Union College
ECE318/CSC318
Assignement 3 Solutions

Problem 1.

Code Listing:

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY nand_cct IS
 PORT(
  A,B,C,D  : IN STD_LOGIC;
  X : OUT STD_LOGIC);
END nand_cct;

ARCHITECTURE data_flow OF nand_cct IS
 SIGNAL C_inv, D_inv, p, q, r: STD_LOGIC; -- internal signals in the
circuit
BEGIN

-- inverters
C_inv <= NOT C after 2 ns;
D_inv <= NOT D after 2 ns;

-- column of NAND gates
p <= NOT(A AND B AND C) after 4 ns;
q <= NOT(C_inv AND D_inv) after 4 ns;
r <= NOT(C_inv AND A AND B) after 4 ns;

-- output NAND gate
X <= NOT(p and q and r) after 4 ns;

END data_flow;
```

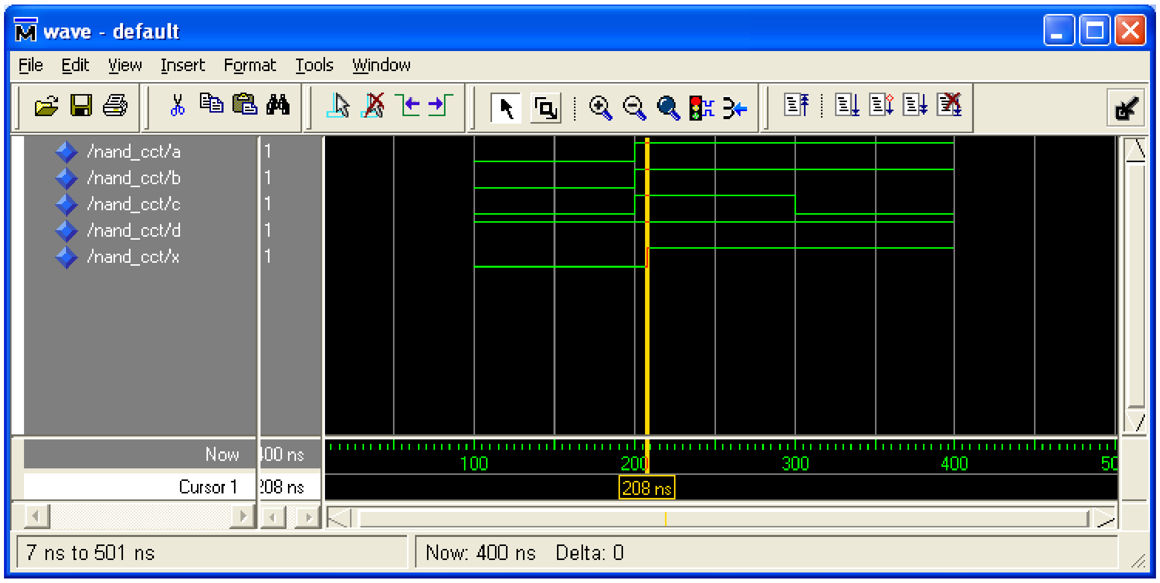


Figure 1. Simulation Results for Question 1.

ABCD 0001, --> 1111 - A, B, and C change. X settles to a final value of 1 after 8 ns since it is the top NAND gate that forces the output high.

ABCD 1111 --> 1101 – there is no change in X, i.e., stays at 1

Problem 2.

```
cout <= "001" when a = '1' else
        "010" when b = '1' else
        "100";
```

Problem 3.

(a)

```
F <= "11" when A(2) = '1' else
     "10" when A(1) = '1' else
     "01" when A(0) = '1' else
     "00";
```

(b)

```
with A select
F <= "00" when "000",
     "01 when  "001",
     "10" when "010" | "011",
     "11" when others;
```

Problem 4. The code in the assignment will not synthesize correctly and will include latches. The code below is corrected.

