Midterm Exam

Please, type your solutions or write very neat, use the provided space, and submit only one single pdf file.

Exam is 90 minutes.

Do not chat with me or with anyone else in private.

If you have a question, chat to public (everyone).

Exam is open notes and textbook.

Kindly, make sure that you submitted the pdf file.

Do not ask me if I received or not. If I do not receive, I will contact you.

Questions are equal weight.

Question 1

Find all of the dependencies in the following assembly code.
Be sure to specify the type of dependency as: data dependency, structural dependency, or control dependency. Assume no forwarding and branch will not be taken.
Also be sure to list what register (if any) is involved in the dependency.

```
        Add $S0, $0, $0          # No dependency. First instruction.
Loop: beq  $S0, $S1, done        # Control dependency on $S0.
        Lw$t0, 0($S2)            # No dependency on previous instruction.
        Addi $S2, $S2, 4         # No dependency on previous instruction.
        Add $t0, $t0, 5          # No dependency on previous instruction.
        Sw $t0, 0 ($S4)          # Data dependency on $t0.
        Addi $S4, $S4, 4         # No dependency on previous instruction.
        Addi $S0, $S0, 1         # No dependency on previous instruction.
        J Loop                   # No dependency on previous instruction.
done: ….
```

Solution

Question 2

Pipelining (assuming any unnecessary stall cycles will be considered wrong answer).

a) Show the sequence of execution (timing diagram IF-ID-EXE-MEM-WB) for the following instruction sequence assuming no forwarding but hazard detection and stalling where necessary. You should assume that we can write back and decode instructions in the same clock cycle. Assume each stage takes a single clock cycle. Assume the branch is not taken and assume separate I/D memory.

Add $S0, $0, $0	**F**	**D**	**X**	**M**	**W**																			
Loop: beq $S0, $S1, done			**F**	**S**		**S**	**D**	**X**	**M**	**W**														
Lw$t0, 0($S2)							**F**	**D**	**X**	**M**	**W**													
Addi $S2, $S2, 4								**F**	**D**		**X**		**M**		**W**									
Add $t0, $t0, 5									**F**		**D**			**X**		**M**	**W**							
Sw $t0, 0 ($S4)										**F**		**S**			**S**	**D**		**X**	**M**	**W**				
Addi $S4, $S4, 4																**F**		**D**	**X**	**M**	**W**			
Addi $S0, $S0, 1																			**F**	**D**	**X**	**M**	**W**	
J Loop																				**F**	**D**	**X**	**M**	**W**

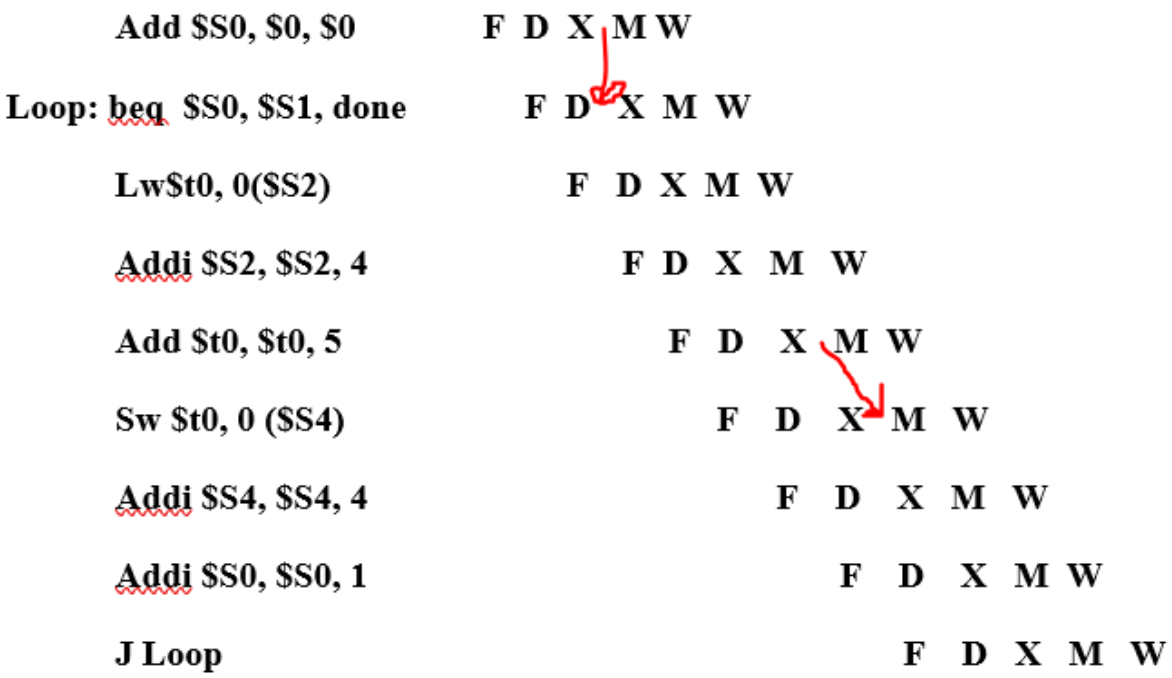
Done: ….

Solution

See the above timing diagram.

b) Assume standard forwarding with the five stage pipeline, show the sequence of execution (timing diagram IF-ID-EXE-MEM-WB) for the above sequence. Show the forwarding by an arrow. Assume the branch is not taken.

