Sample Test 1 - Null text Name: Mark F

Computer Architecture Test 1

Question 1. (10 points) Consider the high-level assignment statement X = (A + B + C) * (C - A / B).

a) As in homework #1, write the LOAD and STORE assembly language instructions for this statement.

```
Load R1,A
Load R2,B
Load R3,C
ADD R4, R1,R2
ADD R4, R4,R3
DIV R5, R1,R2
SUB R5, R3,R5
MUL R4,R4,R5
STORE R4,X
```

b) As in homework #1, write the 3-address assembly language instructions for this statement.

```
ADD T,A,B
ADD T,T,C
DIV T2,A,B
SUB T2,C,T2
MUL X,T,T2
```

Question 2. (15 points) How does each of the following RISC (reduced instruction set computers) characteristics aid in an instruction pipeline?

a) fixed-length instruction formats (e.g., all instructions 4 bytes)

Makes the fetch stage relatively short and consistent with respect to memory access -- probably just one memory read of 4-bytes

b) simple addressing modes

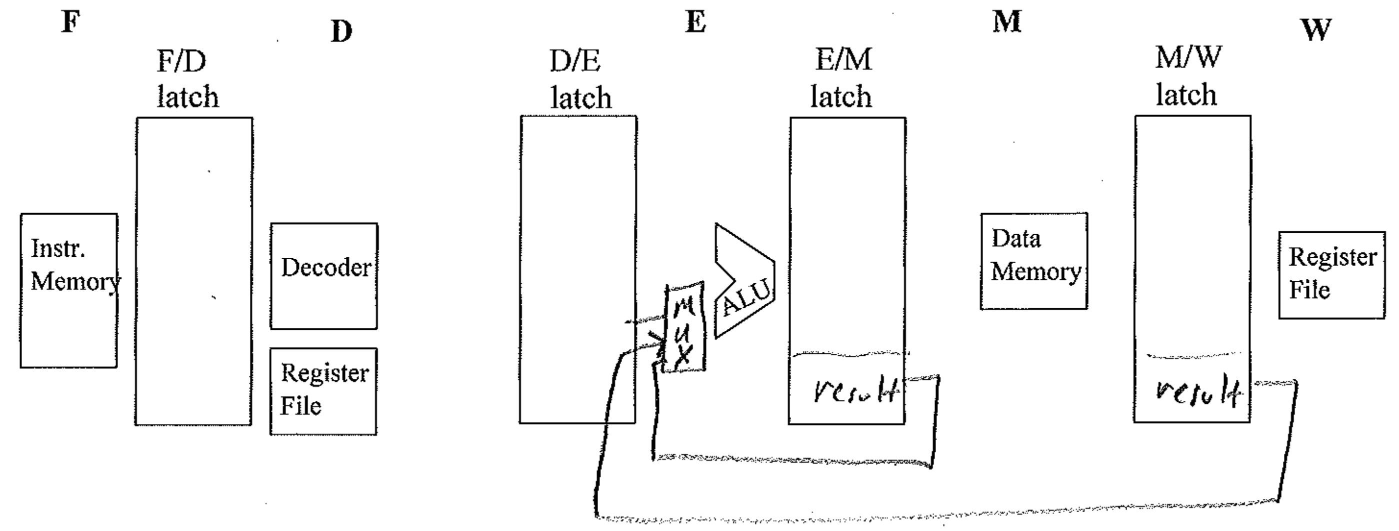
Makes the calculation of effective address and fetching of operands short and consistent

c) register-to-register operations of a LOAD/STORE machine

Makes accessing operands and writing the result short and quick since they are in CPU registers

Sample Test 1 - Null text Name: Mark F.

Question 3. (25 points)



Note that:

- The first register is the destination register, e.g., "ADD R2, R6, R7" performs R2 ← R6 + R7
- LOAD R1, 16(R2) - loads the value from memory at the address 16 + (address in R2) into R1
- STORE R2, 8(R6) - stores the value in R2 to memory at the address 8 + (address in R6)

a) For the five stage pipeline of discussed in class (see above), complete the following timing diagram **assuming NO by-pass signal paths.**

Instructions	**Time →**																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
ADD R2, R3, R4	F	D	E	M	W													
LOAD R5, 8(R2)		F	F	F	F	D	E	M	W									
SUB R6, R5, R2						F	F	F	F	D	E	M	W					
LOAD R7, 4(R6)										F	F	F	F	D	E	M	W	
BGT R7, R8, ELSE														F	F	F	F	D

E M W

b) Complete the following timing diagram **assuming by-pass signal paths.**

Instructions	**Time →**																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
ADD R2, R3, R4	F	D	E	M	W													
LOAD R5, 8(R2)		F	D	E	M	W												
SUB R6, R5, R2			F	D	D	E	M	W										
LOAD R7, 4(R6)				F	F	D	E	M	W									
BGT R7, R8, ELSE						F	D	D	E	M	W							

c) In the diagram at the top of the page, add all by-pass signal paths used in part (b).

