**Objective:**

The objective of this lab is to practice designing using registers in order to store outputs in a circuit regardless of input and to continue using previous designs as modules in more complex designs.

**Design:**

**Design 1:**

A 4-bit register made of four positive-edge triggered flip-flops:

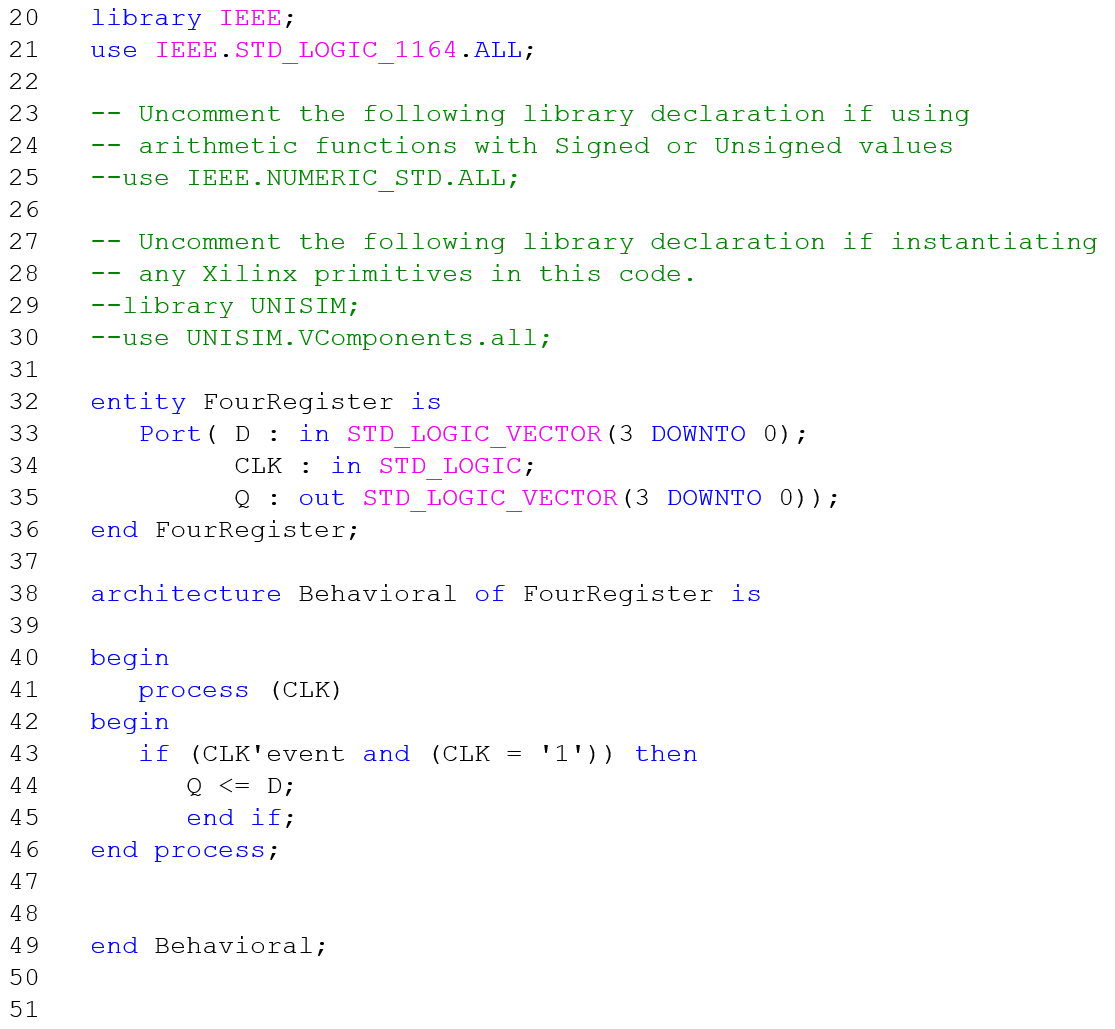


Figure - Four-bit Register VHDL



Table - 4-bit Register Test Vectors

**Design 2:**

A 4x4bit Register Bank using a 4 4-bit registers with a 2:4 decoder to select a register to write to and 4set:1set multiplexer to select the output

