1	/13
2	/18
3	/19
4	/18
5	/17
6	/15

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

6.191 Computation Structures
Spring 2023

Quiz #1

Name	*Athena login name*	*Score*

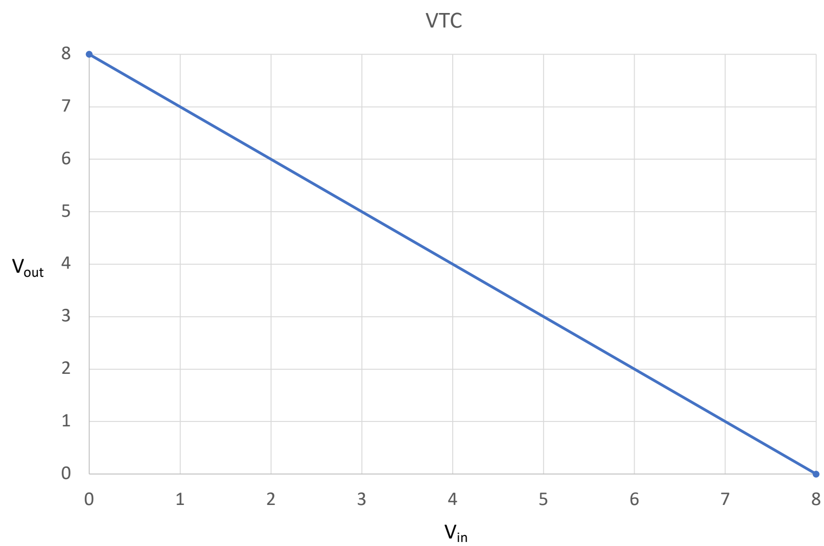
Recitation section

☐ WF 10, 34-302 (Alexandra)	☐ WF 2, 34-302 (Boom)	☐ opt-out
☐ WF 11, 34-302 (Alexandra)	☐ WF 3, 34-302 (Boom)	
☐ WF 12, 34-302(Georgia)	☐ WF 12, 35-308 (Keshav)	
☐ WF 1, 34-302 (Georgia)	☐ WF 1, 35-308 (Keshav)	

Please enter your name, Athena login name, and recitation section above. Enter your answers in the spaces provided below. Show your work for potential partial credit. You can use the extra white space and the back of each page for scratch work.

Problem 1. Digital Abstraction (13 points)

(A) (3 points) The Voltage Transfer Curve of Device X is shown below.

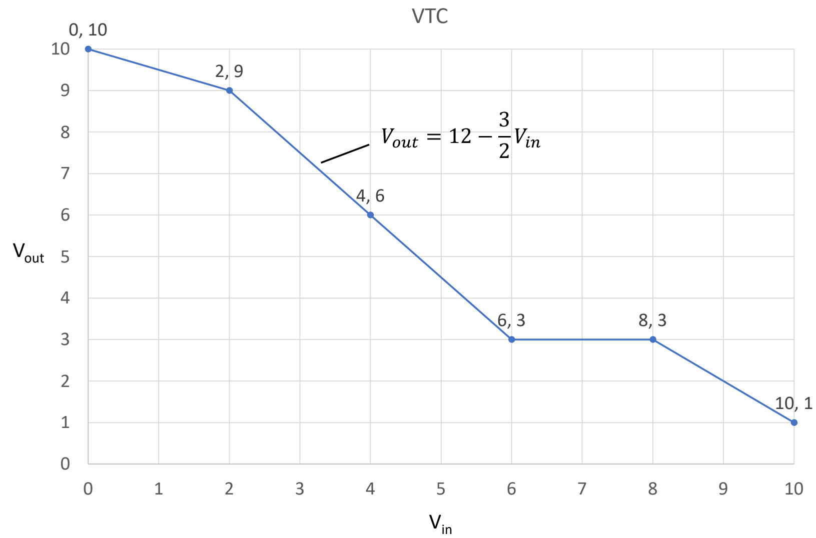


Give a valid specification for Device X or explain why no such specification exists.

