Learning Activity 1

By Alex Clunan

ECE 4435 Computer Architecture & Design

February 2, 2025

1 Problem Statement

Implement three multiplexers (MUXs) using Verilog in the 2-1, 3-1, and 4-1 configurations with a configurable input/output bus width. Also implement an edge triggered register with reset and enable functionalities using Verilog. Must also have a configurable bus width of the inputs/outputs.

2 Analytical Design

The 2-1 MUX consists of 3 inputs, 1 output, and 1 WIDTH parameter. d0 and d1 are data inputs to the MUX, and the other input, sel, is a 1 bit select line. The output y is the output of the MUX. The logic of the output is determined in the line: assign y = sel ? d1 : d0;. This sets the output y to d1 if sel is 1, and selects d0 if the inverse is true. The RTL schematic of the circuit is shown in Fig 1.

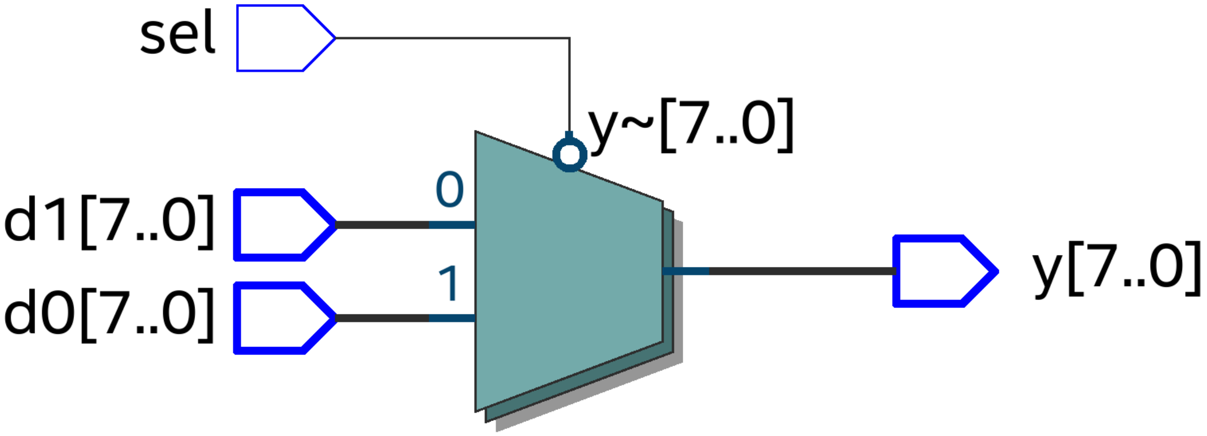


Fig 1. RTL Schematic of Designed 2-1 MUX

The 3-1 MUX was the next component designed. The 3-1 MUX is a combination of two 2-1 MUXs chained together. The first two inputs and first bit of the select line are connected to the first MUX, and the third input, first MUX output and second bit of the select line are connected to the second MUX. This setup allows the two MUXES to be configured as a 3-1 MUX. The RTL schematic is shown in Fig 2.

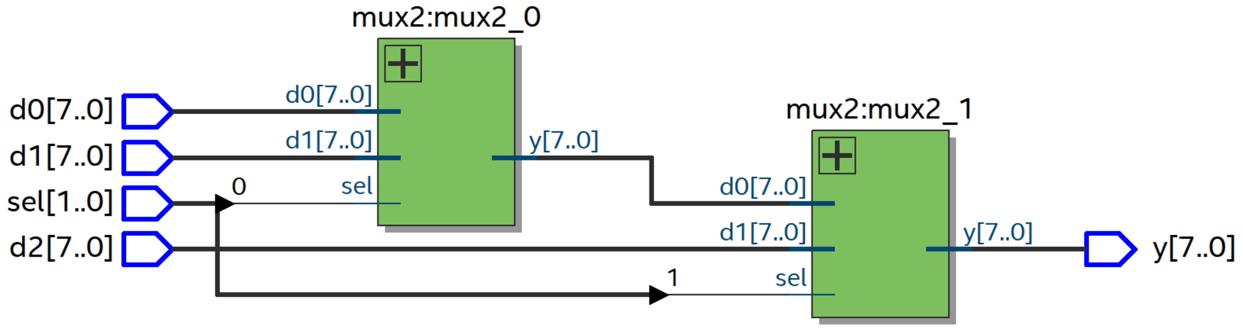


Fig 2. RTL Schematic of Designed 3-1 MUX

Next, the 4-1 MUX was designed. It includes three 2-1 muxes with the output of two MUXs serving as the input for a third MUX. The RTL schematic is shown in Fig 3.