Solution



Question 3

For the given sequence of word addresses, show the sequence hits and misses along with the cache contents at each reference. The cache is fully-associative and supports **2-word block size** (i.e. each block has two words). Assume the cache uses a LRU strategy for replacement and it is initially empty and it can hold only five blocks.

Word address sequence (READ): 0, 3, 10, 1, 4, 12, 20, 21, 5, 9

Solution

Feel free to use the provided table.

The table below shows the cache as 5 locations and you should show for each user request/reference if it is a miss or a hit.

I am not sure what you want us to do here. I know that LRU kicks the least recently used block out of the cache.

Read Sequence or Requests										
Cache 1st block					0	3	10	1	4	12
Cache 2nd block				0	3	10	1	4	12	20
Cache 3rd block			0	3	10	1	4	12	20	21
Cache 4th block		0	3	10	1	4	12	20	21	5
Cache 5th block	0	3	10	1	4	12	20	21	5	9
Hit or Miss	Miss	Miss	Miss	Miss	Hit	Hit	Hit	Hit	Hit	Hit

Question 4

Unroll the following loop with a factor 4 (4 iterations) and optimize the code sequence assuming the following stalling procedures. Assume that: An ALU operation followed by a LW incurs a stall of 2 cycles (when there is a data dependency). An ALU operation whose output is used by another ALU instruction incurs a stall of 1 cycle. An ALU operation whose output is used by a SW incurs a stall of 3 cycles.

```
add $s0, $0, $0                 # i = 0
loop: beq $s0, $s1, done        # Jump to done when $s0=$s1
lw $t0, 0($s2)                  # Load B[i+1]
addi $s2, $s2, 4                # s2+=4
add $t0, $t0, 5                 # B[i+1]+5
sw $t0, 0($s4)                  # A[i]=B[i+1]+5
addi $s4, $s4, 4                # s4+=4
addi $s0, $s0, 1                # i++
j loop                          # repeat
done: ...                       # no instruction here
```

Solution

```
add $s0, $0, $0                 # i = 0
stall
loop: beq $s0, $s1, done        # Jump to done when $s0=$s1
lw $t0, 0($s2)                  # Load B[i+1]
addi $s2, $s2, 4                # s2+=4
add $t0, $t0, 5                 # B[i+1]+5
addi $s4, $s4, 4                # s4+=4
sw $t0, -4($s4)                 # A[i]=B[i+1]+5
addi $s0, $s0, 1                # i++
lw $t0, 0($s2)                  # Load B[i+2]
addi $s2, $s2, 4                # s2+=4
add $t0, $t0, 5                 # B[i+2]+5
addi $s4, $s4, 4                # s4+=4
sw $t0, -4($s4)                 # A[i]=B[i+2]+5
```

```
addi $s0, $s0, 1                # i++
lw $t0, 0($s2)                  # Load B[i+3]
addi $s2, $s2, 4                # s2+=4
add $t0, $t0, 5                 # B[i+3]+5
addi $s4, $s4, 4                # s4+=4
sw $t0, -4($s4)                  # A[i]=B[i+3]+5
addi $s0, $s0, 1                # i++
lw $t0, 0($s2)                  # Load B[i+4]
addi $s2, $s2, 4                # s2+=4
add $t0, $t0, 5                 # B[i+4]+5
addi $s4, $s4, 4                # s4+=4
sw $t0, -4($s4)                  # A[i]=B[i+4]+5
addi $s0, $s0, 1                # i++
j loop                          # repeat
done: ...                       # no instruction here
```

Question 5

Cache hierarchy

You are building a computer system with in-order execution that runs at 1GHz and has a CPI of 1 when no memory accesses.

The memory system is split L1 cache. Both the instruction I-cache and the data D-cache are direct mapped and hold 32KB each with block size 64 bytes. (I means instructions and D means Data).

The I-cache has a 2% miss rate, and the D-cache is a write-through with 5% miss rate.

The hit cycles for both the I-cache and the D-cache take 1 cycle. (1 cycle takes 1ns).

The L2 cache is a unified write-back with a total size of 512KB and a block size of 64 bytes.

The hit cycles of the L2 cache is 15 cycles (15nsec). The local hit rate of the L2 cache is 80%.

Given: Hit time= Hit Rate* Hit Cycles

L2 miss penalty = 100nsec

L2 data miss penalty takes extra 15ns.

Compute the AMAT for both Instruction and Data memories

Solution

L1 cache miss rate = 0.02
L1 cache hit rate = (1 – L1 cache miss rate) = 0.98
L2 cache hit rate = 0.80
L2 cache miss rate = 1 – L2 cache hit rate = 0.20
L1 cache hit time = 1ns
L2 cache hit time = 15ns
L2 cache miss penalty = 100ns

AMAT = Hit Time + (Miss Rate * Miss Penalty)

Instruction AMAT = 0.98 * 1ns + (15ns * 0.80 + (0.20* 100))*0.02 = 1.62ns
Data Memory AMAT = 0.98 * 1ns + (15ns*0.8 + (0.20*100)*0.20)*0.02 = 1.3ns

Total AMAT = 1.62ns + 1.3 + 1= 3.92 ns