German University in Cairo - GUC
Faculty of Media Engineering and Technology- MET
Assoc. Prof. Milad Ghantous

CSEN605 – Digital System Design
Winter Semester 2021/2022
Final Exam - solution

Bar Code

Tick your major

☐ CS ☐ DMET

Instructions: **Read Carefully Before Proceeding.**

1- Closed book examination. Non-programmable calculators are allowed
2- Write your solutions in the space provided
3- The exam consists of **(XX) questions**
4- This exam booklet contains **(XX) pages** including this page, and the scratch sheet.
5- Total time allowed for this exam is **(180) minutes**
6- When you are told that time is up, stop working on the test

Good Luck!

Question	1	2	3	4	5	6	∑
Possible Marks	10	20	20	20	20	10	100
Final Marks							

Question 1

Choose the correct answer for each of the below statements and write your answers in the provided table.

1) Consider the following process(clock). This process becomes active on:
a) positive edge of the clock
b) negative edge of the clock
c) both edges
d) none of the above

2) Consider a component with process(clk). This component
a) is synchronous
b) is asynchronous
c) could be both depending on the code inside the process
d) none of the above.

3) A 120 MHz clock is feeding into a 16 bit counter with output Q(15 downto 0). The frequency produced on output Q(15) is
a) 3.6 Khz
b) 1.8 Khz
c) 60 Mhz
d) 30 Mhz

4) To activate a process on both negative and positive edge we can write:
a) wait until clock='1' and clock'event;
b) wait until clock='0' and clock'event;
c) wait until clock ='0' and clock ='1';
d) wait until clock'event;

5) Consider the entity E with Generic (N: integer:=3). The output Q
a) can be 3 bits and above
b) can be up to 3 bits
c) both a and b are valid
d) none of the above.

6) Which statement is true?
a) In Moore systems, the output depends on the input only
b) In Moore systems, the output depends on the state only
c) In Mealy systems, the output depends on the input only
d) In Mealy systems, the output depends on the state only

7) the type BUFFER:
a) allows several signals to connect to it.
b) allows it to be connected to an output and an input at the same time.
c) is just like OUT
d) is just like INOUT

8) A tri-state buffer with input A, output B, control C, acts like a 2-1 mux with:
a) inputs A and B
b) inputs A and C
c) inputs A and 'Z'
d) inputs C and 'Z'

9) Overflow in unsigned numbers is the same as Cout.
a) True
b) False
c) more information is needed.

10) What's the behavior of the following code assuming output Q was '1' initially.

```
Process (clock)
begin
if( clk'event and clk='1')
Q <='0';
End process;
```

a) On every rising edge of the clock, Q becomes 0 then goes back to '1' if no rising edge is detected.
b) On the first rising edge of the clock, Q becomes 0 and stays 0 forever
c) On every falling edge of the clock, Q becomes 0 then goes back to '1' if no falling edge is detected.
d) On the falling edge of the clock, Q becomes 0 and stays 0 forever.

Write your answers here

1	2	3	4	5
c	a	b	d	c
6	7	8	9	10
b	b	c	a	b

Question 2 (20 pts)

Part A) Construct, in VHDL, a *generic* multiplexer with the generic size N, with N being the number of inputs. (example N=4 means a 4-1 mux).

- Let the default value of N be =2, and assume you have a `log()` function that you can use from the math library, which computes log base 2 of any number.
- The mux input should be a vector called mux_in, the select line(s) called sel, and the output is called mux_out.
- Don't worry about importing any libraries. Assume all of them are available.

(5 pts) please consider the whatsapp conversation about the two possible solutions.

```
Entity genericMux is
     Generic( N: integer:=2)
     Port( mux_in: IN std_logic_vector(N-1 downto 0),
     sel: IN std_logic_vector (0 to (log(N)-1),  (or integer 0
     to N-1)
     mux_out: OUT std_logic_vector)
End genericMux;
```

```
 (5 pts)
Architecture arch of genericMux is
Begin
     Process(mux_in, sel)
     Begin

     Mux_out <= Mux_in(sel);
     End process;
End arch
```

Part B) Write a function in VHDL that accepts a vector A of 8 bits and an integer input N. The function returns a value equal to A divided by 2, N times. For example, if A was 24 and N=3, the output is (24/2)/2)/2) = 3. The function should output a vector of zeros if N is greater than 8.

