UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE AND ENGINEERING

ECE253F – Digital and Computer Systems
Midterm Examination

October 12, 2023 9:10am - 10:40am
Duration: 90 minutes

Examiners: Profs. N. Enright Jerger and M. Jeffrey

Please enter your name and student number in the spaces provided above as it appears on Quercus. It is important that your name exactly match the Quercus gradebook.

Exam Type D: Examiner specified aids: One single sheet of letter size paper (8.5 x 11 inch), both sides may be used.

Calculator Type 4: No calculators or other electronic devices are allowed.

All questions are to be answered on the examination paper. Your answer **MUST** be fully contained on the same page as the question. **Any material written on the back of each page will be ignored.**

Please state any assumptions you make when answering a question.

The number of marks for each question are indicated. The exam has **15 pages**, including this one.

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Total
5	8	10	4	3	8	7	8	53

Question 1 [5 Marks]

[3 marks] (a) Assuming that all numbers given below are unsigned integers, fill in the following table with the appropriate number conversions:

9-bit binary	**decimal**	**hexadecimal**
100110001	305	0x131
110011100	412	0x19C
010101101	173	AD

[2 marks] (b) Convert decimal 51200 to octal (base 8).

$(144000)_8$

$$\begin{aligned} 51200 &= 512 \times 100 \\ &= 2^9 \times (1 \times 8^2 + 4 \times 8^1 + 4 \times 8^0) \\ &= 8^3 \times (1 \times 8^2 + 4 \times 8^1 + 4 \times 8^0) \\ &= 1 \times 8^5 + 4 \times 8^4 + 4 \times 8^3 + 0 \times 8^2 + 0 \times 8^1 + 0 \times 8^0 \end{aligned}$$

Question 2 [8 Marks]

Use Boolean algebra to minimize the following function into sum-of-products (SOP) form:

$$f = \overline{wx\bar{y}z + (w\bar{x} + y)\bar{z}}$$

For full marks, show your work and state the theorems used. If you do not know the theorem name, write the simplified form (e.g., $x + x = x$).

Proof 1

$$
\begin{aligned}
f &= \overline{wx\bar{y}z + w\bar{x}\bar{z} + y\bar{z}} && \text{Distributive Theorem} && (1)\\
&= \overline{(wx\bar{y}z)} \cdot \overline{(w\bar{x}\bar{z})} \cdot \overline{(y\bar{z})} && \text{De Morgan Theorem} && (2)\\
&= (\bar{w} + \bar{x} + y + \bar{z}) \cdot (\bar{w} + x + z) \cdot (\bar{y} + z) && \text{De Morgan Theorem} && (3)\\
&= \bar{w} \cdot (\bar{w} + x + z) \cdot (\bar{y} + z) && &&\\
&\quad + \bar{x} \cdot (x + \bar{w} + z) \cdot (\bar{y} + z) && &&\\
&\quad + y \cdot (\bar{y} + z) \cdot (\bar{w} + x + z) && \text{Distributive and} &&\\
&\quad + \bar{z} \cdot (z + \bar{y}) \cdot (\bar{w} + x + z) && \text{Commutative Theorems} && (4)\\
&= \bar{w} \cdot (\bar{y} + z) && &&\\
&\quad + \bar{x} \cdot (\bar{w} + z) \cdot (\bar{y} + z) && &&\\
&\quad + y \cdot z \cdot (\bar{w} + x + z) && \text{Covering and} &&\\
&\quad + \bar{z} \cdot \bar{y} \cdot (\bar{w} + x) && \text{Absorption Theorems} && (5)\\
&= \bar{w}\bar{y} + \bar{w}z && &&\\
&\quad + \bar{w}\bar{x}\bar{y} + \bar{w}\bar{x}z + \bar{x}\bar{y}z + \bar{x}z && &&\\
&\quad + yz && \text{Commutative and} &&\\
&\quad + \bar{w}\bar{y}\bar{z} + x\bar{y}\bar{z} && \text{Covering Theorems} && (6)\\
&= \bar{w}\bar{y} + \bar{w}z + \bar{x}\bar{y}z + \bar{x}z + yz && \text{Covering Theorem} && (7)\\
&= x\bar{y}\bar{z} + \bar{w}\bar{y} + yz + \bar{x}z && \text{Consensus Theorem} && (8)
\end{aligned}
$$

Question 2 continued . . .