V_{OL}: ______; V_{IL}: _______; V_{IH}:______; V_{OH}:_______

Or explain why a valid specification for Device X does not exist:

Device Y has the Voltage Transfer Curve shown below. Device Y is an inverter.



(B) (6 points) For each partial specification below, complete the table by filling in the missing entries that **maximize the noise immunity given the existing constraints**. If the voltages in a given row cannot be part of valid specification, fill the entries for that row with "X"s. Remember, **a valid specification must have positive noise immunity**.

	V_{OL}	V_{IL}	V_{IH}	V_{OH}	Noise Immunity
Spec 1	1			10	
Spec 2	3			9	
Spec 3	3			6.6	
Spec 4	3			6	

(C) (4 points) Explain why each of these specifications is valid or invalid.

Problem 2. Boolean Algebra (18 points)

(A) (8 points) Zoomba, a diligent student in 6.191, is trying to simplify some combinational logic. Help Zoomba by **finding a minimal sum-of-products** expression for each of the following Boolean expressions. (Note: These expressions can be reduced into a minimal SOP by repeatedly applying the Boolean algebra properties we saw in lecture.) Make sure your answer is in minimal-sum-of products form.

1. $\overline{xy(x + \bar{y}) + \bar{y}}$

2. $(a + b)(a + \bar{b}) + \bar{a}\bar{b}$

3. $\bar{x}\bar{y}\bar{z} + \bar{x}y\bar{z} + \bar{x}yz + xyz + xy\bar{z}$

4. $(ab + \bar{a}\bar{b})\mathrm{c} + b\bar{c} + \bar{a}\bar{c}$

(B) (3 points) Zoomba wants to implement Boolean function G(A, B, C) given by the truth table below. Find the normal form and a minimal sum-of-products expression for G(A, B, C).

A	B	C	G
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

1. **Normal form for G =** ____

2. **Minimal sum of products for G =** ______________________________________

(C) (4 points) Draw the circuit that implements the function G **using 3 or fewer gates**. You **may only use inverters and 2-input OR, NOR, AND,** and **NAND** gates in your circuit. To receive any credit for this question, your circuit must implement the given truth table of function G, regardless of whether your minimal sum-of-product expression is correct or not.

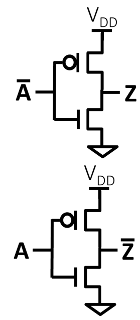
(D) (3 points) Zoomba likes function G so much that they would like to construct all possible Boolean functions using G. Determine whether G is functionally complete. Explain your answer.

Problem 3. DrCMOS Will See You Now (19 points)

Ben Bitdiddle is disappointed with the limitation that CMOS logic is inverting: many common functions need multiple CMOS gates in series, which adds propagation delay.

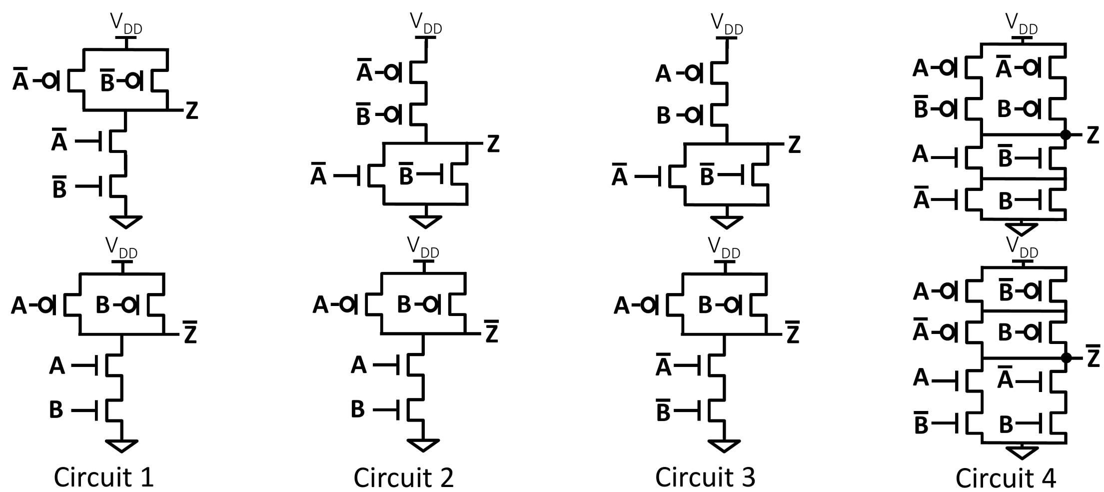
To solve this problem, Ben proposes Dual-rail CMOS (or DrCMOS) logic. In a DrCMOS gate:

- Each input and output $\boldsymbol{X}$ is encoded using two wires $(\boldsymbol{X}, \overline{\boldsymbol{X}})$ that hold the value $\boldsymbol{X}$ and its complement, i.e., these wires can be (0, 1) or (1, 0). This is called dual-rail encoding.
- Each DrCMOS gate consists of two separate CMOS gates, which compute the output $(\boldsymbol{Z}, \overline{\boldsymbol{Z}})$. One of the CMOS gates produces $\boldsymbol{Z}$, and the other produces $\overline{\boldsymbol{Z}}$. The CMOS gates can use any of the input wires as their inputs.



The figure to the right shows a DrCMOS buffer, which is built with two CMOS inverters in parallel. Whereas with the conventional (single-wire) signal encoding, a buffer needs two CMOS inverters in series, this dual-rail encoding allows using a single CMOS gate to get each output, which is faster! By using this dual-rail encoding throughout the circuit and using DrCMOS gates, Ben argues that we can implement faster circuits.

(A) (8 points) For each of the following circuits shown below, find whether the circuit is a valid DrCMOS gate. If so, specify its Boolean function **in sum-of-products form**. If not, briefly explain why.



Circuit 1: Z = F(A,B) = _____________ or NONE and explain why it's not a DrCMOS gate.

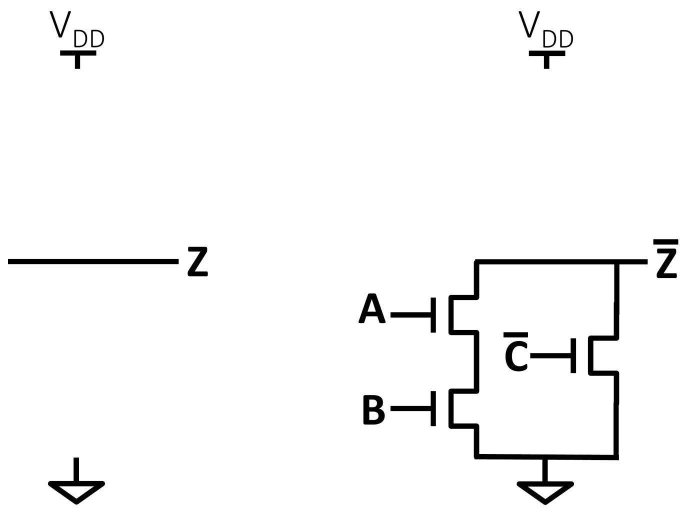
Circuit 2: Z = F(A,B) = _____________ or NONE and explain why it's not a DrCMOS gate.

Circuit 3: Z = F(A,B) = _____________ or NONE and explain why it's not a DrCMOS gate.

Circuit 4: Z = F(A,B) = _____________ or NONE and explain why it's not a DrCMOS gate.

(B) (4 points) For each of the valid DrCMOS gates in part A, suppose that we swap the $\boldsymbol{Z}$ and $\overline{\boldsymbol{Z}}$ signals (so that the CMOS gate at the top produces $\overline{\boldsymbol{Z}}$, and the one at the bottom produces $\boldsymbol{Z}$). What would be their new Boolean function? (Give each function in **sum-of-products form**.)

(C) (4 points) A DrCMOS gate can be derived completely from just the pull-up or pull-down network of one of its two CMOS gates. To see this by example, the figure below shows the pull-down network of the CMOS gate that produces $\overline{\mathbf{Z}}$. Draw the rest of the DrCMOS gate. What is its Boolean expression?



