

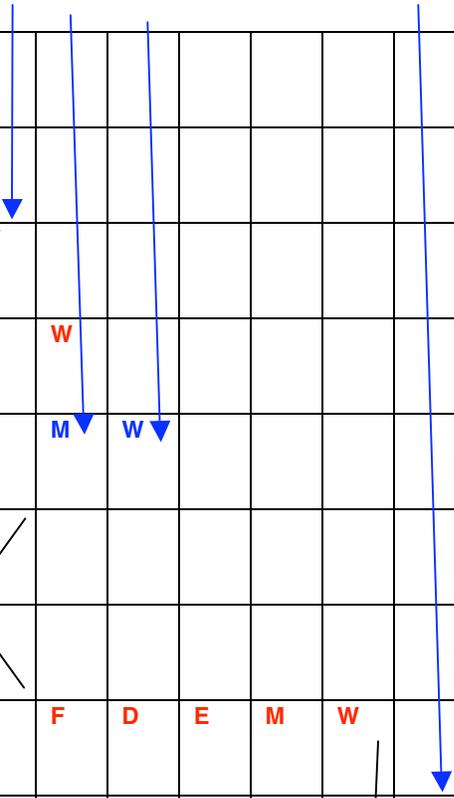
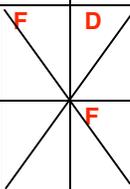
For the sequence of instructions shown below, show how they would progress through the pipeline.

Part 1:

- Assume that **forwarding HAS been implemented**
- We will predict that any branch instruction is **NOT TAKEN**
- Branches or Jumps are resolved after the EX stage.
- Assume that register \$8 does not equal \$1 for the 1<sup>st</sup> Beq instruction
- Assume that register \$17 does equal \$26 for the 2<sup>nd</sup> Beq instruction

Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
<b>SUB</b> \$1, \$2, \$3	F	D	E	M	W												
<b>Add</b> \$8, \$9, \$10		F	D	E	M	W											
<b>Beq</b> \$1, \$8, X			F	D	E	M	W										
<b>Lw</b> \$7, 0(\$20)				F	D	E	M	W									
<b>Add</b> \$11, \$7, \$12					F	D	D	E	M	W							
<b>Sw</b> \$11, 0(\$24)						F	F	D	E	M	W						
<b>X: Addi</b> \$17, \$17, 1							F	D	E	M	W						
<b>Beq</b> \$17, \$26, Y								F	D	E	M	W					
<b>Sub</b> \$5, \$6, \$7										F	D						
<b>Or</b> \$8, \$5, \$5											F						
<b>Y: Addi</b> \$17, \$17, 1												F	D	E	M	W	
<b>Sw</b> \$17, 0(\$10)													F	D	E	M	W
<b>SUB</b> \$1, \$2, \$3														F	D	E	...
<b>Add</b> \$8, \$9, \$10															F	D	...

Technically, nothing done, but can think of instruction as progressing through pipeline



Part 2:

- (i) Assume that this sequence of code is executed 100 times. How many cycles does the pipelined implementation take?
- (ii) How many cycles would this code take in a multi-cycle implementation?
  
- From Part 1, you can see that it takes 17 clock cycles to execute 12 instructions.
- However, we can start the next "iteration" in clock cycle 14. Therefore, it *really* only takes 13 cycles for each iteration and 17 CCs for the last one.
- Therefore, iterations 1 through 99 take 13 CCs each
  - o  $(13 \times 99 = 1287 \text{ CCs})$
- Iteration 100 takes 17 CCs
- Therefore  $1287 \text{ CCs} + 17 \text{ CCs} = 1304 \text{ CCs}$
  
- For the multi-cycle implementation, we have:
  - o 9 instructions that take 4 CCs
  - o 2 instructions that take 3 CCs
  - o 1 instruction that takes 5 CCs
- Therefore, each "iteration" takes:  $(9 \times 4) + (2 \times 3) + (1 \times 5) = 36 + 6 + 5 = 47 \text{ CCs}$
- If there are 100 iterations, then 4700 CCs are required

Pipelining gives us a speed up of  $4700 / 1304 = 3.6$  for this implementation

- Little to no extra HW is needed!