Midterm Review

Plan

- We will go through all the key concepts following the course timeline
  - Current Computing
  - Computer System Evaluation
  - Instruction set architectures and RISC-V
  - Single cycle CPU design
  - Pipelined CPU design
  - Instruction-level Parallelism
- Do some review problems

Current Computing

Moore's Law:

The number of transistor on a chip doubles every period of time (2, 1.5, or 2.5 years).

It is enabled by Dennard Scaling.

Dennard Scaling:

As you reduce the size of the transistors, energy/power goes down proportionally.

However, Dennard Scaling started breaking down around 2003.

How to calculate energy and power of CMOS devices?

Energy = α x Capacitance x (Voltage ^ 2)

Power = Energy / time

= Dynamic Power + Static Power

$= \frac{\alpha \times \text{Capacitance} \times (\text{Voltage} \wedge 2)}{2} \times \text{frequency} + \text{Current} \times \text{Voltage}$

$= \frac{\alpha C (V^2) f}{2} + IV$

How to reduce power consumption?

Lower the voltage $\sim V^2$

Lower the frequency $\sim f$

Computer System Evaluation

Performance: **latency** and **throughput**

Latency

Time a single fixed task costs to finish.

For example, the execution time of a benchmark is the latency.

Throughput

Number of works/operations done in a fixed period of time.

For example, the CPI is the throughput.

Iron Law

Time = # of instructions x cycle per instruction x time per cycle

= architecture x micro-architecture x technology

Speedup

Speedup = old time / new time

Amdahl's Law

"the overall performance improvement gained by optimizing a single part of a system is limited by the fraction of time that the improved part is actually used."

$$S_{\text{latency}}(s) = \frac{1}{(1-p) + \frac{p}{s}}$$

where

- S_{latency} is the theoretical speedup of the execution of the whole task;
- s is the speedup of the part of the task that benefits from improved system resources;
- p is the proportion of execution time that the part benefiting from improved resources originally occupied.

Instruction set architectures and RISC-V

Instruction set architectures (ISA)

ISA is a contract between hardware and software.

Instruction format, virtual memory, number of registers, size of registers, exception, and etc. are parts of the ISA.

Reduced instruction set computing (RISC)

Small # of instructions.

Load/store architecture.

Operating on two operands.

Greatly simplifies implementation of allowed for higher frequency.

Complex instruction set computing (CISC)

Many instructions -> instructions are broken into sub-operations (micro code) by hardware

Single cycle CPU design

Every instruction goes through 5 steps:

