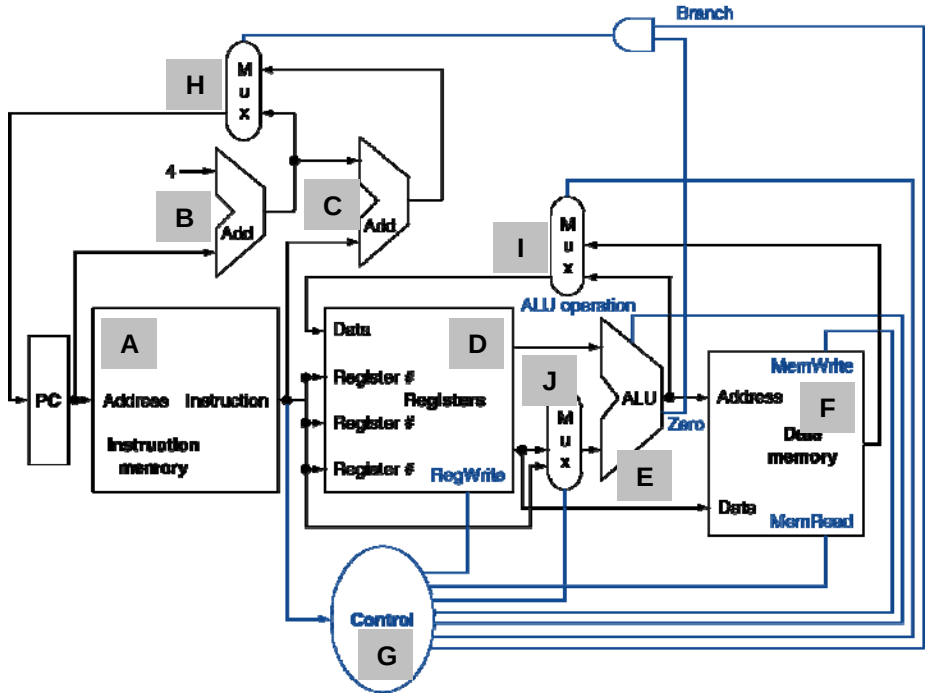
Prepare your answers to the following questions in a plain text file, and submit them to the Curator system by the posted deadline for this assignment. No late submissions will be accepted.

You will submit your answers to the Curator System (www.cs.vt.edu/curator) under the heading `HW5`.

For questions 1 and 2, consider the following simplified MIPS datapath (Fig 4.2 from P&H):



The datapath supports the following instructions: `add`, `sub`, `and`, `or`, `slt`, `beq`, `lw` and `sw`.

Assume that the major components of the given datapath have the following latencies:

Unit	I-Mem	Add	Mux	ALU	Regs	D-Mem	Control
Latency	400ps	100ps	30ps	120ps	200ps	350ps	100ps

1.

a) [20 points] Referring to the labels **A** through **J** in the datapath diagram above, which components are required during the execution of a `lw` instruction? (We ignore the PC.)

The components labeled C and H are used to compute a branch address for the beq instruction, and C is not needed for any of the others. All of the other components are used in executing a `lw` instruction.

b) [20 points] What is the critical path for the `lw` instruction, and what is the total latency for the critical path?

There are several relevant paths for `lw`:

PC→B→H→PC **130 = 100+30**
PC→A→D→J→E→F→I→D **1330 = 400+200+30+120+350+30+200 critical path**
PC→A→G **500 = 400+100**

2.

a) [20 points] Referring to the labels **A** through **J** in the datapath diagram above, which components are required during the execution of a `beq` instruction? (We ignore the PC.)

