EEL 4712: Digital Design

Instructor: Dr. Farimah Farahmandi

Midterm 2 – 11/1/2019

Time: 55 Minutes

Name: ……………………………………………………

UFID: ……………………………………………………

Please be neat and write (or draw) carefully. If we cannot read it with a reasonable effort, it is assumed wrong.

As always, the best answer gets the most points.

By taking this exam, you agree to follow all rules specified below or face receiving a 0% for this exam. We will be implementing a Zero-Strike policy.

- **No electronic devices or printed materials allowed**
- **Communication between students is NOT ALLOWED**
- **Under no circumstances will you share the questions from this test until after the solutions are formally released by the instructor**

Re-Grade Information:

Problem	Points	Score
1	25	
2	10	
3	15	
4	20	
5	15	
6	15	
Total	100	

```
ENTITY __entity_name IS
        PORT(__input_name, __input_name         : IN  STD_LOGIC;
               __input_vector_name              : IN  STD_LOGIC_VECTOR(__high downto __low);
               __bidir_name, __bidir_name       : INOUT      STD_LOGIC;
               __output_name, __output_name     : OUT        STD_LOGIC);
END __entity_name;

ARCHITECTURE a OF __entity_name IS
        SIGNAL __signal_name : STD_LOGIC;
        SIGNAL __signal_name : STD_LOGIC;
BEGIN
        -- Process Statement
        -- Concurrent Signal Assignment
        -- Conditional Signal Assignment
        -- Selected Signal Assignment
        -- Component Instantiation Statement
END a;

SIGNAL __signal_name : __type_name;
__instance_name: __component_name  GENERIC MAP (__component_par =>__connect_par)
                                    PORT MAP  (__component_port => __connect_port,
                                               __component_port => __connect_port);
WITH __expression SELECT
        __signal <= __expression WHEN __constant_value,
                    __expression WHEN __constant_value,
                    __expression WHEN __constant_value,
                    __expression WHEN __constant_value;
```

SELECT assignment statement

```
__signal <= __expression WHEN __boolean_expression ELSE
            __expression WHEN __boolean_expression ELSE
            __expression;
```

Conditional assignment statement

```
IF __expression THEN
   __statement;
   __statement;
ELSIF __expression THEN
   __statement;
   __statement;
ELSE
   __statement;
   __statement;
END IF;
```

```
<optional_label>:
        FOR <loop_id> IN <range> LOOP
                -- Sequential Statement(s)
        END LOOP;
```

```
WAIT UNTIL __expression;
```

```
<generate_label>:
        FOR <loop_id> IN <range> GENERATE
                -- Concurrent Statement(s)
        END GENERATE;
```

```
CASE __expression IS
        WHEN __constant_value =>
           __statement;
           __statement;
        WHEN __constant_value =>
           __statement;
           __statement;
        WHEN OTHERS =>
           __statement;
           __statement;
END CASE;
```

Question 1 (25 points):

a. (12 points) If I wanted to output a clock with a 50 us period, what ratio (scaling factor) would I use if I were using the on-board 50MHz clock? What would be the value of the counter that toggles the derived clock?

Desired: 50us = 50 * 1000ns
Source: 1 / 50MHz = 20ns
Scaling Factor = 50000ns / 20ns = 2500
Toggle Count = Scaling Factor / 2 - 1 = 1249

b. (3 points) What is the maximum number of gates that can be implemented using a 3-input, 2-output LUT?

Infinite

c. (4 points) Create a VHDL type called all_type for a constrained two-dimensional array with 200 rows and 100 columns, where each element is std_logic_vector, 64 bits.

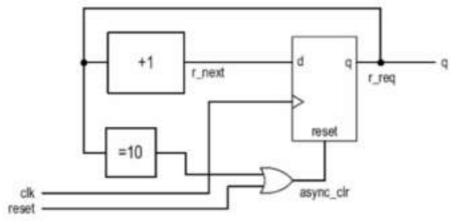
```
type all_type is array(199 downto 0, 99 downto 0) of
                std_logic_vector(63 downto 0);
```

d. (6 points) What are two ways of implementing memory on an FPGA fabric?

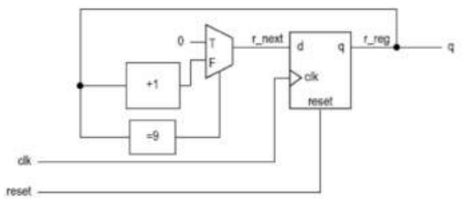
In CLBs / LEs,
or using Block RAM

Question 2 (10 points): For a Binary-Coded-Decimal counter, which of the following is a better design, and why?