1. Fetch
2. Decode instruction
3. Execute
4. Memory
5. Writeback

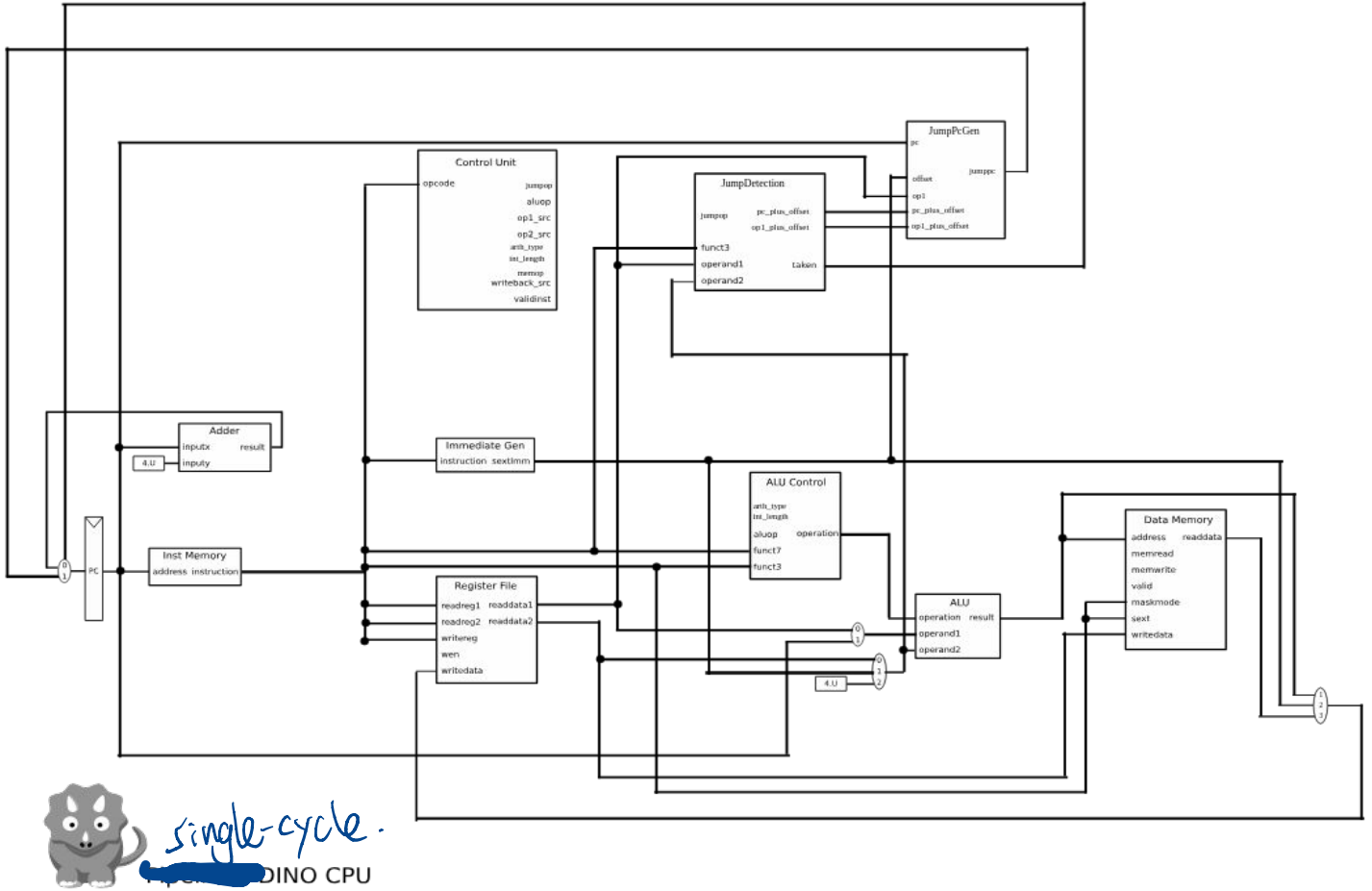
In single cycle CPU design, its cycle time is limited by the longest latency instruction.

Data Path:

Wires the data flows across

Control Path:

Multiplexers which enable multiple instructions to use one data path



Pipelined CPU design

The cycle time of a pipelined CPU design is the latency of its critical stage (the stage with the longest latency).

Theoretically the maximum CPI of a single-issue pipelined CPU design is 1.

However, pipelined CPU design introduces hazards.

Hazard

> Hazard is a dependency that causes the pipelines to stall.

Data dependency

Dependency between two instructions occurs when source of a younger instruction is the destination of an older instruction.

Forwarding the value from the older instruction to the younger instruction can hide or help the hazard.

