Simulation

Design Verification

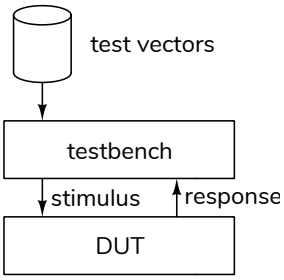
It's necessary to ensure a design will meet its requirements before it's put into use, This requires verifying that it will produce the correct outputs at the correct times.

Often it's more practical to verify a design by simulating it than by building it.

In this lab you will simulate the operation of a median filter circuit that outputs the median of the three most recent inputs. Median filters are useful for removing short noise impulses in signals.

Testbenches

A "testbench" is HDL code that applies inputs to the design being tested (often called the "device under test" or DUT) and checks that the outputs are correct:



The values of the inputs and expected outputs are called "test vectors." Test vectors are often read from a file generated by other software. In this lab you will be supplied with text files containing the test vectors.

In a previous course you may have use a stimulus-only testbench that applied inputs to the DUT and displayed the input and output waveforms. These simulations are useful during the initial design process but are not practical for complex designs or if we want to automate testing to ensure new errors ("regressions") are not introduced. Once the expected output has been determined, a "self-checking" testbench is typically used to check the outputs and flag any differences.

Writing a testbench requires HDL language features and programming skills that are beyond the scope of this course so for this lab you will be supplied with a self-checking testbench and asked to use it to test your design.

Procedure

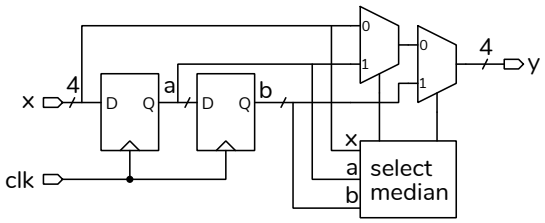
You will use the version of the Modelsim simulator supplied by Intel along with their PLD synthesis software ("ModelSim-Intel® FPGA Starter Edition Software"). See the document and videos on the course web site for instructions on installing and using it.

Create a project directory (folder) for the simulation. Download the **`lab7_tb.sv`** testbench and the **`lab7testdata.zip`** file containing the test vector files from the course web site to this folder.

Design the Median Filter

A median filter is a circuit that outputs the median of previous values. For example the median of the values 5, 3 and 8 would be 5. In this lab you will design and test a circuit that outputs the median of the current input and the two previous inputs. The inputs are 4-bit unsigned integers.

The following diagram shows a possible block diagram of a median filter:



Write a module named **`lab7`** that has the following declaration:

```
// lab7.sv
// median filter
// your name & date here

module lab7
  ( input logic clk,
    input logic [3:0] x,
    output logic [3:0] y ) ;

   // your code here

endmodule ;
```

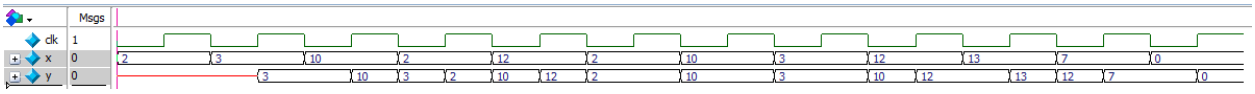


Figure 1: Sample Wave display.

and save it to a file named **lab7.sv**.

Follow the course coding guidelines, including adding a comment at the beginning of the file with your name and the date.

As a hint, the following logical expression is true if **b** is the median value of **a**, **b** and **x**:

```
a <= b && b <= x || a >= b && b >= x
```

you can write a similar expression to test if **a** is the median value (otherwise **x** is the median).

Select the Correct Test Vector File

For this circuit the test vectors are pairs of values of the input, **x**, and the expected output, **y**. The **y** value is the median of **x** and the two previous values of **x** (labelled **a** and **b** in the diagram above). The testbench applies the **x** value and checks the **y** value before generating a rising clock edge.

The supplied testbench reads the test vectors from a text file, **testdata*n*.txt**. Download the **lab7testdata.zip** file from the course web site and extract the file where *n* is the last digit of *your* BCIT ID to your project directory. Change the last digit in the file name in **lab7_tb.sv** to match the last digit of your BCIT ID.

Run the Simulation

Follow the procedure in Software Installation and Use document and the video on the course web site to create a simulation project, add your **lab7.sv** and the edited **lab7_tb.sv** files to the project and compile them. After fixing any errors, run the simulation. The Transcript window should show the messages generated by the testbench and the Wave window should show the signal waveforms.

An example of the waveforms is shown in Figure 1. To show the values in decimal, right-click on a trace and set Radix to Ufixed.

Submit Results

Submit a PDF file to the appropriate Assignment folder that includes the following:

- a listing of your **lab7.sv** file
- a screen capture of the waveforms similar to that shown above
- a screen capture of the Transcript window showing the messages generated by running the simulation. These will resemble those shown below:

```
# run -all
# In DUT l0 x=  2 a=  x b=  x y=  x
# In DUT l0 x=  3 a=  2 b=  x y=  x
# In DUT l0 x= 10 a=  3 b=  2 y=  3
# test passed: input x=10 and output y= 3
# In DUT l0 x=  2 a= 10 b=  3 y=  3
# test passed: input x= 2 and output y= 3
# In DUT l0 x= 12 a=  2 b= 10 y= 10
# test passed: input x=12 and output y=10
# In DUT l0 x=  2 a= 12 b=  2 y=  2
# test passed: input x= 2 and output y= 2
# In DUT l0 x= 10 a=  2 b= 12 y= 10
# test passed: input x=10 and output y=10
# In DUT l0 x=  3 a= 10 b=  2 y=  3
# test passed: input x= 3 and output y= 3
# In DUT l0 x= 12 a=  3 b= 10 y= 10
# test passed: input x=12 and output y=10
# In DUT l0 x= 13 a= 12 b=  3 y= 12
# test passed: input x=13 and output y=12
# In DUT l0 x=  7 a= 13 b= 12 y= 12
# test passed: input x= 7 and output y=12
# In DUT l0 x=  0 a=  7 b= 13 y=  7
# test passed: input x= 0 and output y= 7
# passed          10 and failed           0 test vectors
```