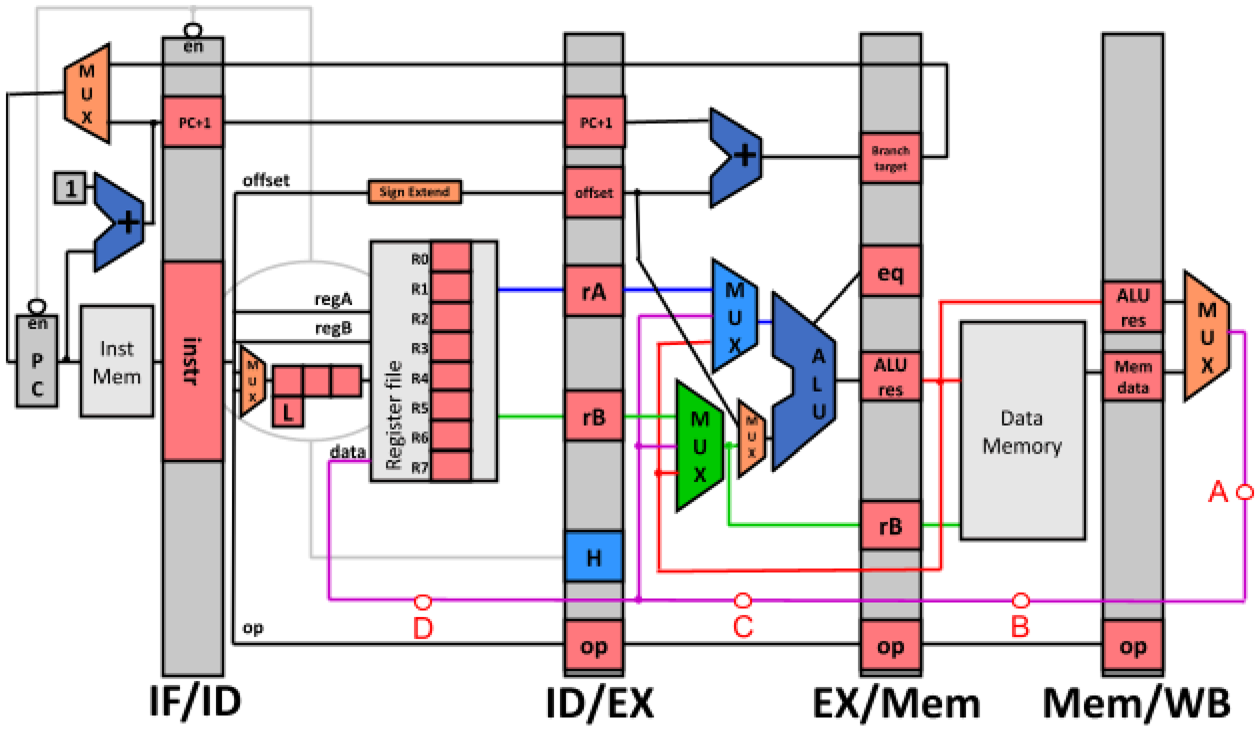
Problem 2: Pipeline Design Modification *[22 points]*

Scribe: **[Scribe's name here]**



Pipeline Datapath for Reference

Consider the pipeline from lecture (above) with times below, using **detect and forward** for data hazards and **speculate taken and squash** for control hazards. The current cycle time is 30ns.

- Data Memory Read/Write: 30 ns
- Instruction Memory Read: 8 ns
- ALU: 20 ns
- Register file read/write: 5 ns
- Backwards wire delay: 2 ns per stage *
- All other wires: 0 ns
- All other delays: 0 ns

* Wires that go backwards across pipeline stages (eg. mux in WB to the Register File, pipeline registers to muxes in EX) have a delay of **2 ns per stage**. Thus the delay from point **A** (the black circle labeled A in WB) to **B** (in MEM) is 2 ns, and the delay from **A** to **D** is 6 ns. This also affects forwarded values: for example, **A** to **C** is 4ns.

The current pipeline is: IF → ID → EX → MEM → WB

Suppose we have the option to add new stages to the pipeline and *evenly* split memory transactions and ALU operations into up to 4 stages each. E.g. we could split the ALU into two stages to create a new pipeline:

IF → ID → EX1 → EX2 → MEM1 → WB

In this new pipeline, the 20 ns ALU operation is divided into two 10 ns operations, one in EX1 and one in EX2. However, the delay from point **A** (in WB) to **C** (in EX) is now 6 ns, so the propagation delay in EX1 is 16ns. And the delay from point **A** (in WB) to **D** (in ID) is now 8ns, so the WB stage delay is 13ns. (Data can only be forwarded from MEM1 and WB, data is always forwarded to EX1, and register reads and writes are concurrent due to internal forwarding).

a) The below pipeline design has the optimal clock period given the above constraints. What is this clock period? Show your work. *[8 points]*

Pipeline: IF → ID → EX1 → EX2 → EX3 → MEM1 → MEM2 → WB

Each mem stage is now 30ns / 2 = 15ns.
The ALU is split evenly into 3 operations of 6.67ns each. EX1 also has the wire delay from A to C to do forwarding, now 10ns. So, the EX1 stage is 16.67ns.
The WB stage is A to D delay (now 12ns) + reg write (5ns) = 17ns.
The ID stage, since it has internal forwarding, is max(reg read, A to D delay) = 12ns. Always less than or equal to WB.

Clock period: __________17_____________ ns

b) This pipeline, though it has more stages, has a higher clock period. What is this clock period? *[6 points]*

Pipeline: IF → ID → EX1 → EX2 → EX3 → EX4 → MEM1 → MEM2 → MEM3 → WB

Each mem stage is now 30ns / 3 = 10ns.
The ALU is split evenly into 4 operations of 5ns each. EX1 also has the wire delay from A to C to do forwarding, now 14ns. So, the EX1 stage is 19ns.
The WB stage is A to D delay, now 16ns, + reg write, 5ns, to get 21ns.
The ID stage, since it has internal forwarding, is max(reg read, A to D delay) = 16ns. Always less than or equal to WB.

Clock period: __________21_____________ ns

c) Destination distance can be used to describe how far away dependencies are from each other. Below showcases two examples of destination distance (the data dependency is **bolded**).

Example 1 – Destination Distance of 1:

lw	*0*	***1***	*data*
add	***1***	*2*	*3*

Example 2 – Destination Distance of 2:

lw	*0*	***1***	*data*
noop			
add	***1***	*2*	*3*

For the optimal design in part a), complete the table to indicate how many *stalls* are needed for register dependencies that cannot be completely solved by forwarding. For example, in the original pipeline, a dependent instruction immediately after an LW would require 1 stall. *[8 points]*

Source Instr. Type	Destination Distance	# Stalls
Ex for OG design: LW	*1 (Immediately after)*	*1*
LW	1 (immediately after)	4
LW	2 (1 in between)	3
LW	3	2
LW	4	1
ADD/NOR	1 (immediately after)	2
ADD/NOR	2 (1 in between)	1