Gate's Boolean expression (in sum-of-products form): Z = F(A,B,C) = _____________

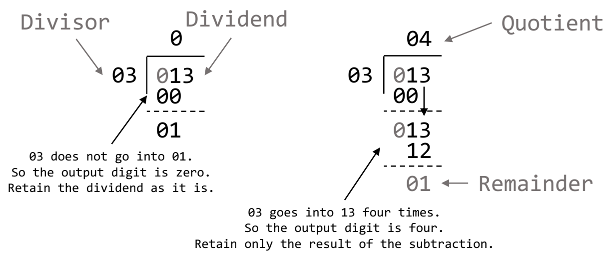
(D) (3 points) Ben claims that any Boolean function can be implemented with a **single** DrCMOS gate. Is he right? If so, explain how to construct the gate from the Boolean expression. If not, give a counterexample, i.e., a Boolean expression that requires multiple DrCMOS gates.

Note: In your explanation, you can use the fact from part C that a single pull-up or pull-down network fully specifies the DrCMOS gate, and explain only how to implement one network from the Boolean expression.

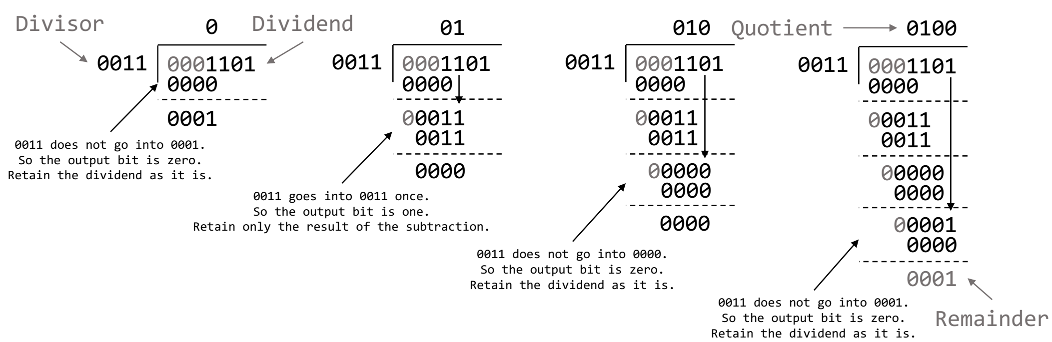
Problem 4: Back to 5^{th} Grade (18 points)

Divya's favorite pastime is to quickly and efficiently divide integers. However, she is frustrated by the fact that CPUs rarely include combinational integer dividers. She wants to fix this problem once and for all by designing a combinational circuit that performs integer division. She remembers how binary operations often use algorithms learned in grade school, such as ripple carry addition for $a + b$ and repeated shifted addition for $a \times b$, so long division cannot be that hard either, right? Let's help Divya *implement integer division* using grade school long division.

To divide two-digit decimal numbers, like $13 \div 03$, long division proceeds as follows:



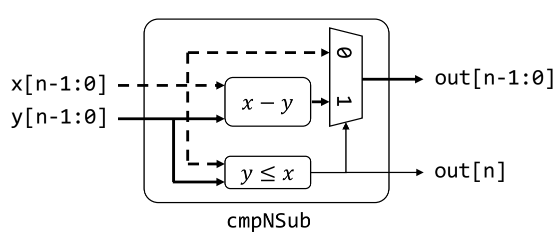
At each stage, we find out how many times the divisor goes into the current digits of the dividend, append that digit to the quotient and retain the remainder. *Note that the highest digit of the remainder is always zero, so we ignore it.* Indeed, $13 = 03 \times 04 + 01$, so the math checks out. We could use the same algorithm to perform this division in binary, i.e., $1101 \div 0011$:



(A) (4 points) Let's start off by thinking about a single comparison step. Implement a combinational circuit called `cmpNSub` that takes in two n-bit numbers, and returns the following:

$$cmpNSub(x, y) = \begin{cases} \{1'b1, x - y\}, & if\ y \leq x \\ \{1'b0, x\}, & otherwise \end{cases}$$

