Midterm Exam
ECE 448
Spring 2012
Thursday Section
(15 points)

Instructions:
Zip all your deliverables into an archive <last_name>.zip and submit it through Blackboard no later than Thursday, March 8, 10:15 PM EST.
Lab Midterm Exam

Introduction:

The circuit described below performs 4-bit binary sequential multiplication. The design takes two 4-bit operands A and B and produces an 8-bit product P as a result. A START signal is used to initiate the operation.

The circuit is specified below as follows:

- Interface
- Table of input/output ports
- Table of control/status signals between the datapath and controller
- Pseudocode
- Block Diagram
- Input/output and waveforms

Interface:
Assume the following interface to your circuit.

Table of input/output ports:

<table>
<thead>
<tr>
<th>Port</th>
<th>Mode</th>
<th>width</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>clock</td>
<td>IN</td>
<td>1</td>
<td>System clock</td>
</tr>
<tr>
<td>reset</td>
<td>IN</td>
<td>1</td>
<td>Asynchronous system reset</td>
</tr>
<tr>
<td>A</td>
<td>IN</td>
<td>4</td>
<td>4-bit Multiplier A</td>
</tr>
<tr>
<td>B</td>
<td>IN</td>
<td>4</td>
<td>4-bit Multiplicand B</td>
</tr>
<tr>
<td>START</td>
<td>IN</td>
<td>1</td>
<td>A signal to initiate multiplication operation</td>
</tr>
<tr>
<td>P</td>
<td>IN</td>
<td>8</td>
<td>8-bit Product P</td>
</tr>
</tbody>
</table>
### Table of control/status signals between datapath and controller:

<table>
<thead>
<tr>
<th>Port</th>
<th>Direction</th>
<th>width</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD_Ph</td>
<td>To_Dp</td>
<td>1</td>
<td>Loads Ph with the sum output of adder</td>
</tr>
<tr>
<td>LD_E</td>
<td>To_Dp</td>
<td>1</td>
<td>Loads E with the carry output of adder</td>
</tr>
<tr>
<td>LD_Pi</td>
<td>To_Dp</td>
<td>1</td>
<td>Loads Pi with the multiplier A</td>
</tr>
<tr>
<td>LD_B</td>
<td>To_Dp</td>
<td>1</td>
<td>Loads B with the multiplier B</td>
</tr>
<tr>
<td>Clr_Ph</td>
<td>To_Dp</td>
<td>1</td>
<td>Clears Register Ph</td>
</tr>
<tr>
<td>LD_Cnt</td>
<td>To_Dp</td>
<td>1</td>
<td>Loads initial value into counter</td>
</tr>
<tr>
<td>Dec_Cnt</td>
<td>To_Dp</td>
<td>1</td>
<td>To decrement the counter value</td>
</tr>
<tr>
<td>Shift_Rt</td>
<td>To_Dp</td>
<td>1</td>
<td>Shifts contents of E, Ph and Pl right by one bit</td>
</tr>
<tr>
<td>zero</td>
<td>To_Cn</td>
<td>1</td>
<td>Zero flag of the down counter</td>
</tr>
</tbody>
</table>

**Notation:**
- To_Dp – A control signal from controller to datapath,
- To_Cn – A status signal from datapath to controller,

**Pseudocode:**

```plaintext
//Step 1: Initialize
i = 0,
Ph ← 0, Pl ← A,
B-reg ← B, Cnt ← n-1, where n is the number of operand bits

//Step 2: Accumulation of partial products a_i * B into (E, Ph) one by one
(E, Ph) ← Ph + a_i * B = Ph + P_o B

//Step 3
Shift (E, Ph, Pl) Right by one bit
Cnt ← Cnt - 1
i = i+1

//Step 4
If Cnt = 0
    STOP
Else
    Loop back to step 2
```

**Note:** In step 3, registers (E, Ph, Pl) are shifted right. For Register E, a zero is shifted into it.

**Notation:**
- B-reg: 4-bit register to hold the multiplicand B
- Ph: 4-bit register, which is initialized to 0
- Pl: 4-bit register, which is initialized to A
- (Ph, Pl): 8-bit register, which holds the product P in the end
$\varepsilon$: 1-bit register, which stores the carry output of the adder. Initially it is cleared.

Cnt: 2-bit down counter, which controls the number of iterations to be performed

**Block diagram:**

![Block Diagram](image)

**Inputs, Outputs and Waveform:**

**Inputs:**

A = 1011  
B = 1101

**Output:**

$P(\varepsilon, \text{Ph}, \text{Pl}) = (0, 1000, 1111) = (\text{Hex value } = x"08F")$
ASM CHART (CONTROLLER):

reset

&box; Initialize

&box; 0

&box; start
0

&box; 1

&box; Clr_Ph
&box; LD_Pl
&box; LD_B
&box; LD_Cnt

&box; Loading

&box; LD_E
&box; LD_Ph

&box; Shifting

&box; Shift_Rt
&box; Dec_Cnt

&box; 0

&box; zero

&box; 1
Functional waveform:

Design Requirements:

The combinational portion of the circuit should be described using the dataflow VHDL code, and the sequential portion of the circuit should be described using the synthesizable behavioral code. Your code should infer a circuit that requires a minimum amount of FPGA resources. The target clock frequency should be 50 MHz.

Tasks:

Perform the following tasks:

1. Write a synthesizable VHDL code representing the datapath of the multiplication circuit (shown in the block diagram above).
2. Translate the ASM chart of the controller to VHDL code.
3. Develop RTL VHDL code for your entire circuit including the controller and top-level circuit.
4. Write a testbench for your entire circuit, and debug any possible errors in your RTL code.
5. Perform functional simulation of your circuit and use it to debug your VHDL code. Take a print out of the waveform showing the entire operation using default PDF conversion tool installed in the lab (Use multiple page option in order to display necessary information on multiple pages, if required).
6. Synthesize your circuit.
7. Implement your circuit using
a. FPGA family: Spartan 3E
b. Device: 3s100cp132
c. Speed Grade: -4

8. Run the static timing analysis of your circuit.
9. Based on the circuit block diagram and the report from the static timing analysis, determine the most critical path in your circuit and the circuit maximum clock frequency.
10. Based on the implementation reports, determine the number of CLB slices, Logic Cells, LUTs, D flip-flops and pins used by the circuit.
11. Perform the timing simulation of your circuit at the **maximum clock frequency** returned by the static timing analysis. Take a screen shot and include that in the report.

**Deliverables:**

1. VHDL code of your entire circuit (including datapath and controller) fulfilling the requirements specified in the *Design Requirements* section above.
2. VHDL code of the ASM Chart of the controller.
3. VHDL code of your testbench for the entire circuit.
4. Timing waveforms from the *functional* and *timing* simulations demonstrating the correct operation of your circuit.
5. Description of the critical path in your circuit
6. FPGA resource utilization (as defined in Task 10 above).
7. Minimum clock period and maximum clock frequency of your circuit.