CMPE 125 Lab 8 Assignments

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

System-level Design (2)

Purpose:

1) To enhance the ability in system-level design, verification, and FPGA validation
2) To familiarize with the procedure and tools for system-level development

Preparation:

Review lecture notes related to this assignment

Description

In this assignment, you will design a 4-bit unsigned integer divider, using the systematic design methodology taught in class. The top-level system configuration is illustrated by Figure 1 shown below, where the data path (DP) consists of properly interconnected functional units for data processing, while the control unit (CU), which is an FSM, generates control signals to steer the DP's cycle-by-cycle operation.

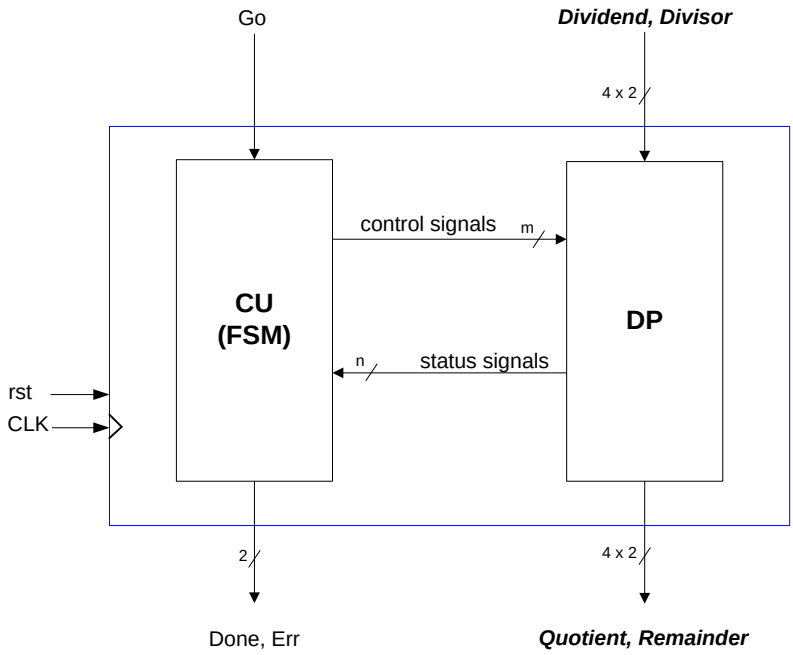

Figure 1: The system's CU – DP model.

In Fig. 1, when the reset signal *rst* is active, all clock triggered components in CU and DP are cleared. As the result, the CU goes to its Idle state, and the shift registers and counter in the DP are reset to zero. The signal *Done* is asserted when the division is completed. The *Err* signal is to flag the "divide- by-zero" error.

Your design will be based on the restoring algorithm for unsigned integer division shown below:

```
//Step 1: load the initial values into the registers
R[4:0] ← 0;
X[3:0] ← dividend;
Y[3:0] ← divisor;

//Step 2: shift the concatenation of {remainder, dividend} by 1 bit
to the left
R[4:0] ← {R[3:0], X[3]}; X[3:0] ← {X[2:0], 0};

//Step 3: repeat step 3 for 4 times
for (i = 3; i >= 0; i = i - 1)
begin
    if (R[3:0] < Y[3:0])  //compare
        begin
        //restore and set quotient bit to 0
          R[4:0] ← {R[3:0], X[3]}; X[3:0] ← {X[2:0], 0};
        end
    else
        begin
        //subtract and set the quotient bit to 1
          R[3:0] ← R[3:0] - Y[3:0];
          R[4:0] ← {R[3:0], X[3]}; X[3:0] ← {X[2:0], 1};
        end
end

//Step 4: right shift the remainder
R[4:0] ← {0, R[4:1]}; //now remainder is in R[3:0],
                      //and quotient is in X[3:0].
```

Tasks:

The lab is split into four tasks across three weeks. At the end of each week, have the instructor check your work. You will be assigned a grade for the task(s) that are due at that time.

Tasks for first week:

1. Design and functionally verify the datapath (DP).

   You need to design the microarchitecture (i.e., functional blocks and their interconnections) of the DP, write register-transfer-level (RTL) Verilog code to describe your design, and then write a Verilog testbench to functionally verify the DP you designed.

   Use a **self-checking testbench** for functional verification of the data path.

   You also must complete the following diagrams and show them BEFORE you demo:
   - Data path block diagram
   - ASM chart
   - State transition diagram (bubble diagram)
   - Output table

Tasks for the second week:

2. Design and functionally verify the control unit (CU).
   The FSM must be tested thoroughly via functional verification

   Use a **self-checking testbench** for functional verification of the control unit.

Tasks for third week:

3. Integrate and functionally verify the entire system.
   You should write a top-level Verilog design code for the divider by connecting the FSM with the DP, and write a Verilog testbench to verify the overall system's functional correctness.

   Use a **self-checking testbench** for functional verification of the integrated system.

4. Validate the designed integer divider using the Nexys4 DDR FPGA board. Note that since the results (quotient and remainder) are in binary, in order to display their decimal values on the 7-segment LEDs, you need to write Verilog code for a *binary to 7-segment decoding* module, attach it to the DP's output, and integrate it to the overall system before launching the FPGA implementation tool.

   Use a debounced pushbutton for the clock so that you can step through states with each button press. Do not use the four second clock or 100 MHz system clock as the clock source.

Contents of Report:

Follow the standard report format as usual. Some highlights include:

1) Cover page.
2) A list of successfully accomplished tasks.
3) For Tasks 1 – 3, include the following:
   a) Test plan for functional verification,
   b) Commented source code (design and verification),
   c) Captured verification results (TCL console output).
4) For Task 1 (DP), include schematics of the DP's microarchitecture.
5) For Task 2 (CU), include the ASM chart describing the system's cycle-by-cycle operation, as well as the *state transition diagram* and the *output table* extracted from the ASM chart.
6) For Task 3 (integration), include a diagram for the system's CU-DP model (refer to Fig. 1), with all signals explicitly shown.
7) For Task 4 (validation), include description and additional source code (Verilog and constraints) for hardware prototyping setup, and your test plan for hardware validation.
8) A conclusion section that summarizes your work and discusses problems encountered and lessons learned, as well as detailed descriptions of the task(s) that you cannot, or did not accomplish, including reason(s) and/or your analysis.