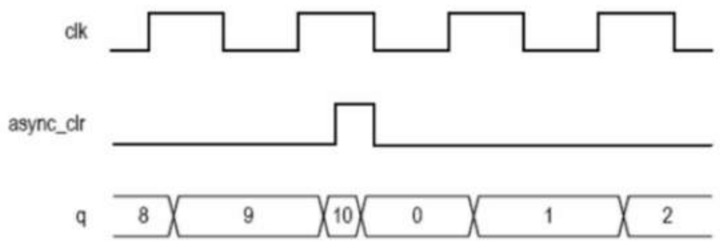
A)



B)



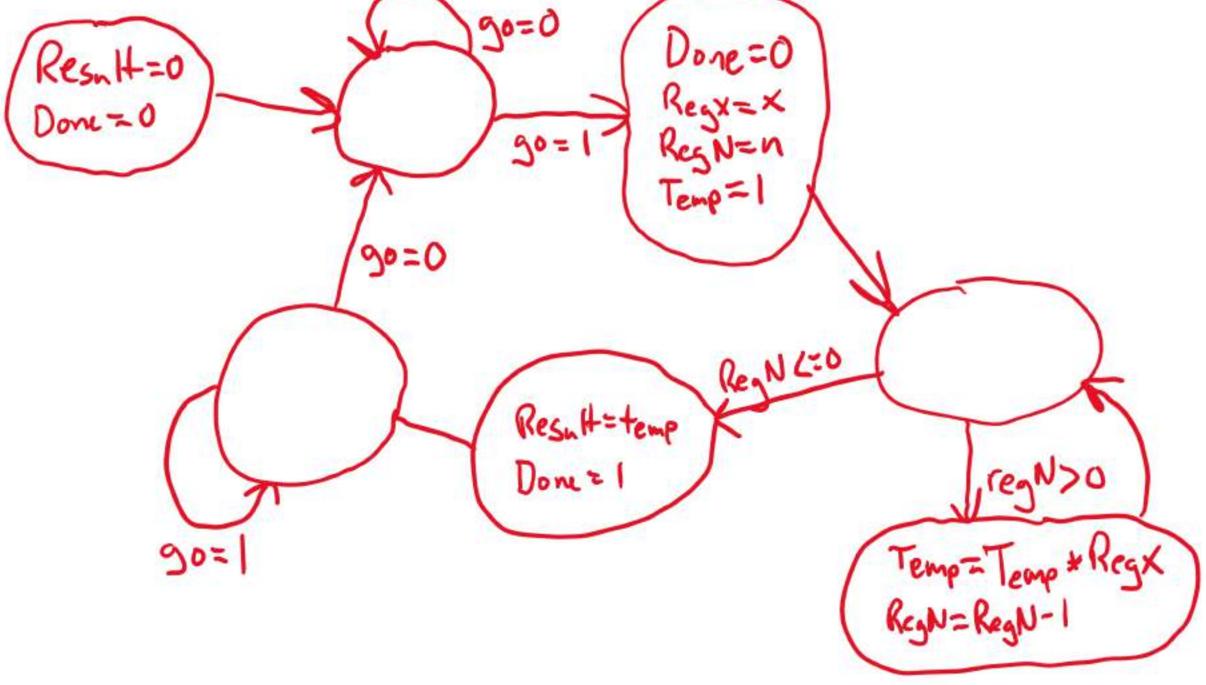
Short interval where Q out is 10 in best case, worse case, Q out could temporarily look like 10, causing premature reset to 0.



