CMP 334 Practice Exam 2 (Spring 2019)

1) For each row in the following table, determine whether the assertion would hold if the indicated operation produced the indicated condition flag values.

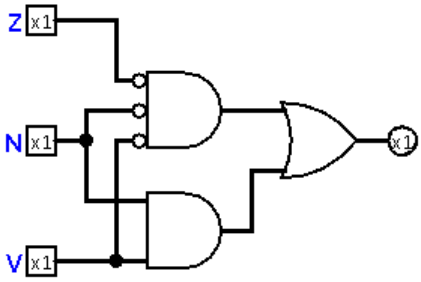
	operation		flags	assertion	T / F	why	A−B
a	A +ⁿ B	unsigned	$\overline{Z}N\overline{C}V$	result is honest	T	$\overline{C}$	?
b	A +ⁿ B	signed	$\overline{Z}N\overline{C}V$	result is honest	T	$\overline{V}$	?
c	A −ⁿ B	unsigned	$\overline{Z}N\overline{C}V$	result is honest	T	C	+
d	A −ⁿ B	signed	$\overline{Z}N\overline{C}V$	result is honest	F	V	–
e	A −ⁿ B	unsigned	$\overline{Z}N\overline{C}V$	A ≤ B	T	$\overline{C}$	–
f	A −ⁿ B	signed	$\overline{Z}N\overline{C}V$	A < B	T	$N\overline{V}$	–
g	A −ⁿ B	unsigned	$\overline{Z}N\overline{C}V$	A ≥ B	T	C	+
h	A −ⁿ B	signed	$\overline{Z}N\overline{C}V$	A < B	F	NV	+
i	A −ⁿ B	unsigned	$Z\overline{N}\overline{C}V$	A ≤ B	T	Z	0
j	A −ⁿ B	signed	$\overline{Z}N\overline{C}\overline{V}$	A > B	F	$N\overline{V}$	–
k	A −ⁿ B	unsigned	$\overline{Z}N\overline{C}V$	A < B	T	$\overline{C}$	–
l	A −ⁿ B	signed	$\overline{Z}N\overline{C}V$	A ≤ B	T	$\overline{N}V$	–
m	A −ⁿ B	unsigned	$\overline{Z}N\overline{C}\overline{V}$	A ≥ B	T	C	+
n	A −ⁿ B	signed	$Z\overline{N}\overline{C}\overline{V}$	A > B	F	Z	0

A−B ≥ 0	C	$\overline{N}V$	(true positive)	NV	(false negative)
A−B = 0	Z			
A−B < 0	$\overline{C}$	$N\overline{V}$	(true negative)	$\overline{N}V$	(false positive)

2) Draw a Boolean circuit, having the condition flags – **Z**, **N**, **C**, and **V** – as inputs, that will be 1, following the execution of the instruction: `sub $0, $5, $6` if the signed integers **A** and **B** in registers $5 and $6 (respectively) satisfy the relationship **A > B**.

(signed) **A > B** ==> **A−B > 0** ==> **A−B > 0 & A−B ≠ 0**

==> $(\overline{NV} + NV)$ **&** $\overline{Z}$ = $\overline{ZNV} + \overline{Z}NV$ = $\overline{ZNV} + NV$



4) Write a **TOY** AL subprogram to accomplish the following assignment:

A = (A + B) & (17 − B)

consistent with the following subprogram interface:

Label: AB17

On entry:

Register **$1** is the return address of the caller.

@A (the address in memory of variable **A**) is in register **$A**.

@B (the address in memory of variable **B**) is **@A+7**.

On exit:

Variable **A** will have been updated; no other values in main memory will have changed.

