Department of Electrical and Computer Engineering
University of California, Davis

Lab 4: A Flip-Flop-Based Shift Register

I. Prelab

Complete the following and submit your work at the beginning of your lab session.

1. Read: *Basic Diagrams* posted on the course web page
2. Read: *Tips for building and debugging your circuits* posted on the course web page
3. Read: *Notes on active high/low inputs/outputs for the DE-10 Lite* posted on the course web page
4. [15 pts] Complete preliminary designs of your Pipelined Block Diagram and hardware verilog (Section III), and verilog testbench (Section IV)

Details not specified in this Lab should be chosen by you and plainly explained in your lab report.

II. Circuit Description

A circuit contains ten positive-edge-triggered D flip-flops (FFs) wired in a chain with one *LEDR* LED connected to the output of each FF, and also some additional circuits to implement the following functions:

- Each time the system is clocked one cycle, the pattern of lit LEDs moves one position to the right when the DE10-Lite board is viewed with the SW switches on the bottom
- The *KEY0* button makes the *clock* signal go high when pressed and makes it go low when released. (Although this violates one of our two clock-related rules, we will do it anyway for this lab only because building this functionality correctly with a true multi-MHz clock is significantly more complex. Tell your TA if you observe strange behavior such as an LED occasionally skipping over one LEDR position.)
- The *KEY1* button initializes the values in the ten FFs if held pressed during an active *clock* edge. The initialized values depends on the position of several SW switches; how you do this is up to you. Consider what happens for all positions that the SW switches may be in and describe in your report.
    a) 00_0000_0000 all LEDs off
    b) 11_1111_1111 all LEDs on
    c) 10_1010_1010 alternating pattern (LEDR9, …, LEDR0)
- In most cases, the input to the leftmost FF injects zeros into the shift register chain. Only in the case when all LEDs are off, the circuit will inject a one into the shift register chain. If built properly, the chain will circulate a single lit LED after ten clock cycles, no matter what the beginning pattern happened to be.

III. Circuit Design

Design the described circuit and draw a Pipelined Block Diagram with simple logic shown (such as AND, OR, NOT, MUX, etc.), and combinational-logic-clouds (with all inputs and outputs properly connected) for more complex logic.

Implement your design in verilog.

IV. ModelSim Testbench

Design and implement in verilog a testbench that uses the Approach #1 style shown on slide 114 in the *Verilog 5: Testing* handout (one large `initial` block).

In a single ModelSim simulation, test all three reset patterns with at least 15 cycles for each, and at least 25 cycles for one case. Generate waveforms or text output that show operation clearly enough to verify correct operation.

V. Demonstration on the DE10-Lite board

After your design is working correctly in simulation, download it onto the DE10-Lite board and verify it works there.

VI. Submitted Work [100 pts total]

With the exception of any instructor-provided code, all work must be yours alone.

[15 pts] **Prelab**

[60 pts] **Lab Checkoff**

- [25 pts] Demonstrate your ModelSim simulation to your TA.
- [35 pts] Demonstrate your design compiling and operating on the DE10-Lite board, to your TA.
- Your TA will ask you to generate and will record a `certutil` hash of your top-level files. This hash must match the hash of the files you upload to canvas so no file modifications are possible after your demo. The hash is calculated using the following commands:
    1. in Quartus, double click a few key modules determined by your TA, into the Project Navigator Hierarchy pane to open its Verilog
    2. right click the file's tab and select Copy Full Path
    3. open a Command Prompt window (by typing "cmd" into the Windows search bar)
    4. type "`certutil -hashfile` *`filepath`*" where *`filepath`* is the file path that was copied earlier

[25 pts] **Lab Report**

Upload the following two files to Canvas by the end of your lab session—this is essential to receive any credit for the entire lab.

I. A single Zip file. Submit all Verilog hardware and testbench code that you wrote in a single zip file. Do not include any code that you did not write such as files generated by Quartus or IP components.

- Make a folder on your computer
- Copy all files to be submitted into the folder
- "zip" the folder into a single .zip file
- Log onto Canvas, click Assignments, find the correct lab number
- Upload the .zip file

II. A single PDF or Word file.

1. Any required diagrams
2. Required hardware verilog
3. Required testbench verilog
4. Text and/or waveform printouts