CS2100 Computer Organization
AY2022/23 Semester 2
Assignment 2

PLEASE READ THE FOLLOWING INSTRUCTIONS VERY CAREFULLY.

1. The **deadline** for this assignment is **12 NOON SUNDAY 16 APRIL 2023** The submission folders will close shortly after 12 noon and no submissions will be allowed after that. You WILL receive 0 mark if you do not submit on time.
2. **We will be enforcing the 0-mark for late submission rule very strictly! You WILL get ZERO (0) if you submit even 1 second after the folder closes!**
3. You should complete this assignment **on your own** without collaborating or discussing with anyone. If you have been found to have cheated in this assignment, you will receive an F grade for this module as well as face other disciplinary sanctions. Take this warning seriously.
4. Please keep your submission **short**. You only need to submit your answers; you do not need to include the questions.
5. If you submit multiple copies, one copy will be graded. This is not necessarily the last copy submitted.
6. Answers may be typed or handwritten. In case of the latter, please ensure that you use **dark ink** and your handwriting is **neat and legible**. Marks may be deducted for untidy work or illegible handwriting.
7. Submit your assignment **only in PDF format**. Please name your file AxxxxxY.pdf, where AxxxxxxY is your student number.
8. Submit your assignment on Canvas in the folder created for your **TUTORIAL GROUP**.
9. The questions are worth **50 MARKS** in total.
10. If you have any queries on this assignment, please post them on the Canvas forums.

=== END OF INSTRUCTIONS ===

Question 0. Submission instructions (-2 marks)

You do not need to answer this question. Your tutor will deduct the marks accordingly if appropriate.

(a) You did not name your file appropriately, i.e., **AxxxxxxxY.pdf** [-1 mark]

(b) You do not have your **name** and **tutorial group number** written at the top of the first page of your submitted answer file. [-1 mark]

Question 1. Boolean Algebra and Combinational Circuits (15 marks)

a. Use the laws/theorems of Boolean Algebra to simplify the following expression, **writing your answer as a sum of products**.

You are not allowed to use K-Maps or any other methods to solve this problem. If you use a K-Map or any other method to solve this question **you will not get any marks.**

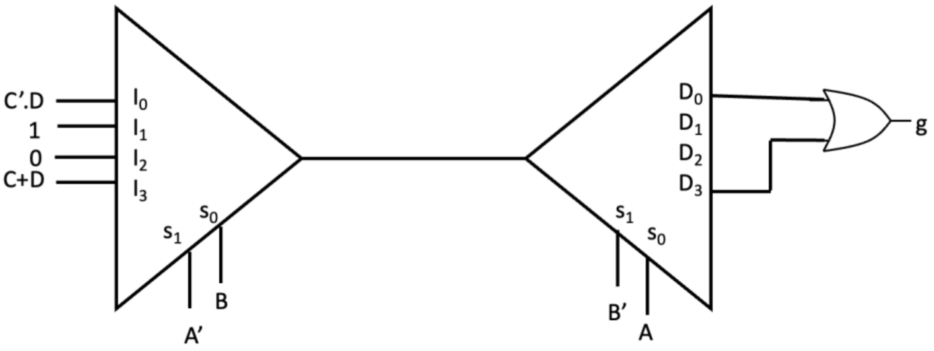
You must write down the laws/theorems you used, or **marks will be deducted.** (4 marks)

$$f(\mathrm{w,x,y,z}) = \sum \mathrm{m}(0,2,4,6,8,10,12-15)$$

(Note: 12 – 15 means “12 to 15”)

b. Draw f as a 2-level NAND circuit. You may assume that complemented inputs (e.g. w'), and literals 0 and 1 are available to you. (2 marks)

c. We are given the following MUX-DEMUX circuit. Write the simplified expression for the function $g(A,B,C,D)$ using only the laws/theorems of Boolean Algebra. K-Maps, truth tables, etc. are not allowed. You must write down the laws/theorems you used or marks will be deducted. (5 marks)

