University of Western Ontario, Computer Science Department CS3350B, Computer Organization

Assignment 2 Due: 11:55 P.M, 25th Feb, 2024.

General Instructions: This assignment consists of 4 pages, 5 exercises, and is marked out of 80. For any question involving calculations you must provide your workings. You may collaborate with other students in the class in the sense of general strategies to solve the problems. But each assignment and the answers within are to be solely individual work and completed independently. Any plagiarism found will be taken seriously and may result in a mark of 0 on this assignment, removal from the course, or more serious consequences.

Submission Instructions: The answers to this assignment are to be submitted to Gradescope. Ideally, the answers are to be typed. At the very least, clearly *scanned* copies/photographs of hand-written work(scanned copies should be clear and there should not be unnecessary doodling on the page). If the person correcting your assignment is unable to easily read or interpret your answer then it may be marked as incorrect without the possibility of remarking.

Useful Facts:

Logism is a good tool for drawing circuits.

draw.io is a good tool for drawing any kind of diagram, including circuits.

You may also consider using OneNote, Photoshop, etc. to draw circuits

$1GHz$=$1\times10^9\,Hz$
$1\ byte$ = $8\ bits$
$1\ Kbyte\ (KB)$ = $1024\ bytes$

Recallthat$XY \equiv XY \equiv X\cdot Y$

Exercise 1. [6 marks] Simplify the following expression. Also Draw the logic diagram for the expression. Do mention any Boolean law in the steps whenever needed.

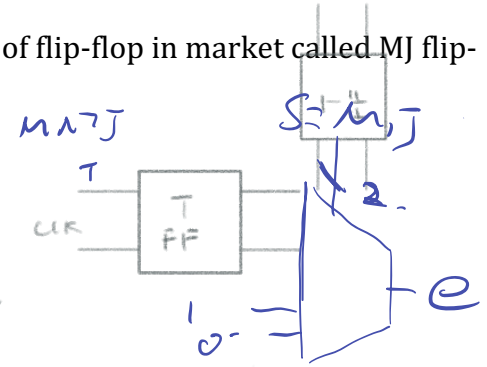
$$\overline{ABC} + \overline{A\bar{B}C} + \bar{A}B\bar{C} + A\bar{B}C + A\overline{BC}$$

Exercise 2. [10 Marks] Why NAND and NOR gates are called functionally complete gates? Draw every logic [AND, OR, NOT, NAND, NOR] gate using:

1. NAND gates only
2. NOR gates only

Exercise 3. [10 marks] Assume we have a new type of flip-flop in market called MJ flip-flop. Following is the truth table of MJ flipflop.

M	J	Qn+1
0	0	1
0	1	0
1	0	$\overline{Qn}$
1	1	Qn



Design a circuit diagram for MJ flip-flop using T Flip-Flop.

Exercise 4. [20 marks] Counter is the circuit (made up of flipflops and logic gates) which is used to count the number of clocks. There are two types of counters: synchronous counters and asynchronous counters.
Synchronous counters are the ones in which all the flipflops are connected to one clock.
Asynchronous counters are the ones in which output of previous flipflop becomes the clock for next flipflop.

Consider we've a synchronous counter which consists of two T flipflops. We are observing an output of these flipflops to get the output of the counter (these represents two bits).

The counter gives us following outputs in the exact same order:

00 → 01 → 11 → 00 → 01 → 11 → 00 … and so on.

Using this sequence of outputs, design a circuit diagram for the counter using T flipflops which gives the same output. (You can use any logic gates along with T flipflops).

Exercise 5. [10 + 8 + 8 + 8 = 34 marks] In this exercise we will explore designing an integrated circuit for use within a coffee grinder. Rather than tackling the entire system, we'll focus on a specific aspect of it. Imagine the coffee maker already has a circuit responsible for managing the heating element and the water pump. Your objective is to design a simple circuit to control the coffee grinder. The grinder is driven by a motor with two input bits:

G: If G≡1, the grinder operates. If G≡0, the grinder is inactive.

