ECE 341 Digital System Design
Final Exam

December 14, 2020

Instructions:
This exam is closed book, closed notes, no calculators. In addition, the use of Aldec or any other logic simulator is not permitted. You are permitted a "cheat sheet" consisting of an 8.5 by 11 inch sheet of paper, with written notes/diagrams or print on one side only with text size 6 point or larger. *Please submit this sheet with your exam.*

This exam will be made available both in Blackboard and in our class Dropbox folder at 3:40pm on December 14th.

This test is organized into two sections:

Section 1: Problems (synchronous, Dec 14 from 3:45-6:45pm)
Section 2: Short answer (synchronous, Dec 14 from 3:45-6:45pm)

For the duration of the exam, leave your cameras turned on throughout the Zoom session and remain within the field of view of the camera.

Your solutions can be written on regular paper and be sure to accurately label each problem. You may use any color pencil or ink other than red. If you have a tablet computer and can enter written answers on this document, you may do so, provided the document you submit can be viewed using any PDF viewer. To be clear, only hand written solutions are permitted. Solutions that are typed, word processed, and/or diagrams composed using drawing tools or drawing aids are not acceptable.

When time is called, within 15 minutes (by 7:00pm on Dec 14), upload an electronic copy (a clear scanned copy) of your test answers & "cheat sheet" in PDF format using the link provided. If you are unable to do so, (1) let me know immediately (email is fine), (2) photograph & upload individual pages from your test, again within the 15 minutes, and (3) upload a clear scanned copy as soon as possible afterwords.

Use the following link to upload your exam. It is the same upload link that was used for the other tests this semester.

`https://www.dropbox.com/request/kGjwI8YeoT6MXkta5Cv5`

Good Luck!

Please include your Honor Pledge.
I pledge to support the Honor System at Old Dominion University. I will refrain from any form of dishonesty or deception, such as lying, cheating, or plagiarizing, which are Honor violations. I am further aware that as a member of the academic community, it is my responsibility to turn in all suspected violators of the Honor System. I will report to an Honor Council hearing if summoned.

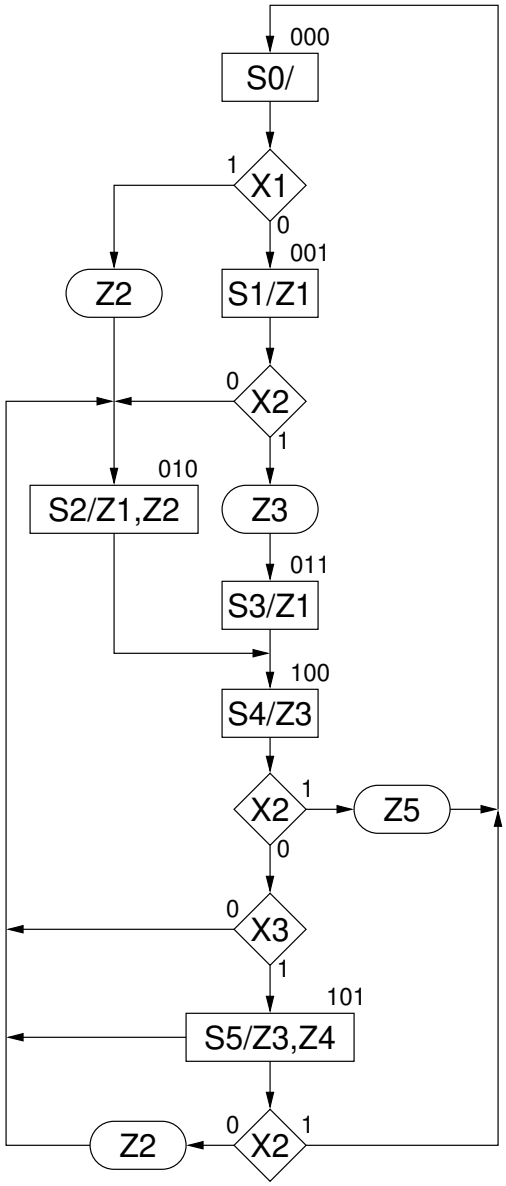
_____ _____

(Print name) (Signature)