10 pts

```
Function div( A: std_logic_vectore(7 downto 0), N: integer)
returns std_logic_vector

Signal B: std_logic_vectore(7 downto 0):= A;
If N>=8
B <= “00000000”;
Else {
For i in 0 to N-1 loop
B(7-i) <= ‘0’;
}
Return B;
```

Question 3 (20 pts)

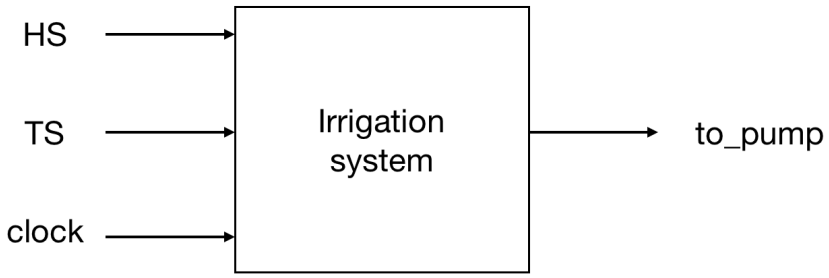
Part A- 14 pts) Design, in VHDL, a small autonomous irrigation system. The system has two digital sensors: a soil humidity sensor **HS** and a soil temperature sensor **TS**, each one is feeding our system with an 8-bit value.

- When the temperature is equal or above 32, regardless of humidity, the irrigation system must enable a water pump by sending a HIGH to it through the **`to_pump`** output. The pump should be kept enabled for up to 15 seconds or until the soil temperature drops below 32.

- When the temperature is below 32 but the humidity is equal or below 63, the pump is enabled also. The pump should be kept enabled for at most 25 seconds in this case, or until humidity rises.

- In all other cases, the pump should be off.

- The whole system is driven by a 5 HZ clock.



Entity irrigation is (2 pts)
port (hs, ts: in std_logic_vector(7 downto 0);
clk: in std_logic;
to_pump: out std_logic)
end irrigation

architecture arch of irrigation is (12 pts)
signal count: integer range 0 to 128 :=0;
begin

process(clk)

```
if(clk'event and clk='1') then

if(ts >= "100000") then
count <= count +1;
if (count<=75) then   // because clock is 5 hz, we need 75 cycles.
   to_pump <= '1';
else
   to_pump<='0';
   count <= 0;
end if

elsif (hs <= "111111") then
count <= count+1;
if(count <= 125) then // 125 cycles to achieve 25 seconds
to_pump <= '1';
else
to_pump <='0';
count <= 0;

else to_pump <='0';
end if
end if
end process
```

Part B – 6 pts) Assume that a clock of 5 HZ was not available, but you have a counter that is driven by a clock of 20 HZ. The counter is called count8 and has a clk input and 8-bit output Q and nothing else: it's always counting.

B.1) Explain, in schematic, how would you connect the above system to the counter.

Answer: (3 pts for connecting C(1))

B.2) Write down the VHDL of the entire system. (assume irrigation entity you wrote in part a is available in the library and no need to write it again, as well as the counter is available also). 3 pts for signal, and two port maps

```
Entity entireSystem port ( hs, ts, clk, to_pump)
Architecture arch
Signal counter_out: std_logic(7 downto 0);
C1: count8 port map(clk, counter_out);
C2: irrigation port map(hs, ts, counter_out(1), to_pump)
End arch
```

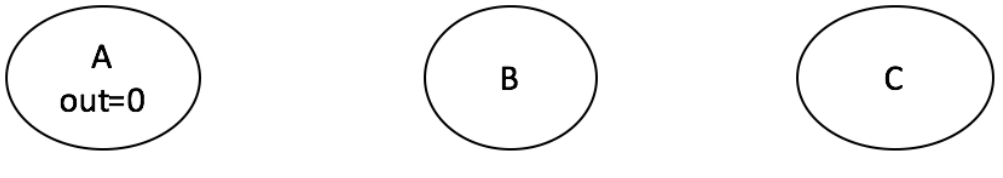
Question 4 (20 pts)

