EssCS - Topic 2

Introduction to Computer Architecture

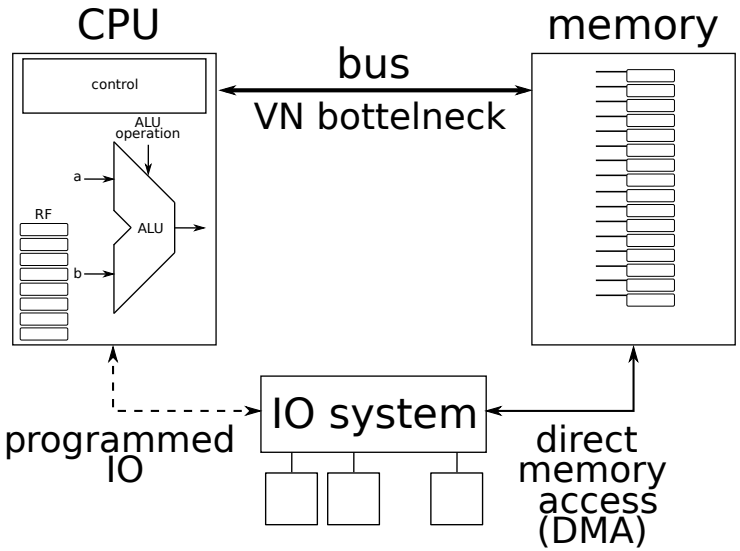
Lecture 6, 08.10.2024

Nuša Zidarič

Group Practical Assignment (Mini-Project) - see Brightspace

without minimum 9 points on the mini-project you can not pass this course !!!

Recap: von Neumann model



- Operation of VN model is completely defined with a sequence of instructions (seq. model)
- CPU reads insns from the main memory and executes them
- memory: array of memory words (address, data)
- bus: address, control (read/write), data
- which insn is next?
  CPU has a program counter (PC) register
  PC holds the address of the next insn
- how is the insn executed?
  CPU has an instruction register (IR)
  used by control to select correct ALU operation

the CPU is **reused** by insns ! (parcel=insn)

machine code	assembly	meaning	CPU action
000000 01000 01001 01010 00000 100000	add $t2, $t0, $t1	add c, a, b	$c = a + b$

other examples: lw (load word), sw(store word), j (jump), bne (branch if not equal)

Examples of instructions

- CPU registerfile: R0 (dedicated constant value 0), R1, R2, . . . , R31
  assume: each reg is 32-bit wide , special registers: `hi`, `lo`
- normal flow: PC $\leftarrow$ PC+4 (32bit = 4 bytes)

insn syntax	operation	meaning	comments
add Rd, Rs, Rt	Rd $\leftarrow$ Rs + Rt	add	arithmetic op.
addi Rt, Rs, Imm	Rt $\leftarrow$ Rs + Imm	add immediate	arithmetic op.
addu Rd, Rs, Rt	Ri $\leftarrow$ Rs + Rt	add unsigned[1]	arithmetic op.
mult Rs, Rt	[hi, lo] $\leftarrow$ Rs $*$ Rt	multiply[1]	arithmetic op.
and Rd, Rs, Rt	Rd $\leftarrow$ Rs AND Rt	bitwise and	logic op.
or Rd, Rs, Rt	Rd $\leftarrow$ Rs OR Rt	bitwise or	logic op.
sll Rd, Rt, shamt	Rd $\leftarrow$ Rt $<<$ shamt	shift[2] left logical	logic op.
lw Rt, offset(Rs)	Rt $\leftarrow$ MEM[Rs+offset]	load word	data transfer
sw Rt, offset(Rs)	MEM[Rs+offset]$\leftarrow$ Rt	store word	data transfer
beq Rs, Rt, Label[1]	if true: PC$\leftarrow$new	branch if equal	conditional
j Label	PC$\leftarrow$new	jump	unconditional

[1] see lecture notes

[2] by shamt = shift amount

Performance

- Execution time $= T = IC \times t_{\mathrm{CLK}} \times CPI = \frac{\#insn}{program} \cdot \frac{\#seconds}{cycle} \cdot \frac{\#cycpes}{insn}$
- Performance $= \frac{1}{T}$
- Speedup$= \frac{Told}{Tnew}$ **speedup is a FACTOR!**

