First Name

Last Name

First letter of last name

UW Userid

ECE-327 Midterm

2017t1 (Winter)

Instructions and General Information

- 100 marks total
- Time limit: 1 hour and 20 minutes (80 minutes)
- No books, no notes, no computers. Calculators are allowed
- If you need extra paper, request some from a proctor.
- Write neatly.
- To earn part marks, you must show the formulas you use and all of your work.
- **The proctors and instructors will not answer questions, except in cases where an error on the exam is suspected. If you are confused about a question, write down your assumptions or interpretation.**
- **Justifications of answers will be marked according to correctness, clarity, and concision.**

			Total Marks	Approx. Time	Page
Q0	!!Almost Free!!		2	2	2
Q1	VHDL Simulation		25	15	3
Q2	The Flu, the Fever, and the Nausea		25	15	5
Q3	DFD		25	25	9
Q4	FSM		25	15	11
Totals			100	72	

Q0 (2 Marks) !!Almost Free!!

(*estimated time: 2 minutes*)

Q0a (1 Mark) Best part

What is the best part of the course?

Q0b (1 Mark) Most improve

What one thing could be done to most improve the course for the remainder of the term?

Q1 (25 Marks) VHDL Simulation

(*estimated time: 15 minutes*)

For the VHDL program below, calculate the values for the signals `c0`, `c1`, `c2`, and at 45ns.

NOTES:

1. All of the processes are in the *same architecture*.
2. The signals `a0`, `a1`, `a2`, `b0`, `b1`, `b2`, `c0`, `c1`, and `c2`, are declared to be `unsigned(15 downto 0)`.
3. Don't panic. Although the code is long, it is systematic:
   - The `a` signals drive the `b` signals
   - The `b` signals drive the `c` signals

Category of each signal			
timed	**comb**	**clk1**	**clk2**
`a0`		`a1`	`a2`
	`b0, b1, b2`		
	`c0`	`c1`	`c2`

4. **For full marks you must justify your answer, using text and/or the waveform diagram. You may, but are *not* required to, show a delta-cycle simulation.**

```
process begin
  clk1 <= '0';
  wait for 10 ns;
  clk1 <= '1';
  wait for 10 ns;
end process;

clk2 <= clk1;

process begin
  reset <= '1';
  wait for 11 ns;
  reset <= '0';
  wait;
end process;

process begin
  a0 <= ( others => '0' );
  wait for 10 ns;
  a0 <= a0 +    1;
  a0 <= a0 +    2;
  wait for 10 ns;
  a0 <= a0 +    4;
  a0 <= a0 +    8;
  wait for 10 ns;
  a0 <= a0 +   16;
  a0 <= a0 +   32;
  wait for 10 ns;
  a0 <= a0 +   64;
  a0 <= a0 +  128;
  wait for 10 ns;
end process;
```

```
process begin
  wait until rising_edge( clk1 );
  if reset = '1' then
    a1 <= ( others => '0' );
  else
    a1 <= a1 +  20;
  end if;
end process;

process begin
  wait until rising_edge( clk2 );
  if reset = '1' then
    a2 <= ( others => '0' );
  else
    a2 <= a2 + 200;
  end if;
end process;

b0 <= a0;
b1 <= a1;
b2 <= a2;

c0 <= b0 + b1 + b2;

process begin
  wait until rising_edge( clk1 );
  c1 <= b0 + b1 + b2;
end process;

process begin
  wait until rising_edge( clk2 );
  c2 <= b0 + b1 + b2;
end process;
```

Answer on the next page

	value at 45ns
c0	
c1	
c2	

Q2 (25 Marks) The Flu, the Fever, and the Nausea

(*estimated time: 15 minutes*)

For each of the code fragments Q2a–Q2e:

1. Answer whether the code is *legal*
2. If the code is *illegal*: explain why, and proceed to the next code fragment.
3. Answer whether the code is *synthesizable*.
4. If the code is *unsynthesizable*: explain why, and proceed to the next code fragment.
5. Answer whether the code adheres to good coding practices, according to the guidelines for ECE 327.
6. If the code does *not follow good coding practices*: explain why, and proceed to the next code fragment
7. Calculate the number of flip-flops that will be synthesized from the code.

NOTES:

1. The signal declarations are:

```
clk, a        : std_logic;
b, c, d, e, f : unsigned( 7 downto 0 );
```

Q2a

```
d <=   b + c when d(0) = '1'
  else b - c;
```

	Yes	No
Legal	☐	☐
Synthesizable	☐	☐
Good Practice	☐	☐
Number of flip-flops		

Explanation if illegal, unsynthesizable, or bad practice:

Answer:

Legal, synth, bad (comb loop through d(0)). If said "Legal, synth, good", then part marks for 0 flops.

