CMPE 140 – Laboratory Assignment 1

Dr. Donald Hung
Computer Engineering Department, San Jose State University

System-Level Design Review

Purpose

Review system-level design by designing, functionally verifying, and FPGA prototyping a digital system for computing the factorial of *n*. The system should start execution upon receiving an external input "*Go*" and should output a "*Done*" signal when the execution is completed.

Background

1) The algorithm for computing the factorial of *n*, i.e., *n*! = 1 x 2 x … x *n*, is as shown below:

```
1. INPUT n
2. product = 1
3. WHILE (n > 1) {
4.         product = product * n
5.         n = n -1
6.         }
7. OUTPUT product
```

Figure 1. The Factorial of *n* Algorithm

2) Functional building blocks to be used in building the datapath of the system is provided below. CNT is a down counter with parallel load control and an enable signal; REG is a data register with a load control signal; CMP is a comparator with a GT (greater-than) output; MUL is a combinational multiplier for unsigned integers; MUX is a 2-to-1 multiplexer; BUF is a tri-state buffer for output control, but is recommended to use another 2-to-1 MUX for this purpose.

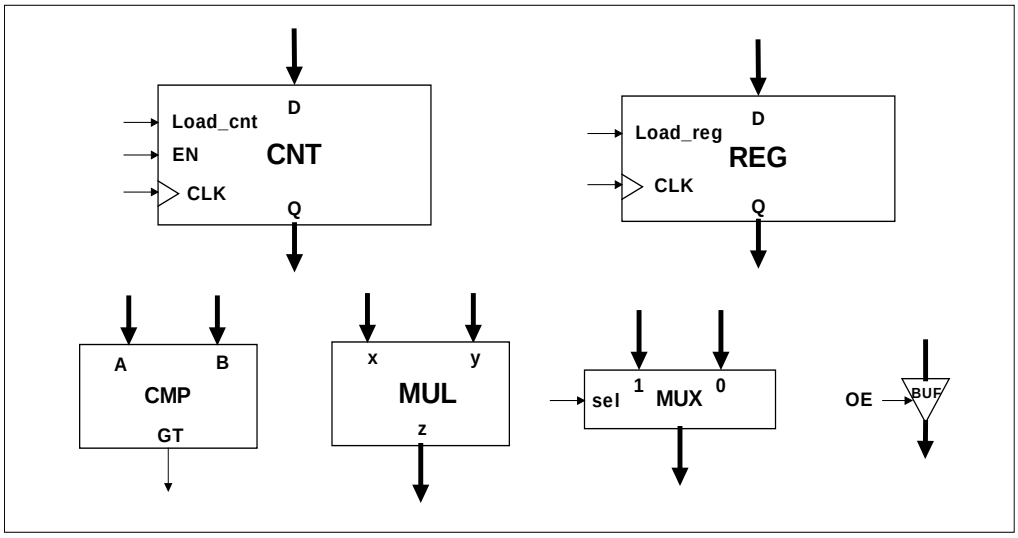

Figure 2. Functional Building Blocks of the Factorial Datapath Module

3) The factorial generator shall calculate and display results up to 10! (which is 3,628,800). Inputs for *n* shall be unsigned. Except the 2-to-1 MUX (a second one for replacing the BUF), only one of each building block shall be used. No other submodule shall exist within the datapath. Parameterize the data path elements to make it easier to resize the data path width in order to determine an optimal size to support large factorial results.
4) To display large decimal values a 32-bit binary to BCD converter module is supplied in the Appendix.

Tasks

1) Design the system's datapath using the given building blocks
2) Draw the two-piece (CU-DP) system block diagram that shows all relevant signals
3) Draw the ASM chart which describes the cycle-by-cycle operations of the datapath
4) Draw bubble diagram for the "next state logic" (NS) part of the control unit
5) Construct an output table for the "output logic" part of the control unit
6) Based on the above, write Verilog design code for the factorial generator, as well as testbench code to functionally verify your design
7) Implement your design on the FPGA board and test it using the Nexys4 DDR's on-board resources.
8) Write a report based on your work, and including the following components:
   (1) Cover page (use photo copy of the cover page signed by the lab TA or the course instructor)
   (2) Purpose of the assignment
   (3) A list of tasks that you have successfully accomplished
   (4) A list of tasks that you cannot or did not accomplish (if there is any), with your explanations and/or analysis.
   (5) Appendices including the following (**Do not** cut and paste from this assignment):
      - diagrams and table mentioned in Tasks 1 – 5, generated by Visio or other tools
      - source code for both design and verification
      - waveforms captured from your simulation results, and photos taken from FPGA validation results

Lab Checkup Schedule

Week	Due to be Checked by TA
Week #1 (9/5 & 9/7)	1. All diagrams/table completed (Tasks 1-5). 2. Verilog code drafted
Week #2 (9/12 & 9/14)	Demonstrate results of 1) functional verification (Task 6) 2) FGPA validation (Task 7)

Appendix

```
module bin2bcd32(
    input wire [31:0] value,
    output wire [3:0] dig0,
    output wire [3:0] dig1,
    output wire [3:0] dig2,
    output wire [3:0] dig3,
    output wire [3:0] dig4,
    output wire [3:0] dig5,
    output wire [3:0] dig6,
    output wire [3:0] dig7
    );

    assign dig0 = value % 10;
    assign dig1 = (value / 10) % 10;
    assign dig2 = (value / 100) % 10;
    assign dig3 = (value / 1000) % 10;
    assign dig4 = (value / 10000) % 10;
    assign dig5 = (value / 100000) % 10;
    assign dig6 = (value / 1000000) % 10;
    assign dig7 = (value / 10000000) % 10;

endmodule
```