NYU Tandon School of Engineering

Fall 2025, ECE 6913

Homework Assignment 5

---

1. In this exercise, we examine in detail how an instruction is executed in a single-cycle datapath. Problems in this exercise refer to a clock cycle in which the processor fetches the following instruction word: **`0x00c6ba23`**.

1.1 What are the values of the ALU control unit's inputs for this instruction?

1.2 What is the new PC address after this instruction is executed? Highlight the path through which this value is determined.

1.3 For each mux, show the values of its inputs and outputs during the execution of this instruction. List values that are register outputs at `Reg [xn]`.

1.4 What are the input values for the ALU and the two add units?

1.5 What are the values of all inputs for the register's unit?

2. Problems in this exercise assume that the logic blocks used to implement a processor's datapath have the following latencies:

I-Mem / D-Mem	Register File	Mux	ALU	Adder	Single gate	Register Read	Register Setup	Sign extend	Control
250 ps	150 ps	25 ps	200 ps	150 ps	5 ps	30 ps	20 ps	50 ps	50 ps

"Register read" is the time needed after the rising clock edge for the new register value to appear on the output. This value applies to the PC only. "Register setup" is the amount of time a register's data input must be stable before the rising edge of the clock. This value applies to both the PC and Register File.

2.1 What is the latency of an `R-type` instruction (i.e., how long must the clock period be to ensure that this instruction works correctly)?

2.2 What is the latency of `ld`? (Check your answer carefully. Many students place extra muxes on the critical path.)

2.3 What is the latency of `sd`? (Check your answer carefully. Many students place extra muxes on the critical path.)

2.4 What is the latency of `beq`?

2.5 What is the latency of an I-type instruction?

2.6 What is the minimum clock period for this CPU?

3. (a) Suppose you could build a CPU where the clock cycle time was different for each instruction.

3.a1 What would the speedup of this new CPU be over the CPU presented in Figure 4.21 (in RISC-V text) given the instruction mix below? (assuming instruction latencies from the problem 2)

R-type/I-type (non-ld)	ld	sd	beq
52%	25%	11%	12%

3 (b) Consider the addition of a multiplier to the CPU shown in Figure 4.21. This addition will add 300 ps to the latency of the ALU, but will reduce the number of instructions by 5% (because there will no longer be a need to emulate the multiply instruction).

3.b1 What is the clock cycle time with and without this improvement?

3.b2 What is the speedup achieved by adding this improvement?

3.b3 What is the slowest the new ALU can be and still result in improved performance?

3 (c) When processor designers consider a possible improvement to the processor datapath, the decision usually depends on the cost/performance trade-off. In the following three problems, assume that we are beginning with the datapath from Figure 4.21, the latencies from **`Problem 2`** in this assignment, and the following costs:

I-Mem	Register File	Mux	ALU	Adder	D-Mem	Single Register	Sign extend	Single gate	Control
1000	200	10	100	30	2000	5	100	1	500

Suppose doubling the number of general-purpose registers from 32 to 64 would reduce the number of `ld` and `sd` instruction by 12%, but increase the latency of the register file from 150 ps to 160 ps and double the cost from 200 to 400. (*Use the instruction mix [from 3(a) above] and ignore the other effects on the ISA*)

3.c1 What is the speedup achieved by adding this improvement?
3.c2 Compare the change in performance to the change in cost.
3.c3 Given the cost/performance ratios you just calculated, describe a situation where it makes sense to add more registers and describe a situation where it doesn't make sense to add more registers.

4. `ld` is the instruction with the longest latency on the CPU from Section 4.4 (in RISC-V text). If we modified `ld` and sd so that there was no offset (i.e., the address to be loaded from/stored to must be calculated and placed in `rs1` before calling `ld/sd`), then no instruction would use both the ALU and Data memory. This would allow us to reduce the clock cycle time. However, it would also increase the number of instructions, because many `ld` and `sd` instructions would need to be replaced with `ld/add` or `sd/add` combinations.

4.1 What would the new clock cycle time be?

4.2 Would a program with the instruction mix presented *in Problem 2* run faster or slower on this new CPU? By how much? (For simplicity, assume every ld and sd instruction is replaced with a sequence of two instructions.)

4.3 What is the primary factor that influences whether a program will run faster or slower on the new CPU?

4.4 Do you consider the original CPU (*as shown in Figure 4.21 of RISC-V text*) a better overall design; or do you consider the new CPU a better overall design? Why?

5. (a) Examine the difficulty of adding a proposed `lwi.d rd, rs1, rs2` ("Load With Increment") instruction to RISC-V. Interpretation: `Reg[rd]=Mem[Reg[rs1]+Reg[rs2]]`

5.a1 Which new functional blocks (if any) do we need for this instruction?

5.a2 Which existing functional blocks (if any) require modification?

5.***a3*** Which new data paths (if any) do we need for this instruction?

6. In this exercise, we examine how pipelining affects the clock cycle time of the processor. Problems in this exercise assume that individual stages of the datapath have the following latencies:

IF	ID	EX	MEM	WB
250 ps	350 ps	150 ps	300 ps	200 ps

Also, assume that instructions executed by the processor are broken down as follows:

ALU/Logic	Jump/Branch	Load	Store
45%	20%	20%	15%

6.1 What is the clock cycle time in a pipelined and non-pipelined processor?
6.2 What is the total latency of an `ld` instruction in a pipelined and non-pipelined processor?
6.3 If we can split one stage of the pipelined datapath into two new stages, each with half the latency of the original stage, which stage would you split and what is the new clock cycle time of the processor?
6.4 Assuming there are no stalls or hazards, what is the utilization of the data memory?
6.5 Assuming there are no stalls or hazards, what is the utilization of the write-register port of the "Registers" unit?

7. What is the minimum number of cycles needed to completely execute n instructions on a CPU with a k stage pipeline? Justify your formula.

8. (a) Assume that `x11` is initialized to `11` and `x12` is initialized to `22`. Suppose you executed the code below on a version of the pipeline from Section 4.5 that does not handle data hazards (i.e., the programmer is responsible for addressing data hazards by inserting NOP instructions where necessary). What would the final values of registers `x13` and `x14` be?

```
addix11, x12, 5

addx13, x11, x12

addix14, x11, 15
```

(b) Assume that `x11` is initialized to `11` and `x12` is initialized to `22`. Suppose you executed the code below on a version of the pipeline from Section 4.5 *that does not handle data hazards* (i.e., the programmer is responsible for addressing data hazards by inserting NOP instructions where necessary).

What would the final values of register `x15` be? Assume the register file is written at the beginning of the cycle and read at the end of a cycle. Therefore, an `ID` stage will return the results of a `WB` state occurring during the same cycle. See Section 4.7 and Figure 4.51 for details.

```
addix11, x12, 5
addx13, x11, x12
addix14, x11, 15
addx15, x11, x11
```

(c) Add `NOP` instructions to the code below so that it will run correctly on a pipeline that does not handle data hazards.

```
addix11, x12, 5
addx13, x11, x12
addix14, x11, 15
addx15, x13, x12
```