```
library ieee;
use ieee.std_logic_1164.all;

entity counter is
    port (reset, clk: in std_logic;
         F: buffer std_logic_vector(3 downto 0));
end counter;

architecture hw_3 of counter is
shared variable C : std_logic_vector(3 downto 0);
begin

comb: process(F) is
begin
   C := F;
   for i in 0 to 3 loop
      if C(i) = '1' then
       C(i) := '0';
     else
       C(i) := '1';
       exit;
   end if;
  end loop;
end process comb;

seq: process(clk,reset) is
begin
   if (reset = '1') then
     F <= "0000";
  elsif (rising_edge(clk)) then
     F <= C;
  end if;
end process seq;

end hw_3;
```

Simulation:

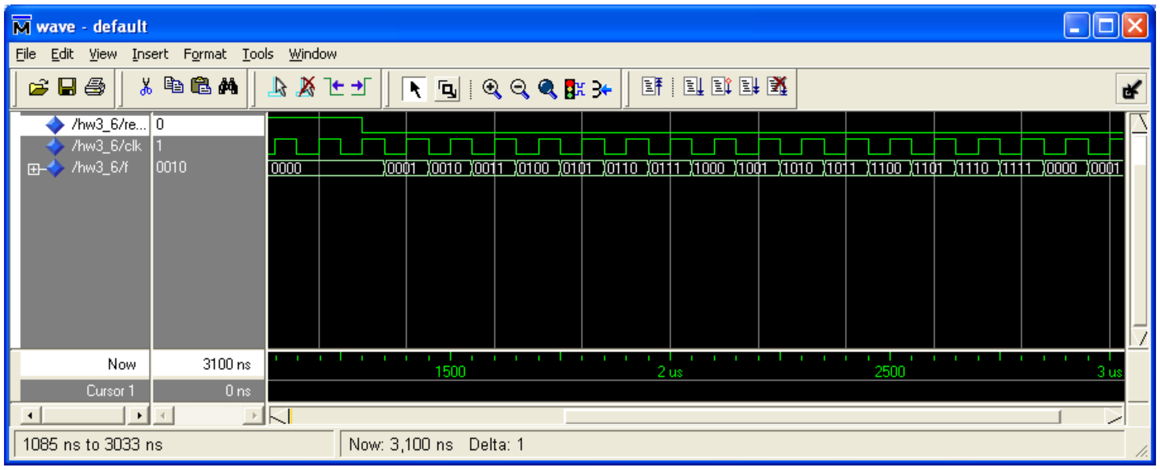
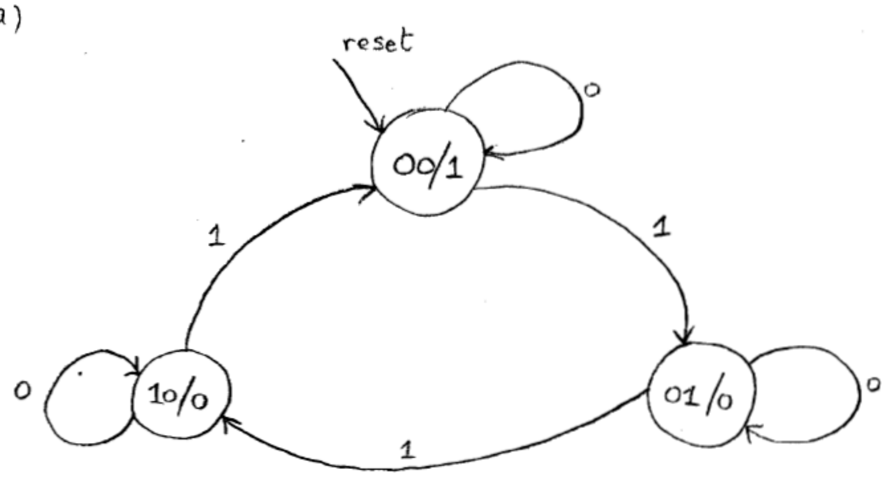


Figure 2. Simulation Results for Question 4.

The reset is synchronous, since it does not appear in the sensitivity list of the process. The “for” loop implements the "increment" algorithm. From the simulation it is obvious that this is a counter circuit, and the count changes on the **negative** edge of the clock. Notice that the reset does not take effect until there is a negative clock edge.

Q5

(a)



(b)

Input x	Current State S_1	S_0	Next State S_1^+	S_0^+	Output Z
0	0	0	0	0	1
1	0	0	0	1	0
0	0	1	0	1	0
1	0	1	1	0	0
0	1	0	1	0	0
1	1	0	0	0	1

S_1^+

$x \backslash S_1S_0$	00	01	11	10
0	0	0	X	1
1	0	1	X	0

S_0^+

$x \backslash S_1S_0$	00	01	11	10
0	0	1	X	0
1	1	0	X	0

Z :

$x \backslash S_1S_0$	00	01	11	10
0	1	0	X	0
1	0	0	X	1

Don't Cares don't buy us too much here so instead make them ϕ $\Rightarrow$ reset to 00 state if we get to a 11 state. In this case