Problems (95 points total)

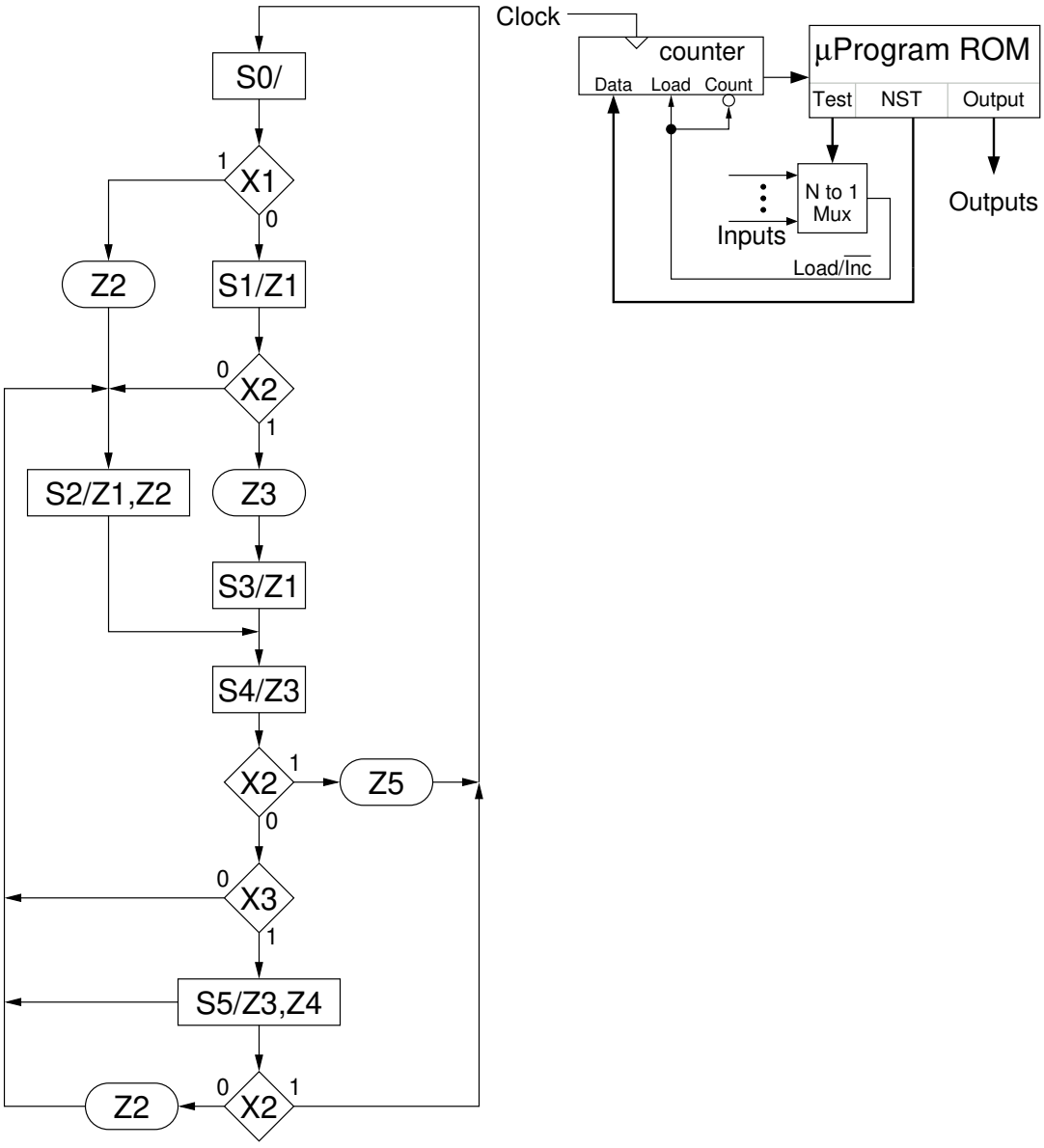
P1 (15 points) Use the following SM Chart for this problem.