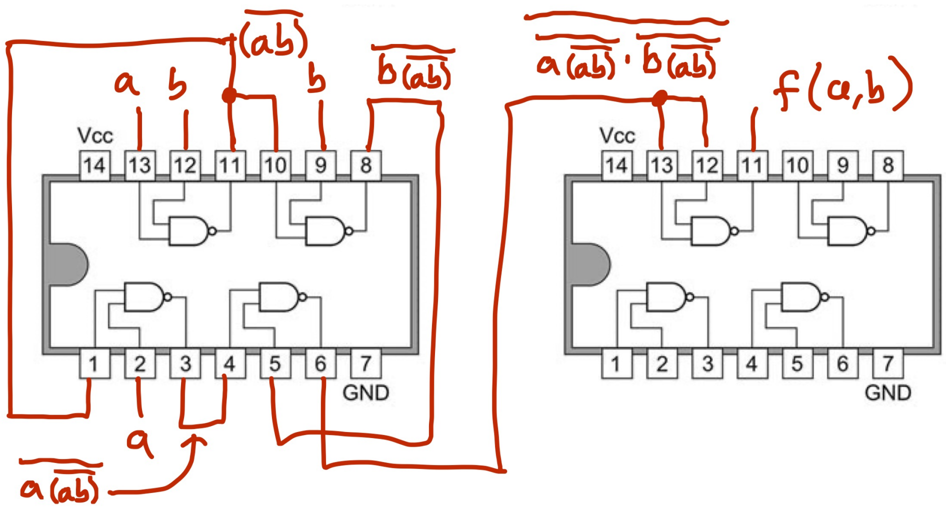
Proof 2

$$
\begin{aligned}
f &= \overline{wx\bar{y}z + w\bar{x}\bar{z} + y\bar{z}} && \text{Distributive Theorem} \\
&= \overline{(wx\bar{y}z)} \cdot \overline{(w\bar{x}\bar{z})} \cdot \overline{(y\bar{z})} && \text{De Morgan Theorem} \\
&= (\bar{w} + \bar{x} + y + \bar{z}) \cdot (\bar{w} + x + z) \cdot (\bar{y} + z) && \text{De Morgan Theorem} \\
&= (\bar{w} + x + z) \cdot (\bar{w} + \bar{x} + \bar{z} + y) \cdot \bar{y} && \text{Distributive and} \\
&\quad + (\bar{w} + \bar{x} + y + \bar{z}) \cdot (\bar{w} + x + z) \cdot z && \text{Commutative Theorem} \\
&= (\bar{w} + x + z) \cdot (\bar{w} + \bar{x} + \bar{z}) \cdot \bar{y} && \text{Absorption and} \\
&\quad + (\bar{w} + \bar{x} + y + \bar{z}) \cdot z && \text{Covering Theorem} \\
&= (\bar{w} + (x + z) \cdot (\bar{x} + \bar{z})) \cdot \bar{y} && \text{Distributive and} \\
&\quad + (\bar{w} + \bar{x} + y) \cdot z && \text{Absoprtion Theorem} \\
&= \bar{w}\bar{y} + \bar{x}z\bar{y} + x\bar{y}\bar{z} && x \cdot \bar{x} = 0 \text{ and} \\
&\quad + \bar{w}z + \bar{x}z + yz && \text{Distributive Theorem} \\
&= \bar{w}\bar{y} + x\bar{y}\bar{z} + \bar{w}z + \bar{x}z + yz && \text{Covering Theorem} \\
&= x\bar{y}\bar{z} + \bar{w}\bar{y} + yz + \bar{x}z && \text{Consensus Theorem}
\end{aligned}
$$

Question 3 [10 Marks]

A Machine Intelligence upper year is excited about Binary Neural Networks, which replace expensive multiplications with cheap 2-input XNOR operations. XNOR inverts the output of XOR.

