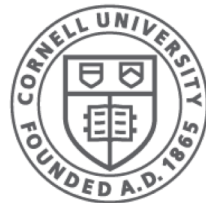




Performance

CS 3410

Computer System Organization & Programming



Cornell CIS
COMPUTING AND INFORMATION SCIENCE

[K. Bala, A. Bracy, E. Siner, and H. Weatherspoon]

Performance

Complex question

- How fast is the processor?
- How fast your application runs?
- How quickly does it respond to you?
- How fast can you process a big batch of jobs?
- How much power does your machine use?

Performance: Latency vs. Throughput

Latency (execution time): time to finish fixed task

Throughput (bandwidth): # of tasks in fixed time

- Different: exploit parallelism for throughput, not latency (e.g., bread)
- Often contradictory (latency **vs.** throughput)
 - Many examples of this at many levels
 - Consider: execution of a single instruction on single-cycle vs. 5 stage pipelined processor (faster or slower?)
- Use performance definition to match your goals
 - Scientific program: latency
 - Web server: throughput?

iClicker Question #1: Car vs. Bus

Car: speed = 60 miles/hour, capacity = 5

Bus: speed = 20 miles/hour, capacity = 60

Task: transport passengers 10 miles

	Latency (min)	Throughput (PPH)
Car	10 min	
Bus	30 min	



iClicker Question #1: Car vs. Bus

Car: speed = 60 miles/hour, capacity = 5

Bus: speed = 20 miles/hour, capacity = 60

Task: transport passengers 10 miles

	Latency (min)	Throughput (PPH)
Car	10 min	
Bus	30 min	

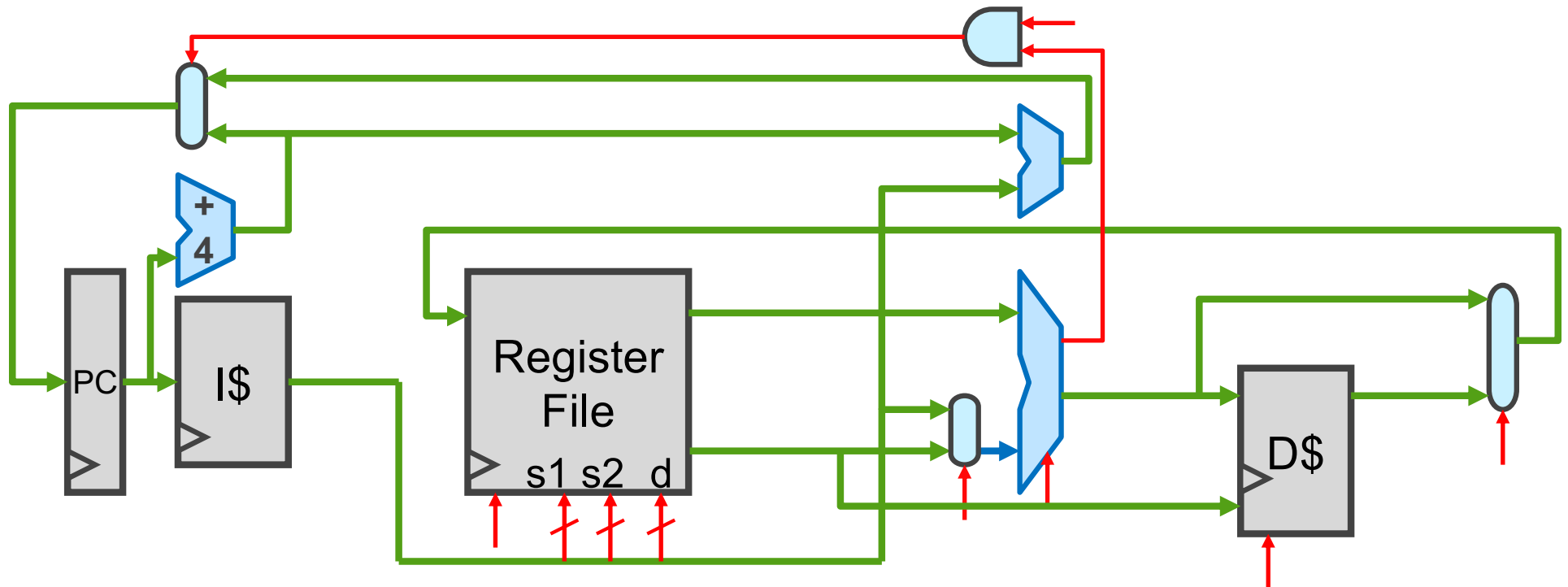


**CLICKER
QUESTIONS:**

#1 Car Throughput

- A. 10
- B. 15
- C. 20
- D. 60
- E. 120

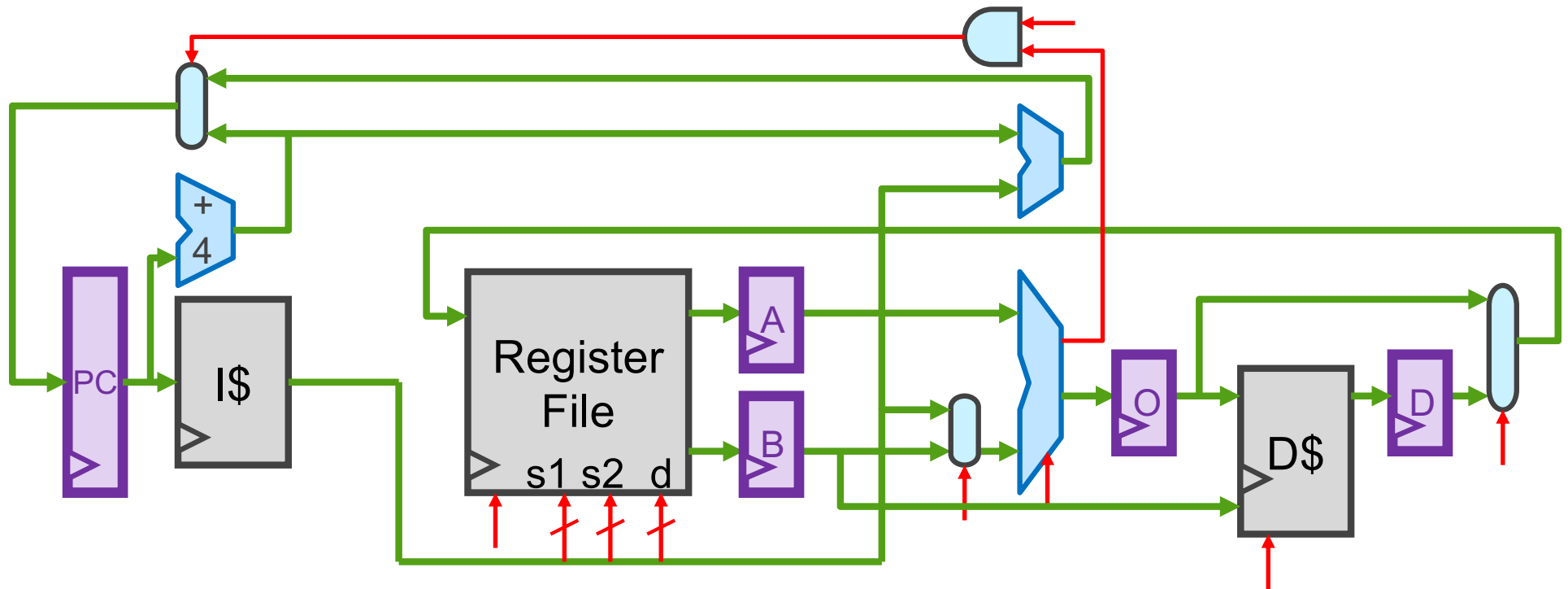
Review: Single-Cycle Datapath



Single-cycle datapath: true “atomic” F/EX loop

- Fetch, decode, execute one instruction/cycle
- + Low CPI (later): 1 by definition
- Long clock period: accommodate slowest insn
(PC → I\$ → RF → ALU → D\$ → RF)

