EECS 470 Winter 2025
HW1 solutions

1a)

```
Loop:   DADDI   R2, R2, #4
        DSUB    R4, R3, R2
        LD      R1, 0(R2)
        DADDI   R1, R1, #1
        SD      0(R2), R1
        BNEZ    R4, Loop
```

* denotes stall in stage.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
DADDI R2, R2, #4	IF	ID	EX	MEM	WB											
DSUB R4, R3, R2		IF	ID*	ID*	ID	EX	MEM	WB								
LD R1, 0(R2)			IF*	IF*	IF	ID	EX	MEM	WB							
DADDI R1, R1, #1						IF	ID*	ID*	ID	EX	MEM	WB				
SD 0(R2),R1							IF*	IF*	IF	ID*	ID*	ID	EX	MEM	WB	
BNEZ R4, Loop										IF*	IF*	IF	ID	EX	MEM	WB

It takes 16 cycles for one iteration of this loop to execute.

1b)

	1	2	3	4	5	6	7	8	9	10	11
DADDI R2, R2, #4	IF	ID	EX	MEM	WB						
DSUB R4, R3, R2		IF	ID	EX	MEM	WB					
LD R1, 0(R2)			IF	ID	EX	MEM	WB				
DADDI R1, R1, #1				IF	ID*	ID	EX	MEM	WB		
SD 0(R2),R1					IF*	IF	ID	EX	MEM	WB	
BNEZ R4, Loop							IF	ID	EX	MEM	WB

This loop takes 11 cycles to execute

2)

(assume Q=0 is left state)

Q	A	B	Q*
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Q*= A + QB
X = Q

3a)

Cache has 1MB/64B=16384 lines. 8-way set associative means 2048 sets.
#bits of byte offset=lg(64)=6
#bits of set index=lg(2048)=11
#bits of tag= 40-11-6=23

3b) 23(tag bits)*16384(lines)/8(bits/byte)=47,104 bytes (46 kB)

3c)
Fully associative means tags are: 40-6=34 bits → 34*16384/8=69,632 bytes (68 kB)
Dir. mapped mans tags are:40-14(log2 16384)-6=20 bits → 20*16384/8=40,960 bytes (40kB)

4a)

RISC	**CISC**
Fixed instruction format	Many formats and lengths
Single cycle execution	Many multicycle operations
Hardwired control	Microcoded multi-cycle operations
Load/store architecture	Register-memory and memory-memory
Few memory addressing modes	Many modes
Reliance on compiler optimizations	Hand assemble to get good performance

4b)

- must save/restore on procedure calls
- cannot take address of register
- fixed size (cannot fit FP, strings, structures/records)
- compilers must manage
- longer register specifiers
- slower registers (increased access time)
- more state slows context switches

5) Use DVFS to evaluate if the tradeoff is worth it.

Method 1 - Reduce original V and F to match performance, see if power is lower
Doing DVFS to match the 85% performance of reducing cache lines leads to a power draw of ((0.85)^3) * 100W = 61 W
Since this is < 75W, making staying with the original design still the better choice.
Method 2 - Reduce original V and F to match power, see if performance is higher
Doing DVFS to match the 75W of reducing cache size leads to a performance of (75/100)^(⅓) = 90.8,
Since this is a performance > the 85% of the proposed mode, it makes staying with the original still the better choice.
Method 3 - Calculate ED2P (MIPS^3/W) and see if any are better
Start assuming this original gets 100 MIPS (any number here just so we can compare)
Baseline: MIPS^3/W = 100^3 / 100 = 10,000
New design: It sees a 15% lower MIPS at 75W, so 85^3 / 75 = 8,188
Baseline gets more MIPS^3/W, so Baseline is better

6a) The quad core performance is:
1 / ((1 – 0.9) + (0.9 / 4)) * 0.8 * 100 trans/sec = 246 trans/sec
The 8-core performance is:
1 / ((1 – 0.9) + (0.9 / 8)) * 0.65 * 100 trans/sec = 306 trans/sec
The 8-core is the fastest

6b) The single-core energy per transaction is (recall: 1W = 1J/s):
80 J/s / 100 trans/sec = 0.8 J/trans
The quad-core:
100 J/s / 246 trans/sec = 0.41 J/trans
The 8-core:
135 J/s / 306 trans/sec = 0.44 J/trans
→ The 4-core is most energy-efficient.

7a) Use geometric mean to summarize ratios (speedups).

7b) $\sqrt[3]{1.42 * 1.13 * 1.77}$
= 1.47