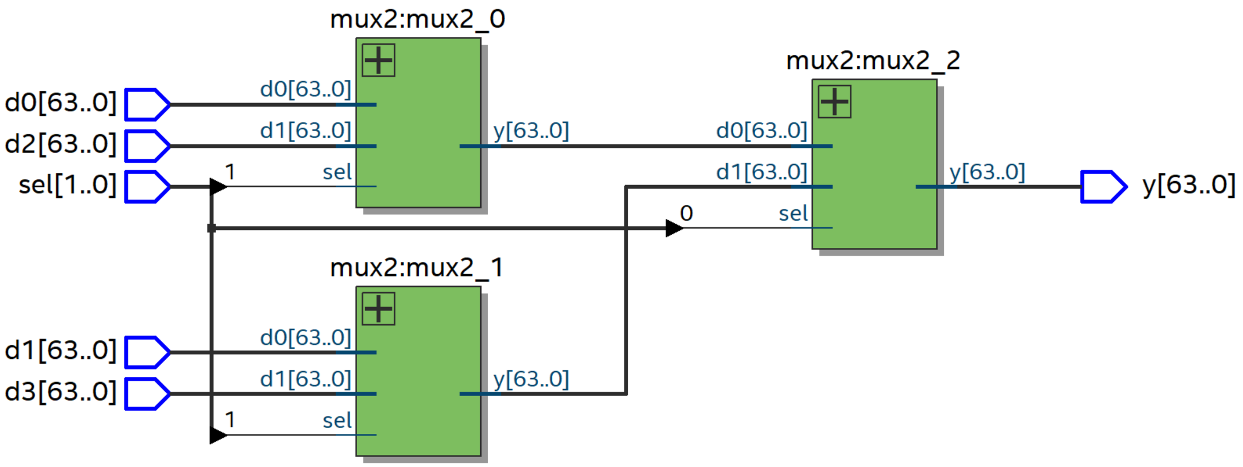


Fig 3. RTL Schematic of Designed 4-1 MUX

Fig 3 shows 64 bit width input/output of each MUX. This shows that the WIDTH parameter can be figured for each module. There are also no latches in the RTL schematic which verifies the no-latch condition.

Finally the n-bit register was designed. The register takes 4 inputs: reset, clock, enable, and D. It has one output: Q. D and Q are both designed to take a parameterized WIDTH variable that determines the bit size of the bus. All logic is defined within an always statement that checks for the positive edge of the clock or reset. Once inside the always statement an if-else statement lays out the logic for the register. The logic is as follows: if reset is high, set Q to 0, else if enable is high, set Q to D, otherwise set Q to Q. This logic outputs the RTL schematic in Fig 4 which matches the course slides.

