CS2100 (AY2020/21 Semester 2)
Assignment #3

PLEASE READ THE FOLLOWING INSTRUCTIONS VERY CAREFULLY.

1. The **deadline** for this assignment is **9 April 2021, Friday, 1pm**. The submission folders will close shortly after 1pm. Late submission will not be accepted -- you will receive 0 mark.
2. Complete this assignment **on your own** without collaborating or discussing with anyone. If 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.
3. Please submit only **ONE pdf file** called **AxxxxxxxY.pdf**, where **AxxxxxxxY** is your student number. Marks will be deducted if you deposit multiple files in the submission folder or your submitted file is not a pdf file.
4. Please keep your submission **short**. You only need to submit your answers; you do not need to include the questions.
5. 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.
6. Upload your file to LumiNUS > Files > Assignment 3 > (your tutorial group) > (your personal folder).
7. There are THREE (3) questions (excluding question 0) on FIVE (5) printed pages in this paper.
8. The questions are worth **40 marks** in total.
9. You do not need to show your working in this assignment.

=== END OF INSTRUCTIONS ===

Question 0. Submission instructions (3 marks)

(a) You have named your file with your student number, i.e., **AxxxxxxxY.pdf**. (1 mark)

(b) You have submitted your assignment as a **single PDF file** and no multiple copies. (1 mark)

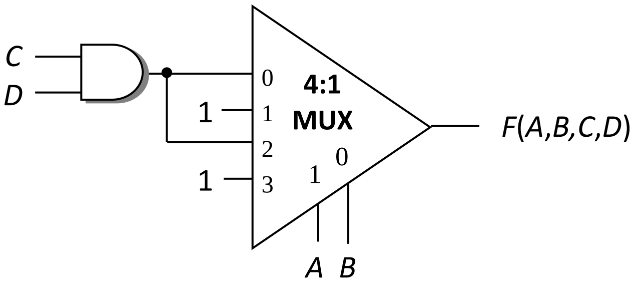
(c) Your assignment submission has your **tutorial group number**, **student number** and **name** at the top of the first page of your file. (All three items must be present.) (1 mark)

Question 1. MSI Devices (16 marks)

Note:

- Boolean constants (0 and 1) are always available;
- Complemented literals are <u>not</u> available.

(a) Given the following circuit showing a 4:1 multiplexer and an AND gate, what is the simplified SOP expression of the function *F*(*A*,*B*,*C*,*D*)? [3 marks]



(b) Implement the following Boolean function using a single 4:1 multiplexer without any logic gate. Once you have chosen the 2 variables for the selector lines, those 2 variables are <u>not</u> to appear in the 4 multiplexer inputs.

No mark will be awarded if any logic gate besides the multiplexer is used.

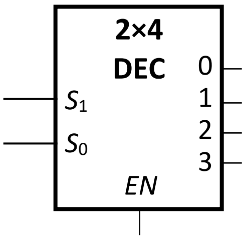
$$H(A,B,C,D) = \Sigma m(2, 7, 10, 12, 14, 15)$$

The block diagram of a 4:1 multiplexer is given in part (a) above. [4 marks]

(c) Implement the following Boolean function using a single 2×4 decoder with one-enable and active high outputs, and at most one logic gate.

$$G(A,B,C,D) = \Sigma m(2, 11)$$

The block diagram of a 2×4 decoder with one-enable and active high outputs is shown below. [4 marks]



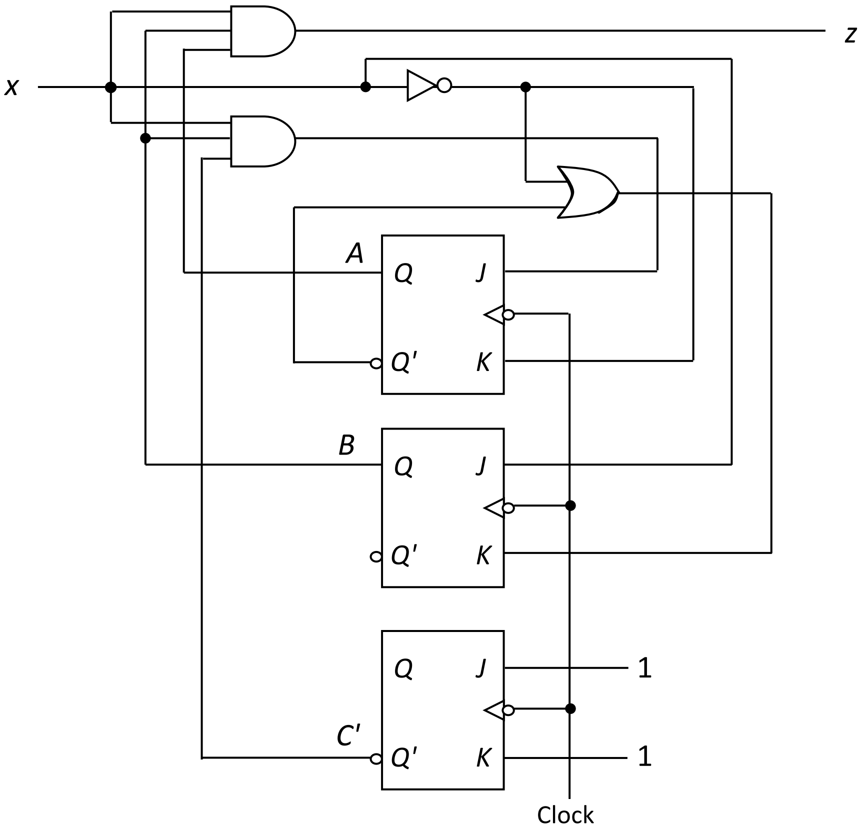
(d) Below is a partial function table of an 8-to-3 priority encoder. It consists of 8 inputs (A_7 to A_0) and 3 outputs (F_2 to F_0). A_i has higher priority than A_j for $i > j$. The only invalid input combination occurs when all inputs are zeroes. 'X' represents don't-care.