Performance

- Execution time $= T = IC \times t_{\text{CLK}} \times CPI = \frac{\#insn}{program} \cdot \frac{\#seconds}{cycle} \cdot \frac{\#cycpes}{insn}$
- Performance $= \frac{1}{T}$
- Speedup$= \frac{Told}{Tnew}$

When running a given benchmark with the frequency of instructions given in table below, we measured the CPI for individual instructions and recorded the data below.
(a) Compute the average CPI.
(b) With modifications we were able to reduce the following:
CPI'$_{\text{load}-\text{store}} = 1.2$ and CPI'$_{\text{ALU}} = 1.1$. Compute the speedup for the given benchmark.

CPI$_{\text{load}-\text{store}} = 1.4$
CPI$_{\text{ALU}} = 1.2$
CPI$_{\text{control}} = 2.0$
CPI$_{\text{other}} = 1$

Instructions	Average frequency
load	30%
store	5%
arithmetic	15%
comparison	5%
logical	10%
control	5%
other	30%

Amdahl's Law

- make the common case fast
- total speedup depends on:
  - fraction of X being used in original system $\Rightarrow f$
  - enhancement/speedup of X $\Rightarrow S$

$S_{\text{total}} = \frac{1}{\frac{f}{S}+(1-f)}$

We want to enhance a GPS application by using a new CPU. The new CPU is 10$\times$ faster then the original CPU on GPS computations. Assuming that the original system was busy with GPS tasks 40% of the time, what is the total speedup by using the new CPU ?

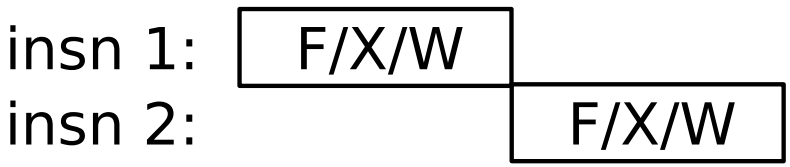
- fraction of GPS being used in original system $\Rightarrow f =$
- enhancement/speedup of GPS $\Rightarrow S =$

$S_{\text{total}} = \frac{1}{\frac{f}{S}+(1-f)} =$

Recap: CPU datapath

- recall CPU operation viewed in 3 steps: fetch (F), execute (X), write-back (W)

single cycle datapath



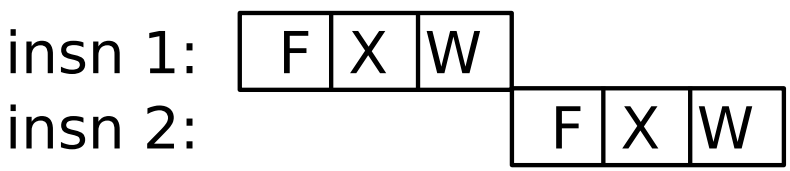
- true atomic F/X/W loop

  CPI $= 1$ (cycles-per-instruction)

  $Tput = 1$ (instructions-per-cycle)

  t_{clk} =large

multicycle datapath

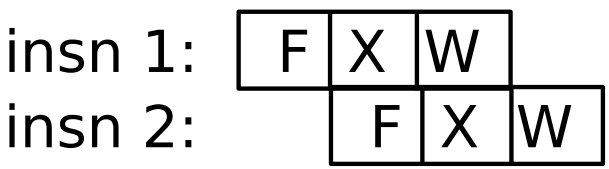


- one insn over multiple cycles

  CPI > 1 (and different for different insns)