Registers **$4** to **$F** may have been changed; registers **$0** to **$3** will not have been.

```
AB17
     lis  $4, @A        :
     lih  $4, @A        :    $4 = @A
     l    $A, $4,  0    :    $A =  A
     l    $B, $4,  7    :    $B =  B
     lis  $C,     17    :    $C = 17
     add  $5, $A, $B    :    $5 = A + B
     sub  $6, $C, $B    :    $6 = 17 - B
     and  $7, $5, $6    :    $7 = (A + B) & (17 - B)
     st   $7, $4,  0    :     A = (A + B) & (17 - B)
     bcl  ALL,$1, $0    :    return to caller
```

5) Write a **TOY** AL subprogram to negate each of the elements of an array **A**, consistent with the following subprogram interface:

Label: `Negate`

On entry:

Register **$1** is the return address of the caller.

Register **$B**. contains n the number of elements in the array **A**.

@B (the address in memory of variable **B**) is **@A+7**.

On exit:

The elements of **A** will have been negated; other memory values will be unchanged.

Registers **$4** to **$F** may have been changed; registers **$0** to **$3** will not have been.

```
Negate
     lis  $4,      @A
     lih  $4,      @A          :      $4 = @A
     lis  $5,      @B
     lih  $5,      @B          :      $5 = @B
     l    $B, $5,   0          : n = $B
     lis  $8,       0          : j = $8 = 0
     lis  $9,       1          :      $9 =  1
Loop
     sub  $0, $8, $B           : j ? n
     bc   UGE, Done

     add  $7, $4, $8           :      $7 = @A[j]
     l    $6, $7,   0          : t = $6 =  A[j]

     sub  $6, $0, $6           : t =      -A[j]

     st   $6, $7,   0          :   A[j] = -A[j]
     bc   ALL, Loop
Done
     bcl  ALL,$1, $0           : return to caller
```

3) Write a **TOY** AL subprogram to determine the number of distinct values in three registers (that is: **3**, if all three values are different; **2**, if two registers have the same value but the other is not; and **1**, if all three resisters are equal) consistent with the following subprogram interface:

Label: `Diff`

On entry:

Register **`$1`** is the return address of the caller.

The registers in question are **`$A`**, **`$B`**, and **`$C`**.

`@B` (the address in memory of variable `B`) is **`@A+7`**.

On exit:

The result will be in register **`$F`**.

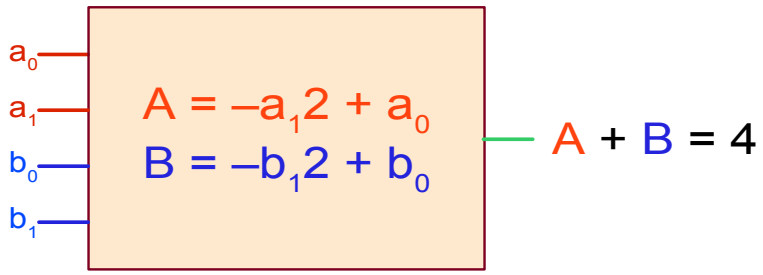
Registers **`$4`** to **`$F`** may have been changed; registers **`$0`** to **`$3`** will not have been.

Main memory will be unchanged.

```
Diff
     sub  $0, $A, $B          : a ? b ? c ? a
     bc   EQ, NotThree
     sub $0, $B, $C           : a≠b ? c ? a
     bc  EQ, Two
     sub $0, $C, $A           : a≠b≠c ? a
     bc  EQ, Two
     lis $F,       3          : a≠b≠c≠a
     bcl ALL,$1, $0           : return 3
Two                           : a≠b=c, c=a≠b, or a=b≠c
     lis $F,       2
     bcl ALL,$1, $0           : return 2
NotThree
     sub $0, $B, $C           : a=b ? c ? a
     bc  NE, Two
     lis $F,       1          : a=b=c
     bcl ALL,$1, $0           : return 1
```

6) Follow the four step combinational circuit design process outlined below to design a circuit for a Boolean function that takes as input two 2-bit ***signed*** integers, **A** and **B**, and outputs 1 if, and only if, their difference, **A – B**, is **4**.

a) Draw a black box for the circuit that specifies its inputs and output.



