Lab 3
Implementing Sequential Logic in VHDL: Square Root Unit based on CORDIC

Task 1 (40%)

Develop a VHDL description of the given design, which calculates the integer square root of an input value using a CORDIC approach. The equation for the output sqrt_x is given by:

\[ \text{sqrt}_x = \lfloor \sqrt{x} \rfloor \]

The interface for the square root circuit is shown below:

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>Input value of generic width N.</td>
</tr>
<tr>
<td>in_valid</td>
<td>Indicates a new input value is available to begin calculation. Held high for one clock cycle.</td>
</tr>
<tr>
<td>clk</td>
<td>Clock signal</td>
</tr>
<tr>
<td>sqrt_x</td>
<td>Output signal, valid only when out_valid is high.</td>
</tr>
<tr>
<td>out_valid</td>
<td>Held high for one clock cycle to indicate the calculation has completed.</td>
</tr>
</tbody>
</table>

N is a generic, specifying the bus width of the input X. The output width is half of that of the input width. The input in_valid is held high for one clock cycle to indicate that a new input value is available for calculation. Similarly, out_valid is held high for one clock cycle when a calculation completes.

Note that this operation takes multiple clock cycles to complete, and a new calculation cannot begin until the previous one completes.
The block diagram for the design is shown below.

The block diagram for the design is shown below.

- All clocked components are shown in a rectangular box. All combinational components have rounded edges.
- “$A \geq B$” has a value of ‘1’ if $A \geq B$, and ‘0’ otherwise.
- All values are unsigned.
- The load value for the shift register, denoted “1000...000,” has a ‘1’ in the MSB, and a ‘0’ in all other bits. It is loaded into the register when ld_en = ‘1’. When ld_en = ‘0’, the shift register shifts right one bit, shifting a ‘0’ into the MSB.
- The down counter loads the value $(N/2-1)$ when ld_en = ‘1’. When ld_en = ‘0’, the value of the counter decrements by 1.
- The shift register and counter loads and all register resets are synchronous operations.

Task 2 (30%)

Write a testbench to verify the correctness of your design for an input width of 8. Your testbench should contain a set of test vectors that sufficiently evaluate your design (using minimum 32, and possibly even all 256 input values).
**Task 3 (30%)**

Synthesize your code and perform implementation and static timing analysis using the Active-HDL design flow for the following values of your generic parameters:

\[ N = 8, 16, 32, 64, \text{and} 128. \]

For each of these parameter values, check the synthesis and implementation reports and record resource utilization, minimum clock period, and maximum clock frequency.

In addition to the functional simulation performed in Task 2, perform post-synthesis and timing simulations for the input width specified in Task 2.

Summarize your findings in the report specified in the list of Deliverables.

**Task 4 – Bonus (20%)**

Modify your design so that it contains a second generic, \( M \), which specifies the length of the calculated output. Assume that \( M \geq N/2 \). The block diagram for the modified design is shown below:

\[ L = \ceil(\log_2(2M)) \]
Task 5 – Bonus (10%)

Modify your testbench from Part 2 to test the operation of the circuit from Part 4. Note that your outputs will be of the form:

$$\sqrt[N]{2^M \cdot N/2} \cdot X$$

Test your design first for the case of N = 8 and M = 4, which is the same configuration as in Task 2. Then verify your design for N = 8 and M = 8.

**Deliverables**

- All source files used for synthesis and implementation of your circuit(s).
- All testbench files used for verifying your design(s).
- All synthesis and implementation report files.
- Simulation waveforms from the functional, post-synthesis, and timing simulations, proving the correct operation of your circuit for N=8 (and M=8 for Tasks 4 and 5)
- Reports from static timing analysis.
- Your own report containing the following major metrics for each input width:
  - Resource utilization
  - Minimum clock period and maximum clock frequency after synthesis and after implementation.
  - Graphs and short conclusions detailing the relationship between the bus widths and
    - Resource utilization
    - Minimum clock period/ maximum clock frequency
    - Latency (the time it takes to complete a calculation).

**Important Dates**

<table>
<thead>
<tr>
<th>Hands-on Session and Introduction to the Experiment</th>
<th>Tuesday Section</th>
<th>Wednesday Section</th>
<th>Thursday Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>02/07/2012</td>
<td>02/07/2012</td>
<td>02/08/2012</td>
<td>02/09/2012</td>
</tr>
<tr>
<td>Demonstration and Deliverables Due</td>
<td>02/14/2012</td>
<td>02/15/2012</td>
<td>02/16/2012</td>
</tr>
</tbody>
</table>