CMP 334 practice final (Spring 2019)

1) Given: X = 0x6E and Y = 0x9B,

a) Convert X and Y to 8-bit binary numbers. X=01101110 Y=10011011

b) Compute the 8-bit sum $\mathbf{X \ddot{+} Y}$ of X and Y.

```
 01101110
 10011011 +
11111110
100001001
```

c) Compute $\ddot{\mathbf{Y}}$ the 8-bit two's complement of Y. 01100100

d) Compute the 8-bit difference $\mathbf{X \ddot{-} Y}$ of X and Y. (Use two's complement addition.)

```
 01101110
 01100100 +
01101100
011010010
```

e) Convert $\mathbf{X \ddot{+} Y}$, $\ddot{\mathbf{Y}}$, and, $\mathbf{X \ddot{-} Y}$ to hexadecimal. 0x09 0x64 0xD2

f) What are the values of the condition flags **z n c v** upon computing $\mathbf{X \ddot{+} Y}$? **0 0 1 0**

g) What are the values of the condition flags **z n c v** upon computing $\mathbf{X \ddot{-} Y}$? **0 1 0 1**

h) T or **F** The *unsigned* 8-bit sum $\mathbf{X \ddot{+} Y}$ is honest. **c=1**

i) **T** or F The *signed* 8-bit sum $\mathbf{X \ddot{+} Y}$ is honest. **v=0**

j) T or **F** The *unsigned* 8-bit difference $\mathbf{X \ddot{-} Y}$ is honest. **c=0**

k) T or **F** The *signed* 8-bit difference $\mathbf{X \ddot{-} Y}$ is honest. **v=1**

2) Use the combinational circuit design process:

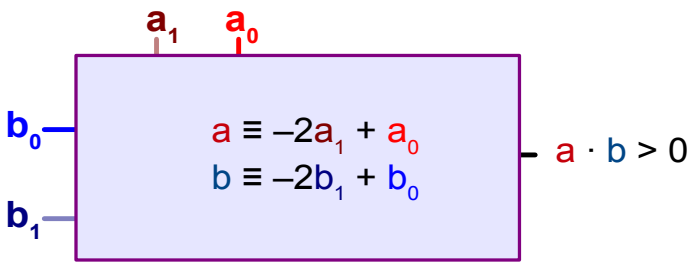
a) Draw a black box for the circuit that specifies its inputs and output.

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

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

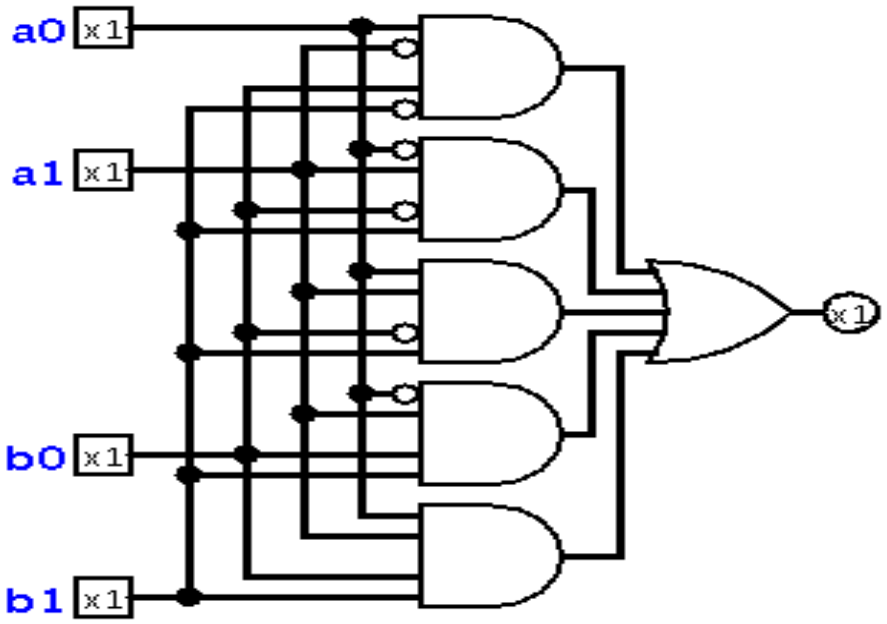
d) Draw the circuit corresponding to the boolean formula.

to design the circuit that takes as input two 2-bit ***signed*** integers **a** and **b** and is **1** if **a•b** > 0



b_1	b_0	a_1	a_0	b	a	a · b	a · b > 0	
0	0	0	0	0	0	0	0	
0	0	0	1	0	1	0	0	
0	0	1	0	0	–2	0	0	
0	0	1	1	0	–1	0	0	
0	1	0	0	1	0	0	0	
0	1	0	1	1	1	1	1	$\bar{b}_1 b_0 \bar{a}_1 a_0$
0	1	1	0	1	–2	–2	0	
0	1	1	1	1	–1	–1	0	
1	0	0	0	–2	0	0	0	
1	0	0	1	–2	1	–2	0	
1	0	1	0	–2	–2	4	1	$b_1 \bar{b}_0 a_1 \bar{a}_0$
1	0	1	1	–2	–1	2	1	$b_1 \bar{b}_0 a_1 a_0$
1	1	0	0	–1	0	0	0	
1	1	0	1	–1	1	–1	0	
1	1	1	0	–1	–2	2	1	$b_1 b_0 a_1 \bar{a}_0$
1	1	1	1	–1	–1	1	1	$b_1 b_0 a_1 a_0$

$$\bar{b}_1 b_0 \bar{a}_1 a_0 + b_1 \bar{b}_0 a_1 \bar{a}_0 + b_1 \bar{b}_0 a_1 a_0 + b_1 b_0 a_1 \bar{a}_0 + b_1 b_0 a_1 a_0 = b_1 a_1 + \bar{b}_1 b_0 \bar{a}_1 a_0$$



3) Write **TOY** AL subprogram that implements the following subprogram interface:

Label: `ArrayAnd`

On entry:

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

A, **B**, and **C** are non overlapping arrays in memory.