Next Step: **Multi-Cycle** Datapath

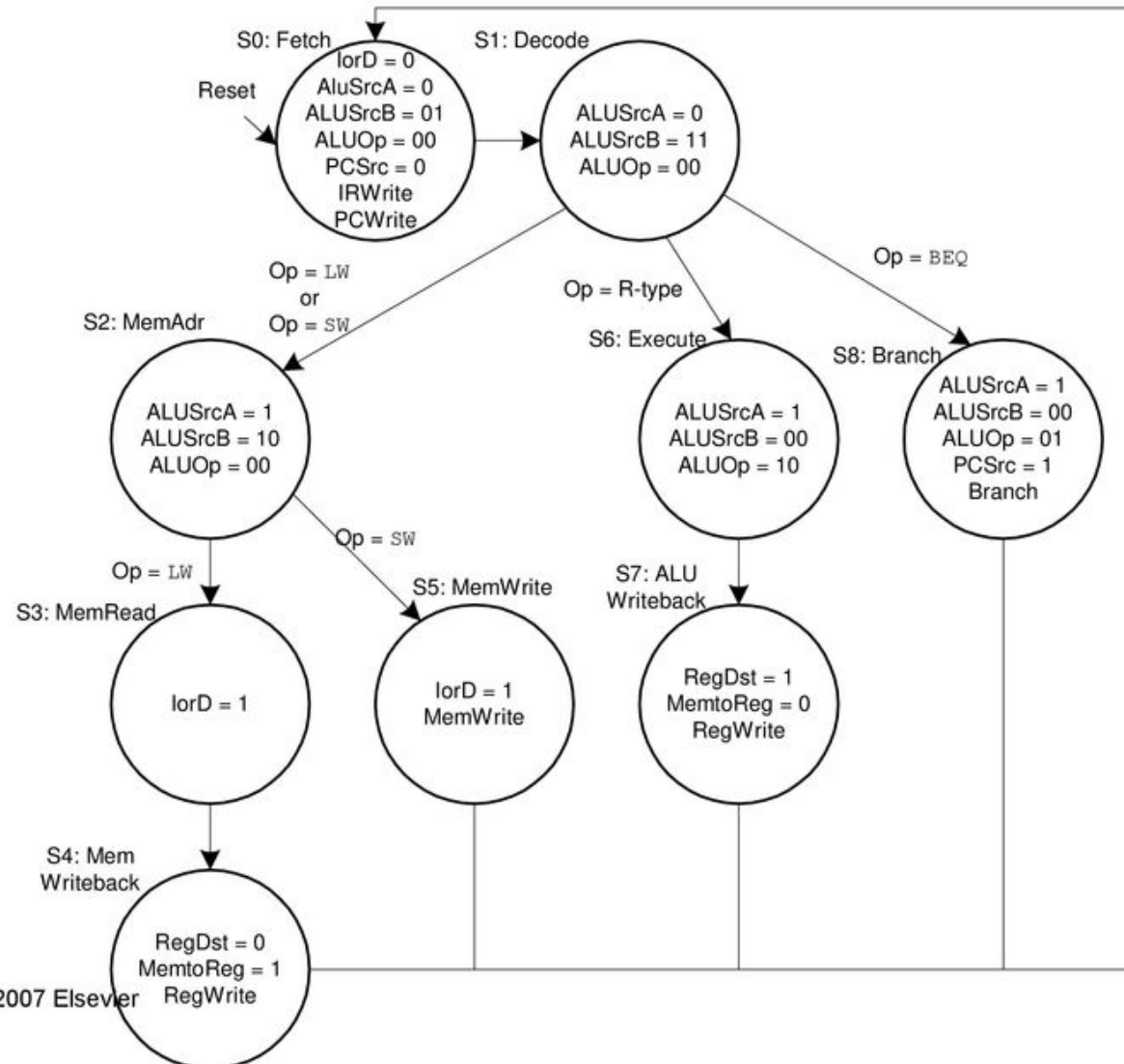


Multi-cycle datapath: attacks slow clock, separate each stage of logic with **registers**

- Fetch, decode, execute one insn over multiple cycles
 - **Allows insns to take different number of cycles**
- ± Opposite of single-cycle: short clock period, high CPI

FSMs in a Processor?

- multi-cycle (non-pipelined) processor



Single- vs. Multi-cycle Performance

Single-cycle

- Clock period = 50ns, CPI = 1
- Performance = **50ns/insn**

Multi-cycle: opposite performance split

- + Shorter clock period
- Higher CPI

Example

- branch: 20% (**3** cycles), ld: 20% (**5** cycles), ALU: 60% (**4** cycle)
- Clock period = **11ns**, $CPI = (20\% * 3) + (20\% * 5) + (60\% * 4) = 4$
 - Why is clock period 11ns and not 10ns?
- Performance = **44ns/insn**

Aside: CISC makes perfect sense in multi-cycle datapath

Cycles per Instruction (CPI)