Q2b

```
process begin
  c <= b;
  wait until rising_edge( clk );
end process;
```

	Yes	No
Legal	☐	☐
Synthesizable	☐	☐
Good Practice	☐	☐
Number of flip-flops		

Explanation if illegal, unsynthesizable, or bad practice:

Answer:

Legal, unsynth (statement before wait). If said "Legal, synth, good", then part marks for 8 flops.

The question continues on the next page

Q2c

```
process begin
  wait until rising_edge( clk );
  d <= b + 2;
  f <= e + 4;
end process;

e <= d;
```

	Yes	No
Legal	☐	☐
Synthesizable	☐	☐
Good Practice	☐	☐
Number of flip-flops		

Explanation if illegal, unsynthesizable, or bad practice:

Answer:

Legal, synth, good. 16 flops.

Q2d

```
process begin
  wait until rising_edge( clk );
  if a = '1' then
    c <= b + 2;
    d <= b + c;
  else
    d <= b - c;
  end if;
end process;
```

	Yes	No
Legal	☐	☐
Synthesizable	☐	☐
Good Practice	☐	☐
Number of flip-flops		

Explanation if illegal, unsynthesizable, or bad practice:

Answer:

Legal, synth, good. 16 flops

Q2e

```
process ( b, c ) begin
  d <= b + 2;
  d <= c + 5;
end process;
```

	Yes	No
Legal	☐	☐
Synthesizable	☐	☐
Good Practice	☐	☐
Number of flip-flops		

Explanation if illegal, unsynthesizable, or bad practice:

Answer:

Legal, synth, good. 0 flops

Marking:

3 marks *Legal, synth, good.*

2 marks *Number of flops or justification*

Q3 (25 Marks) DFD

(*estimated time: 25 minutes*)

In this question, you will design and analyze a dataflow diagram for the equation:

```
z = a + a*d + 5*c + b*c + d*e
```

NOTES:

1. Inputs shall be combinational and outputs shall be registered.
2. The delay of a multiplier is approximately twice that of an adder.
3. Optimization goals in order of decreasing importance:
   (a) minimize number of *multipliers*
   (b) minimize *clock period*
   (c) minimize *latency*
   (d) minimize number of *adders*
   (e) minimize number of *registers*
   (f) minimize number of *input ports*
   (g) minimize number of *output ports*
4. Input values may be read in any clock cycle, but each input value shall be read exactly once.
5. Algebraic optimizations are allowed, as long as the final value of `z` is correct.
6. You do *not* need to perform allocation.

Scratch work

The next page is for your answer

Analysis:

Latency:		Number of muls:	
Clock period:		Number of adds:	
		Number of regs:	
		Number of input ports:	
		Number of output ports:	

Q4 (25 Marks) FSM

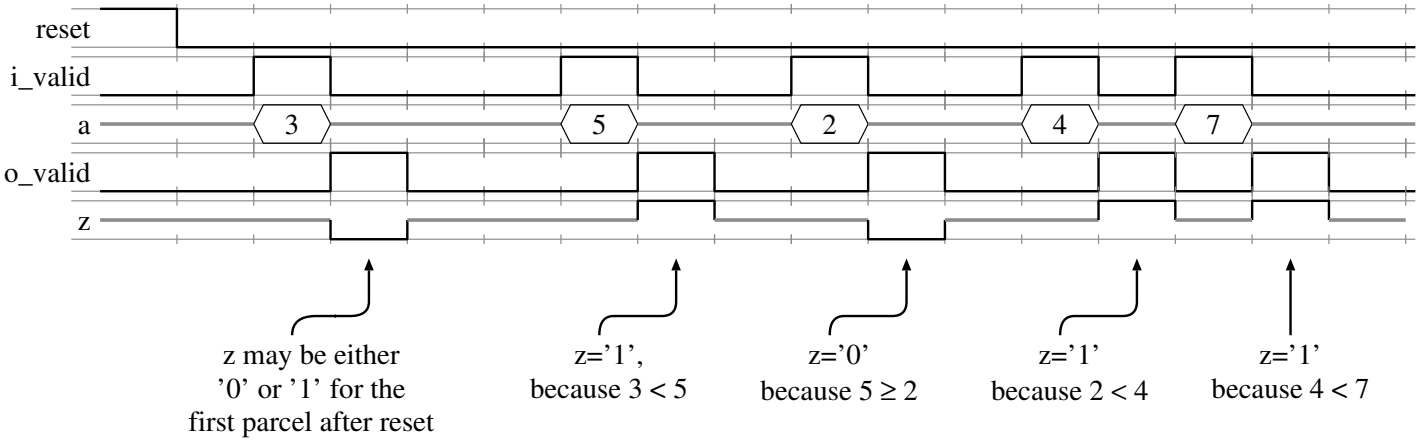
(*estimated time: 15 minutes*)

You've just been promoted to a manager and have been assigned the exciting project of designing a new state machine. Because you are manager, you delegate the design work to the three employees you supervise. Your task is to evaluate each design to determine whether it is correct.