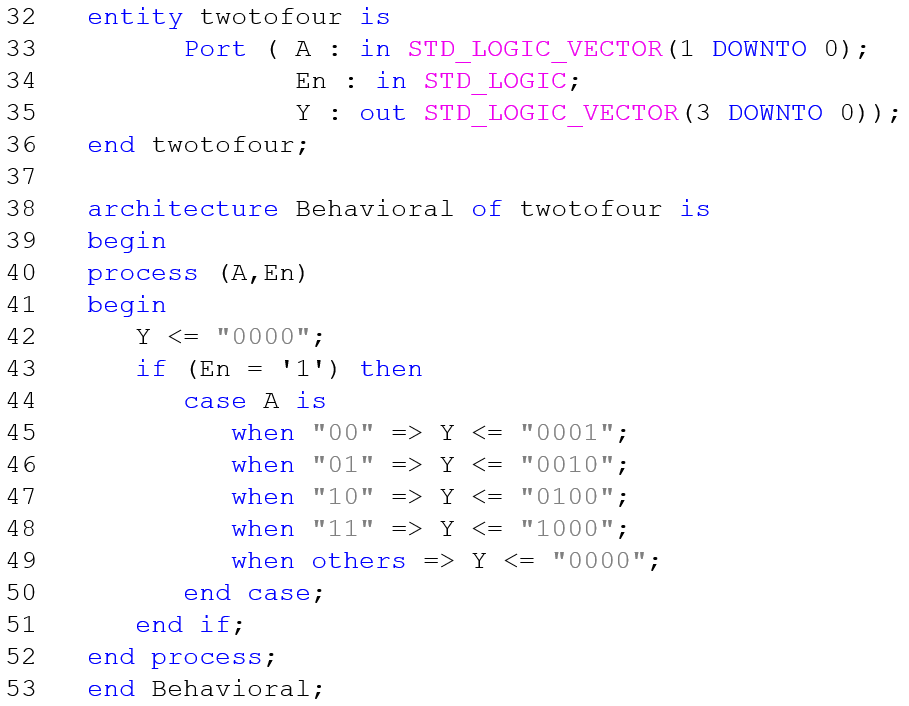


Figure - 2:4 Decoder VHDL



Table - 2:4 Decoder Test Vector

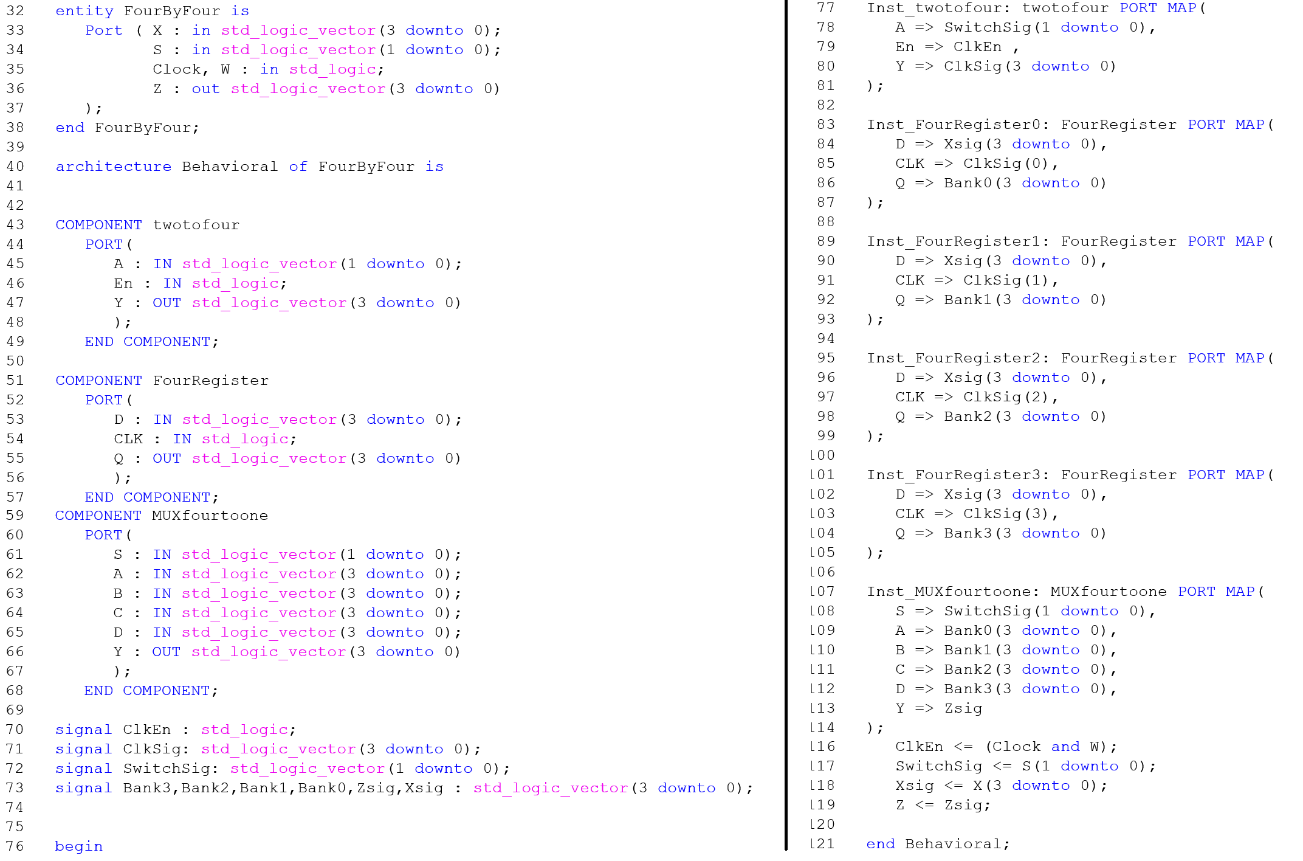


Figure - 4x4bit Register Bank VHDL



Table - 4x4bit Register Bank Test Vectors

**Procedure:**

**Part 1:**

* Design a 4-bit register using VHDL using the same steps as in the previous lab (Fig 1)
* Run a behavioral simulation for the test vectors in Table 1
* Download to FPGA Board and test the same combinations
  + The clock will not be automatic, so connect the Clock input to the leftmost switch on the FPGA
  + Since it is being driven manually add the below to the end of your .ucf file
    - NET "XXXXX" CLOCK\_DEDICATED\_ROUTE = "FALSE";
    - Replace XXXXX with the clock input name

**Part 2:**

* Write the VHDL file for a 4x4bit Register Bank using a 2:4 Decoder, 4set:1set MUX, and 4 4-bit Registers as components (Fig 3)
  + Write the VHDL for a 2:4 Decoder
    - Run behavioral simulation for all outputs using test vectors in Table 2
  + Copy the 4set:1set MUX VHDL file from Lab 7 into the Lab 8 project folder
  + Add the Decoder, MUX, and four copies of the 4-bit register as components to the 4x4bit Register bank as done last week
    - To make a component
      * Select 4:16 file
      * Expand Design Utilities
      * Double click VHDL Instantiation Template
    - To implement the template in another VHDL File
      * Copy and paste the component section between the Architecture and Begin statement
      * Copy and paste the component section between the Begin and End Behavioral statement
* Run a behavioral simulation with the given test vectors (Table 3)
* Download to FPGA Board and test the same combinations
  + Connect the “Write” input to the leftmost switch
  + The clock will not be automatic, so connect the “Clock” input to the switch immediately to the right of the “Write” switch
  + Since it is being driven manually add the below to the end of your .ucf file
    - NET "XXXXX" CLOCK\_DEDICATED\_ROUTE = "FALSE";
    - Replace XXXXX with the clock input name