  $Tput$ = small and t_{clk} =small

pipelined datapath

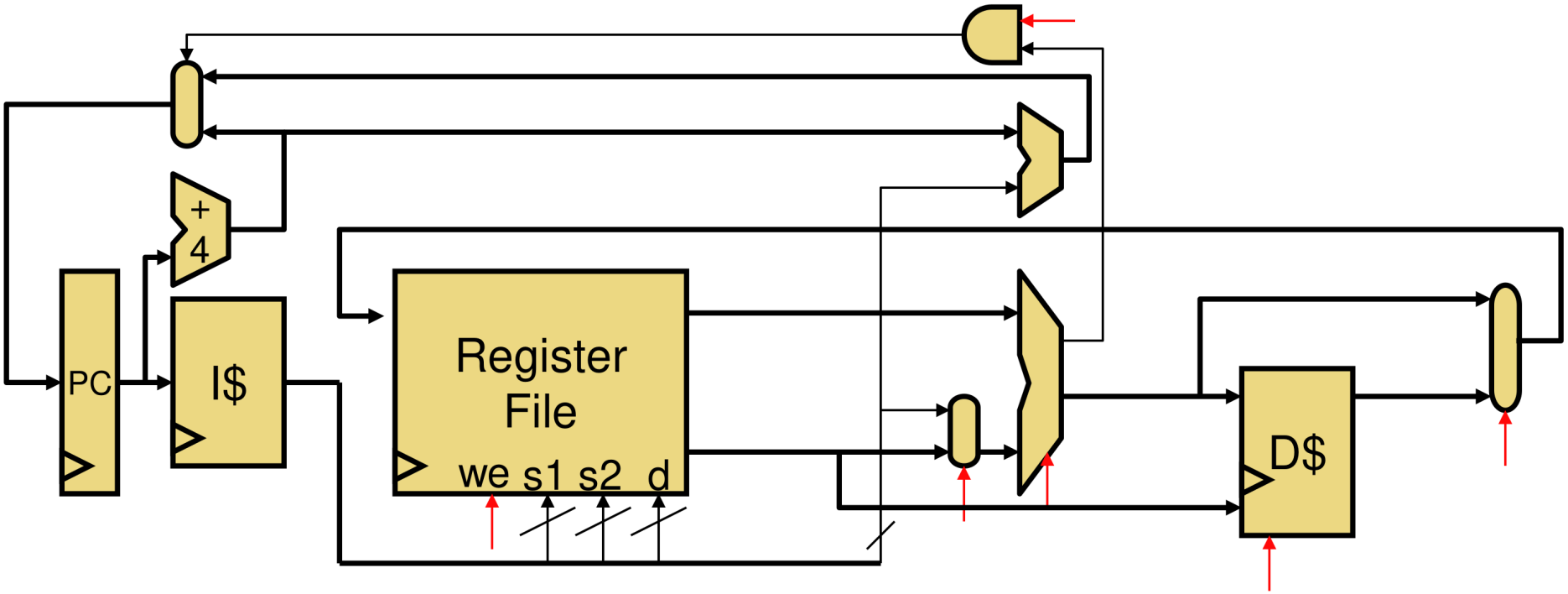


- overlap as much as possible

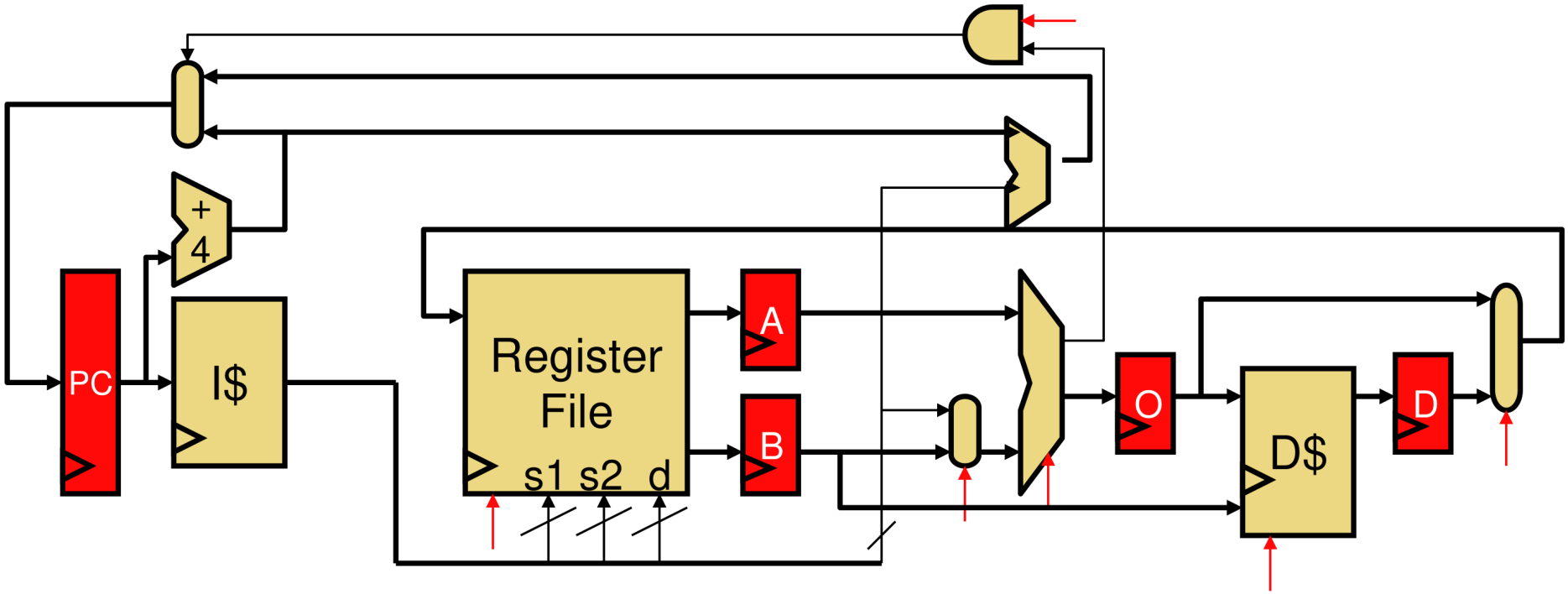
  CPI $= 1$ and $Tput = 1$ and t_{clk} =small

