

3. Describe the components of the single-cycle datapath shown above.

4. Describe the additional logic required to implement jump instructions.

5. Describe the basic operation of a pipelined datapath.

6. Does pipelining improve latency or throughput?

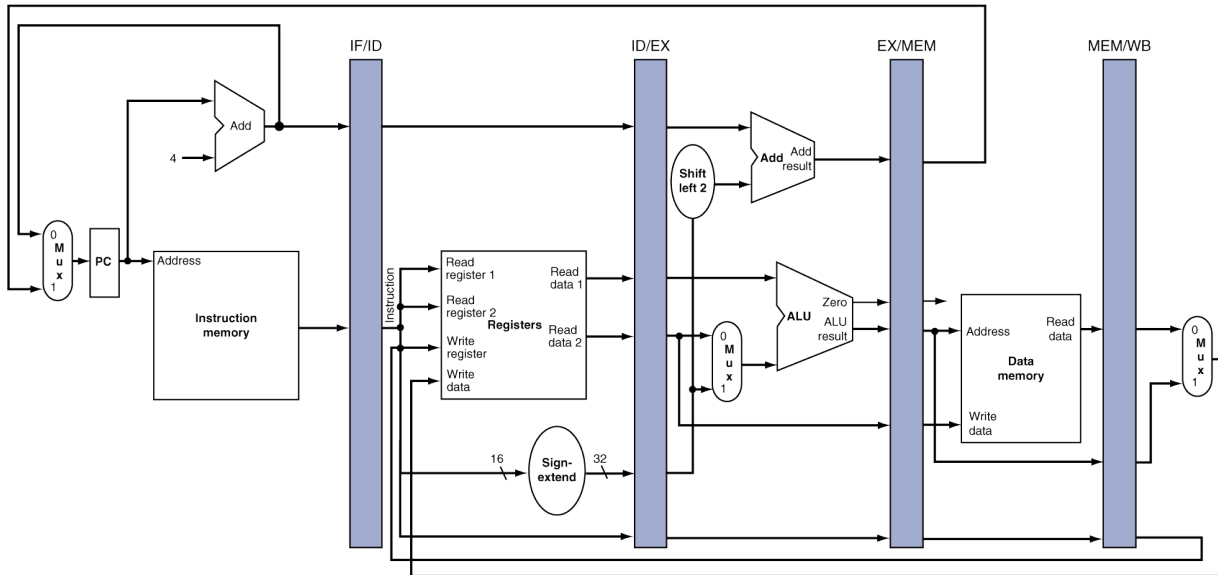
7. What is the maximum potential speedup of pipelining?

8. If one pipeline stage can run faster than the others, how does that affect the speedup?

9. Draw a basic pipeline diagram and describe the 5 stages.

10. Every pipelined instruction takes the same number of cycles. Why?

12. Describe how a pipelined datapath is divided into stages and how data is transferred between stages. (Refer to the figure below.)



13. Describe the three different types of pipeline hazards.

14. Example: What are no-ops? Given the following code, where are no-ops needed?

```
add $t2, $t3, $t4  
sub $t5, $t1, $t2  
or  $t6, $t2, $t7  
slt $t8, $t9, $t5
```

15. Explain how forwarding works in a pipelined datapath.

16. Describe a case in which forwarding alone is not enough to resolve a data hazard.

17. Example: Let's revisit the following code:

```
loop:      add $t1, $t2, $t3
           lw  $t4, 0($t1)
           beq $t4, $t3, end
           sw  $t3, 4($t1)
           add $t2, $t2, 8
           j   loop
end:       ...
```

Again, assume each pipeline stage takes 4 ns.

a. How many cycles would one loop iteration take in a pipelined datapath without forwarding?

b. How many cycles would one loop iteration take in a pipelined datapath with forwarding?

18. Describe the reason processors may experience branch delays.