EEL 4712
Midterm 3 – Spring 2014
VERSION 1

Name: _____

UFID: _____

Sign your name here if you would like for your test to be returned in class:

_____

IMPORTANT:
• 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.**

COVER SHEET:

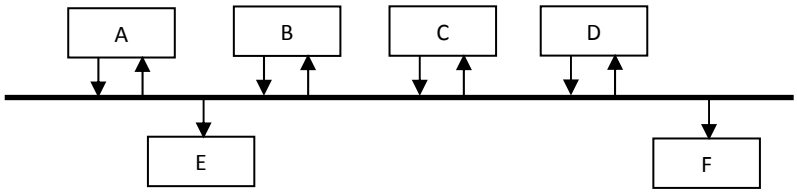
Problem#:	Points
1 (8 points)	
2 (4 points)	
3 (4 points)	
4 (4 points)	
5 (8 points)	
6 (4 points)	
7 (4 points)	
8 (4 points)	
9 (4 points)	
10 (4 points)	
11 (10 points)	
12 (4 points)	
13 (8 points)	
14 (30 points)	

Total:

Regrade Info:

```
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;
BEGIN
-- Process Statement
-- Concurrent Signal Assignment
-- Conditional Signal Assignment
-- Selected Signal Assignment
-- Component Instantiation Statement
END a;

__instance_name: __component_name 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;

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

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

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

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

1) (8 points) Draw a schematic of the FPGA circuit that would be synthesized for the following bus structure. Show how the A-F components connect to the inputs and outputs. You can omit control signals.

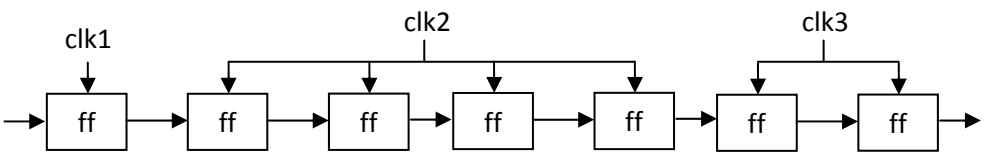


2) (4 points) Name the one FPGA resource that supports tri-states.

3) (4 points) Name two situations where the input to a flip-flop may change during the setup and hold window.

4) (4 points) Synchronizers avoid metastability by waiting some amount of time before storing potentially metastable data. How long does a dual-flop synchronizer wait?

5) (8 points) Identify all the locations in the following circuit that will become metastable:



6) (4 points) You are designing a pipelined circuit that requires high bandwidth from an external memory running at a different clock frequency. What synchronizer is most effective for this situation?

7) (4 points) What is the advantage of the handshake synchronizer compared to the mux-recirculation synchronizer?

8) (4 points) What error/warning will Quartus report when compiling this code? Note that there are no syntax errors.

```
architecture BHV of test is

  signal mux1_out, mux2_out : std_logic_vector(7 downto 0);

begin

  U_MUX1 : entity work.mux
    port map (in1    => in1,
              in2    => mux2_out,
              sel    => sel1,
              output => mux1_out);

  U_MUX2 : entity work.mux
    port map (in1    => in2,
              in2    => mux1_out,
              sel    => sel2,
              output => mux2_out);

  out1 <= mux1_out;
  out2 <= mux2_out;

end BHV;
```

9) (4 points) When subtracting two 8-bit numbers, what instruction must precede the SBCR (subtract with borrow) instruction to ensure that the result is correct?

10) (4 points) What is the *conceptual* difference between the carry flag and the overflow flag? Do not explain their implementation differences.

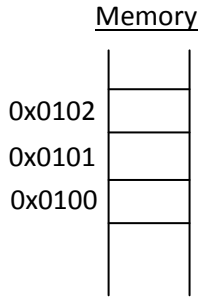
11) (10 points) Create a memory initialization file for the following assembly code. Add a comment to show the beginning of each instruction and each variable in memory. *You will likely need to break your answer up into two columns to fit on the page.*

```
OUTPORT0        EQU     $FFFE
TWO             EQU     $02

BEGIN:
        LDAI    TWO
        STAA    COUNT
AGAIN:
        LDAA    VALUE
        CLRC
        RORC
        STAA     VALUE
        LDAA    COUNT
        DECA
        STAA    COUNT
        BNEA    AGAIN
        LDAA     VALUE
        STAA    OUTPORT0