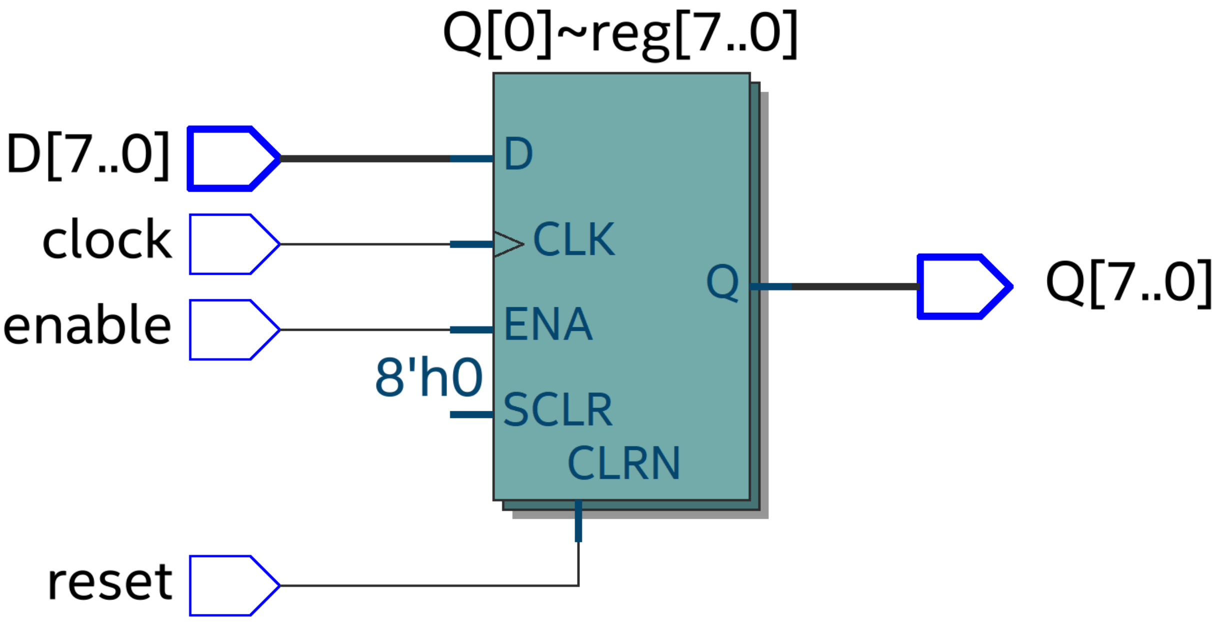


Fig 4. RTL Schematic of Designed 8-bit Register

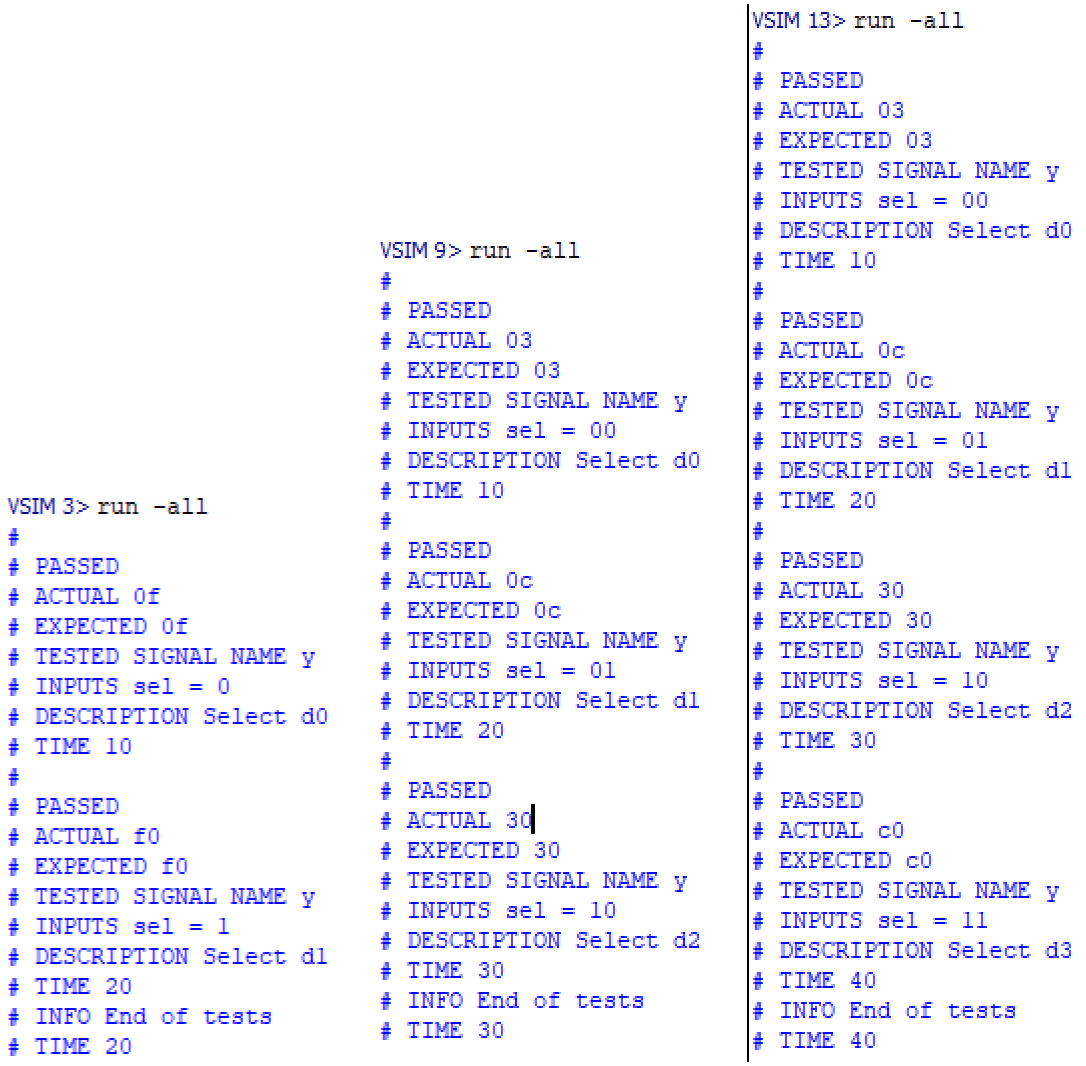
3 Numerical Verification

The MUXs were tested using testbenches that checked every select value for every MUX. If the output matched the expected input, then the test passed. Fig 5. shows the verification message for the MUXs, and Figs 5-7 show the verification waveform.

4 Summary

The problem requirement for this assignment was to design a 2-1 MUX, 3-1 MUX, 4-1 MUX, and register with enable. Each design needed an adjustable bus width for the inputs and outputs.

A singular 2-1 MUX was first created, and then the 3-1 and 4-1 MUXs used chained implementations of the designed 2-1 MUX. The register used if statements inside an always block to define its logic. “The circuit was implemented using Verilog, a hardware implementation was synthesized from the Verilog implementation, and