# CSE260 Exam II

Name: _____

ID Number: _____

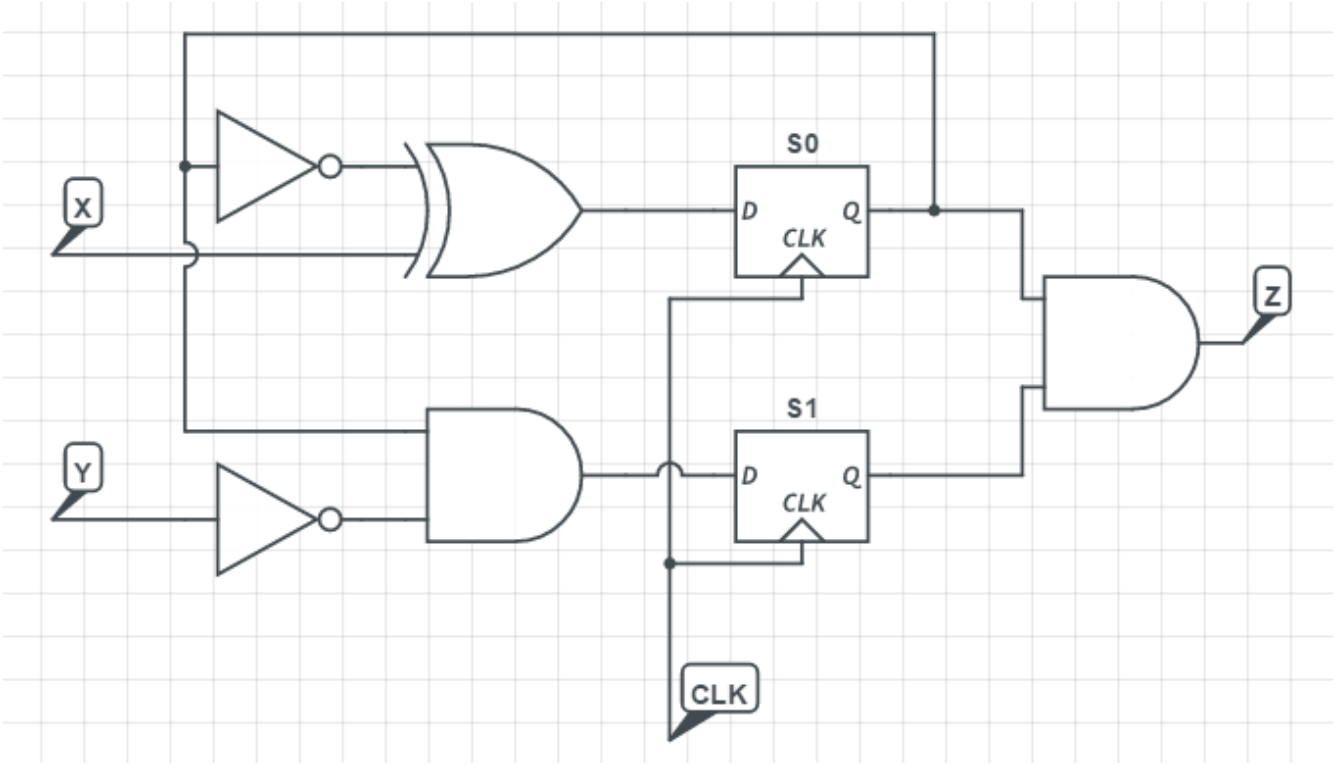
- This exam is open book and open notes. You may only use materials that you brought with you. You may not use someone else's notes.
- If the question requires you to provide code, you must use correct syntax and semantics to receive full credit. We should be able to type your answer into Xilinx and verify that it works.
- This exam is worth 15% of your final grade.
- The exam is broken up into 6 questions and is worth 100 points. It is your responsibility to make sure you have all of the pages.