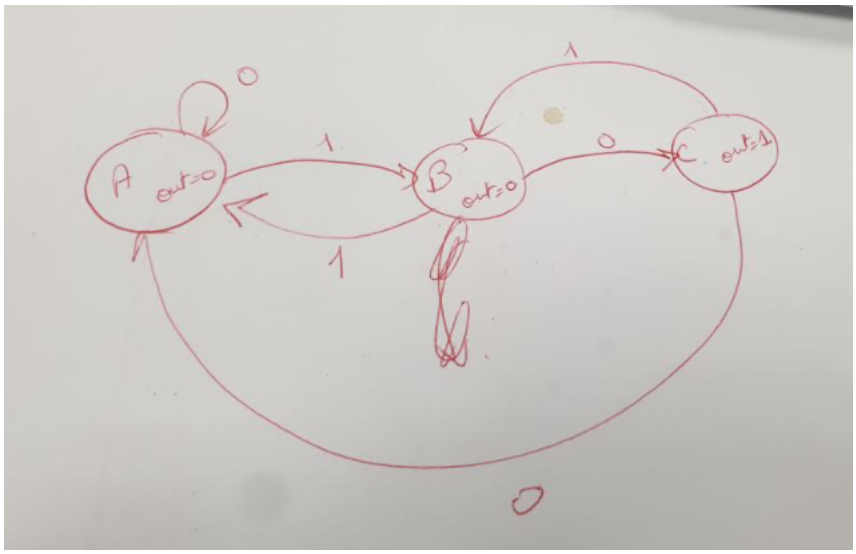
Design the Moore FSM of a system that outputs a '1' only after a sequence of an odd number of 1s is detected in a row. If an even number of 1s is detected, the outputs remains zero.
Note that the output is not ready until a zero is detected after the sequence of 1s to denote the end of the 1s sequence.

Example:

Stream	0	0	**1**	**1**	0	**1**	0	**1**	**1**	**1**	0	0
output	0	0	0	0	0	0	0	**1**	0	0	0	**1**

Part a) Complete the diagram. Consider a Reset input that resets the FSM to state A. (9 pts)





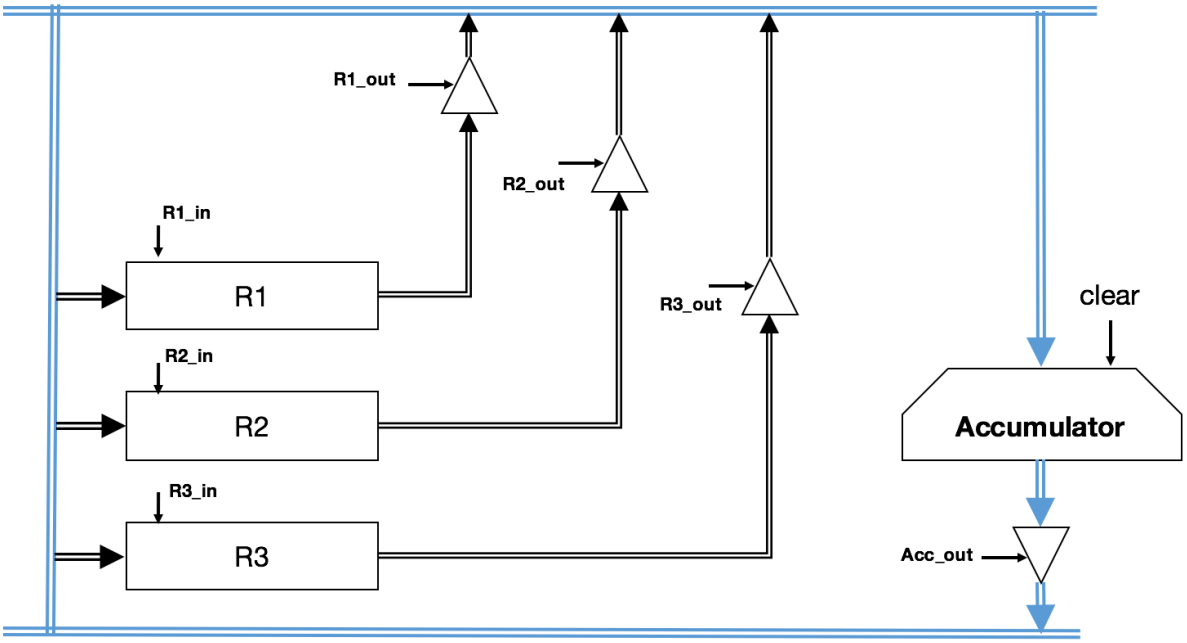
Part b) Write the corresponding VHDL code. (11 pts, 1 for each)

```
ENTITY oddchecker IS
PORT ( Clock, Resetn, w : IN STD LOGIC ;
 z : OUT STD LOGIC ) ;
 END oddchecker ;

 ARCHITECTURE Behavior OF sequenceDetector IS
 TYPE myState IS (  A,B,C) ;
 SIGNAL y: MYSTATE;
 BEGIN

PROCESS (RESETN,CLOCK)
 BEGIN
 If (  resetn='1'      ) then
 y <= A   ;
 elsif (clock'event and clock='1') then
CASE y IS
 WHEN A =>
If w='1' then y <= B; end if;  (the else here is redundant but
not false)
WHEN B =>
If w='1' then y<= A; else y<= C end if;
 WHEN C =>
If w='0' then y<= A; else y<= B; end if;
 END CASE ;
END PROCESS;
Z <= '1' when y=C else '0';

 END Behavior
```