Control dependency

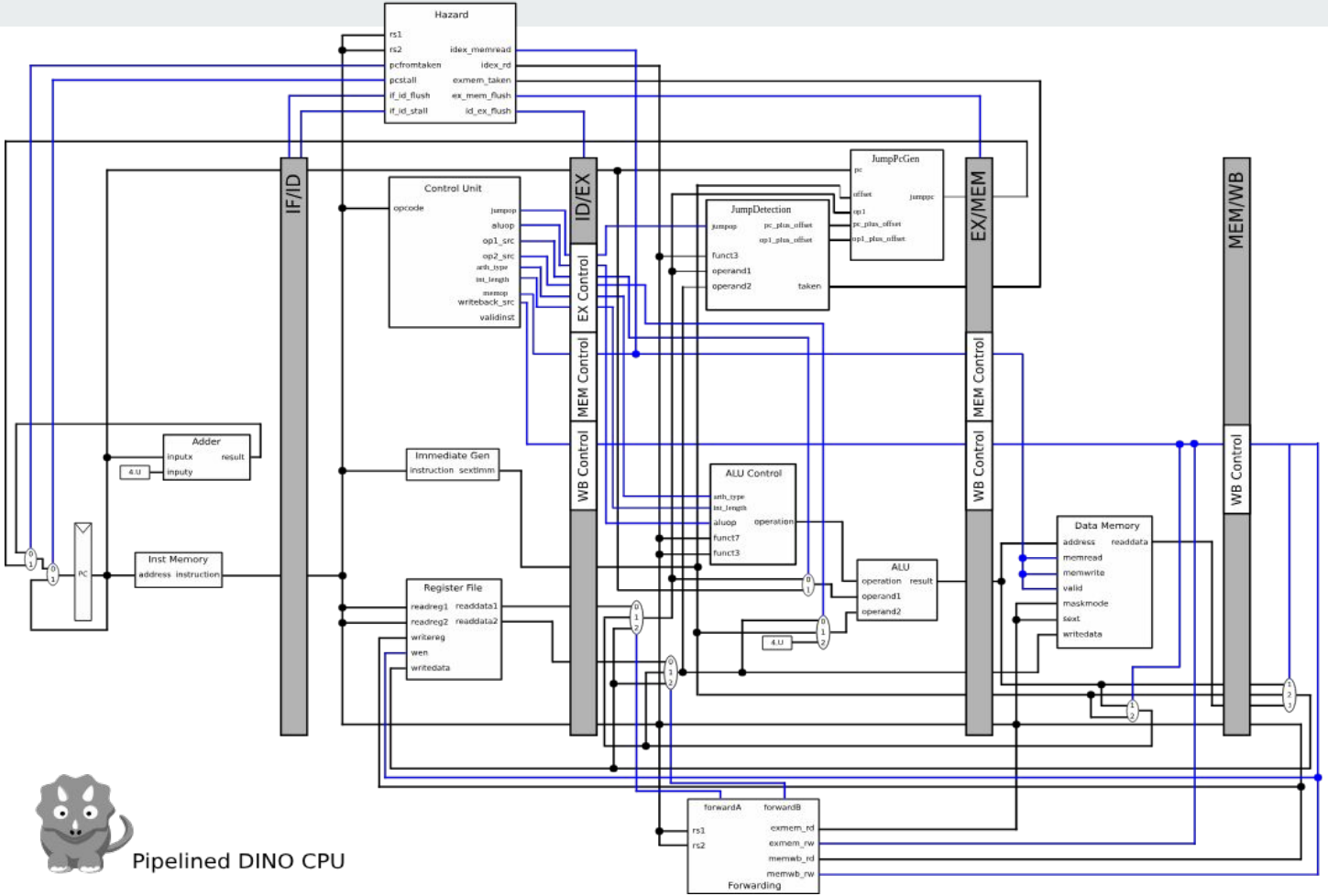
Dependency caused by waiting for the decision and target address of the branch instruction.

Branch prediction can hide this hazard.

Structure dependency

Dependency caused by multiple instructions having conflicts in resources.

Fetch Decode Execute Memory Writeback



Pipelined DINO CPU

Instruction-level Parallelism

Static ILP

- Static scheduling
- Loop unrolling
- Very long instruction word (VLIW)

Dynamic ILP

- Out of order execution

Do some review problems

- Pipelined CPU on paper simulation

This program is executed in the five stage pipelined DinoCPU

The branch predictor will predict not taken, and the branch result will be not taken.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
addi x6, x7, 16	F	D	E	M	W										
sub x8, x9, x10		F	D	E	M	W									
lw x10, 8(x6)			F	D	E	M	W								
add x6, x10, x8				F	D	stall	E	M	W						
bne x7, x8, 400					F	stall	D	E	M	W					
sw x6, 8(x10)							F	D	E	M	W