finally the design was verified numerically to confirm that the specified digital circuit design solution satisfies the problem requirements.” (from Example Submission for learning activity 0)

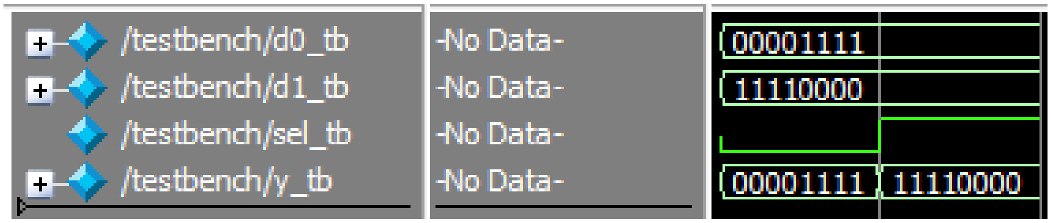


```
VSIM 3> run -all
#
# PASSED
# ACTUAL 0f
# EXPECTED 0f
# TESTED SIGNAL NAME y
# INPUTS sel = 0
# DESCRIPTION Select d0
# TIME 10
#
# PASSED
# ACTUAL f0
# EXPECTED f0
# TESTED SIGNAL NAME y
# INPUTS sel = 1
# DESCRIPTION Select d1
# TIME 20
# INFO End of tests
# TIME 20
```

```
VSIM 9> run -all
#
# PASSED
# ACTUAL 03
# EXPECTED 03
# TESTED SIGNAL NAME y
# INPUTS sel = 00
# DESCRIPTION Select d0
# TIME 10
#
# PASSED
# ACTUAL 0c
# EXPECTED 0c
# TESTED SIGNAL NAME y
# INPUTS sel = 01
# DESCRIPTION Select d1
# TIME 20
#
# PASSED
# ACTUAL 30
# EXPECTED 30
# TESTED SIGNAL NAME y
# INPUTS sel = 10
# DESCRIPTION Select d2
# TIME 30
# INFO End of tests
# TIME 30
```