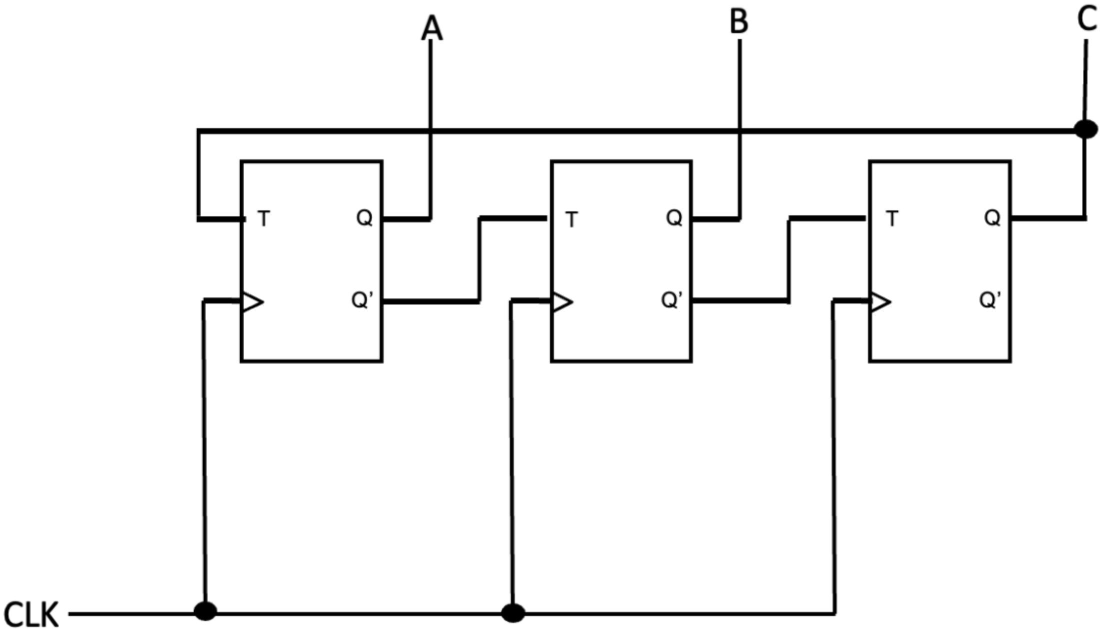


d. Given two three-bit inputs $X = x_2x_1x_0$ and $Y = y_2y_1y_0$, with $X, Y \geq 0$ implement the function $h(X,Y) = 2X - Y$ using a single four-bit parallel adder and no other components.

You may assume that complemented inputs (e.g. x_2'), and literals 1 and 0 are available, and you do not have to handle any overflow situations. (4 marks)

Question 2. Sequential Circuit Design (12 marks)

a. We are given the following circuit made up of three T flip-flops:



i. Draw the state diagram for this circuit. Assume that this circuit is initially in state 0b000. (2 marks)

ii. List all unused states, if any. If all states are used, put "Nil" as your answer. (2 marks)

b. We are given the following state table for a state machine. We want to implement this using T flip-flops. We begin initially at state 0b000

A	**B**	**C**	**A+**	**B+**	**C+**
0	0	0	0	1	0
0	1	0	1	0	0
1	0	0	1	1	0
1	1	0	0	0	1
0	0	1	0	1	1
0	1	1	1	0	1
1	0	1	1	1	1
1	1	1	0	0	0

i. Write down the simplified expressions for the flip-flop inputs to implement this state machine, in sum-of-products. (6 marks)

ii. Draw your circuit using T flip-flops and AND, OR and NOT gates only, as necessary. You do not need to draw the clock connection. (2 marks)

Question 3. Pipelining (9 marks)

The following MIPS code was given in Assignment #1. It reads an integer array ***A***, whose base address is stored in **$s0**, and computes some answer in **$s1** which is associated with the integer variable **ans**.

```
      addi $s1, $zero, 0     # Inst1
      addi $t0, $s0, 16      # Inst2
                             #
Here: addi $t0, $t0, -4      # Inst3
      lw   $t1, 0($t0)       # Inst4
      slti $t2, $t1, 8       # Inst5
      bne  $t2, $zero, Skip  # Inst6
      addi $s1, $s1, 1       # Inst7
      sll  $s1, $s1, 1       # Inst8
Skip: beq  $t0, $s0, End     # Inst9
      j    Here              # Inst10
End:
```

For the parts below, you are to consider only the first iteration of the above MIPS code in a 5-stage MIPS pipeline, that is, instructions 1 through instruction 10. You need to count until the last stage (WB stage) of instruction 10. Assume that array *A* contains the following 16 integers:

8, 2, 10, 23, 5, 6, 7, 8, 9, 10, 1, 2, 3, 4, 5, 6. (i.e., *A*[0] = 8, *A*[15] = 6.)