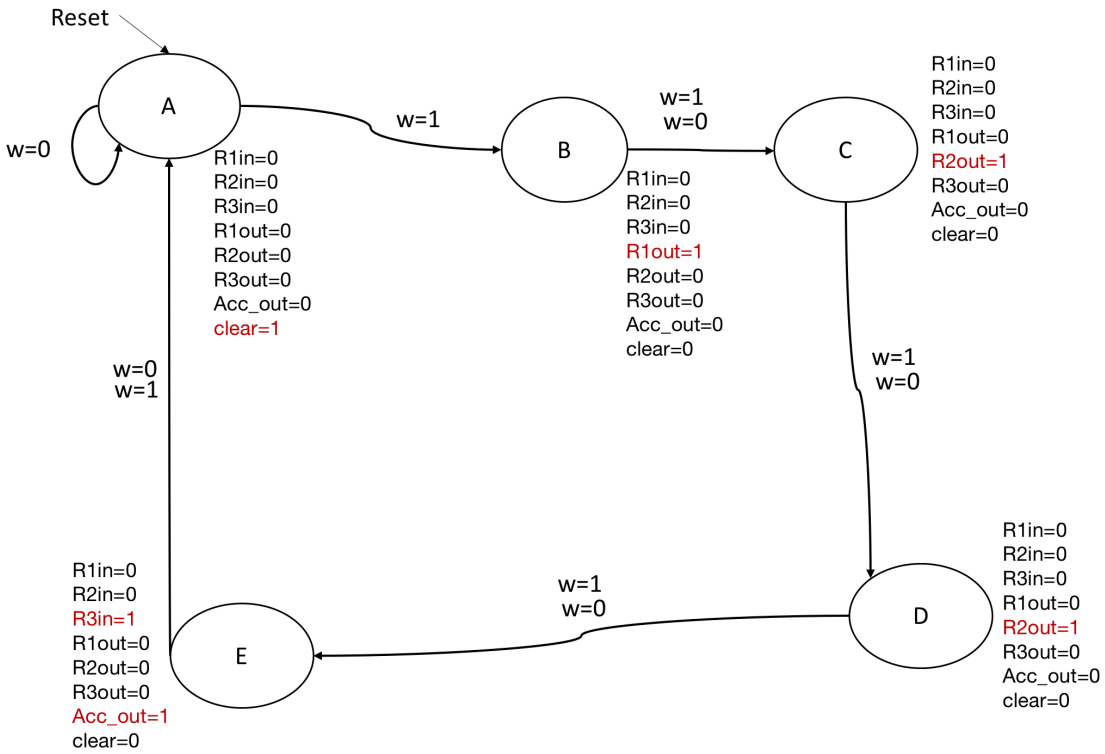
Question 5 (20 pts)

Consider this accumulator circuit comprising 3 registers and one accumulator that has a clear input, that, when asserted to 1, the accumulator internal values are set to 0. The accumulator accumulates the value present on its input every clock cycle, when clear =0. In other words, it adds the input to its current value.