INFINITE_LOOP:
        CLRC
        BCCA    INFINITE_LOOP

* Data Area
VALUE:  dc.b    $FF
COUNT:  ds.b    1
```

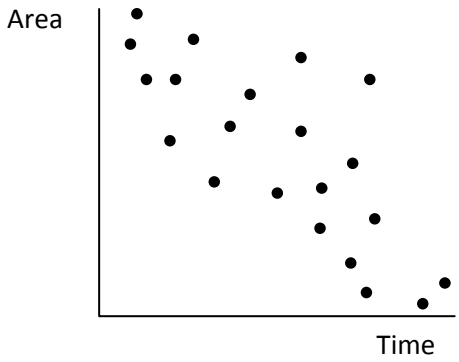
```
Depth = 256;
Width = 8;
Address_radix = hex;
Data_radix = hex;
% Program RAM Data %
Content
  Begin
```

```
[       ..00FF] : 00;
End;
```

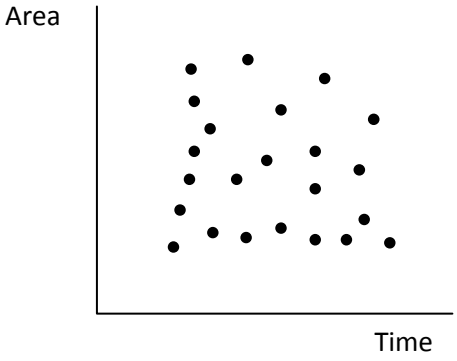
12) (4 points) Assuming that the stack pointer is set to address 0x0102, show the state of the stack immediately after a call instruction that occurs at address 0x5376. Make sure to show the new value of the stack pointer.



13) a. (4 points) For the set of implementations shown below, circle the implementations that are Pareto optimal. List any assumptions.



b. (4 points) Do the same for the following set of implementations.



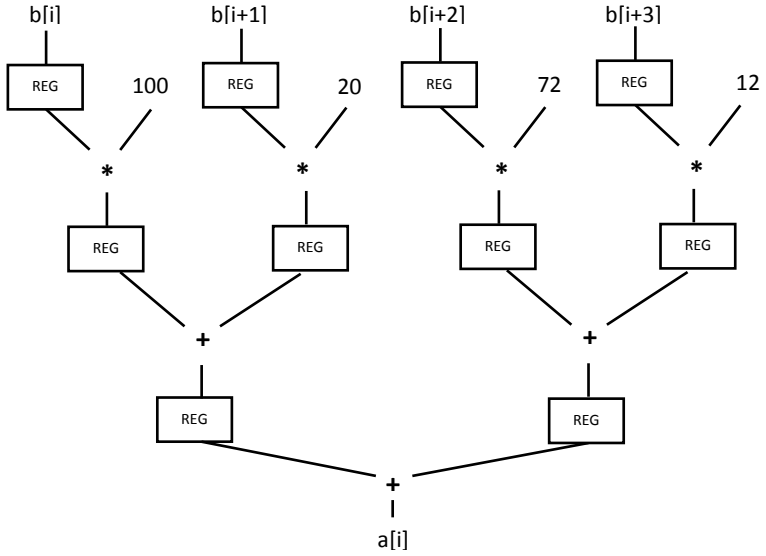
14) a. (8 points) For the following code, create a schedule for the provided datapath. Ignore muxes and other glue logic. Like the examples in class, assume that address calculations are done *without* using the specified resources (i.e., address calculations cost nothing). Do not change the code. List any assumptions.

```
for (int i=0; i < 5000000; i++) {
   a[i] = b[i]*100 + b[i+1]*20 + b[i+2]*72 + b[i+3]*12;
}
```

Datapath
2 multipliers
1 adder
1 comparator
1 memory for b[] (can read 2 elements/cycle)
1 memory for a[] (can write 1 element/cycle)

b. (4 points) What is the execution time in total cycles based on your schedule from part a? Show your work.

c. (8 points) Estimate the total cycles when using the following pipeline. Show your work:



d. (4 points) Assuming the entire design space consisted of the implementations in *a* and *c*, is *a* Pareto optimal? Explain your answer.

e. (4 points) Assuming the entire design space consisted of the implementations in *a* and *c*, is *c* Pareto optimal? Explain your answer.

f. (2 points) What is the name of the common loop optimization that enables additional wide/arithmetic parallelism?