[6 marks] (a) Building on your Lab 1 experience with 7400-series chips, help them out: implement the logic function $f(a, b) = a \text{ XNOR } b$ by drawing a schematic of as many connected 7400 (i.e., NAND) chips as you need. 4 chips are given to you; you can use more or fewer as needed. Carefully label the inputs, outputs, and VCC/GND. You must invert any variable yourself.


$\overline{(ab)}$
a
b
b
$\overline{b\overline{(ab)}}$
$\overline{\overline{a\overline{(ab)}} \cdot \overline{b\overline{(ab)}}}$
f(a,b)
Vcc
14
13
12
11
10
9
8
1
2
3
4
5
6
7
GND
a
$\overline{a\overline{(ab)}}$

Five NANDs are required

$$f = a \text{ XNOR } b = \overline{a \text{ XOR } b} = \overline{a\bar{b} + \bar{a}b} = \overline{\overline{\overline{a\bar{b} + \bar{a}b}}}$$

$$= \overline{\overline{\overline{(a\bar{b})} \cdot \overline{(\bar{a}b)}}} \qquad \text{De Morgan theorem}$$

$$= \overline{\overline{\overline{(a\bar{b} + a\bar{a})} \cdot \overline{(\bar{a}b + \bar{b}b)}}} \qquad x\bar{x} = 0$$

$$= \overline{\overline{\overline{(a(\bar{b} + \bar{a}))} \cdot \overline{((\bar{a} + \bar{b})b)}}} \qquad \text{Distributive}$$

$$= \overline{\overline{\overline{(a\overline{(ab)})} \cdot \overline{(\overline{(ab)}b)}}} \qquad \text{De Morgan}$$

Alternatively,

$$f = a \text{ XNOR } b = \overline{a \text{ XOR } b} = \overline{a\bar{b} + \bar{a}b}$$

$$= \overline{(a\bar{b})} \cdot \overline{(\bar{a}b)} \qquad \text{De Morgan}$$

$$= (\bar{a} + b) \cdot (a + \bar{b}) \qquad \text{De Morgan}$$

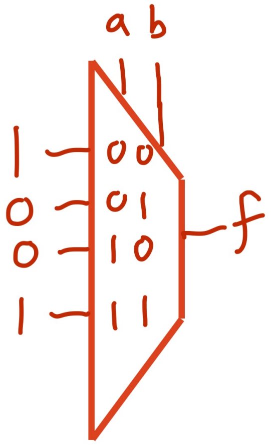
$$= ab + \bar{a}\bar{b} \qquad \text{Distributive and Complements}$$

$$= \overline{(ab) \cdot (\bar{a}\bar{b})} \qquad \text{De Morgan}$$

$$= \overline{\overline{(ab)} \cdot \overline{(\bar{a}\bar{b})}} \qquad \text{De Morgan}$$

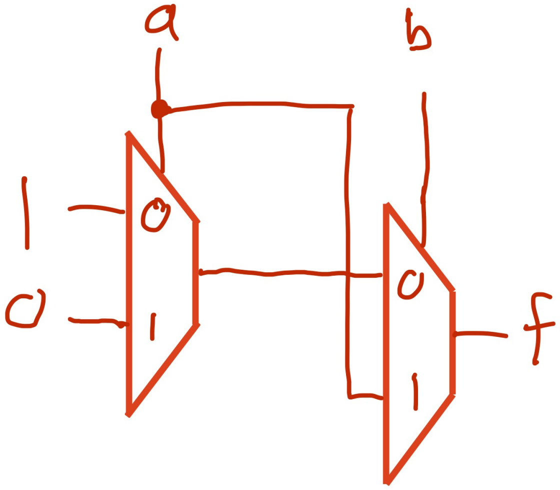
$$= \overline{\overline{(ab)} \cdot \overline{(\overline{aa}\,\overline{bb})}} \qquad x \cdot x = x$$

