Lab Four: Shift registers, Counters

Ben Smith

***Abstract*—This document introduces concepts related to the implementation of Registers and Timers in the System Verilog hardware descriptive language.**

I. Introduction

THIS lab will reinforce the idea of modularity as we design modules which depend on a clock. Clocking circuits allows the synchronization of large blocks of logic. The concepts this lab introduces are at the core of design with Verilog. The sensitivity list will allow you to communicate with a number of different devices on different system clocks greatly expanding the usefulness of the language. It is important to remember that each one of these blocks evaluates simultaneously, he behavioral block is at the core of Verilog's parallel nature.

A. Clocked designs in Verilog

Procedural blocks are one of the core constructs of Verilog.

1) Posedge and Negedge in the sensitivity list:

B. Blocking and non blocking operators

Verilog statements can be structured by using two different assignments. These two types of statements must be used to avoid race conditions when your code is synthesized. Once again Clifford Cummings offers a great insight into the operation of these language constructs.

1) Blocking assignments (=): Blocking assignments occur in one step, all other operations are "paused" while waiting for the operation to complete.Evaluate the RHS (right-hand side equation) and update the LHS (left-hand side expression) of the blocking assignment without interruption from any other Verilog statement." [**?**].

2) Blocking assignments (<=): The evaluation of nonblocking assignments is a bit more complex than the blocking operator and requires special consideration. The execution of these statements is defined as follows

1) Evaluate the RHS of nonblocking statements at the beginning of the time step.
2) Update the LHS of nonblocking statements at the end of the time step.

[**?**].

C. Blocking and non blocking operators

D. Clock division with Counters

The idea of a clock divider in verilog is identical to a divider with discrete logic. Because of the constraints on FPGA design a synchronus counter is the best option for our purpose. If one d flip-flop divides a clock by two, then the addition of another will divide the input clock by four and so on. this can be generalized to

$$f_{out} = \frac{f_{in}}{\text{number of bits in register}}$$

II. Lab Procedure

A. Application of registers: Timers

The DE0-Nano has a 50MHz crystal oscillator and this is input to the FPGA clock pin. This clock must used to trigger operations every nanosecond and every 2 milliseconds. Measure the frequency of this trigger signal using a debugging tool like Signaltap or output the signal to pins and measure the output with an oscilloscope.

Laboratory Demo

Physical deliverable:	Constructed circuit on breadboard, the Nano programmed with your AOI module.
Documentation deliverable:	Labeled schematic of circuit, truth table from your experimental result.
Process :	The student will be asked the result of a few random numbers to verify the correct operation of their circuit.

B. Application of Shift Registers: IO expansion

This sections focuses on a application of the shift register. It can be found in a number of uses like the serialization and de-serialization of data streams. A common trick to expand IO on a device is to use the venerable HC595 shift register. The data for the output pins can be output serially to the shift register instead of directly parallel. This trades update speed and code complexity for IO pin usage. Let's take some indicator LED's for example. They do not need to be updated frequently and we can save the complex IO pins on the FPGA for other purposes.

Laboratory Demo

Physical deliverable:	Constructed circuit on breadboard, the Nano programmed with your AOI module.
Documentation deliverable:	Labeled schematic of circuit, truth table from your experimental result.
Process :	The student will be asked the result of a few random numbers to verify the correct operation of their circuit.

III. Lab Report

IV. Conclusion