The output of this circuit is an (n+1)-bit quantity, where the highest (n^{th}) bit represents the quotient bit, i.e., whether or not $y \leq x$, and the lower n bits represent the current remainder, i.e., $x - y$ or x, depending on the result of the comparison. A schematic for this circuit is as follows:



Complete the following function definition in Minispec.

```
function Bit#(n+1) cmpNSub#(Integer n)(Bit#(n) x, Bit#(n) y);



endfunction
```

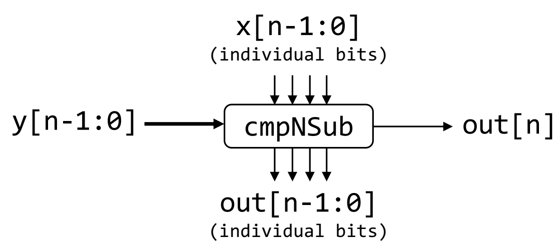
(B) (2 points) What is the propagation delay of this function? Use $\Theta(.)$ notation. Assume that the subtractor is implemented with a Ripple-Carry Adder, and the comparator is implemented as a chain comparator, *i.e.*, both their delays scale as $\Theta(n)$. Provide a brief explanation.

t_{pd} of an n-bit `cmpNSub` is Θ(_________).

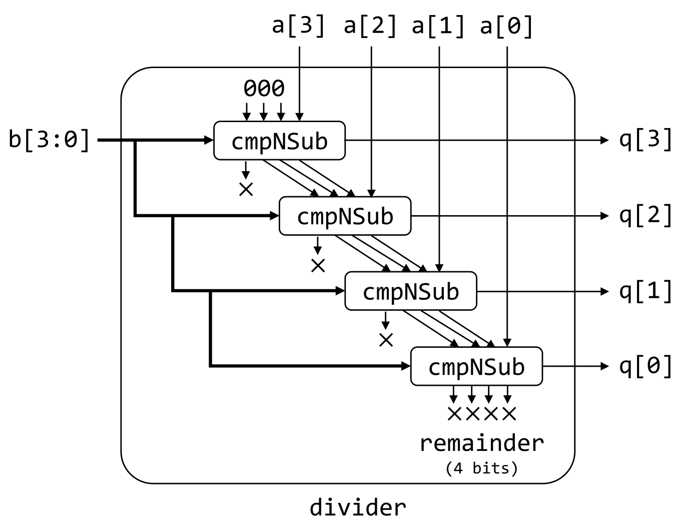
Explanation: ____

____

(C) (10 points) Consider the following symbol for the `cmpNSub` circuit. The individual bits are shown on some signals for visual clarity of connections in the overall `divider` circuit.



We can connect n instances of `cmpNSub` in order to determine the final quotient and remainder. An example of the `divider` circuit is shown for n = 4 bits (× indicates no connection):



Using the reference connections provided for n = 4 bits, implement a generalized `divider` function for two n-bit inputs and one n-bit output in Minispec that computes $a \div b$.

```
function Bit#(n) divider#(Integer n)(Bit#(n) a, Bit#(n) b);
```

```
endfunction
```

(D) (2 points) Finally, using your work from parts (A), (B) and (C), calculate the propagation delay of this n-bit `divider` circuit. Provide a brief explanation.

t_{pd} of the n-bit divider is Θ(___________).