Register **`$D`** contains **n**, the size of all three arrays.

On exit:

C[i] = **A**[i] & **B**[i] for all **i** such that 0 ≤ **i** < **n**.

Otherwise, no values in memory have changed.

Any of the registers may have changed value.

```
ArrayAnd  lis  $9,  1          :    $9   =   1
          lis  $8,  1          :    $8   =   1 = j
          lis  $A, @A
          lih  $A, @A          :    $A   =  @A
          lis  $B, @B
          lih  $B, @B          :    $B   =  @B
          lis  $C, @C
          lih  $C, @C          :    $C   =  @C
Loop      sub  $0, $8, $D      :     j   ?  $D
          bc   UGE, Done
          add  $2, $A, $8      :    $2   =  @A[j]
          add  $3, $B, $8      :    $3   =  @B[j]
          add  $4, $C, $8      :    $4   =  @C[j]
          l    $5, $2,  0      :    $5   =   A[j]
          l    $6, $3,  0      :    $6   =   B[j]
          add  $7, $5, $6      :    $7   =   A[j] + B[j]
          st   $7, $4,  0      :    C[j] =   A[j] + B[j]
          add  $8, $8, $9      :     j   =   j + 1
          bc   ALL, Loop
Done      bcl  ALL, $1, $0     :    return
```

4) Write **TOY** AL subprogram that implements the following subprogram interface:

Label: `Mult`

On entry:

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

On exit:

Register **`$F`** contains the (unsigned) product of the unsigned values in registers **`$A`** and **`$B`**. (Note: your result need not be accurate unless these values are less than 256.)

Main memory will not have changed, but any of the registers may have.

```
Mult    lis   $F,  0          :    $F   =   0
        lis   $9,  1          :    $9   =   1
Loop    sub   $0, $B, $0      :    $B   ?   0
        bc    EQ, Done
        add   $F, $F, $A      :    $F   =   $F + $A
        sub   $B, $B, $9      :    $B   =   $B - 1
        bc    ALL, Loop
Done    bcl   ALL, $1, $0     :    return
```

5 a) Computer A had a processor that connects to an L_1 cache with a hit-cost of 1 cycle and a miss-rate of 4%. The L_1 cache connects to a main memory with hit-cost of 600 cycles. What is the ***memory access cost*** of computer A?

$MAC_A =$ 1 + 0.04 * (600) = 25

b) Computer B is computer A with a L_2 cache between the L_1 cache and main memory with hit-cost of 15 cycles and a miss-rate of 10%. What is the ***memory access cost*** of computer B?

$MAC_B =$ 1 + 0.04 * (15 + 0.1*(600)) = 1 + 0.04*75 = 1 + 3 = 4

c) Computer C is computer B with an L_3 cache between the L_2 cache and main memory with a hit-cost of 30 cycles and a miss-rate of 20%. What is the ***memory access cost*** of computer C?

$MAC_C =$ 1 + 0.04 * (15 + 0.1*(30 + 0.2*(600))) = 1 + 0.04*(15 + 0.1*150) = 2.2

d) On program P with an execution consisting of 50% ALU instructions, 20% load instructions, 5% store instructions, and 25% branch instructions, how many memory accesses per instruction are there on computer C? How many ***stall cycles per instruction*** are due to memory accesses on execution of program P on computer C?

SCPI = MAPI * (MACC – 1) = 1.25 *1.2 = 1.5

6) Consider a disk with the following specifications:

block size	512	bytes
sectors per track	400	
overhead	2.5	ms
seek time	20	ms
rotation speed	12000	RPM

a) How big is a track on this disk in bytes?

1 track = 512 bytes ~~per block~~ · 400 ~~blocks~~ = 204800 bytes

b) What is its rotation time in ms?

1/12000 · 60/1 * 1000/1 = 60/12 = 5 ms

c) What is its bandwidth in megabytes per second?

204800 / 5 * 1000 /1000000 = 40960 / 1000 = 40.96 Megabytes per second

d) What would be the data transfer time in ms of a record 768,000 bytes long?

76800 / 4096 = 4800 / 256 = 300 /16 = 75 ms

e) What would be the total disk access time in ms for this record?

2.5 + 20 + 5/2 + 75 = 100 ms

7) Consider the following TOY Assembly Language instructions:

A)	`add,`	B)	`and,`	C)	`bc,`	D)	`l,`
E)	`lis,`	F)	`nor,`	G)	`st,`	H)	`sub,`

a) Which of these instructions can change the value of a register? A, B, D, E, F, H

b) Which can change the value of a location in memory? G

c) Which can change the program counter? C

d) Which can change the value of a condition flag? A, B, F, H

e) Which use an immediate value contained in the instruction? C, D, E, G