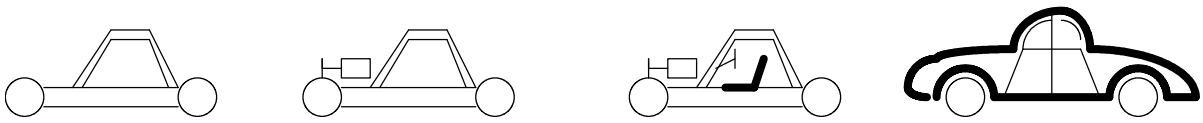
Computer Arch. (CS 2420) Lecture 3 Name:_____

1. Assume that an automobile assembly process takes 4 hours.

Chassis Motor Interior Exterior



a) If the stages take the following amounts of time, then what is the time between completions of automobiles?
Chassis 1 hour
Motor 1 hour
Interior 1 hour
Exterior 1 hour

b) If the stages take the following amounts of time, then what is the time between completions of automobiles?
Chassis 45 minutes
Motor 1 hour
Interior 1 hour and 15 minutes
Exterior 1 hour

2. We could follow the instruction/machine cycle into stages for instruction pipelined.

- Fetch Instruction - read instruction pointed at by the program counter (PC) from memory into Instr. Reg. (IR)
- Decode Instruction - figure out what kind of instruction was read
- Fetch Operands - get operand values from the memory or registers
- Execute Instruction - do some operation with the operands to get some result
- Write Result - put the result into a register or in a memory location

Two approaches for designing a computer is CISC (Complex Instr. Set Computer - pre-1980) and RISC (Reduced Instruction Set Computer post 1985). A CISC philosophy was to make assembly language (AL) as much like a high-level language (HLL) as possible to reduce the "*semantic gap*" between AL and HLL. The rational for CISC at the time was to:

- reduce compiler complexity and aid assembly language programming. Compilers were not too good during the 50's to 70's, (e.g., they made poor use of general purpose registers so code was inefficient) so some programs were written in assembly language.
- reduce the program size. More powerful/complex instructions reduced the number of instructions necessary in a program. Memory during the 50's to 70's was limited and expensive.
- improve code efficiency by allowing complex sequence of instructions to be implemented in microcode. For example, the Digital Equipment Corporation (DEC) VAX computer had an assembly-language instruction "MATCHC *substrLength, substr, strLength, str*" that looks for a substring within a string.

The architectural characteristics of CISC machines include:

- complex, high-level like AL instructions
- variable format machine-language instructions that execute using a variable number of clock cycles
- many addressing modes (e.g., the DEC VAX had 22 addressing modes)

a) Why are complex instructions of CISC (Complex Instr. Set Computer) machines difficult to pipeline?

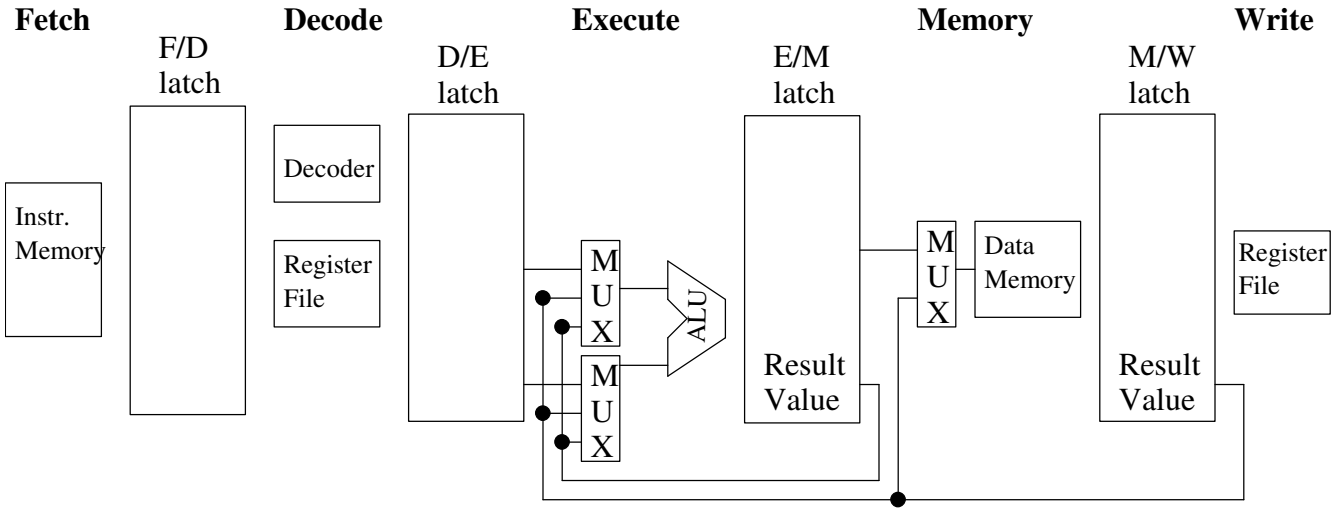
b) Why are RISC machines usually Load & Store machines (i.e., only Load and Store instructions access memory)?

3. The whole question refers to a pipelined, RISC machine with five stages:

- F, fetch - fetch the instruction from memory
- D, decode - determine the type of instruction and read any necessary register values
- E, execute - perform ALU operation or memory address calculation for LOAD or STORE instructions
- M, memory - access memory on LOAD or STORE instruction
- W, write - write register values

a) Complete the following timing diagram assuming NO by-pass signal paths.

Without by-pass signal paths	**Time →**																			
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
ADD R1, R3, R4	F	D	E	M	W															
ADD R2, R4, R5																				
ADD R3, R2, R1																				
LOAD R2, 12(R3)																				
STORE R2, 16(R2)																				



b) Complete the following timing diagram assuming by-pass signal paths as shown above.

With by-pass signal paths	**Time →**																			
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
ADD R1, R3, R4	F	D	E	M	W															
ADD R2, R4, R5																				
ADD R3, R2, R1																				
LOAD R2, 12(R3)																				
STORE R2, 16(R2)																				