

CME 435

Digital Systems Architecture

Lab #2 Report

October 15, 2021

Thomas Slotboom

Student #:11221182

### Running Test Bench

To run the testbench, run ./script/<phase\_name>/run.csh from the cme435\_ex4 folder.

For phase 7, command line arguments specify which test case is to be run. If no argument or an invalid argument is given, the sanity check test case is run by default. A number can be provided as an argument to select which test file is to be run. The correspondence between test numbers and what each test does is specified below.

Example running phase 7:

* ./script/phase\_7\_scoreboard/run.csh 1
* ./script/phase\_7\_scoreboard/run.csh 2
* ./script/phase\_7\_scoreboard/run.csh 3

### Test cases

In the test bench, currently there are three test files:

(Note - “X”s in outputs specify don’t cares - when nothing is assigned to an output port in a test case we don’t care what value is at that port)

* Test file 1 - Sanity check - Designed to test basic functionality of the xswitch with a single test case.
  + valid\_in = 0b0101
  + addr\_in = 0x0102
  + data\_in = 0x0A08
  + Expected data\_out: 0xX8AX
  + Expected addr\_out: 0xX02X
  + Expected data\_rdy: 0b0110
* Test file 2 - Port assignments - Designed to test that the xswitch assigns data\_in to the correct ports on data\_out.
  + Test case 1:
    - valid\_in = 0b1111
    - addr\_in = 0x3210
    - data\_in = 0xABCD
    - Expected data\_out: 0xABCD
    - Expected addr\_out: 0x3210
    - Expected data\_rdy: 0b1111
  + Test case 2:
    - valid\_in = 0b1111
    - addr\_in = 0x0123
    - data\_in = 0x1234
    - Expected data\_out: 0x4321
    - Expected addr\_out: 0x0123
    - Expected data\_rdy: 0b1111
  + Test case 3:
    - valid\_in = 0b1111
    - addr\_in = 0x2130
    - data\_in = 0x2130
    - Expected data\_out: 0x3210
    - Expected addr\_out: 0x1320
    - Expected data\_rdy: 0b1111
* Test file 3 - Overlapping addresses - Designed to test what happens when there are one or more of the same address specified in addr\_in.
  + Test case 1:
    - valid\_in = 0b1111
    - addr\_in = 0x0000
    - data\_in = 0x4321
    - Expected data\_out: 0xXXX1
    - Expected addr\_out: 0xXXX0
    - Expected data\_rdy: 0b0001
  + Test case 2:
    - valid\_in = 0b1111
    - addr\_in = 0x2200
    - data\_in = 0x4321
    - Expected data\_out: 0xX3X1
    - Expected addr\_out: 0xX2X0
    - Expected data\_rdy: 0b0101
  + Test case 3:
    - valid\_in = 0b1111
    - addr\_in = 0x3303
    - data\_in = 0x4321
    - Expected data\_out: 0x1XX2
    - Expected addr\_out: 0x0XX1
    - Expected data\_rdy: 0b1001

All outputs matched the expected outputs listed above in the actual tests, therefore no bugs have been discovered yet.

### Potential Additional Test Cases to be Added

* Test the validity of the data\_rdy signal, EG what happens if an output is read from DUT before data\_rdy is high at the port.
* Test the validity of the data\_read signal, EG what happens if more data is fed to the DUT before data\_read is high.
* Test what happens if there is a value in addr\_in that is higher than three (invalid port number)

### Potential Improvements to be Made

* Have more flexible control over timing to get more information on what happens to outputs of the DUT on each and every clock cycle.
* More modular way to compile different test cases - right now there are three different .f and .do files to compile the different test files, would be better to only use one of each and have some logic to pick which test file to compile.
* Have randomized test cases to cover a wide variety of test cases with no bias in inputs being fed to the DUT.