$$S_1^+ = X.\overline{S_1}.S_0 + \overline{X}.S_1.\overline{S_0}$$

$$S_0^+ = \overline{X}.\overline{S_1}.S_0 + X.\overline{S_1}.\overline{S_0}$$

$Z = \overline{X}.\overline{S_1}.\overline{S_0} + X.S_1.\overline{S_0}$ [not asked for in the problem]

Q6

(a)

Clock Frequency = 166MHz

$$\Rightarrow \quad t_\phi = \frac{1}{166 \times 10^6} \simeq 6\text{ns}.$$

$t_{su} = 1\text{ns}$

$t_p = 2\text{ns}.$

From class :- $t_p + t_{su} + t_{c_{max}} \le t_\phi$

$$\Rightarrow \quad t_{c_{max}} = 3\text{ns}.$$

(b) Longest path back to the input is through NOR and 2 NANDs

$$\Rightarrow \quad 3\,t_d = 3\text{ns}$$

t_d = NOR/NAND delay = 1ns.

(c) Mealy FSM.

Q7

(a)

$t_{c_{max}} = 3\,ns$

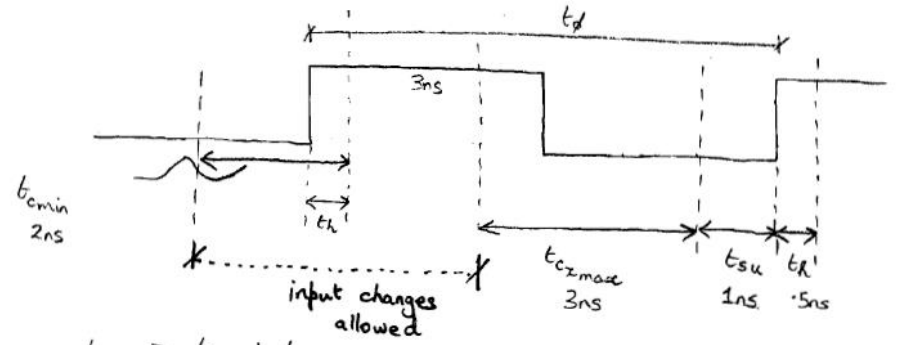
$t_{c_{min}} = 2\,ns.$

Say an input changes, when does this appear at D?

For A : 3 ns

B : 2.5 ns

C : 2 ns.



$t_x \geqslant t_{su} + t_{c_{x_{max}}}$

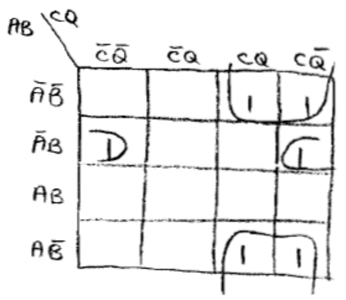
$\Rightarrow t_x \geqslant 4\,ns.$

7/(b)

$$\mathcal{D} = \overline{\left[\overline{\overline{(A+Q)}.B}\right].\left(\overline{\overline{B}.C}\right)}$$

$$= \overline{(A+Q)}.B + (\overline{B}.C)$$

$$= \overline{A}.\overline{Q}.B + \overline{B}.C$$



AB \ CQ	$\overline{C}\overline{Q}$	$\overline{C}Q$	CQ	$C\overline{Q}$
$\overline{A}\overline{B}$			1	1
$\overline{A}B$	1			1
AB				
$A\overline{B}$			1	1

Since all sets of adjacent 1's are not circled a hazard is possible.

(c) a hazard is possible but if the inputs change only in the allowed region then the glitch will not affect the operation of the FSM.
[i.e., any glitch does not appear at $\mathcal{D}$ during t_{su} or t_h]