[2 marks] (b) You have run out of 7400 chips! Draw a schematic to implement the XNOR function, $f(a, b) = a$ XNOR b, using only 4-to-1 multiplexers.



[2 marks] (c) The 4-to-1 multiplexers all exploded! Draw a schematic to implement the XNOR function, $f(a, b) = a$ XNOR b, using only 2-to-1 multiplexers.

```
f = b ? a : (a ? 0 : 1)
```



Question 4 [4 Marks]

Consider the function $f(x_1, x_2, x_3, x_4) = \prod M(1, 4, 5, 6, 7, 8, 11, 13, 14)$ that operates on boolean variables. This is a product-of-sums notation.

[2 marks] (a) Fill in the K-map below for f.

x_3x_4 \ x_1x_2	00	01	11	10
00	**1**	**0**	**1**	**0**
01	**0**	**0**	**0**	**1**
11	**1**	**0**	**1**	**0**
10	**1**	**0**	**0**	**1**

[2 marks] (b) Report the cost of f in its canonical product-of-sums form, where cost is defined as the number of gates plus the number of gate inputs. Unlike in lecture, assume that inverters are free (zero cost).

$$\begin{aligned} \mathit{cost} &= 9 \times (1 + 4) \\ &\quad + 1 + 9 \\ &= 55 \end{aligned}$$

9 four-input OR gates

1 nine-input AND gate

Assuming only 2-input OR and 2-input AND gates are available:

$$\begin{aligned} \mathit{cost} &= 9 \times 3 \times (1 + 2) \\ &\quad + 8 \times (1 + 2) \\ &= 105 \end{aligned}$$

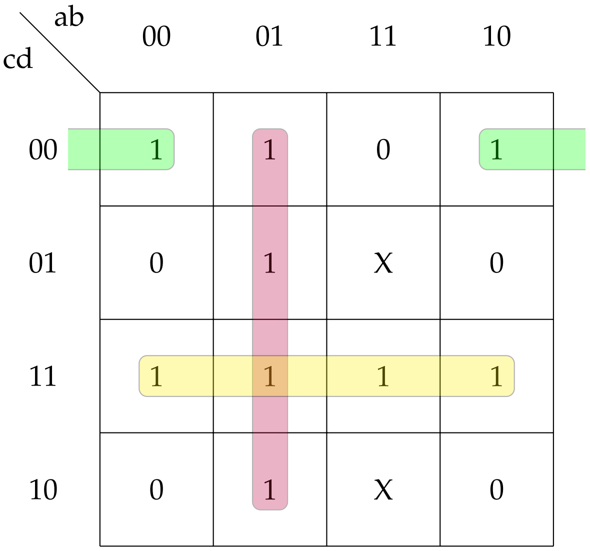
9 sets of 3 sets of two-input OR gates

8 two-input AND gates

Question 5 [3 Marks]

For the K-map given below, derive the minimum-cost cover as a sum of products expression. ‘X’ in the K-map indicates a “don’t care”.