F: If F≡1 and G≡1, the grinder operates in a fine setting. If F≡0 and G≡1, the grinder operates in a coarse setting.

The manufacturer of this coffee maker requests a synchronous circuit that will control the grinder to switch between fine and coarse grinding each time the coffee maker is turned on. The circuit to be designed will receive a single input bit

I from the circuit controlling the heating element.

When I≡1, the grinder should operate.

When I≡0, the grinder should remain inactive.

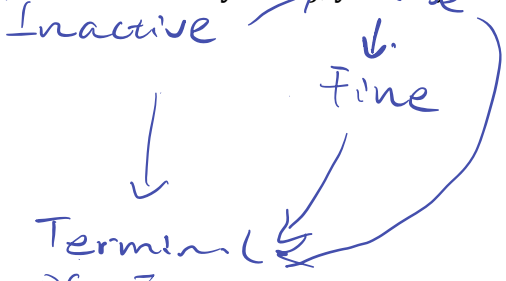
To guide your design process, complete parts (a) to (d) below.

(a) Draw a Finite State Machine that describes this state circuit. Since this circuit has 4 possible combinations of outputs, it should have 4 states. Be sure to include the inputs and outputs in the FSM. Order the outputs as GF. Therefore, a transition in the FSM will be labeled as *I*/*GF*. The change in grinding setting should be triggered by the coffee maker turning on.

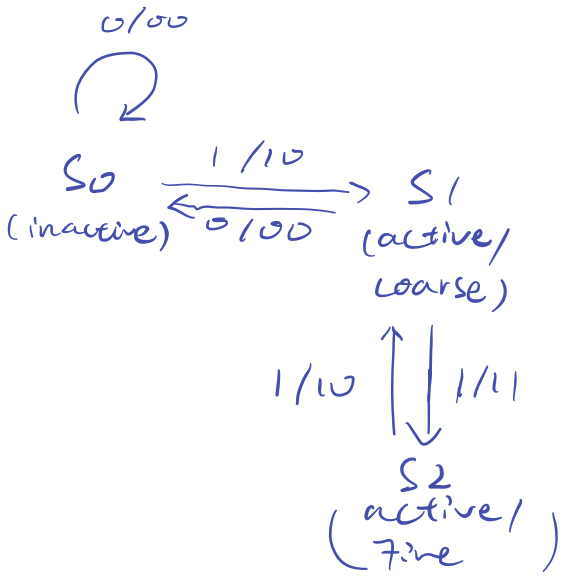
(b) Create a truth table to describe the inputs, outputs, and transitions of your FSM. That is, create a truth table describing the combinational logic of your eventual state circuit. Since there are 4 states, you must have 2 bits of "input" to encode the current state, and 2 bits of "output" to encode the next state. (Hint: your table should have 8 rows and 7 columns.)

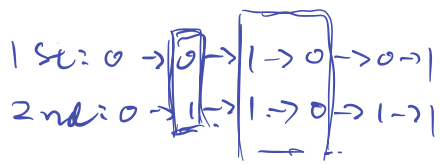
(c) Give the DNF for each output of the combinational part of your state circuit (there are 4 of them; recall that the "next state" is part of the combinational logic output). Then, simplify each DNF into a reduced sum of products. For this particular question, you do not need to specify the Boolean law being applied at each step of the simplification.

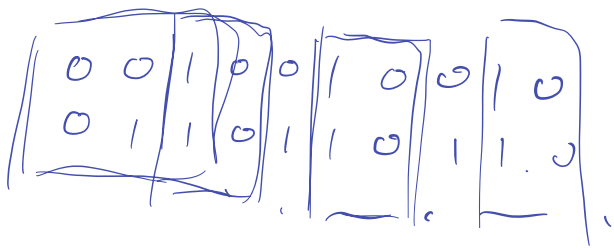
(d) Draw a circuit which implements the combinational logic of your turntable controller circuit. That is, draw a circuit implementing your 4 simplified formulas from part **(c)**. Use gates with *arity* at most 2. For clarity (and to avoid crossing too many lines), you may draw 4 independent circuits, one for each output bit.