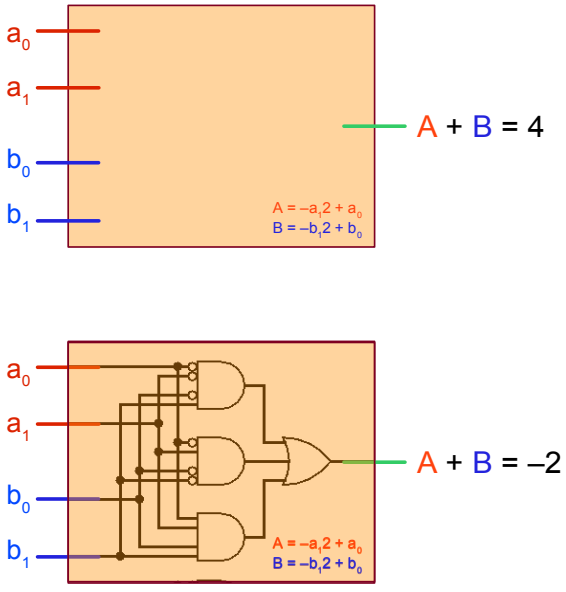
b) Formalize the informal semantics of this circuit with a truth table.

b_1	b_0	B	a_1	a_0	A	A+B	A+B = 4	A+B = –2	*term*
0	0	**0**	0	0	**0**	**0**	0	0	
0	0	**0**	0	1	**1**	**1**	0	0	
0	0	**0**	1	0	**–2**	**–2**	0	**1**	$\bar{b}_1\bar{b}_0a_1\bar{a}_0$
0	0	**0**	1	1	**–1**	**–1**	0	0	
0	1	**1**	0	0	**0**	**1**	0	0	
0	1	**1**	0	1	**1**	**2**	0	0	
0	1	**1**	1	0	**–2**	**–1**	0	0	
0	1	**1**	1	1	**–1**	**0**	0	0	
1	0	**–2**	0	0	**0**	**–2**	0	**1**	$b_1\bar{b}_0\bar{a}_1\bar{a}_0$
1	0	**–2**	0	1	**1**	**–1**	0	0	
1	0	**–2**	1	0	**–2**	**–4**	0	0	
1	0	**–2**	1	1	**–1**	**–3**	0	0	
1	1	**–1**	0	0	**0**	**–1**	0	0	
1	1	**–1**	0	1	**1**	**0**	0	0	
1	1	**–1**	1	0	**–2**	**–3**	0	0	
1	1	**–1**	1	1	**–1**	**–2**	0	**1**	$b_1b_0a_1a_0$

c) Construct the boolean formula corresponding to the truth table.

$$\bar{b}_1\bar{b}_0a_1\bar{a}_0 + b_1\bar{b}_0\bar{a}_1\bar{a}_0 + b_1b_0a_1a_0$$

d) Draw the circuit corresponding to the boolean formula.



7) For the following distribution of instructions

40% ALU instructions
20% Load instructions
15% Store instructions
20% Conditional branch instructions
5% Unconditional branch instructions

a) How many memory accesses are required for each instruction (on average)?

MAPI = # fetches + # Loads + # Stores = 1 + 0.2 + 0.15 = 1.35

b) On a multi-cycle per instruction processor: ALU instructions require 5 cycles; loads, 9 cycles; stores, 7 cycles; **taken** conditional branches, 8 cycles; not taken conditional branches, 6 cycles; and unconditional branches, 4 cycles. If one fifth of all conditional branches are taken, what is the CPI for this distribution of instructions?

CPI = 0.4·5 + 0.2·9 + 0.15·7 + 0.2·0.2·8 + 0.2·0.8·6 + 0.05·4
CPI = 2 + 1.8 + 1.05 + 0.32 + 0.96 + 0.2
CPI = 4 + 0.8 + 0.05 + 0.32 + 0.96 + 0.2
CPI = 5 + 0.05 + 0.32 + 0.96
CPI = 6.33

For the parts **c**, **d**, and **e**, assume that the processors execute instructions in the following stages:

IF — 100 ps
ID — 100 ps
EX — 200 ps
MEM — 250 ps
WB — 150 ps.

c) What would be the cycle time of a single-cycle-per-instruction-execution processor having these stages?

