CST456 – Lab 4

Constrained Pseudo-random Testbench

Objective

The objective of this lab is to implement a constrained pseudo-random testbench to test the operations and operands of a synchronous arithmetic and logic unit (ALU). The testbench will contain pseudo-random stimulus generator (stimulus), a monitor, a scoreboard, and a checker.

Design Under Verification (DUV)

The ALU contains the following ports:

Port Name	Signal Type	Direction	Number of Bits
clk	Clock	Input	1
op_start	Control	Input	1
operation	Control	Input	2
operand_a	Data	Input	8
operand_b	Data	Input	8
result	Data	Output	8

At the rising edge of the clock, if op_start is asserted (== 1’b1) then the operation, operand_a, and operand_b signals are latched into the ALU. Exactly two clock cycles later the upper eight bits of the result are read from the result port. The lower eight bits of the result are read from the result port one clock cycle later. The op_start signal is ignored during the middle of an operation. The ALU supports the following operations:

Name	Value	Description
ADD	2’b00	operand_a is added to operand_b. Only the lower nine bits of the 16-bit result are used.
MULT	2’b01	operand_a is multiplied with operand_b. This is an unsigned operation and all sixteen bits of the result are used.
OR	2’b10	operand_a is bitwise ORed with operand_b. Only the lower eight bits of the 16-bit result are used.
AND	2’b11	operand_a is bitwise ANDed with operand_b. Only the lower eight bits of the 16-bit result are used.

Figure 1 shows a timing diagram for the addition operation, where 255 + 255 = 510.

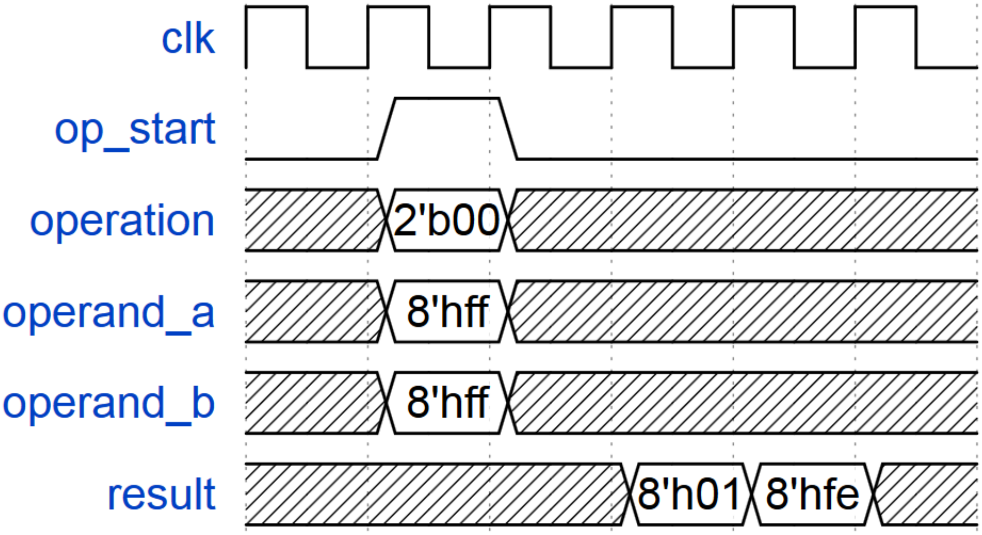

Figure 1 - Timing diagram for an addition operation.

Stimulus

The stimulus block generates constrained pseudo-random stimuli for the DUV. Signals and scenarios that can be randomized include:

- The operation and operand signals to generate a valid operation.

Scoreboard

The scoreboard is a location for storing temporary test data. The scoreboard should contain fields for the current operation, operands, and result.

Monitor

The monitor block passively observes the DUV signals and updates the scoreboard based on the stimuli driven into the DUV and the responses from the DUV.

Checker

The checker block checks the scoreboard once an operation has completed. Note that **checker** is a SystemVerilog keyword, so don’t use it as a task name.

Testbench

Edit the tb.sv file to create the testbench with the stimulus, scoreboard, monitor, and checker structures. Use the `RND_CHECK macro to check that randomization was successful. Use the `FAIL_UNLESS_EQUAL and `FAIL macros to check the expected result with the tested result.

To run the simulation in Xilinx Vivado for Windows execute following commands in cmd window under SIM directory:

```
call C:\Xilinx\Vivado\2023.2\bin\xvlog --sv ../SRC/duv.sv ../SRC/tb.sv

call C:\Xilinx\Vivado\2023.2\bin\xelab -debug typical -top tb -snapshot duv_tb_snapshot

call C:\Xilinx\Vivado\2023.2\bin\xsim duv_tb_snapshot -R

call C:\Xilinx\Vivado\2023.2\bin\xsim  duv_tb_snapshot --tclbatch xsim_cfg.tcl

call C:\Xilinx\Vivado\2023.2\bin\xsim --gui  duv_tb_snapshot.wdb
```

Lab Submission

Zip the entire contents of the lab directory and submit it to Canvas.