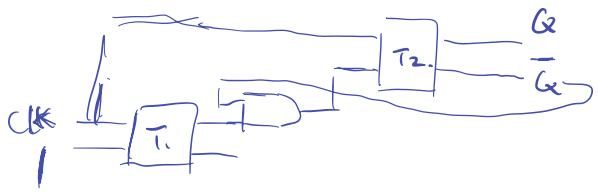


PS	In	NS	out.
S0	0	S0	00
S0	1	S1	10
S1	0	S0	00
S1	1	S2	11
S2	0	S0	00
S2	1	S1	10









Ex-5.

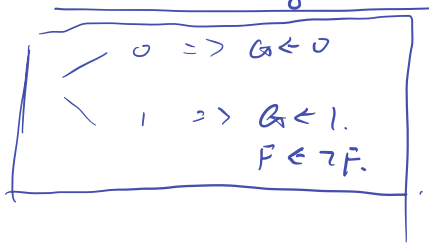
G=1: grinder operating.

G=0: grinder closed.

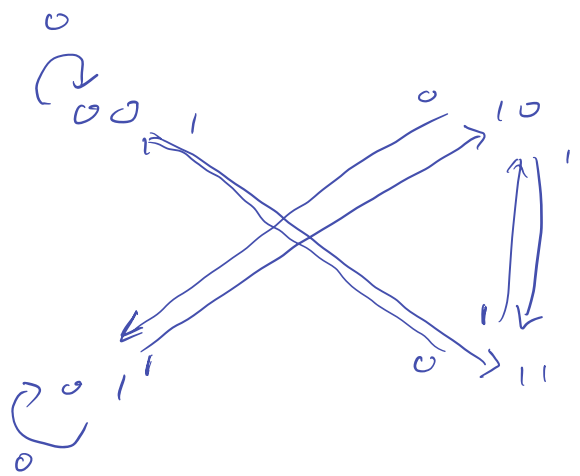
grinder operation $\begin{cases} 1 \\ 0 \end{cases}$

grinder setting (G=1) $\begin{cases} 1 & \text{fine} \\ 0 & \text{coarse.} \end{cases}$

I => control grinder keep operating or not.



Gf	00	01
coarse		fine.
	10	11



PS

DNF for G:

$= !PS0 \cdot !PS1 \cdot In + !PS0 \cdot PS1 \cdot In$

PS	In	NS	G	F
00	0	00	0	0
00	1	11	1	0
01	0	01	0	0
01	1	10	1	1
10	0	01	0	0
10	1	11	1	0
11	0	00	0	0
11	1	10	1	1

$$\neg PS0 \cdot \neg PS1 \cdot I + (PS0 \cdot \neg PS1 \cdot \neg I + PS1 \cdot \neg PS0 \cdot \neg I) + PS1 \cdot \neg PS0 \cdot I$$

$$\equiv \neg PS0 \cdot I + (\neg I) \cdot (PS0 \cdot \neg PS1 + \neg PS0 \cdot PS1)$$

$$\equiv \neg PS0 \cdot I + \neg PS0 \cdot \neg I \cdot PS1 + PS0 \cdot \neg PS1 \cdot \neg I$$

$$\equiv \neg PS0 \cdot (I + \neg I \cdot PS1) + PS0 \cdot \neg PS1 \cdot \neg I$$

$$\equiv \neg PS0 \cdot (I + PS1)$$

$$\equiv I \cdot \neg PS0 + PS1 \cdot \neg PS0 + \neg PS1 \cdot PS0 \cdot \neg I$$