<https://www.youtube.com/watch?v=pbY2D_0GwGE>

1. Consider a non-pipelined, single-cycle CPU (i.e., all instructions complete in one CPU cycle).

Suppose you are asked to design a similar CPU with a two-stage pipeline. List the factors that

may prevent you from making the cycle time of the new pipelined machine half of the cycle time

of the non-pipelined machine.

*Several factors that prevent us from achieving the ideal speedup include:*

*- Not being able to divide the pipeline evenly*

*- The time needed to empty and flush the pipeline*

*- Overhead needed for pipelining*

*- Structural, Data, and Control* ***hazards***

2. Consider two CPUs that differ only in that one is pipelined and one is not. Is the total time needed to completely execute a single instruction greater on the pipelined or non-pipelined machine? Why? (In other-words, does pipelining tend to increase or decrease the latency of individual instructions?)

*Pipelining decreases execution time but can increase cycle time*

*- Throughput, is decreased since a single instruction (ideally) finishes every clock.*

*- However, it usually increases the latency of each instruction.*

*Why?*

*- Imbalance among the pipe stages*

* + *The slowest stage determines the clock cycle time*

***- Pipelining overhead***

*- pipeline register delay, that is, adding registers, adds logic between each of the stages.*

*- pipelining does not help the latency of a single instruction. It helps the throughput of the entire workload.*

3. Look at the two tables below.

1. Assume that your processor is a 5-state pipeline with both data and register forwarding.

Which table is correct? Why?

The first table is correct because the stall must happen at the same time throughout the pipeline.

(b) Draw a diagram showing how the instructions would flow through the pipeline if it didn’t

have any data or register forwarding.

4. What are structural hazards? What causes them? How are they resolved?

2 or more instructions in the pipeline need to use the same resource (or stage in pipeline) at the same time.

These can be caused when say MEM and IF have to access cache memory at the same time (hardware resource). The IF stage is going there to receive data about the instruction, while a MEM stage could be produced by many instructions. This happens if the 2 instructions are 3 instructions apart.

These are resolved by inserting a stall for the instruction fetch stage to wait one cycle. Another would be to design memory so that it can handle 2 memory requests at the same time (double ported memory, haven’t talked about) or, by having a cache for just instructions, and one for just memory. So, basically, by throwing more hardware at it.

5. Are structural hazards unique to pipelined CPUs? (As opposed to non-pipelined CPUS?) Why

or why not?

No, they are not. Notice that the single-cycle CPU requires several ALUs and separate data instructions.

6. What are data hazards? What causes them? How are they resolved?

An instruction depends on the data result of a prior instruction that is still in the pipeline.

These are resolved by again inserting stalls in the instructions that are causing the data hazard. The EX stage of the second instruction must occur after the WB stage from the first. This can also be solved by forwarding the data ahead of the memory stage so it is ready for the next instruction’s execution.

7. Are data hazards unique to pipelined CPUs? (As opposed to non-pipelined CPUs?) Why or why not?

Yes, they are because they only occur when several instructions are being processed simultaneously, which only happens in pipelined CPUs.

8. What are control hazards? What causes them? How are they resolved?

A hazard that arises due to control transfer hazards like jumps, branches, etc.

These are resolved by

9. Are control hazards unique to pipelined CPUs? (As opposed to non-pipelined CPUs?) Why or

why not?

10. (Problem 6.4 from Patterson and Hennessy): Identify all of the data dependencies in the following code. Which dependencies are data hazards that will be resolved by forwarding? Which dependencies are data hazards that will cause a stall?

1: add $3, $4, $2

2: sub $5, $3, $1

3: lw $6, 200($3)

4: add $7, $3, $6

11. The performance of a pipelined computer, in terms of how long it takes to run a program, can be expressed as:

seconds/program =instructions/program × cycles/instruction × seconds/cycle

(a) What term in the equation above corresponds to n?

instructions/program

(b) What term in the equation above corresponds to p?

seconds/cycle

(c) What does cycles /instruction represent with respect to a pipelined CPU?

N and p correspond the same way. The cycles/instruction term accounts for the stalls and branch delay slots.

(d) How does k affect this formula?

The larger k is, the smaller p is.

The larger values of k tend to produce larger CPI values

12. Consider the instruction moveqz R1, R2, R3 that sets R1 to be equal to R2 if, and only if,

R3 == 0.

(a) Explain how adding this instruction to a CPU can potentially improve performance.

(b) Explain how adding this instruction to a CPU can potentially decrease performance

1. Implementing this instruction in hardware would save time by allowing us to replace the following 2 instructions with just this 1 new instruction:
   1. Bneq r3, r0, 1
   2. R1 = r2

B. The additional hardware can reduce the clock period for all instructions, but only if it ends up on the critical path.

13. Explain the purpose of each component and control wire in the figure below:

14. Highlight the forwarding paths used by the following sequence of instructions:

add $t0, $t1, $t2

sub $t3, $t4, $t5

add $t5, $t0, $t3

15. Highlight the forwarding paths used by the following sequence of instructions:

add $t0, $t1, $t2

sub $t3, $t4, $t5

beq $t0, $t3, END