$f(a, b, c, d) = \bar{a}b + cd + \bar{b}\bar{c}\bar{d}$



cd \ ab	00	01	11	10
00	1	1	0	1
01	0	1	X	0
11	1	1	1	1
10	0	1	X	0

Question 6 [8 Marks]

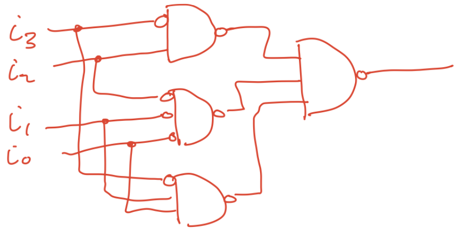
Design a 4-bit input ($i[3:0]$), 1-bit output (out) combinational logic circuit with the following specifications:

- Input will only be values between 1 and 12 inclusive
- Output is 1 when input is between 3 and 8 inclusive.

[2 marks] (a) Complete the truth table for your circuit.

i_3	i_2	i_1	i_0	out
0	0	0	0	x
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	x
1	1	1	0	x
1	1	1	1	x

[6 marks] (b) Derive a minimal implementation of this circuit using only NAND gates and inverters.



Question 7 [7 Marks]

Match each of the System Verilog code snippets shown below with the equivalent circuit on the next page (A-I). Note: input and output names in the Verilog code may be different from those shown in the circuits. Each circuit may be used more than once, and logic simplification may be required.

(a)
```
always_latch
  if (clock)
    out = a | (~b & b);
```

Answer: **A**

(b)
```
assign z = (~x) & (~y);
```

Answer: **E**

(c)
```
assign out = a & b & c;
```

Answer: **I**

(d)
```
logic [4:0] out;
assign out = a[3:0] + b[3:0];
```

Answer: **G**

(e)
```
assign f = ~s[1] & ~s[0] & a | ~s[1] & s[0] & b |
           s[1] & ~s[0] & c | s[1] & s[0] & d;
```

Answer: **H**

(f)
```
assign w = (x ^ z) + ~x & ~y;
```

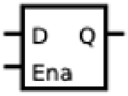
Answer: **I**

(g)
```
always_ff @(posedge clk, posedge R)
  if (R)
    Q <= 0;
  else
    Q <= D;
```

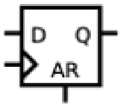
Answer: **B**

Use these circuits to answer the questions on the previous page. Note: In (A) Ena is an Enable signal; in (B) AR is an asynchronous reset; in (C) R is a synchronous reset.

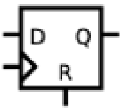
A



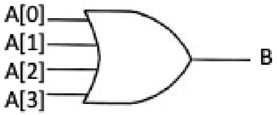
B



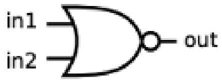
C



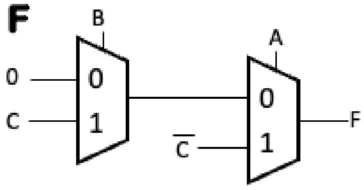
D



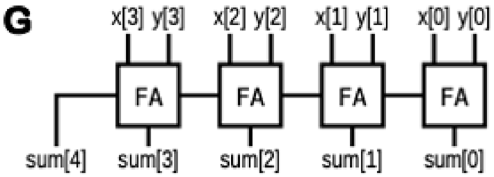
E



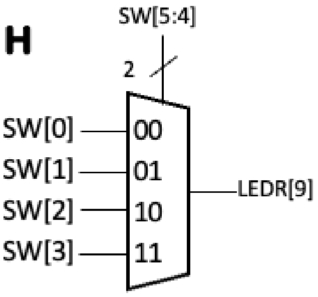
F



G



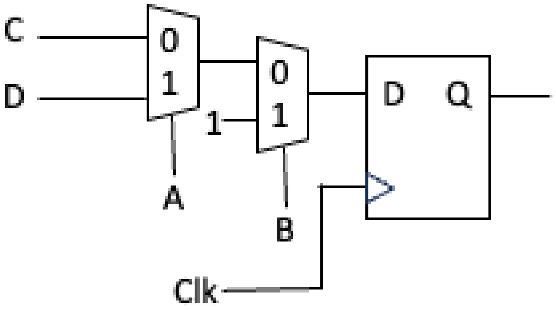
H



I **None of the above**

Question 8 [8 Marks]

