle load-use hazard check
  - register read

Spring 2015

CSE 471: In-Order Processors

5



## In-order instruction issue: UltraSparc 1

### **Shift register model**

- can issue up to 4 instructions per cycle
- shift in new instructions after every group of instructions is issued