(a) In an ideal pipeline with no delays, how many cycles are needed to complete the first iteration of the code? [1 mark]

For parts (b) – (d) below, list out the cycles of delay at each instruction, and the total cycles of delay for the first iteration, that is, instruction 1 through instruction 10.

For example, for part (b), if you think that two cycles of delays are incurred at Instruction 2, one cycle of delay at Instruction 5, and three cycles of delays at Instructions 7, then you write:

Inst2: +2
Inst5: +1
Inst7: +3
Total: +6

(b) Assume without forwarding and branch is resolved at stage 4 (MEM). No branch prediction is made and no delayed branching is used. [2 marks]

(c) Assume with forwarding and branch is resolved at stage 2 (ID). No branch prediction is made and no delayed branching is used. [2 marks]

(d) Assume with forwarding and branch is resolved at stage 2 (ID). Branch predicted not taken and no delayed branching is used. [2 marks]

(e) Assume the condition in part (b) (no forwarding and branch resolved at stage 4). Refer to the first instruction in the code that incurs some delay. To reduce the delay, two of the instructions can be swapped. Indicate which two instructions can be swapped, and the number of cycles of delay that are reduced as a result of this swap. [2 marks]

Question 4. Cache (14 marks)

Refer to the following MIPS code. Arrays *A* and *B* are integer arrays and their base addresses are stored in **$s0** and **$s1** respectively. The size of array *A* (the number of elements in array *A*) is stored in **$s2**. Array *B* has twice the number of elements as array *A*, and it follows immediately after array *A* in the memory. That is, if the last element of array *A* is at address *x*, then the first element of array *B* is at address (*x*+4).

```
      addi $t0, $s0, 0       # Inst1
      addi $t1, $s1, 0       # Inst2
      addi $t8, $zero, 0     # Inst3
Loop: slt  $t9, $t8, $s2     # Inst4
      beq  $t9, $zero, Exit  # Inst5
      lw   $t2, 0($t0)       # Inst6
      lw   $t3, 0($t1)       # Inst7
      add  $t3, $t3, $t2     # Inst8
      sw   $t3, 0($t1)       # Inst9
      addi $t0, $t0, 4       # Inst10
      addi $t1, $t1, 8       # Inst11
      addi $t8, $t8, 1       # Inst12
      j    Loop              # Inst13
Exit:
```

For parts (a) to (g): You are given a **direct-mapped data cache** with 1024 words in total and each block contains 4 words. You may assume that array *A* starts at address **0x3000FE10**.

(a) How many bits are there in the index field? Write your answer as a single number (such as 32, instead of an expression such as 2^5 or $30 - 4(3) + 14$). [1 mark]

(b) How many bits are there in the tag field? Write your answer as a single number. [1 mark]

(c) If *A*[7] is loaded into the cache, which cache block will it be in? [1 mark]

(d) Assuming that there are 256 elements in array *A*. What is the hit rate of the data cache for array *A*? Write your answer as a simple fraction if it is not zero. [1 mark]

(e) Assuming that there are 256 elements in array *A*. What is the hit rate of the data cache for array *B*, considering only the **lw** instruction for array *B* and not the **sw** instruction? Write your answer as a simple fraction if it is not zero. [2 marks]

(f) Assuming that there are 1024 elements in array *A*. What is the hit rate of the data cache for array *A*? Write your answer as a simple fraction if it is not zero. [2 marks]

(g) Assuming that there are 1024 elements in array *A*. What is the hit rate of the data cache for array *B*, considering only the **lw** instruction for array *B* and not the **sw** instruction? Write your answer as a simple fraction if it is not zero. [2 marks]

For parts (h) and (i): You are given a **2-way set-associative instruction cache** with LRU replacement policy. Instruction 1 (`addi $t0, $s0, 0`) is stored at address **0xC000FFFC** and there are 1024 elements in array *A*.

(h) The instruction cache contains 16 words in total and each block contains 4 words. How many misses in total are there in the instruction cache during the execution of the code? [1 mark]

(i) The instruction cache contains 8 words in total and each block contains 2 words. How many misses in total are there in the instruction cache during the execution of the code? [3 marks]

=== END OF PAPER ===