Explanation: ____

____

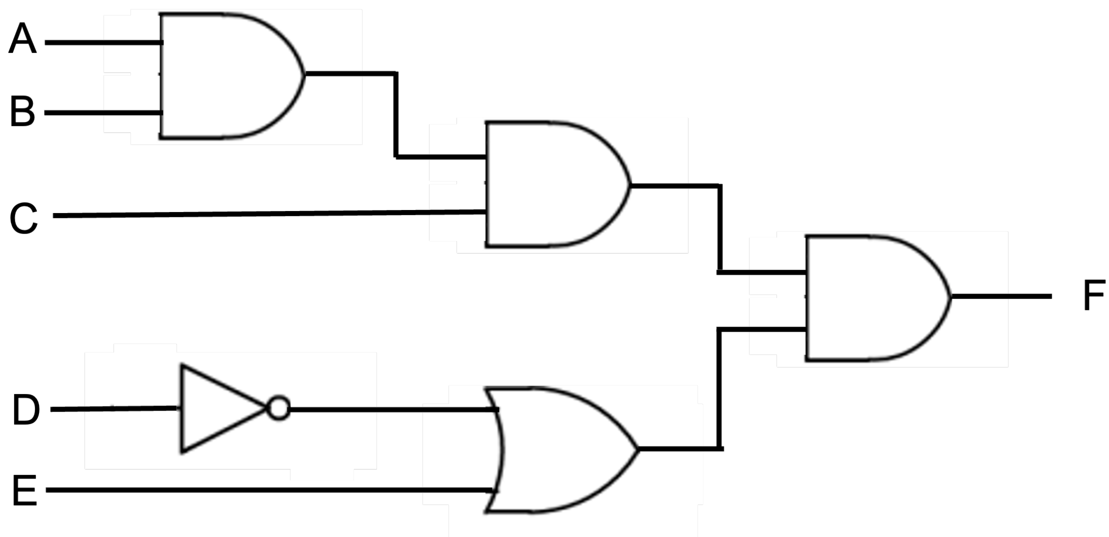
When you get your quiz back, try to synthesize the divider function for n = 32 and n = 64.

This will provide some insight into why combinational dividers are almost never included in processor ALUs, and are often entirely omitted out of lower-end processors!

Poor Divya will have to wait.

Problem 5. Combinational and Sequential Logic Timing (17 points)

(A) (4 points) Given the timing parameters of each logic gate in the table below, what are the propagation delay (t_{pd}) and contamination delay (t_{cd}) of the circuit below.

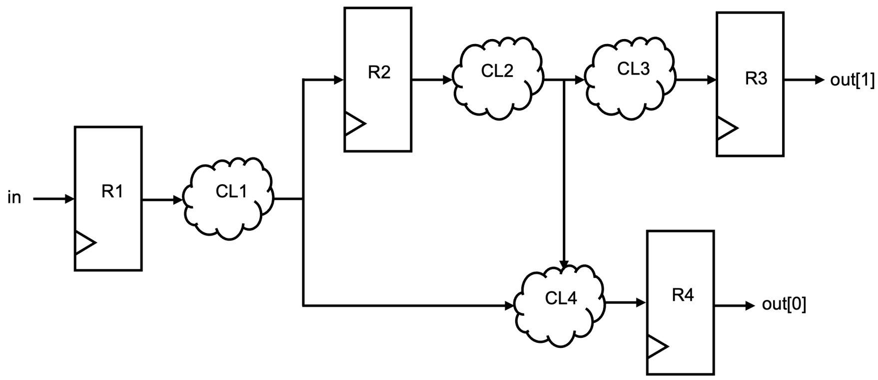


Gate	t_{pd}	t_{cd}
AND2	1.5ns	1ns
INV1	1ns	0.5ns
OR2	3ns	0.5ns

t_{pd} **(ns):** _______________

t_{cd} **(ns):** _______________

For the rest of the problem, consider the sequential circuit below, as well as the timing specifications. Registers R1, R2, R3, and R4 are driven by a common clock with period t_{clk} = 15ns. CL1, CL2, CL3, and CL4 are combinational circuits.



	t_{pd}	t_{cd}	t_{setup}	t_{hold}
Register (R1, R2, R3, R4)	3ns	1ns	5ns	2ns
CL1	3ns	1ns	--	--
CL2	2ns	1.5ns	--	--
CL3	4ns	2ns	--	--
CL4	5ns	1ns	--	--

(B) (2 points) What are the $\boldsymbol{t_{pd}}$ and $\boldsymbol{t_{cd}}$ of this sequential circuit?