  NOTE: actual CPI and Tput are worse due to stalls

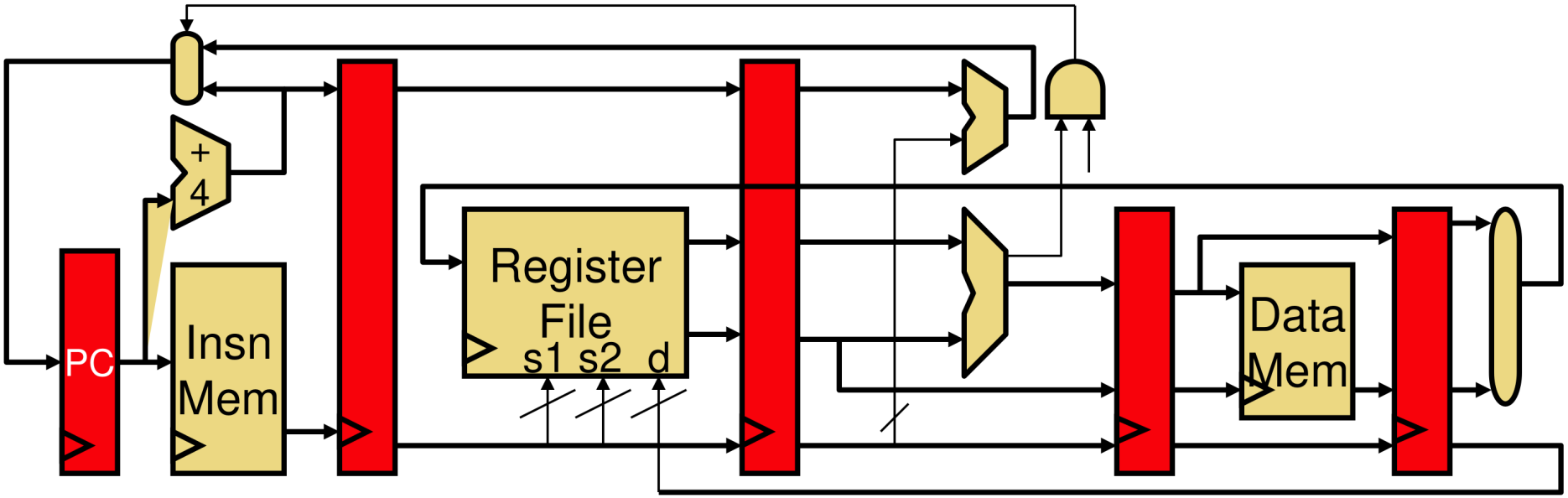
Single Cycle Datapath



Multi Cycle Datapath



Pipelined Datapath[2]



[2] Figures: H. Patel, University of Waterloo