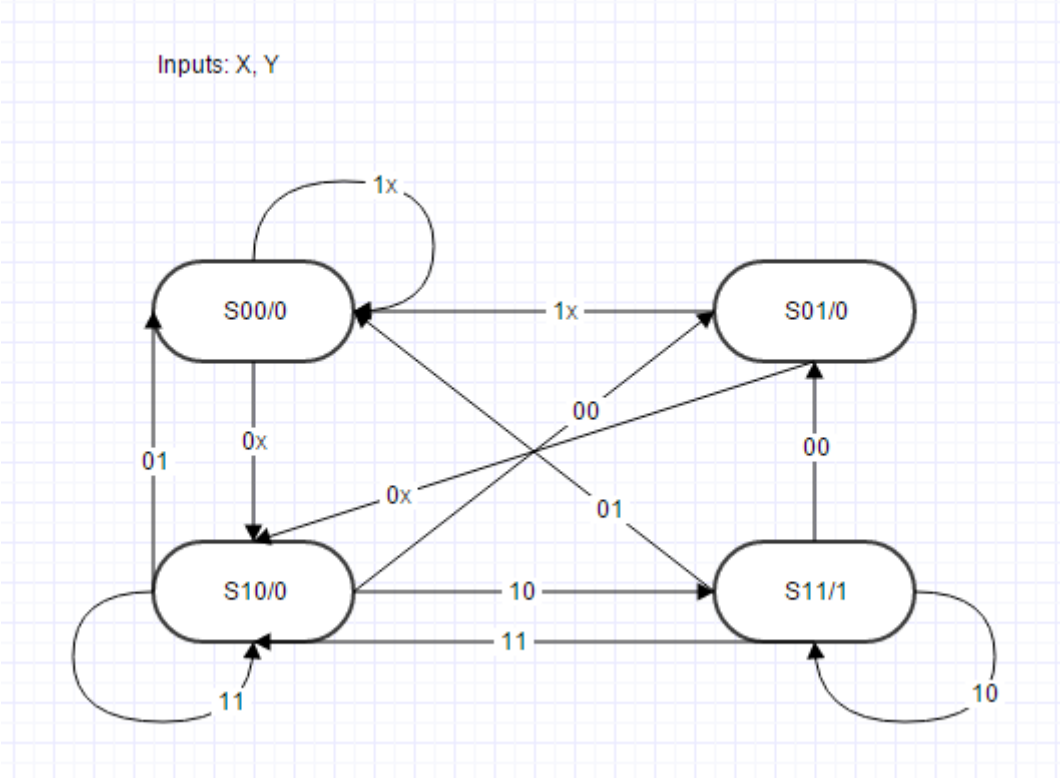
| Section | Possible Points | TA Initials | Score |
|---|---|---|---|
| I | 15 | | |
| II | 10 | | |
| III | 25 | | |
| IV | 10 | | |
| V | 15 | | |
| VI | 25 | | |

**Pledge:** On my honor, I have neither given nor received any unauthorized aid on this exam.

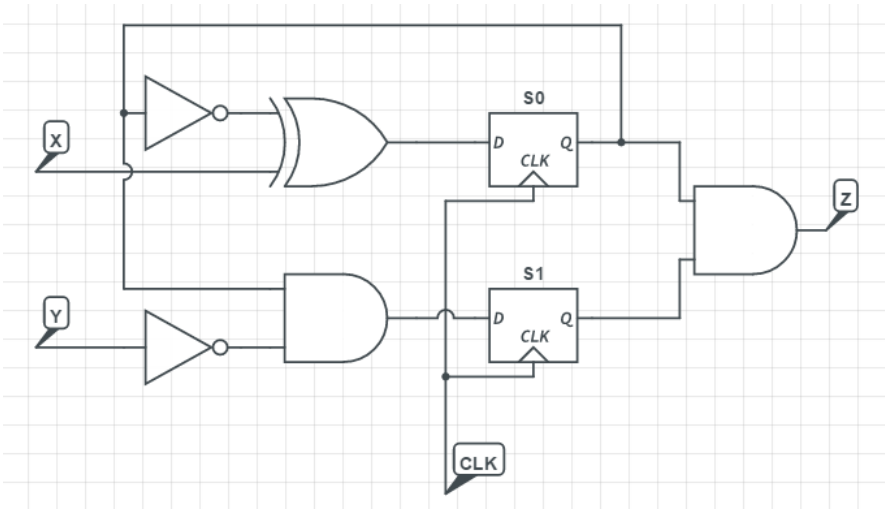
Signed: _____

1. (15 points) Draw a finite state machine for the following circuit:



2. (10 points) For the following circuit, assume a hold time of 1ns, a setup time of 2ns, a flip flop propagation delay of 1-3ns, and a gate delay of 0.5-2 ns.




Are there any hold time violations? Show your work.

1 (prop delay) + 0.5 (gate delay) > 1 (hold time) so there is no violation.

~~Are there any setup time violations? Show your work.~~

What is the minimum clock period of this circuit? Show your work.

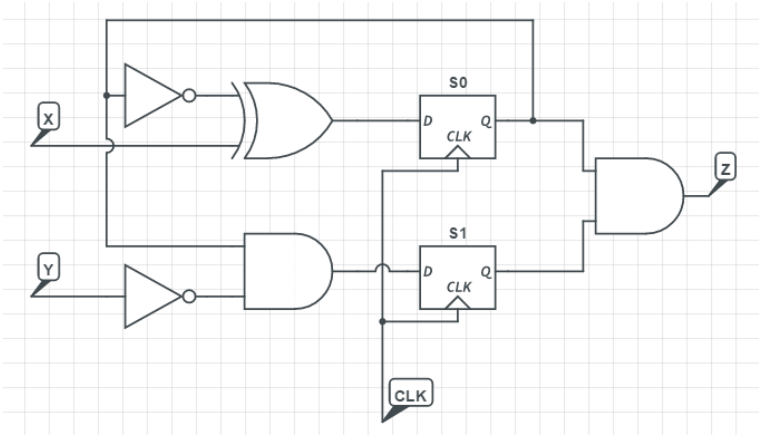
3 (prop delay) + 4 (gate delay) + 2 (setup time) = 9ns

If the clock rises at time *t* when must the inputs X and Y remain stable? Show your work.