100 + 100 + 200 + 250 + 150 = 800 ps

d) What would be the cycle time of a pipeline processor with these stages?

max (100, 100, 200, 250, 150) = 250 ps

e) What would be the speedup of the pipeline processor in **d** over the single-cycle processor of **a**?

800 / 250 = 3.2

8) Assuming the same distribution of instructions as in problem 7, compute the stall cycles per instruction SCPI due to each of the following four hazards:

a) a 1 cycle delay for 1 of every 20 load instructions

$0.2 \cdot 0.05 \cdot 1 = 0.01$

b) a 2 cycle delay for a quarter of the conditional branch

$0.2 \cdot 0.25 \cdot 2 = 0.1$

c) a 40 cycle delay for 1 of every 900 memory accesses

$1.35 \cdot 40 \cdot 1/900 = 54/900 = 0.06$

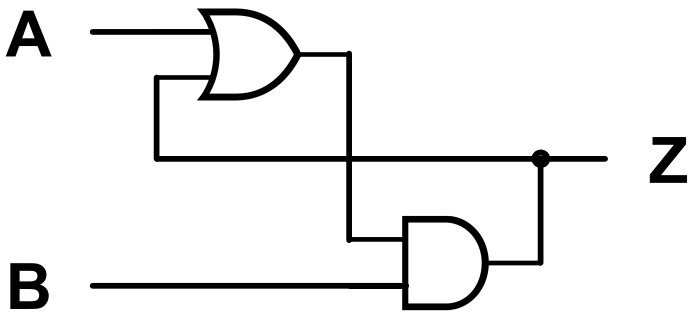
d) a 5,000 cycle delay for 3 of every 100,000 instructions

$0.00003 \cdot 5000 = 0.03 \cdot 5 = 0.15$

e) What is would be the actual CPI for a pipeline processor experiencing these four hazards (assuming that its perfect pipeline CPI = 1)?

$1 + 0.01 + 0.1 + 0.06 + 0.15 = 1.32$

9) Complete the transition truth table for the following circuit:



A	B	$Z_{(old)}$	Z (new)	state
0	0	0	0	RESET
0	0	1	0	
0	1	0	0	LATCH
0	1	1	1	
1	0	0	0	RESET
1	0	1	0	
1	1	0	1	SET
1	1	1	1	

10) Program Q takes 45 seconds to run on processor X and 18 seconds to run on processor Y. The clock rate for X is 2 GHz and for Y is 4 GHz. The CPI of Y is 3.6. Both processors implement the same instruction set architecture.

processor	seconds	instructions	CPI	clock rate
X	45	I	?	2 GHz
Y	18	I	3.6	4 GHz

a) What is the relative performance of Y to X?

$$\frac{P_Y}{P_X} = \frac{T_X}{T_Y} = \frac{45}{18} = 2.5$$

Y is 2.5 times as fast as X.

b) How many instructions are executed running program Q?

$$18 = T_Y \text{ seconds} = \frac{\#\text{ instructions}_Y \cdot \text{CPI}_Y}{\text{clockRate}_Y} = \frac{I \cdot 3.6 \text{ cycles/instruction}}{4 \cdot 10^9 \text{ cycles / second}}$$

$$I = \frac{18 \cdot 4 \cdot 10^9}{3.6} = 20 \cdot 10^9 \text{ instructions}$$

c) What is the CPI for X?

$$45 = T_X \text{ seconds} = \frac{\#\text{ instructions}_X \cdot \text{CPI}_X}{\text{clockRate}_X} = \frac{20 \cdot 10^9 \text{ instructions} \cdot \text{CPI}_X}{2 \cdot 10^9 \cdot \text{cycles / second}}$$

$$\text{CPI}_X = \frac{45 \cdot 2 \text{ cycles}}{20 \text{ instructions}} = 4.5 \text{ cycles/instruction}$$