P1(a) (5) For the SM chart on the right, give the SM table for the indicated state assignment.



P1(b) (4) Give the next state equations and output equations. Note that the state variables are A, B, and C. Do not simplify your next state or output equations.

P1(c) (6) Modify your SM Chart as needed to make it suitable for a one-address microcoded controller. Be sure to provide your new state assignment *annotated on the diagram* and the rational for any added states. Furthermore, provide all additional implementation details for the one-address controller including 1) the microinstruction format, 2) the configuration of test multiplexer, and 3) the microprogram memory contents.



P2 (15 points) Design a state machine that implements a 6-bit bit serial incrementer with carry. The incrementer inputs the bits one at a time on its input `X`, least significant bits first. On the same clock it inputs each bit, it outputs the respective bit for the incremented value on the output `Z`. If the counter wraps around, on the last bit of output, the counter also reports a carry on a second output `C`.

The following gives the outputs for each input

Time	0	6	12	18	24
	|	|	|	|	|
X	0 0 0 0 0 0	1 1 1 0 0 0	1 1 1 1 1 1	1 0 1 0 1 0	0 1 0 1 0 1
Z	1 0 0 0 0 0	0 0 0 1 0 0	0 0 0 0 0 0	0 1 1 0 1 0	1 1 0 1 0 1
C	0 0 0 0 0 0	0 0 0 0 0 0	0 0 0 0 0 1	0 0 0 0 0 0	0 0 0 0 0 0
Notes	input=0 output=1 carry=0	input=7 output=8 carry=0	input=63 output=0 carry=1	input=21 output=22 carry=0	input=42 output=43 carry=0

P2(a) (10) Devise a SM Chart and SM Table this device.

P2(b) (5) Give a one-hot finite state machine implementation for the counter.

P3 (15 points) Your colleague is trying to code a VHDL model for a 74194 4-bit bidirectional universal shift register. Review the code on the next page and make any and all corrections that are necessary to produce a correct working model. The code includes both syntactic (i.e. won't compile) and semantic errors (i.e. logical errors in code). Note the function table and additional information regarding the problem that you need to solve this problem.

Problem grading: for each appropriate fix, you receive a point; for each change that introduces an error, you lose a point; where maximum number of points awarded is 15 and the minimum awarded is 0.

Control Signals			Outputs				
	Mode						
$ClrN$	$S1$	$S0$	Q_3^+	Q_2^+	Q_1^+	Q_0^+	
0	-	-	0	0	0	0	(asynchronous clear)
1	0	0	Q_3	Q_2	Q_1	Q_0	(hold)
1	0	1	$RSIN$	Q_3	Q_2	Q_1	(right shift)
1	1	0	Q_2	Q_1	Q_0	$LSIN$	(left shift)
1	1	1	D_3	D_2	D_1	D_0	(parallel load)

Notes:

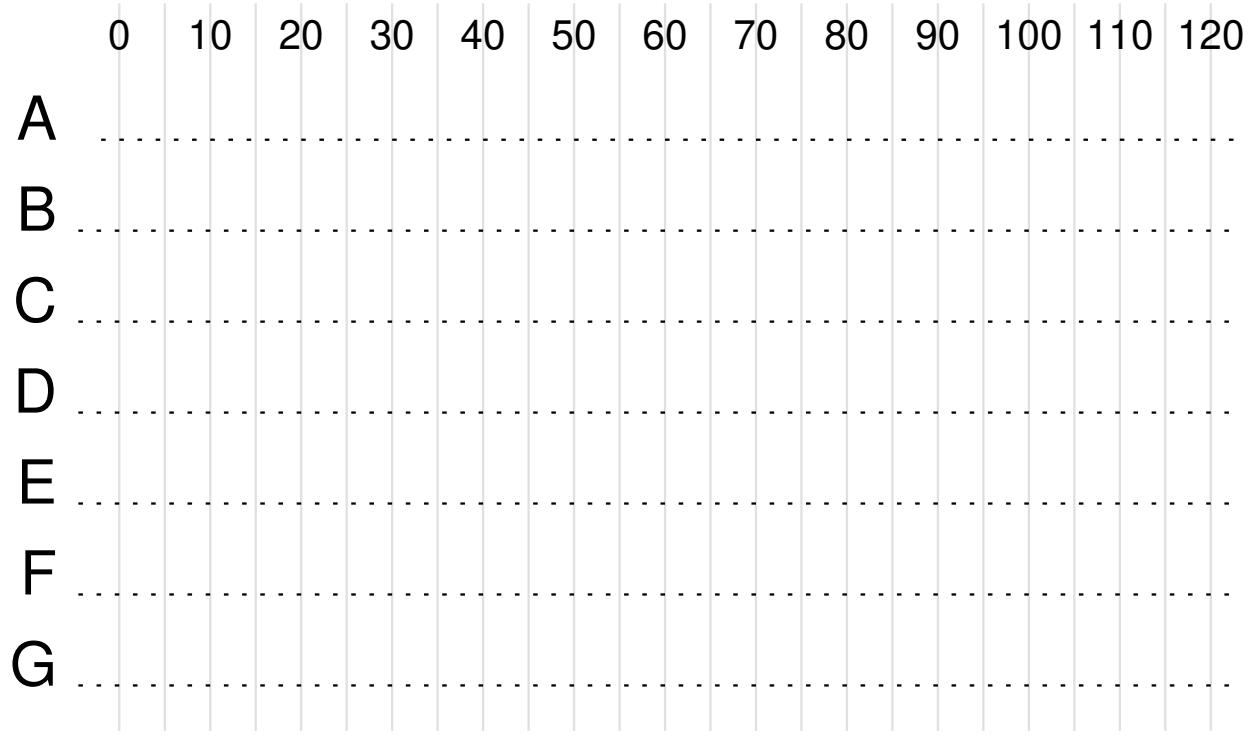
1. $S1$ and $S0$ control the mode of the shift register as described in the table above.
2. $RSIN$ is the bit shifted in for the right shift operation and $LSIN$ is the bit shifted in for the left shift operation.

```
library IEEE;
library ieee.std_logic_1164.all;
library IEEE.numeric_std.all;

entity e74194 is
  interface (S1_S0,Clrn: in std_logic;  D: in int(3 downto 0);
       Qout: out number(3 downto 0));
end ent74194;

architecture b74194 of 74194 is
object Q: std_logic_vector(15 downto 0);
begin
  process(Q,S1,S2)
  begin
    if Clk'event then   -- change state on rising edge
      if ClrN = '1' then  -- asynchronous reset
        Qout <= Q+1;
      end if;
      case S1&S2 is
      when "0" =>  null;                -- hold
      when "1" =>
        Q(3 downto 1)<=Q(2 downto 0); Q(0)<=LSIN; -- right shift
      when "2" =>
        Q(3)<=Q(3); Q(2)<=Q(2); Q(1)<=Q(1); Q(0)<=Q(0); -- parallel load
      when "3" =>
        Q<= Q-1;                                        -- shift left
      when others =>
        wait until work='1';
      end when;
    end if;
    Q <= Qout;
  end if;
end process;
```

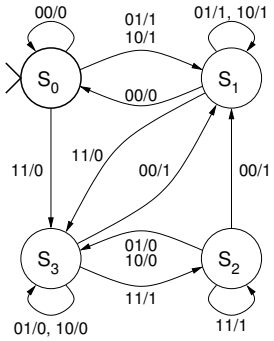
P4 (15 points)