**Data:**

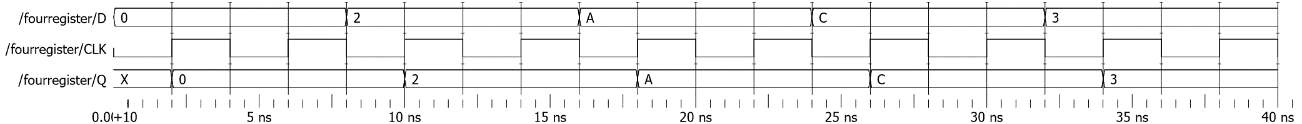


Figure - 4-bit Register Behavioral Simulation

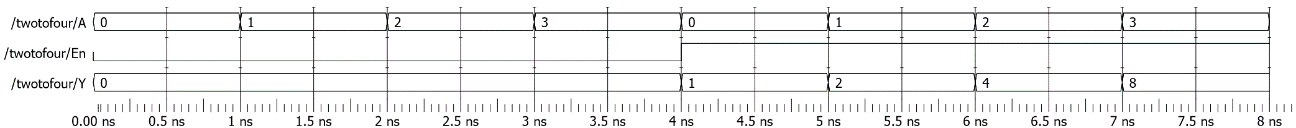


Figure - 2:4 Decoder Behavioral Simulation

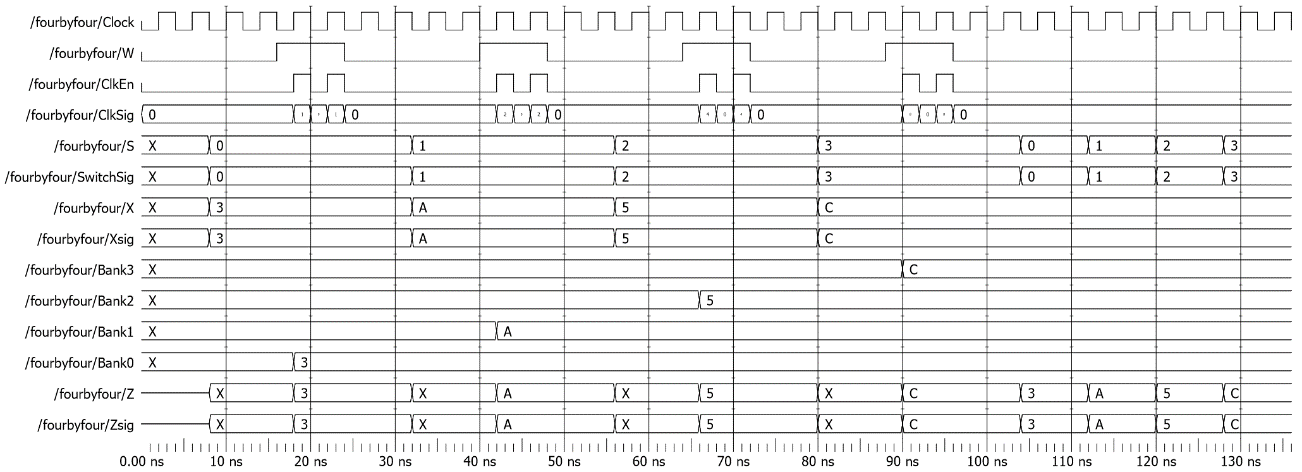


Figure - 4x4bit Register Bank Behavioral Simulation

**Data Analysis:**

The downloaded design performed as simulated without discrepancy. I did test additional inputs on the 4x4bit Register to verify that the “Write” input did determine when new values can be stored.

**Conclusion:**

This experiment was intended to demonstrate the ability to store and recall specific outputs as well utilizing edge triggered clock to allow inputs to propagate as needed. These two capabilities allow us to minimize errors that can occur in unintentional fluctuations in circuits as well as store values (outputs) to be recalled as needed.

The main difficulties in the experiment arose from unfamiliarity with VHDL and simple syntax errors. For example, in my original 4x4bit Register Bank VHDL file I wrote that the final output vector was supposed to write to the output signal instead of the other way around. Troubleshooting guided by the software allowed me to find and correct these mistakes.