$\boldsymbol{t_{pd}}$ **(ns):** _______________

$\boldsymbol{t_{cd}}$ **(ns):** _______________

(C) (4 points) Given the timing parameters in the table above, please indicate whether each constraint is satisfied by the circuit and explain your answer.

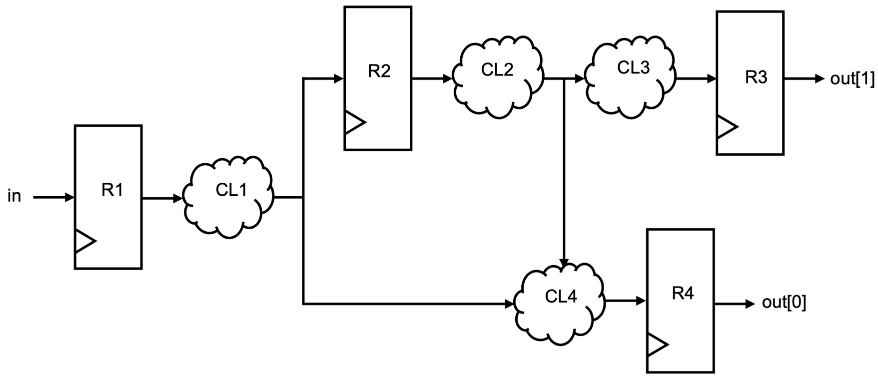
Setup time constraint (circle one): **SATISFIED** **NOT SATISFIED**

Explanation:

Hold time constraint (circle one): **SATISFIED** **NOT SATISFIED**

Explanation:

The sequential circuit and the original timing specifications have been copied here for your convenience.



	t_{pd}	t_{cd}	t_{setup}	t_{hold}
Registers	3ns	1ns	5ns	2ns
CL1	3ns	1ns	--	--
CL2	2ns	1.5ns	--	--
CL3	4ns	2ns	--	--
CL4	5ns	1ns	--	--

We now find that our supplier has the following alternative circuits for CL1, CL2, CL3, and CL4 available with the following specifications.

(D) (3 points) Our supplier replaces all registers with new registers with t_{hold} **= 2.5ns**, all driven by a common clock with period t_{clk} = 15ns. Your job is to make sure that the setup and hold time constraints are satisfied with these new registers. You may replace *only* one of the combinational circuits with its alternative specification. Please indicate which combinational circuit you are replacing. If the constraints are satisfied without any changes, please write **NONE**.

	t_{pd}	t_{cd}	t_{setup}	t_{hold}
Registers-New	3ns	1ns	5ns	**2.5ns**
CL1-New	2ns	1.5ns	--	--
CL2-New	1ns	0.5ns	--	--
CL3-New	3ns	2ns	--	--
CL4-New	3ns	1ns	--	--

Combinational circuit replaced (or NONE): ________________

(E) (4 points) Show that your new circuit satisfies the setup and hold time constraints below. If no changes are needed, write **NO CHANGES REQUIRED** below.