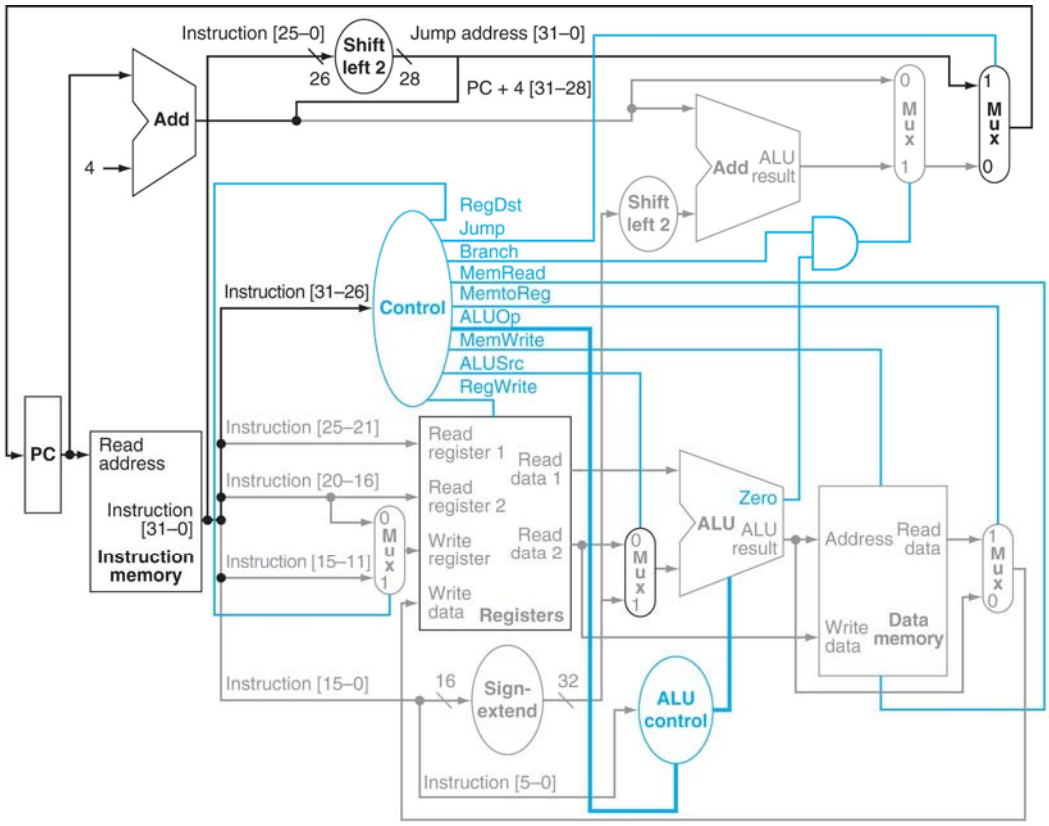
`beq` does not access the data memory F (only `lw` and `sw` do that), and `beq` does not write a value into data memory, so the multiplexor I is also not needed. All the other components are used in executing `beq`.

b) [20 points] What is the critical path for the `beq` instruction, and what is the total latency for the critical path?

Again, there are several relevant paths:

PC→B→C→H→PC	**230 = 100+100+30**
PC→A→D→J→E	**750 = 400+200+30+120 critical path**
PC→A→G	**500 = 400+100**

Question 5 refers to the following MIPS datapath diagram (Fig 4.24 from P&H):



The datapath supports the following instructions: `add`, `sub`, `and`, `or`, `slt`, `beq`, `j`, `lw` and `sw`.

5. [20 points] A *stuck-at-0 fault* occurs when, due to a manufacturing defect, a signal is mis-connected so that it always carries a logical value of 0. A *stuck-at-1 fault* is defined analogously.

Suppose that processor testing is done by loading the PC, registers and data and instruction memories with some values (you can choose which values), letting a single instruction execute, and then reading the PC, memories and registers and examining those values to determine if a fault exists.

Design a test (i.e., specify values for the PC, memories and registers) that would determine whether there is a stuck-at-1 fault for the signal MemWrite. Explain clearly how the resulting values (PC, memories, registers) would indicate that the specified fault does exist and how they would indicate that the specified fault does not exist.

Be very specific with your answers.

If MemWrite is stuck-at-1, the data memory will be written unexpectedly. Here is one test:

Pre-test setup:
PC = PC0
$s0 = 0
Memory location referenced by 0($s1) has a value: 55

Test:
execute `lw $s0, 0($s1)` # could use a different instruction as long as it's not `sw`

After-test check:
There is no MemWrite fault when:
(1) current PC is PC0+4
(2) $s0 = 55
(3) memory location referenced by 0($s1) is unchanged, still 55

There is a MemWrite fault when:
(1) current PC is PC0+4
(2) $s0 = random number or 55 (could be either depending on timing of memory write completion)
(3) memory location referenced by 0($s1) is changed to a random number

Here is another:

Pre-test setup:
PC = PC0 (address of instruction shown below)
$s0 = 1
$s1 = 2
Memory location referenced by 0($s1) has a value: 55

Test:
execute `add $s0, $s1, $s2`

So, what happens is that the sum of $s0 and $s2
After-test check:
There is no MemWrite fault when:
(1) current PC is PC0+4
(2) $s0 = 55
(3) memory location referenced by 0($s1) is unchanged, still 55

There is a MemWrite fault when:
(1) current PC is PC0+4
(2) $s0 = random number or 55 (could be either depending on timing of memory write completion)
(3) memory location referenced by 0($s1) is changed to a random number