X: t - (2 (gate delay) + 2 (setup time)) to t + (1 (hold time) – 0.5 (gate delay) = t – 4 to t + 0.5

Y: t – (4 (gate delay) + 2 (setup time)) to t + (1 (hold time – 1 (gate delay) = t – 6 to t

3. (25 points) Complete the VHDL implementation for the following circuit:




```
entity exam2FSM is
      port(
            X, Y: in STD_LOGIC;
            Z: out STD_LOGIC;
      );
end exam2FSM;

architecture a1 of exam2FSM is
--your code goes here
--Two solutions are given:
type stateType is (s00, s01, s10, s11);
signal state: stateType;
begin
      process (clk) begin
            if rising_edge(clk) then
                  --reset logic is not needed, but don't count off for it if
                  --it is present.
                  case state is
                  when s00 =>
                        if X = '0' then
                              state <= s10;
                        end if;
                  when s10 =>
                        if X = '0' and Y = '1' then
                              state <= s00;
                        elsif X = '0' and Y = '0' then
                              state <= s01;
```

```
                    elsif X = '1' and Y = '0' then
                        state <= s11
                    end if;
                when s11 =>
                    if X = '0' and Y = '0' then
                        state <= s01;
                    elsif X = '0' and Y = '1' then
                        state <= s00;
                    elsif X = '1' and Y = '1' then
                        state <= s10;
                    end if;
                when s01 =>
                    if X = '1' then
                        state <= s00;
                    else
                        state <= s10;
                    end if;
                when others =>
                end case;
            end if;
        end process;
        Z <= '1' when state = s11 else '0';
end a1;
```

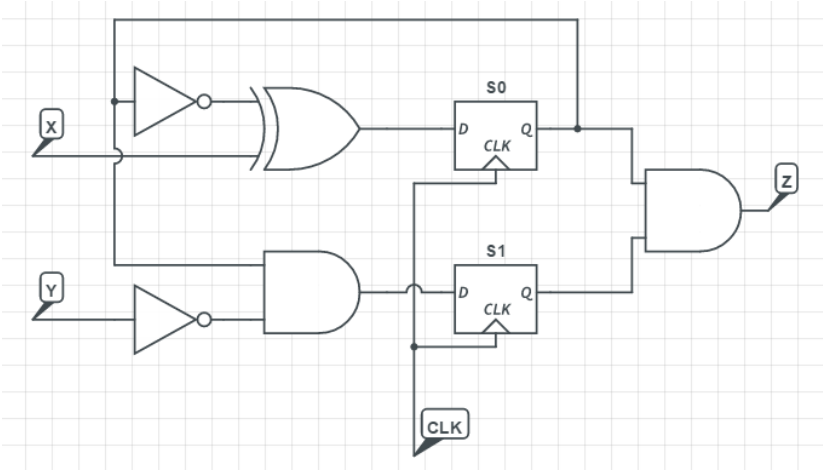
---

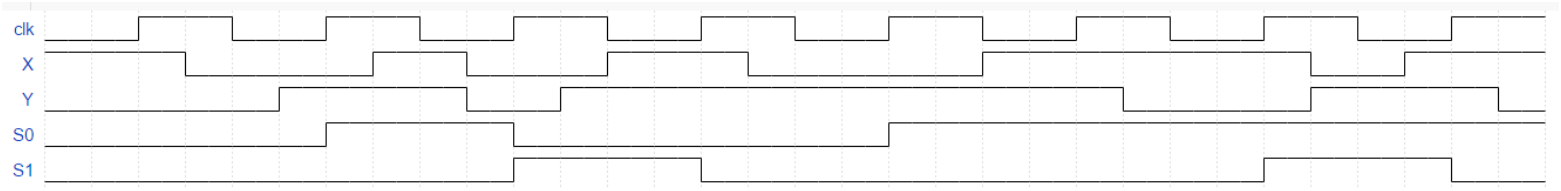
Here is an alternative which is also acceptable (and simpler):

```
signal s0, s1: std_logic;
begin
        process (clk) begin
            variable s0now: std_logic := s0;
            if rising_edge(clk) then
                --reset logic is not needed, but don't count off for it if
                --it is present.
                s0 <= (not s0now) xor X;
                s1 <= s0now and (not Y);
            end if;
        end process;
```