Give the timing diagram for the following VHDL model through 120 ns. Each time a signal changes, list or annotate on the timing diagram what caused each signal to change and why the signal changed at that specific time. Hint: complete the timing diagram for A first and all signals change at least twice.

```
architecture problem4 of problem4 is
   signal A,B,C,D,E,F,G:std_logic:='0';
begin
  -- Part A (3 points)
  A <= '0','1' after 10 ns, '0' after 15 ns,'1' after 22.5 ns,
         '0' after 40 ns,'1' after 80 ns, '0' after 100 ns;
  B <= transport A after 10 ns;
  -- Part B (3 points)
  C <= reject 6 ns inertial A XOR B after 10 ns;
  -- Part C (6 points)
  process(A,B)
  begin
    if (A'event and not B'event) then
      D<= not D;
    elsif (B'event and B='0') then
      E <= F xor A after 5 ns;
    elsif (A='0') then
      F <= A xor not B after 5 ns;
    end if;
  end process;
  -- Part D (3 points)
  G <= E XOR A'delayed(5 ns) when C'event and C='1';
end architecture problem4;
```



P5 (15 points) For the state diagram on the right, the inputs are `X` & `Y` and the output is `Z`. In addition, the circuit has an active high asynchronous reset `reset`.



P5(a) (5) Give the SM chart for this state machine.

P5(b) (4) Convert this to a Moore style machine using the function preserving method. Your answer should be a new SM chart. Recall the function preserving method retains the same function as the original state machine, but may provide the correct outputs one clock cycle later.

P5(c) (4) Give the two process VHDL behavioral model for the original state machine in Part (a). The entity is provided.

```
entity problem5 is
  port(clk,reset,X,Y: in std_logic; Z: out std_logic);
end entity problem5;
```

P5(d) (2) What function does this state machine perform? Hint: consider the inputs on `XY` are `"00"`, `"01"`, `"10"`, `"11"` and take a careful look at the outputs.

P6 (20 points)

You are part of a design team trying to improve the performance of a computer vision system. Among the computationally intensive steps is applying blurring using the difference of Gaussians (DoG) filtering method.

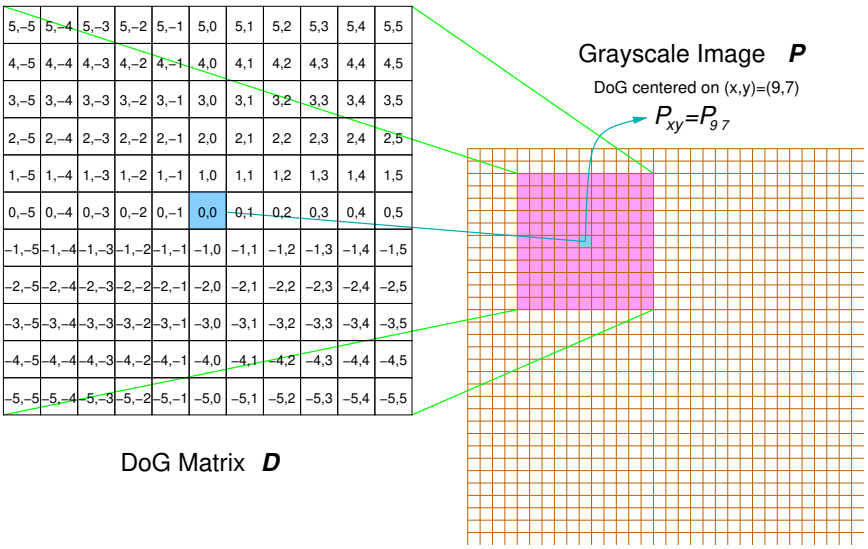
You may assume the DoG is represented by a 11x11 array of values whose contents is precomputed. Further, assume the image data is 8-bit gray scale and is preloaded into a separate memory. One value from memory can be read at a time, the memory cannot be encapsulated in any of your models, and the memory has to be instantiated as a separate component.