Sample Test 1 - Null text Name: Mark F.

Question 4. (25 points) Consider the following sequential search algorithm that searches an array for a specified "target" value. The index of where the "target" value is found is returned. If the "target" value is not in the array, then -1 is returned.

```
SequentialSearch (integer numberOfElements, integer  target, integer array numbers[]) returns an integer
    integer  test;
    for test = 0 to (numberOfElements-1) do  <-- conditional - predict NOT TAKEN
        if number[test] == target then  <-- conditional - predict TAKEN
            return test;
        end if
    end for  <-- unconditional
    return -1;
end SequentialSearch
```

(a)

a) Where in the code would unconditional branches be used and where would conditional branches be used?

b) If the compiler could statically predict by opcode for the conditional branches (i.e., select whether to use machine language statements like: "BRANCH_LE_PREDICT_NOT_TAKEN" or "BRANCH_LE_PREDICT_TAKEN"), then which conditional branches would be "PREDICT_NOT_TAKEN" and which would be "PREDICT_TAKEN"?

c) Under the below assumptions, answer the following questions.

- numberOfElements = 100 and the "target" is **not** found in the array (i.e., an unsuccessful search)
- the five-stage pipeline from class (F, D, E, M, W)
- the target address (address of label being jumped to) of all branches is known at the end of the D stage
- the outcome of conditional branches is known at the end of the E stage

i) If static predict-never-taken is used by the hardware, then what will be the total branch penalty (# cycles wasted) for the algorithm? (Here assume NO branch prediction buffer) For partial credit, explain your answer.

For		if		end for	
2	+	2 x 100	+	1 x 100	= 302
loops 100 times but penalty only when drop out		for unsuccessful search wrong 100 times		loop back 100 times uncond. br. penalty is 1 each.	

ii) If a branch prediction buffer with one history bit per entry is used, then what will be the total branch penalty (# cycles wasted) for the algorithm? (Assume predict-not taken is used if there is no match in the branch prediction buffer) For partial credit, explain your answer.

For	if	end for	
2	2	1	= 5
wrong only when drop out of loop	wrong 1st time when branch is not in branch prediction buffer	wrong 1st time when br. not in BPB.	

iii) If a branch prediction buffer with two history bit per entry is used, then what will be the total branch penalty (# cycles wasted) for the algorithm? (Assume predict-not taken is used if there is no match in the branch prediction buffer) For partial credit, explain your answer.

For	if	end for	
2	2	1	= 5

no nested-loops so 2-bit prediction does not help.

Sample Test 1 - Null text Name: Mark F.

Question 5. (15 points) On a 32-bit computer, suppose we have a 2 GB (2^{31} bytes) memory that is byte addressable, and has a 1 MB (2^{20} bytes) cache with 32 (2^5) bytes per block.

a) How many total lines are in the cache?

$$\#\text{cache lines} = \frac{\text{cache size}}{\text{block size}} = \frac{2^{20}}{2^5} = 2^{15}\text{ lines}$$

b) If the cache is direct-mapped, how many cache lines could a specific memory block be mapped to? 1

c) If the cache is direct-mapped, what would be the format (tag bits, cache line bits, block offset bits) of the address? (Clearly indicate the number of bits in each)

12	15	5
tag	line #	block offset

d) If the cache is fully-associative, how many cache lines could a specific memory block be mapped to?

any line, so 2^{15}

e) If the cache is fully-associative, what would be the format of the address?

27	5
tag	offset

f) If the cache is 8-way set associative, how many cache lines could a specific memory block be mapped to? 8

g) If the cache is 8-way set associative, how many sets would there be?

$$\frac{2^{15}\text{ line}}{2^3\text{ per set}} = 2^{12}\text{ sets}$$

h) If the cache is 8-way set associative, what would be the format of the address?

15	12	5
tag	set #	offset

Question 6. (10 points) Consider the results of running two programs (A and B) on identical processors, except for their cache types:

Cache Type	Program A's Execution Time	Program B's Execution Time
Direct-mapped cache	20 seconds	300 seconds
2-way set-associative cache	15 seconds	302 seconds
4-way set-associative cache	14 seconds	305 seconds

a) For program A, explain why changing from a direct-mapped cache to a 2-way set-associative cache improved performance.

Inside the main loop of A, two (or more) memory blocks must get mapped to the same cache line. This causes these blocks to be re-read from memory each loop.

b) For program B, explain why changing from a direct-mapped cache to a 2-way set-associative cache did not improve performance.

Program B must not map any two blocks within its main loop to the same cache lines, so having a choice multiple lines for a block is not needed.