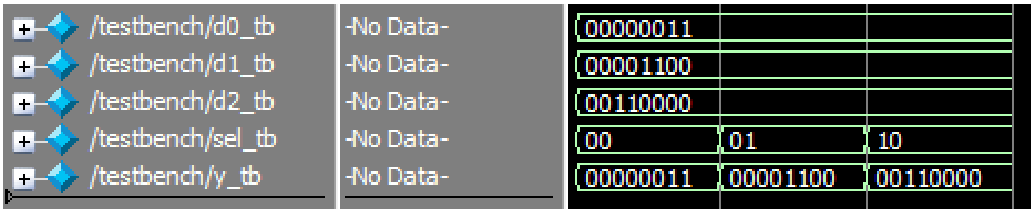
```
VSIM 13> run -all
#
# PASSED
# ACTUAL 03
# EXPECTED 03
# TESTED SIGNAL NAME y
# INPUTS sel = 00
# DESCRIPTION Select d0
# TIME 10
#
# PASSED
# ACTUAL 0c
# EXPECTED 0c
# TESTED SIGNAL NAME y
# INPUTS sel = 01
# DESCRIPTION Select d1
# TIME 20
#
# PASSED
# ACTUAL 30
# EXPECTED 30
# TESTED SIGNAL NAME y
# INPUTS sel = 10
# DESCRIPTION Select d2
# TIME 30
#
# PASSED
# ACTUAL c0
# EXPECTED c0
# TESTED SIGNAL NAME y
# INPUTS sel = 11
# DESCRIPTION Select d3
# TIME 40
# INFO End of tests
# TIME 40
```

Fig 4. Verification Messages for each MUX (2-1 on left, 4-1 on right, 3-1 in the middle)



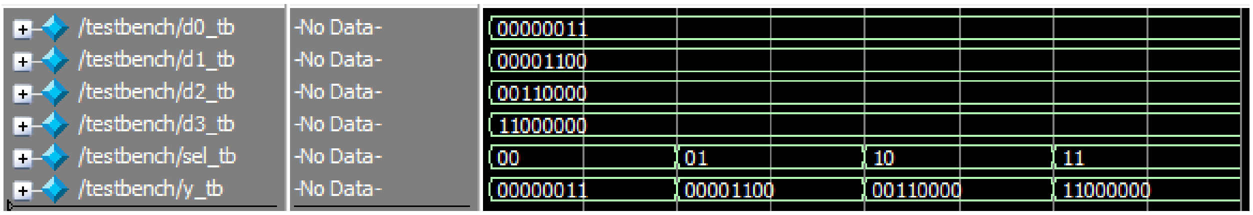
```
/testbench/d0_tb   -No Data-   00001111
/testbench/d1_tb   -No Data-   11110000
/testbench/sel_tb  -No Data-
/testbench/y_tb    -No Data-   00001111   11110000
```

Fig 5. Verification Waveform for 2-1 MUX



```
/testbench/d0_tb   -No Data-   00000011
/testbench/d1_tb   -No Data-   00001100
/testbench/d2_tb   -No Data-   00110000
/testbench/sel_tb  -No Data-   00         01         10
/testbench/y_tb    -No Data-   00000011   00001100   00110000
```

Fig 6. Verification Waveform for 3-1 MUX



```
/testbench/d0_tb    -No Data-   00000011
/testbench/d1_tb    -No Data-   00001100
/testbench/d2_tb    -No Data-   00110000
/testbench/d3_tb    -No Data-   11000000
/testbench/sel_tb   -No Data-   00         01         10         11
/testbench/y_tb     -No Data-   00000011   00001100   00110000   11000000
```

Fig 7. Verification Waveform for 4-1 MUX