CPI: Cycle/instruction for **on average**

- **IPC** = $1/\text{CPI}$
 - Used more frequently than CPI
 - Favored because “bigger is better”, but harder to compute with
- Different instructions have different cycle costs
 - E.g., “add” typically takes 1 cycle, “divide” takes >10 cycles
- Depends on relative instruction frequencies

CPI example

- Program has equal ratio: integer, memory, floating point
- Cycles per insn type: integer = 1, memory = 2, FP = 3
- What is the CPI? $(33\% * 1) + (33\% * 2) + (33\% * 3) = 2$
- *Caveat:* calculation ignores many effects
 - Back-of-the-envelope arguments only



iClicker: Multi-Cycle CPI

Given a CPU with instruction frequencies and costs:

- Integer ALU: 50%, 4 cycles
- Load: 20%, 6 cycles (2 cycles to go to memory)
- Store: 10%, 4 cycles
- Branch: 20%, 3 cycles

Which change would improve performance more?

A: “Branch prediction” to reduce branch cost to 1 cycle?

B: “Faster Memory” to reduce load cost to 5 cycles?

Compute CPI:

- A. A better
- B. B better
- C. equal
- D. can't say

	INT	LD	ST	BR	CPI
Base					
A					
B					

Amdahl's Law

Execution time after improvement =

$$\frac{\text{execution time affected by improvement}}{\text{amount of improvement}} + \text{execution time unaffected}$$

Or: Speedup is limited by popularity of improved feature

Corollary: **build a balanced system**

- Don't optimize 1% to the detriment of other 99%
- Don't over-engineer capabilities that cannot be utilized

Caveat: Law of diminishing returns

iClicker Question

Goal: Make Multi-Cycle CPU run 2x faster

Instruction mix (for workload P):

- 25% load/store, 3 cycles
- 60% arithmetic, 2 cycles
- 15% branches, 1 cycles

Approach: Arithmetic 2 \rightarrow 1?

	(1)	(2)
A.	2.0	1.0
B.	2.1	1.3
C.	2.2	1.4
D.	2.3	1.5
E.	2.4	2.0

(1) Baseline CPI?

(2) Faster Arithmetic CPI?

(3) How many cycles would you have to bring Arithmetic down to to get 2x speedup?



iClicker: Pipelining CPI

CPU with a control delay slot, these insn frequencies & costs:

- Integer ALU: 50%
- Load: 20%, 50% of them followed by a use
- Store: 10%
- Branch: 20%, resolved at execute, 90% of them taken

Which change would improve performance more?

A: Resolve branch at decode

B: Better compiler that can remove all the load followed by uses

Compute CPI:

- A. A better
- B. B better
- C. equal
- D. can't say

Baseline CPI:

A CPI:

B CPI:

Processor Performance Equation

Program runtime:

$$\frac{\text{seconds}}{\text{program}} = \frac{\text{instructions}}{\text{program}} \times \frac{\text{cycles}}{\text{instruction}} \times \frac{\text{seconds}}{\text{cycle}}$$

Instructions per program: “dynamic instruction count”

- Runtime count of instructions executed by the program
- Determined by program, compiler, ISA

Cycles per instruction: “CPI” (typical range: 2 to 0.5)

- How many *cycles* does an instruction take to execute?
- Determined by program, compiler, ISA, micro-architecture

Seconds per cycle: clock period, length of each cycle

- Inverse metric: cycles/second (Hertz) or cycles/ns (Ghz)
- Determined by micro-architecture, **technology parameters**

For lower latency (=better performance) minimize all three

- Difficult: ***often pull against one another***

Mhz (MegaHertz) and Ghz (GigaHertz)

1 Hertz = 1 cycle/second

1 Ghz = 1 cycle/nanosecond, 1 Ghz = 1000 Mhz

General public (mostly) ignores CPI

- Equates clock frequency with performance!

Which processor would you buy?

- Processor A: CPI = 2, clock = 5 GHz
- Processor B: CPI = 1, clock = 3 GHz
- Probably A, but B is faster (assuming same ISA/compiler)

Classic example

- 800 MHz PentiumIII faster than 1 GHz Pentium4!
- Example: Core i7 faster clock-per-clock than Core 2
- Same ISA and compiler!

Meta-point: danger of partial performance metrics!

