University of California, Davis

Department of Electrical and Computer Engineering

Lab 6: Coin Counter

This lab involves the design and implementation of circuits that operate on the DE10-Lite FPGA board. All circuits are synchronously clocked by only the 50 MHz standard clock. Details not specified in this lab should be chosen by you and stated in your lab report.

I. Prelab

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

1. [5 pts] Perform and document a preliminary design for bcd2driver.v including a block diagram and preliminary verilog including items such as wire and reg declarations and basic code blocks.
2. Read Handout 19: Interfacing with Input Signals.

II. bcd2driver.v

Design a fully-combinational module that inputs a 7-bit binary number and outputs the corresponding value onto two HEX displays in base 10 format. If the input value is greater than 99 base 10, display "– –" on the HEX displays and assert output *gt99* high.

The module has the following I/O ports:

- *in*[6:0] - input, unsigned binary
- *out0*[6:0] HEX output, drives "ones" base 10 digit to 7-segment display
- *out1*[6:0] HEX output, drives "tens" base 10 digit to 7-segment display
- *gt99* - output, is high when the input value is greater than 99

III. Main controller

The main controller adds the value of coins as they are entered into a coin counter. It uses a main *state* register and a 7-bit *balance* register.

The controller is told when coins are entered by one and only one of the SW3, SW2, SW1, SW0 switches being high, and KEY1 being pressed (for an arbitrary amount of time). The corresponding coin value entered with the four SW switches is shown immediately on HEX1 (tens) and HEX0 (ones) using an instantiation of bcd2driver.v . The display should show `00, 01, 05, 10,` or `25` if zero or one switch is high, respectively. If more than one switch is high, the display should show "– –".

If more than one SW3–SW0 switch is high when KEY1 is pressed, a hardware malfunction is indicated and the machine should lock up in an Error state, and assert the output *error* and light LEDR4–0 until *reset* is pressed.

Each time a coin is entered, the value is accumulated into the *balance* register. The value of the *balance* register is displayed on HEX5 (tens) and HEX4 (ones) using an instantiation of bcd2driver.v . If the value of balance ever goes above 99 cents, the machine should lock up in a Full state, and assert the output *full* and light LEDR9−5 until *reset* is pressed.

Register (with FFs) all inputs as soon as they enter and immediately before they leave the top level module.

The module has the following I/O ports:

- *reset* KEY0 input, sets all state to initial conditions, balance to zero
- *enter* KEY1 input, enters the value of the coin on SW3−SW0
- *penny* SW0 input, indicates a 1 cent coin has been inserted when high
- *nickel* SW1 input, indicates a 5 cent coin has been inserted when high
- *dime* SW2 input, indicates a 10 cent coin has been inserted when high
- *quarter* SW3 input, indicates a 25 cent coin has been inserted when high
- *error* LEDR4−0 output, invalid coin input
- *full* LEDR9−5 output, balance greater than 99 cents
- *balance* HEX5−4 output, HEX displays
- *coin* HEX1−0 output, HEX displays

V. Testing in Simulation

[10 points] Design and write a testbench for your bcd2driver.v module that exercises all functions.

[20 points] Design and write a testbench for your top-level module that exercises all functions.

Once your test benches are working correctly, demonstrate them to your TA and have them checked off.

Suggestion: consider implementing multiple independent tests that are shorter in length and focus on specific features.

VI. Implementation and Verification on the DE10-Lite

Download your design onto the DE10-Lite board and verify it works correctly.

[55 points] Demonstrate it compiling, downloading, and operating to your TA and have it checked off. Your TA will record a `certutil` hash of your top-level files and this must match the hash of the files you upload to canvas so no file modifications are possible after your demo.

Submitted Work [100 pts total]

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

[5 pts] **Prelab**

[85 pts] **Lab Checkoffs: Simulation and on FPGA board**

[10 pts] **Lab Report**

Submit all Verilog hardware and testbench code that you wrote. Do not include any code that you did not write such as files generated by Quartus or IP components.

a) [10 points] Print and submit a paper copy during your lab session. In addition to your code, include:

   a. A block/circuit diagram of bcd2driver.v

   b. A block/circuit diagram of your top-level module

   c. A timing diagram of example operation of your design including KEY1 operation details

b) Upload a copy to Canvas by performing the following steps by the end of your lab session—this is essential to receive credit for the entire lab.

   1. Make a folder on your computer
   2. Copy all verilog files you wrote into the folder—only the ones you wrote
   3. "zip" the folder into a single .zip file
   4. Log onto Canvas, click Assignments, find the correct lab number
   5. Upload the .zip file