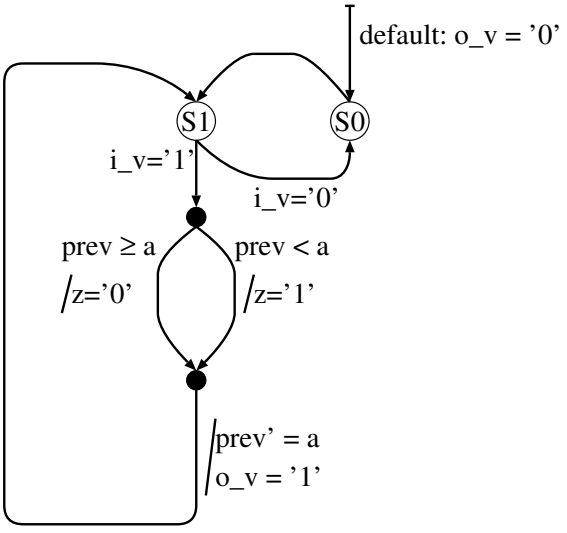
The state machine has one data input (`a`) and one data output (`z`). For each parcel, the value of `z` shall be `'1'` if the current parcel's `a` is greater than the previous parcel's `a`.

NOTES:

1. The system shall use a parcel schedule of unpredictable number of bubbles.
2. There is no requirement for the value of `z` for the first parcel after reset is deasserted. In other words, for the first parcel after reset, the value of `z` may be either `'0'` or `'1'`.
3. There is no requirement for the latency between `i_valid='1'` and `o_valid='1'`. In other words, the latency may be any number of clock cycles.
4. If the state machine is correct, answer what the latency is and what is the minimum number of bubbles between parcels.
5. If the state machine is incorrect, explain either how the machines behaviour differs from the system description or how the state machine could be modified to fix the incorrect behaviour.
6. In the diagrams below, "`i_valid`" has been shortened to "`i_v`" and "`o_valid`" has been shortened to "`o_v`".



Q4a



Yes No

Correct ☐ ☐

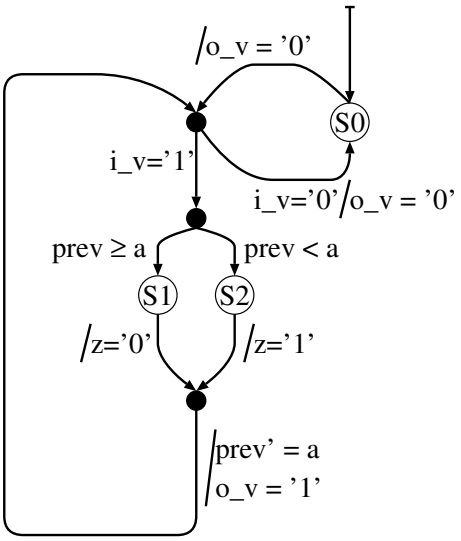
If correct:

latency = ☐

min bubbles = ☐

If incorrect, explanation:

Q4b



Yes No

Correct ☐ ☐

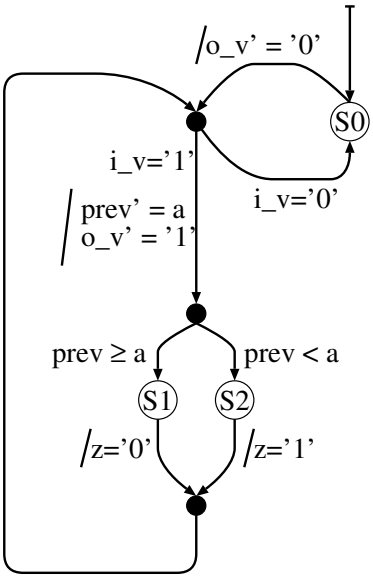
If correct:

latency = ☐

min bubbles = ☐

If incorrect, explanation:

Q4c



Yes No

Correct ☐ ☐

If correct:

latency = ☐

min bubbles = ☐

If incorrect, explanation:

Marking:

+1 mark *gave complete answers for all fsms*

+8 marks *each fsm*

- **+2 marks** *correct/incorrect*
- **+6 marks** *justification, if answered "incorrect"*
- **+3 marks** *latency, if answered "correct"*
- **+3 marks** *bubbles, if answered "correct"*