Lecture 6 Pipelining In-class Exercises

Consider the following code, how many cycles will one iteration through the loop (from lw to bnez) take?

```
Loop: lw   r2, 40(r6)
      mult r4, r3, r2
      sw   r4, 80(r7)
      addi r6, r6, 4
      addi r7, r7, 4
      addi r1, r1, -1
      bnez r1, loop
```

Assumptions:

- In-order pipeline (i.e, no out-of-order execution), no branch prediction
- The pipeline has all kinds of data bypassing available (but stalls on data hazards that cannot be bypassed).
- The pipeline stalls if a structural hazard would occur.
- `mult` has 4 pipelined execution stages P1, P2, P3, P4, which takes 4 cycles in total; `add` has a 1-cycle eXecution stage. `mult` and `add` use different ALUs.
- `mult` does not have M stage; all other instructions have M stage.
- `bnez` comparison is done in the Decode stage (rather than X stage).

Insn	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
lw r2, 40(r6)																						
mult r4, r3, r2																						
sw r4, 80(r7)																						
addi r6, r6, 4																						
addi r7, r7, 4																						
addi r1, r1, -1																						
bnez r1, loop																						

Solution: 15 cycles

Insn	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
lw r2, 40(r6)	F	D	X	M	W																	
mult r4, r3, r2		F	D	D	P1	P2	P3	P4	W													
sw r4, 80(r7)				F	D	D	D	X	M	W												
addi r6, r6, 4							F	D	X	M	W											
addi r7, r7, 4								F	D	X	M	W										
addi r1, r1, -1									F	D	X	M	W									
bnez r1, loop										F	D	D	X	M	W							

Q2:
Given the same assumptions as Q2 of Lecture 5 Exercises (no pipeline), i.e.,
In both of the following ISAs, Operand 1 is register, Oprand 2 and Oprand 3 can be register or immediate.

(1) The first is a **fixed length** ISA that has the following instruction encoding:

Opcode | Oprand 1 (Destination) | Oprand 2 (Source 1) | Oprand 3 (Source 2)

An opcode is 1 byte. Each operand is 1 byte. All register/register and register/immediate operations take 1 cycle, while all load and store instructions take 4 cycles.

(2) The second is a **variable length** ISA that the following instruction encoding:

Opcode | Oprand 1 (Destination) | Oprand 2 (Source 1) | Oprand 3 (**Optional**, Source 2)

The opcode is 1 byte. Each operand (register/immediate) is 1 byte. Note that Oprand 3 is optional. If an instruction does not need the third oprand, it is not used. In other words, the third oprand is not a part of every instruction. The variable length of this ISA increases its decode complexity. Therefore, all instructions take one cycle longer than the previously described fixed length ISA, i.e., All register and register/immediate operations take 2 cycles, while all load and store instructions take 5 cycles.

Consider the following assembly code sequence (please see comment about what each line of code does.)

```
ADD r3, r1, r2 // r3 = r1+r2
SLL r3, 0x2 // r3 = r3 << 2
MOV r5, 0xa // r5 = 0x0a
STW r3, (r5) //MEMORY[r5] = r3
```

What is the number of cycles taken to execute the code sequence for each of the two ISAs?

A. Fixed length ISA 7 cycles, variable length ISA 11 cycles
B. Fixed length ISA 8 cycles, variable length ISA 12 cycles
C. Fixed length ISA 8 cycles, variable length ISA 11 cycles
D. Fixed length ISA 17 cycles, variable length ISA 12 cycles

Solution:
Fixed Length ISA -- 3 × 1 (other instructions) +1 × 4 (STW instruction) = 7 cycles
Variable Length ISA -- 3 × 2 (other instructions) +1 × 5 (STW instruction) = 11 cycles
Answer is A