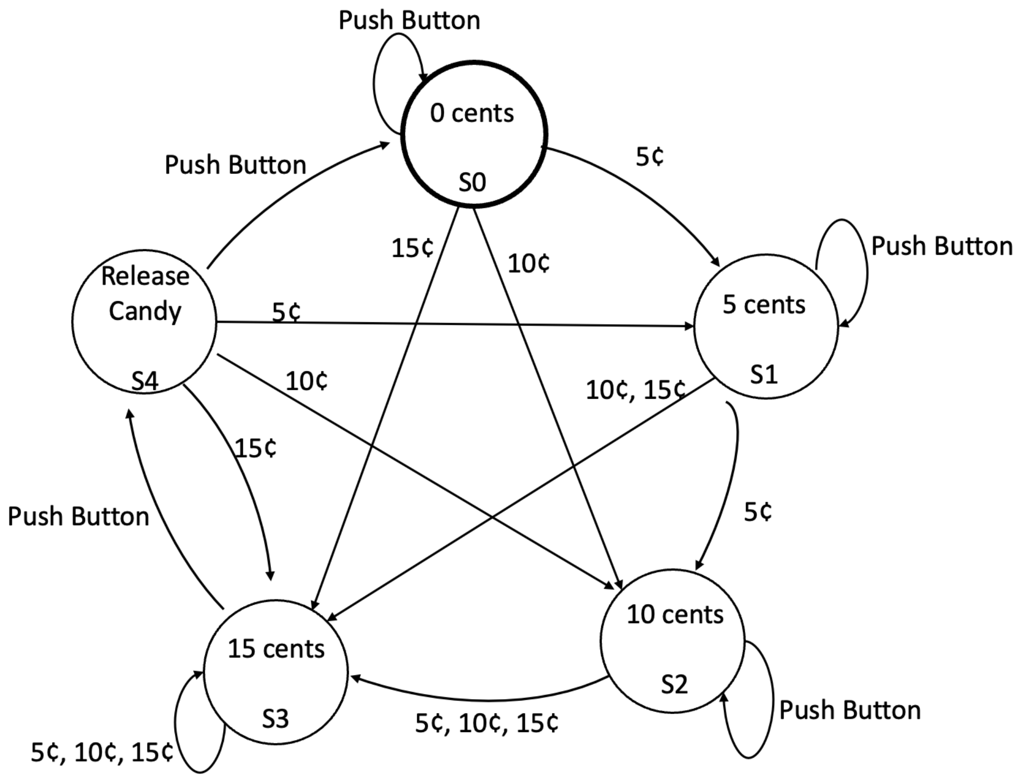
Setup time constraint satisfied:

Hold time constraint satisfied:

Problem 6. Finite State Machines (15 points)

Sid the cookie monster is trying to get candy bars out of a vending machine. He has 5¢, 10¢, and 15¢ coins, and one candy bar is worth 15¢. However, this machine is pretty rudimentary, and if more than 15¢ are put in without exchanging it for candy, the vending machine will stay at 15¢ and not record additional coins. If Sid tries to get a candy bar before putting in enough money, he won't get the candy but the amount recorded by the vending machine will stay the same.

To better understand how Sid can get candy, he draws the following finite state machine. Note that at every cycle of the FSM, there will always be an input of either a coin or a button press.



To represent this, Sid encodes the inputs into the machine using 2-bit values:

- Push Button: 2'b00
- 5¢: 2'b01
- 10¢: 2'b10
- 15¢: 2'b11

(A) (1 point) If the vending machine is in state S3 and Sid puts in any coin, what state does this FSM go to?

Next State: ________________

(B) (1 point) What state must the vending machine be in for Sid to successfully get candy?

State: ________________

(C) (1 point) How many flip flops does the vending machine FSM require to encode all possible states?

Number of Flip Flops: ________________

Before Sid starts using his hard-earned coins to get candy, he wants to know how the vending machine will behave given a series of inputs.

(D) (6 points) What is the resulting state (S0 through S4) of the vending machine at the end of each sequence of inputs provided? **Assume the vending machine is in the default state of 0 cents, S0, at the beginning of each sequence.**

i. 01 01 01 00 ________________

ii. 10 11 01 00 10 ________________

iii. 11 10 00 11 01 00 01 00 01 00 ________________

(E) (6 points) Fill out the following truth table for Sid's finite state machine based on his state transition diagram. The output should be 1 whenever the FSM outputs a candy bar, and 0 otherwise.

State	**Input**	**Next State**	**Output**
S0	00 (Push Button)		
S0	01 (5¢)		
S0	10 (10¢)		
S0	11 (15¢)		
S1	00 (Push Button)		
S1	01 (5¢)		
S1	10 (10¢)		
S1	11 (15¢)		
S2	00 (Push Button)		
S2	01 (5¢)		
S2	10 (10¢)		
S2	11 (15¢)		
S3	00 (Push Button)		
S3	01 (5¢)		
S3	10 (10¢)		
S3	11 (15¢)		
S4	00 (Push Button)		
S4	01 (5¢)		
S4	10 (10¢)		
S4	11 (15¢)		

END OF QUIZ 1!