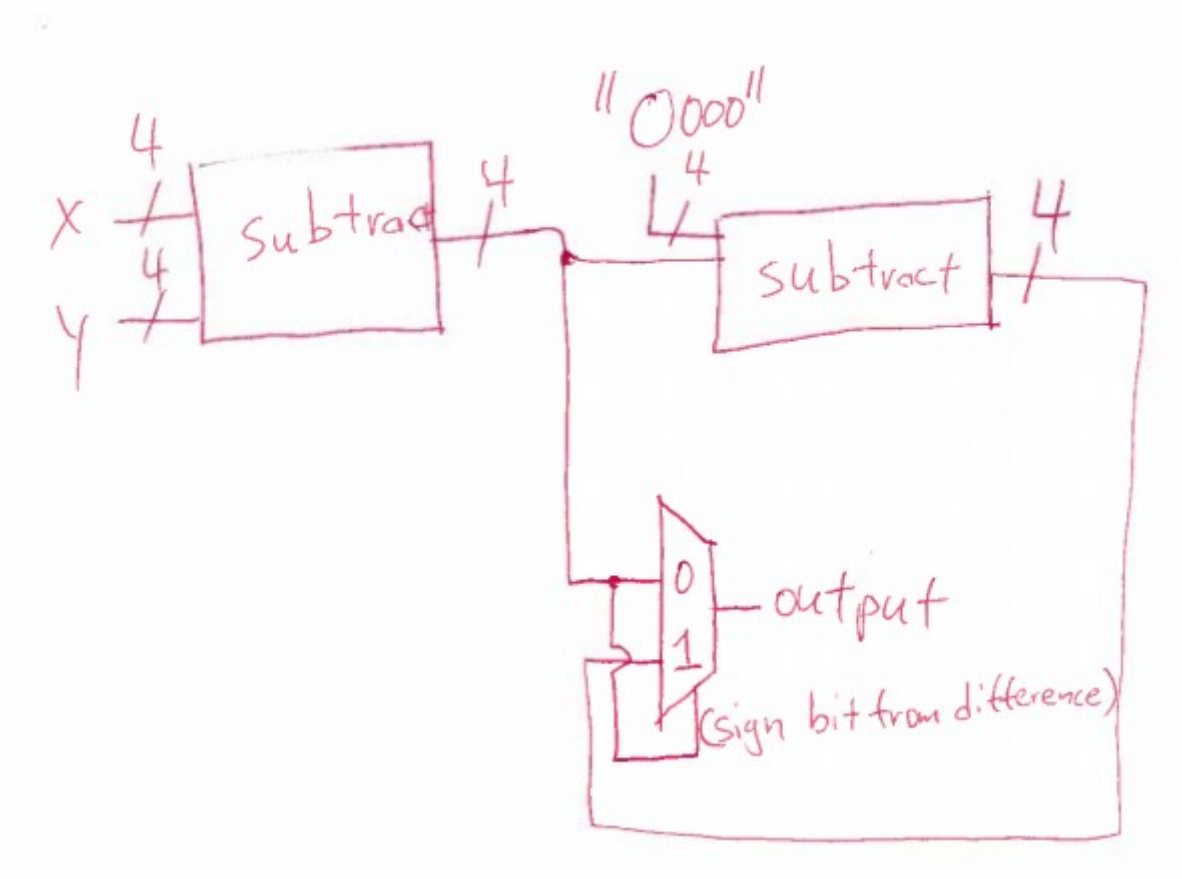
```
      Z = s0 and s1;
end a1;
```

4. (10 points) Complete the following timing diagram for this circuit. Do not worry about hold times, setup times, etc. when answering this problem. You should assume that S0 and S1 are initially storing '0'.

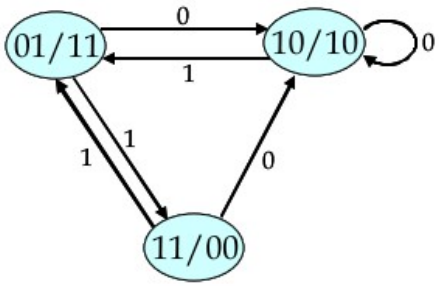


5. (15 points) Design a circuit that computes the absolute difference of two 4-bit signed binary values. If the two binary values are *x* and *y,* then absolute difference is given as $|x - y|$.Your circuit should use at least one subtractor.



6. (25 points). Construct a circuit for the following FSM:




| AB | X | $D_A D_B$ | YZ |
|---|---|---|---|
| 01 | 0 | 10 | 11 |
| 01 | 1 | 11 | 11 |
| 10 | 0 | 10 | 10 |
| 10 | 1 | 01 | 10 |
| 11 | 0 | 10 | 00 |
| 11 | 1 | 01 | 00 |

$D_A = A' + X'$

$D_B = X$

$Y = A' + B'$

$Z = A'$