We wish to implement a control unit that can perform (R1+2xR2) and save the result into R3, when an signal w is asserted to 1 for one clock cycle to initiate the beginning of the operation.



Part a) Design the control unit as a Moore state machine (FSM) diagram. Show the outputs in each state. The control unit should have 8 outputs: Rx_in, Rx_out, Acc_out and clear. (15 pts)



Part b) Implement it in VHDL. (5 pts)

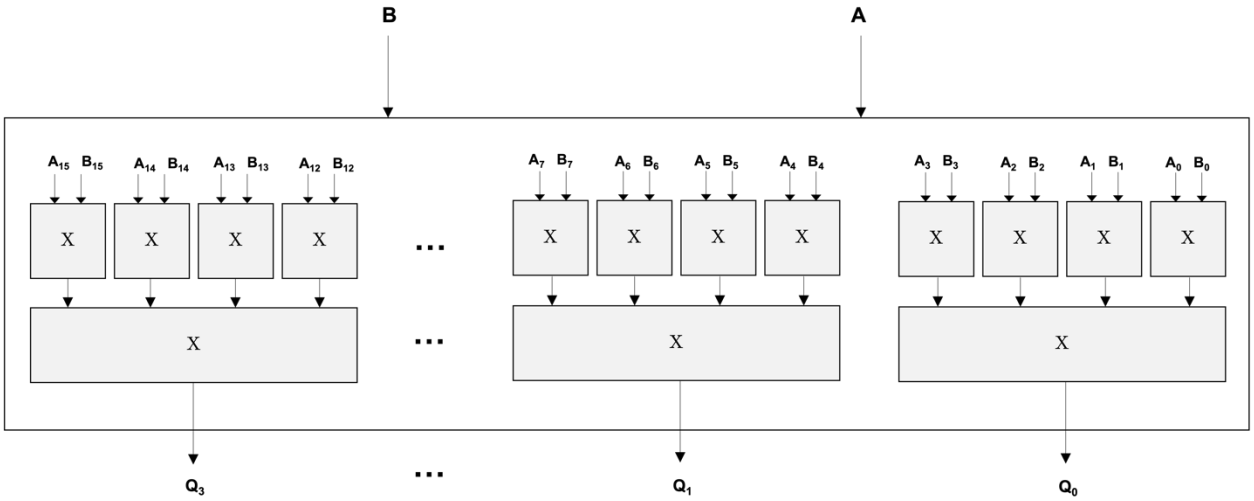
Straight forward (like the previous exercise)

Question 6 (10 pts)

Consider the following component X:

```
Entity X IS
     Generic map(N: integer=4)
     Port map( X_in: IN STD_LOGIC_VECTOR( N-1 downto 0),
               Q: OUT STD_LOGIC)
End X;
```

Construct, in VHDL, the following component called Y with inputs A and B, each of them is 16 bits and a 4-bit output Q using `FOR GENERATE` (and `if generate` if needed).



Entity Y is (2 pts)
Port (A,B: in std_logic_vectore(15 downto 0), Q: out std_logic_vector(15 downto 0))
End Y;

Architecture Arch of Y is (8 pts: using generic 2, for generate 2, special cases, 2, right connections with variable i 2))
begin
For i in 0 to 15 generate

X generic map (N => 2) port map ((A(i)&B(i)), S(i))
If (i=3 or i=7 or i=11 or i=15) generate
X port map (S(i-3 to i), Q((i-3)/4)

End generate
End generate
End arch;

Alternative solution
Can be done in two for generates

For i in 0 to 15 generate

X generic map (N => 2) port map ((A(i)&B(i)), S(i))
End generate

For I in 0 to 3 generate
X port map (S(4*i to 4*i+3), Q(i))
End generate.