(b) Timing diagram

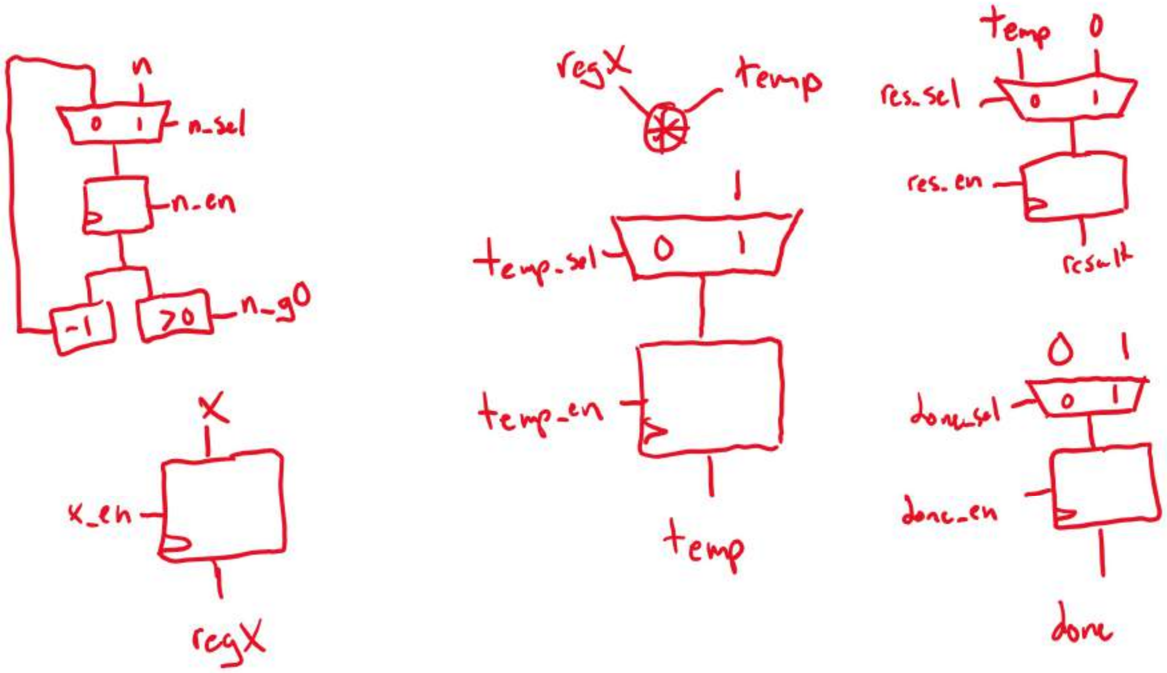
Question 3 (15 points): Create an FSMD that implements the following pseudo-code. **Do not write VHDL and instead leave the FSMD in graphical form** (i.e., state machine with corresponding operations in each state). Make sure to specify all operations and state transitions. Note that *result, go, n, x*, and *done* are I/O.

```
Input: x, n, go
Output: done, result // (xⁿ integer exponentiation)

        result = 0;
        done = 0;

        while(1) {

            // wait for go to start circuit
            while (go == 0);
            done = 0;

            // store input in register
            regX = x;
            regN = n;
            temp = 1;

            while (regN > 0)
            {
                temp = temp * regX;
                regN = regN - 1;
            }

            // output result and assert done
            result = temp;
            done = 1;

            // wait for go to clear so circuit doesn't constantly repeat
            while (go == 1);
        }
```



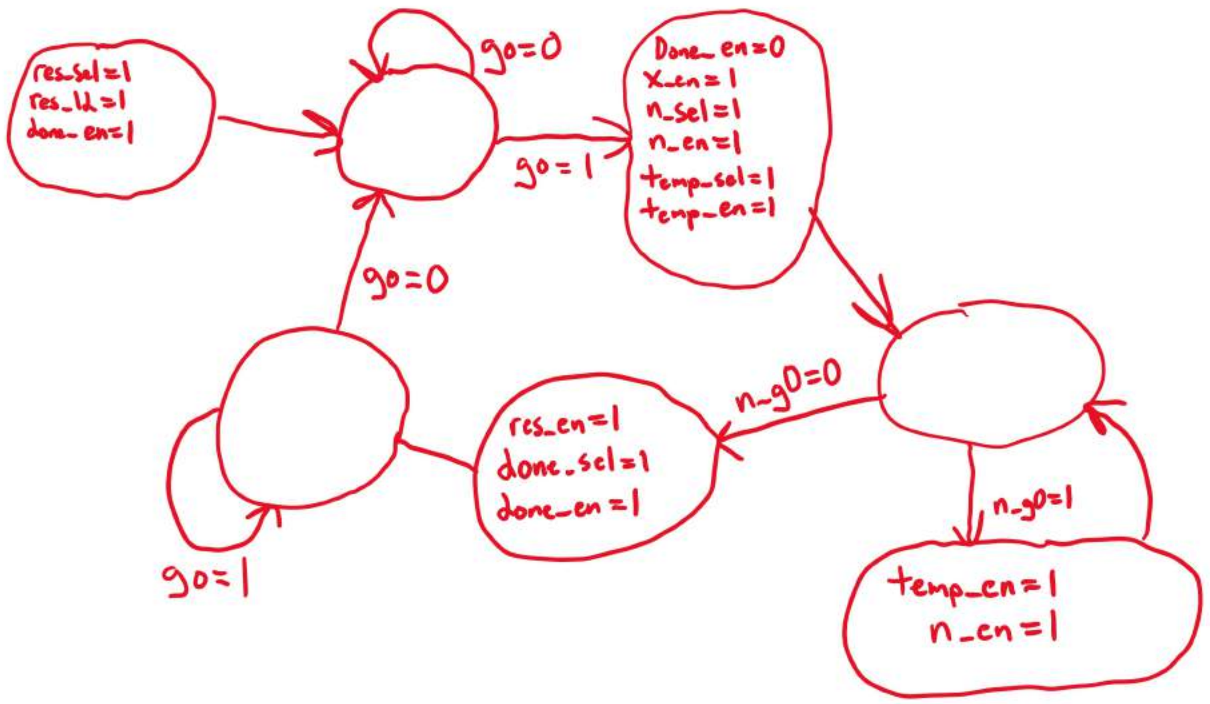
Question 4 (20 points): For the previous pseudo-code (we copied it here again for your convenience), create a datapath capable of executing the code (ignore the controller in this step). Make sure to show all control signals (i.e., mux select signals, register load signals, comparator output signals). Hint: to make things easier, do not try to share resources. Do not write any code, just show the datapath. If you do any non-obvious optimization, make sure to explain. Please include done in the datapath.

```
Input: x, n, go
Output: done, result // (result = x^n integer exponentiation)

       result = 0;
       done = 0;

       while(1) {

           // wait for go to start circuit
           while (go == 0);
           done = 0;

           // store input in register
           regX = x;
           regN = n;
           temp = 1;

           while (regN > 0)
           {
               temp = temp * regX;
               regN = regN - 1;
           }

           // output result and assert done
           result = temp;
           done = 1;

           // wait for go to clear so circuit doesn't constantly repeat
           while (go == 1);
       }
```