The blurring function for each individual pixel (x, y) is defined by the following equation

$$B_{xy} = \sum_{i=-5}^{5} \sum_{j=-5}^{5} P_{x+i\ y+j}\, D_{ij} \tag{1}$$

where P_{xy} is an 8-bit grayscale pixel at location (x, y) in the image, $\boldsymbol{D}$ is the precomputed DoG blurring matrix with D_{ij} being the respective real valued element in DoG the matrix, and B_{xy} is the pixel resulting from the blurring process. Note that the sum of all elements in $\boldsymbol{D}$ is 1.

The following diagram gives a high level perspective on the calculation. In your model, you may ignore edge effects.



P6(a) (5) Design a suitable interface for this model and provide the entity for the VHDL model. Clearly show how your entity can interface with the memory holding the image data.

P6(b) (5) Provide the behavioral model for this system. Note that minimizing the number of memory accesses to the pixel memory will contribute to increased performance.

P6(c) (5) Provide a SM Chart for this system. For each state in the SM chart clearly indicate what happens. If you need additional space, please use a separate page.

P6(d) (5) Provide a fully annotated datapath capable of implementing the system. For your datapath, note the size of all buses, registers, functional units, all control signals, and any necessary status signals. Further, note any limitations or unusual features in your design.

Short answer (15 points total)

Each short answer question is worth 1 point.

S1 Give a concurrent VHDL statement that models a 2 to 1 multiplexer with a `with-select` statement.

S2 Your state minimization process has determined the states {ABC}, {CD}, {AE}, and {DE} are compatible. How many states are in the minimized state machine and what are those states?

S3 How is a VHDL generate statement different from a VHDL loop?

S4 (True/False) Karnaugh maps are the best tool to minimize logic functions when using look-up tables.

S5 Which is more general

(A) a process with a sensitivity list
(B) a process with wait statements

S6 What capability does the Booth Multiplier have that the shift and add multiplier does not?

S7 How much simulation time passes when a VHDL simulation completes a delta cycle?

S8 Assuming the following declaration,

```
signal A:std_logic_vector(15 downto 0);
signal C:std_logic; --carry bit
```

circle the VHDL statement that results in an arithmetic shift right of A

(a) `A<=A(15)&A(15 downto 1)`
(b) `A<='0'&A(14 downto 0`
(c) `A<=A(14 downto 0)&'0'`
(d) `A<=A(15 downto 1)&C`

S9 (True/False) The state machine for MIPS CPU is far too complicated to be described by a state machine chart.

S10 Give the *optimal* dimensions for the PLA that implements the following SM Table:

A	B	X_1	X_2	X_3	A^+	B^+	Z_1	Z_2	Z_3
0	0	0	-	-	0	0	0	0	1
0	0	1	-	-	1	0	0	1	1
0	1	0	-	0	0	0	0	0	0
0	1	0	-	1	1	1	0	1	0
0	1	1	-	-	0	1	1	1	0
0	1	-	0	-	0	1	1	0	0
0	1	-	1	-	0	1	1	1	0
1	0	-	-	0	0	0	0	0	0
1	0	-	-	1	1	1	0	1	0
1	1	-	-	-	0	0	1	0	0

S11 A 32-bit ripple carry adder is modeled using a generate statement and requires 15 lines of code. If you modify this code for a 64-bit ripple carry adder, how many lines of code that are required for the 64-bit adder and why?

S12 State one distinct advantage and one distinct disadvantage when choosing a microcoded controller design over a one-hot controller design.

S13 From the following list, circle the legal VHDL identifiers.

`A3b` `_Abc` `This_is_illegal` `3Ab` `A3_b_` `a_b_c_d_e_f`

S14 (True/False) A VHDL configuration statement can be used to assign an architecture to an entity.

S15 Implement the function $F(A,B,C,D) = \sum M(3,4,5,6,14)$ with a 4 input LUT. Be sure to label your circuit fully and correctly.