Digital Systems II - DGD 2

Question 1

Explain the difference in the output of these two pieces of code. Assume that a,b,c and y are signals. Temp is a signal in the code on the left and it is a variable in the code on the right.

Process (a,b,c,temp)
Begin
  Temp <= '0';
  Temp <= temp or a;
  Temp <= temp or b;
  Y <= temp;
End process;
 	
Assignment isn't done until the end of the process.
Only the last <= would count for Y and Temp.
Final value of
temp = temp or b;
Y = temp;
	Process (a,b,c)
	  Variable temp: std_logic;
	Begin
	  Temp <= '0';
	  Temp <= temp or a;
	  Temp <= temp or b;
	  Y <= temp;
	End process;
	
	Allocation is performed right away and hold a value throughout the function.
	Final value of
	temp = a or b;
	Y = a or b;

':=' is an assignment to a variable
'<=' clears value and assigns variable

Question 2

Consider a 2-by-2 search. It has two input ports x0, and x1 and a 2-bit control signal, ctrl. The input ports are routed to output ports y0, and y1. According to the ctrl signal. The function table is specified below.

a) Draw the conceptual diagram
b) Use concurrent signal assignment statements to derive the circuit

Input	Output	Fn
00	X1 X0	Pass
01	X1 X1	Broadcast x1
10	X0 X0	Broadcast X0
11	X0 X1	Cross

Output could be a 4-to-1 multiplexer with two control bits. Simplifying could reduce it to a 2-to-1 multiplexer and only one control bit.

Y = out std_logic_vector(1 downto 0);
X = in std_logic_vector(1 downto 0);
Ctrl: in std_logic_vector(1 downto 0);

```
Y <= (X1 & X0) when (ctrl = “00”)
      else (X1 & X1) when (ctrl = “01”)
      else (X0 & X0) when (ctrl = “10”)
      else (X0 & X1) when (ctrl = “11”)
```

Question 3

VHDL has ben given, Draw block diagram:

```
libary ieee;
use ieee.std_logic_1164.all;

entity hundred_counter is
    port (
        clk, reset; in std_logic;
        en: in std_logic;
        q_ten, q_one: out std_logic_vector(3 downto 0);
        p100: out std_logic;
    )
end hundred_counter;

architecture str_arch of hundred_counter is
    component dec_counter
    port(
        clk, reset: in std_logic;
        en: in std_logic;
        q: out std_logic_vector(3 downto 0);
        pulse: out std_logic;
    )

    signal p_one, p_ten: std_logic;
begin
    one_digit: dec_counter port map(clk=>clk, reset=>reset, en=>en,
pulse=> p_one, q=>p_ten);
    ten_digit: dec_counter port map(clk=>clk, reset=>reset, en=>p_one,
pulse=>p_ten, q=>q_ten);

    p100 <- p_ont and p_ten;

end str_arch;
```

Question 4

Assume that a and y are 8-bit signals with the std_logic_vector(7 downto 0) data type. We want to perform a mod 8 and assign the result to y. Write a signal assignment statement using only the & operator.

We can bit shift 3 to the right and export those bits that we shifted padded with “00000” to the right
Y <= “000000” & a(2 downto 0)

Question 5

Make corrections to the code below:

```
signal s1, s2, s3, s4, s5, s6, s7: std_logic_vector(3 downto 0);
signal u1, u2, u3, u4, u5, u6, u2: unsigned(3 downto 0);
signal sg: signed(3 downto 0);

u1 <= 2#0001#;
u2 <= u2 and u4;
u5 <= s1 + 1;
u6 <= u3 + u4 + 3;
u7 <= (others => '1');

s2 <- s3 + s4 - 1;
s5 <- (others -> '1');
s6 <- u3 and u4;
sg <= u3 - 1;
s7 <= not sg;
```

Line 1: Since u1 is unsigned, the first line is an incorrect operation:
- u1 <= “0001”

Line 2: We can not perform logic operations on unsigned variables:
- u2 <= unsigned(std_logic_vector(u3) and std_logic_vector(u4))

Line 3: Assigning a signed to an unsigned
- U5 <= unsigned(s1) + 1

Line 4: This line has no problem

Line 5: This line has no problem

Line 6:
- S2 <= std_logic_vector(unsigned(s3) + unsigned(s4) - 1);

Question 6

Consider a circuit described by the following code segment:

```
process(a)
begin
    q <= d
end process;
```

Whenever the value changes, the value q take the value d.

Question 7

Consider the following code segment:

```
process(a,begin)
begin
    if a='1' then
        q <= b;
    end if;
end process;
```

Lab 2

Part one will be running a half adder in ModelSIM and quartus.

Code will be need to be changed for quartus.