Write out the simplified SOP expressions for F_2, F_1 and F_0. [5 marks]

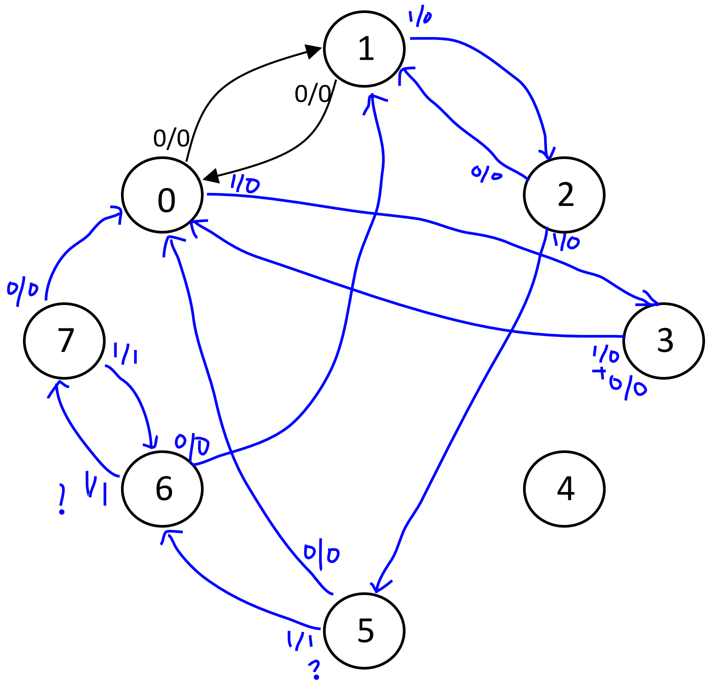
A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	F_2	F_1	F_0
0	0	0	0	0	0	0	0	X	X	X
1	X	X	X	X	X	X	X	1	1	1
0	1	X	X	X	X	X	X	1	1	0
0	0	1	X	X	X	X	X	1	0	1
:								:		
0	0	0	0	0	0	1	X	0	0	1
0	0	0	0	0	0	0	1	0	0	0

Question 2. Sequential Circuit (14 marks)

Analyze the following sequential circuit with JK flip-flops, an input *x* and an output *z*. The states are represented by *ABC*.



Complete the state diagram for the circuit below. States are shown in decimal and the labels are in *x*/*z* format, where *x* is the input and *z* the output. Two arrows have been drawn for you. For ease of tracing, place a label near the start of its associated arrow. Use the same positions of the states in the diagram below to ease grading.



Question 3. Pipelining (7 marks)

(Past year's exam question)

The following MIPS code reads an integer array ***A***, whose base address is stored in **$s0**, and computes some answer in **$s2**. The register **$s1** is associated with the integer variable **cutoff**.

```
2         addi $s2, $zero, 0      # Inst1: count = 0
          addi $t0, $s0, 40       # Inst2: $t0 pts to A[10]
                                  #        i = 9,8,...,0
   Here:  addi $t0, $t0, -4       # Inst3: $t0 pts to A[i]
          lw   $t1, 0($t0)        # Inst4: $t1 = A[i]
7         slt  $t2, $t1, $s1      # Inst5: if (A[i]<cutoff)
          beq  $t2, $zero, Skip   # Inst6: {
          addi $s2, $s2, 1        # Inst7:   count++;  }
   Skip:  beq  $t0, $s0, Fin      # Inst8: $t0 pts to A[0]?
          j    Here               # Inst9
   Fin:
```

For parts (a) to (c) below, you are to calculate the total number of cycles required to complete the first iteration of the above code in a 5-stage MIPS pipeline, that is, instruction 1 through instruction 9. You need to count until the last stage (WB stage) of instruction 9. You may assume that all elements in array *A* have values that are smaller than **cutoff**.

(a) In an ideal pipeline with no delays, how many cycles are needed to complete the first iteration of the code? I+N-1 = 9+5-1 = 13 (1 mark)

(b) Assume without forwarding and branch is resolved at stage 4 (MEM) stage. No branch prediction is made and no delayed branching is used. What is the total number of cycles required to complete the first iteration of the code? (2 marks)

28/29

(c) Assume with forwarding and branch is resolved at stage 2 (ID) stage. No branch prediction is made and no delayed branching is used. What is the total number of cycles required to complete the first iteration of the code? (2 marks)

15

(d) For a general array (where the array elements have random values instead of all being smaller than **cutoff**), how would the choice of branch prediction (that is, choosing between "branch taken" and "branch not taken") affect the performance of this code with respect to the beq instruction in instruction 6? (2 marks)

If there are more elements which are bigger than the cutoff, then taking branch taken will be more efficient

But if there are more elements which are smaller than the cutoff, then taking branch not taken as the prediction will be more efficient

=== END OF PAPER ===