Consider the circuit below:



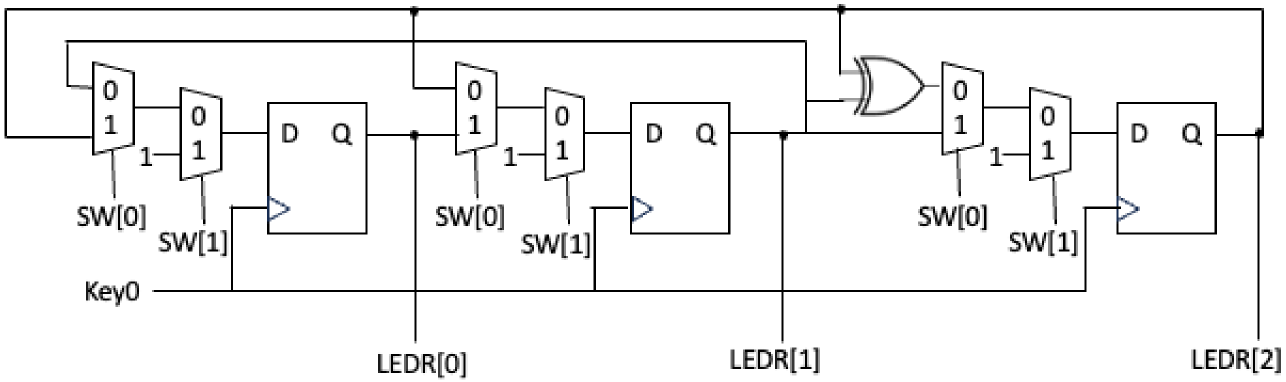
(a) Write a Verilog for this module named Mux2DFF for this circuit. The Flip Flop has an active low asynchronous reset.

```
module Mux2DFF(input logic A, B, C, D, Clk, resetn, output logic Q);

   always_ff @ (posedge Clk, negedge resetn)

     if (!resetn)
       Q <= 1’b0;
     else
         if (B)
           Q <= 1b’1;
         else
           if (A) Q <= D;
           else Q <= C;


 endmodule
```



(b) Now write a top-level System Verilog Module that implements the full circuit shown on the here and instantiates the Mux2DFF module. Connect the A, B inputs to SW 0 and 1, connect the Clk to Key0. Connect resetn to Key1. Connect the Q outputs to the red LEDs. Label the figure with any additional signals you create.

```
    module top(input logic [1:0] SW, KEY,
               output logic [2:0] LEDR);
    // Add additional signals here and label them on the figure

      logic [2:0] Q;
      logic X1; // output of XOR

      assign LEDR = Q;
      assign X1 = Q[1] ^ Q[2];

    // Instantiate Mux2DFF -- FILL IN THE CONNECTIONS
    Mux2DFF  u1     (
      .Clk(  Key[0]        ),
      .resetn(  ~Key[1]     ),
      .A(    SW[0]         ),
      .B(    SW[1]         ),
      .C(    Q[1]         ),
      .D(    Q[2]         ),
      .Q(    Q[0]         )
    ); // Typo in Figure -- accept both Key0 and Key[0] in solution

    Mux2DFF   u2    (
      .Clk(   Key[0]       ),
      .resetn(  ~Key[1]     ),
      .A(    SW[0]         ),
      .B(    SW[1]         ),
      .C(    Q[2]         ),
      .D(    Q[0]         ),
      .Q(    Q[1]         )
    );

// Module continues on the next page
```

```
Mux2DFF   u3    (
  .Clk(    Key[0]      ),
  .resetn(  ~Key[1]     ),
  .A(   SW[0]         ),
  .B(   SW[1]         ),
  .C(    X1        ),
  .D(    Q[1]        ),
  .Q(    Q[2]        )
);

endmodule
```

This page has been left blank intentionally