MIPS (perf. metric, not the ISA)

(Micro) architects often ignore dynamic instruction count

- Typically have one ISA, one compiler → treat it as fixed
- CPU performance equation becomes

$$\text{Latency: } \frac{\text{seconds}}{\text{insn}} = \frac{\text{cycles}}{\text{insn}} \times \frac{\text{seconds}}{\text{cycle}}$$

$$\text{Throughput: } \frac{\text{insn}}{\text{seconds}} = \frac{\text{insn}}{\text{cycles}} \times \frac{\text{cycles}}{\text{second}}$$

MIPS (millions of instructions per second)

- **Cycles / second:** clock frequency (in MHz)
- Ex: CPI = 2, clock = 500 MHz → $0.5 * 500 \text{ MHz} = 250 \text{ MIPS}$

Pitfall: may vary inversely with actual performance

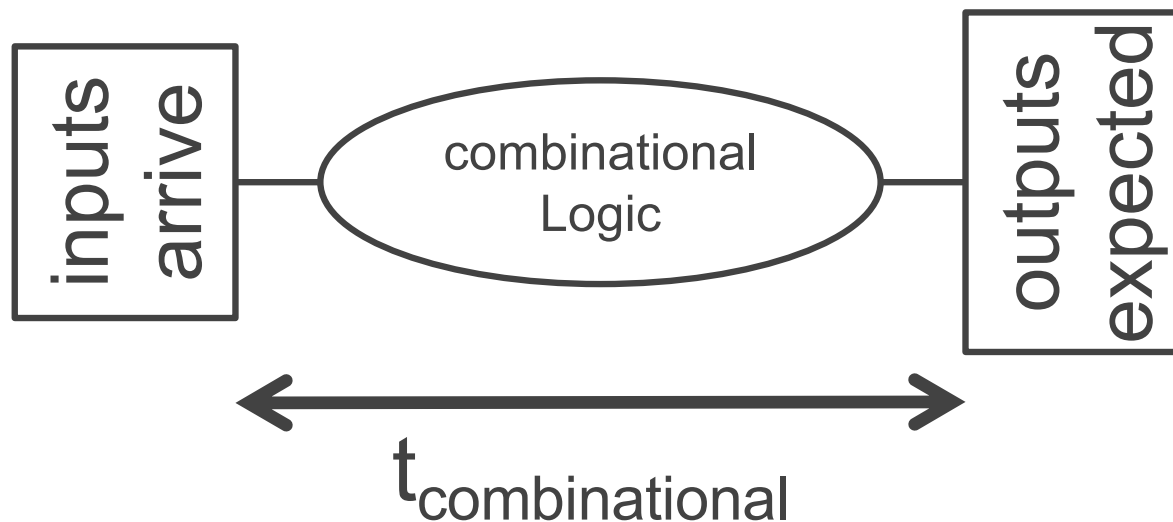
- Compiler removes insns, program faster, but lower MIPS
- Work per instruction varies (multiply vs. add, FP vs. integer)

How to make the computer faster?

Decrease latency

Critical Path

- Longest path determining the minimum time needed for an operation
- Determines minimum length of clock cycle i.e. determines maximum clock frequency

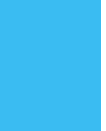


Past Prelim Question

Processor A and Processor B execute the same program in the same number of cycles

What is the minimal, additional metric(s) that you need to decide which processor is faster?

(If 1 metric is enough, only list 1. Include more if needed.)

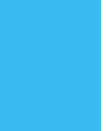
- A. MIPS
 - B. CPI
 - C. Dynamic Instruction Count
 - D. Clock Rate
 - E. Nothing. Enough information has been given
- 

Past Prelim Question

Processor A and Processor B have the same clock rate, but support different ISAs

What is the minimal, additional metric(s) that you need to decide which processor is faster?

(If 1 metric is enough, only list 1. Include more if needed.)

- A. MIPS
 - B. CPI
 - C. Dynamic Instruction Count
 - D. Clock Rate
 - E. Nothing. Enough information has been given
- 

Past Prelim Question

Processor A and Processor B support the same ISA

What is the minimal, additional metric(s) that you need to decide which processor is faster?

(If 1 metric is enough, only list 1. Include more if needed.)

- A. MIPS
 - B. CPI
 - C. Dynamic Instruction Count
 - D. Clock Rate
 - E. Nothing. Enough information has been given
- 