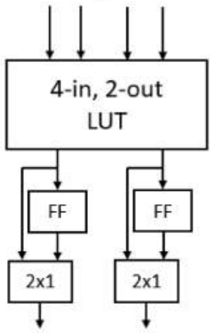


Question 5 (15 points): For the datapath in the previous step, draw an FSM capable of controlling the datapath to perform the pseudo-code. In each state of the FSM, show the values of your control signals from the previous step that configure the datapath to do the corresponding operations. Hint: Do not write any code, just show the FSM and control signals. Be sure to mention default signal values to save space.

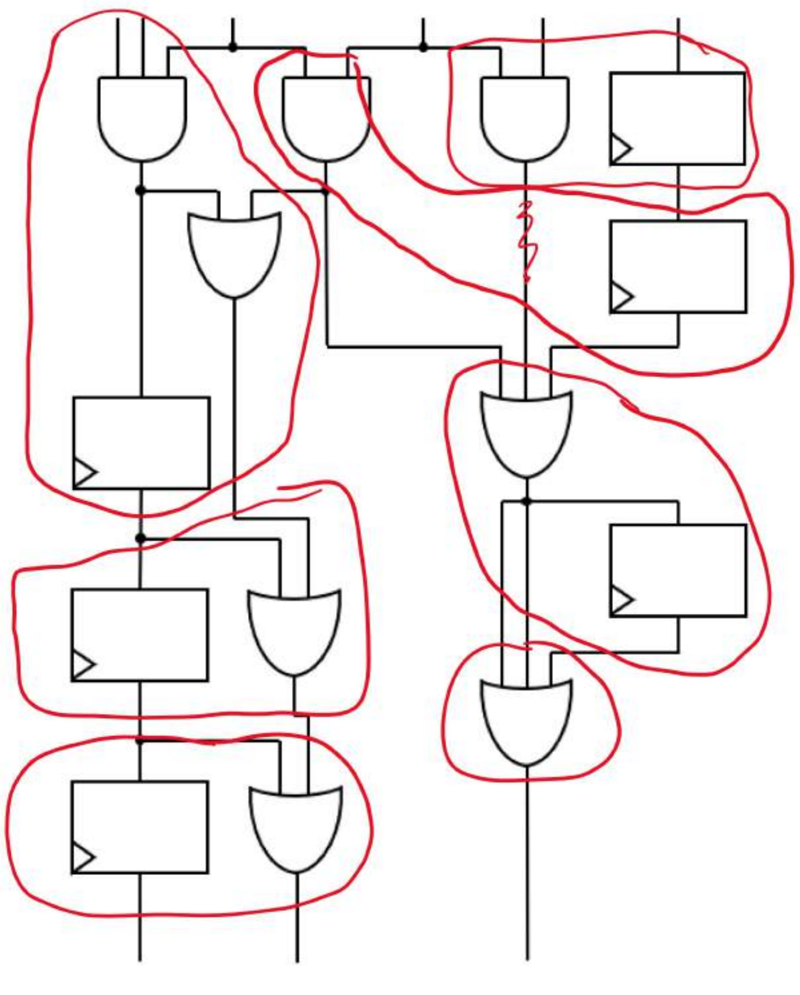
All signals default to 0



Question 6 (15 points): (a) (10 points) Assume you are given an FPGA that consists of the following CLB structures with one 4-input, 2-output LUT and optional flip flops on each output.



Map the following circuit into these CLBs by drawing boxes to represent CLBs. Assume that the rectangle components are flip flops and that everything else is combinational logic. Use the minimum number of CLBs for full credit.



(b) (5 points) If each of the 4-inputs, 2-outputs CLBs have 3ns delay, what would be the propagation delay (from inputs to outputs) of the logic shown in part (a)?

Max comb. CLB path: 3

$3 \times 3nS = 9nS$