University of California, Davis
Department of Electrical and Computer Engineering

EEC180B DIGITAL SYSTEMS II Spring 1999

Lab 6: Register File Design in VHDL

Objective: In this lab you will design a small register file in VHDL and verify it by functional simulation. The register file will be a component in the final processor lab.

Pre-lab: Write a *complete*, *syntactically correct* VHDL process that implements a 4:1 bus multiplexer using a case statement. The multiplexer will direct one of four 8-bit buses to the 8-bit output, under the control of the two-bit select input. Show the declaration of all signals used in your process.

<u>Note: this lab and pre-lab are to be done individually. That is, you may not work with anyone else or use anyone else's work.</u>

Problem Specification

Design a register file consisting of four eight-bit registers, R0 - R3. The external interface of the register file is shown in Figure 1.

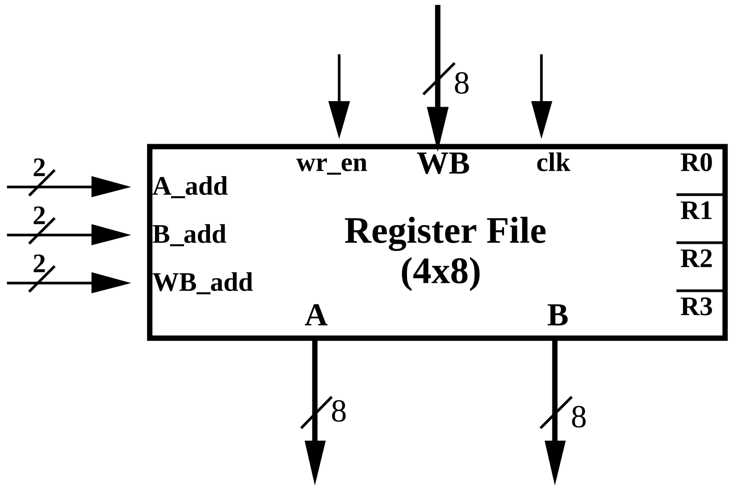

Figure 1 : External Interface of the Register File

There are two *read* buses, A and B, and one *write* bus, WB. There are also three two-bit *address* buses, A_add, B_add, and WB_add. Each of these address buses is used to specify one of the four registers for either reading or writing. The write operation takes place on the rising edge of the clk signal when the wr_en signal is logic 1. The read operation, however, is not clocked - it is combinational. Thus, the value on the A bus should always be the contents of the register specified by the A_add bus. Similarly, the value on the B bus should always be the contents of the register specified by the B_add bus. So, with this register file, you can write into a register and read *two* registers *simultaneously*. It is also possible to read a single register on both of the read buses simultaneously. Your VHDL model ***must not infer tri-state buffers.*** Your code must generate combinational logic to multiplex signals rather than tri-state buffers.

Lab Guidelines

The entity of the VHDL model should be as follows:

```
entity regfile is
port (clk, wr_en : in std_logic;
        A_add, B_add, WB_add : in std_logic_vector(1 downto 0);
        WB : in std_logic_vector(7 downto 0);
        A, B : out std_logic_vector(7 downto 0));
end regfile;
```

Note that you do not need a reset signal in your design. The registers in the register file will be undefined until data is written into them. (In the final processor design, you will use a special reset buffer, STARTBUF, to drive the GSR signal in the Xilinx FPGA so that you can reset all the flip-flops in your design by pressing the reset switch.)

The registers in the register file can be declared using an array. This is not required, but it is a convenient way to reference the registers.

Lab Requirements

1. Complete the pre-lab assignment, working entirely alone, before your first lab period.
2. Write a *synthesizable* VHDL model of the register file. Use a separate process for the clocked logic that implements the write operation. Use one or more processes or concurrent statements for the combinational logic that implements the read operation.
3. Write a test bench to verify the operation of the circuit. Verify that you can correctly read from and write to each register in the register file. Also verify that the wr_en signal works correctly. Demonstrate your simulation to your TA for verification.
4. Synthesize the sequential machine for the Xilinx library. Make sure you are using the .synopsys_dc.setup file for the Xilinx library. Print the gate-level schematic from the Synopsys Design Analyzer. Since you will not be downloading this design to a Xilinx board, you do not need to use an attribute file.
5. Examine the synthesis log file and make sure you did not infer latches in your design. The log file should report how many flip-flops were inferred.

Lab Report

Submit the following items in your lab report: (Due in your first lab period of the following week - i.e. one week after the start of the lab.)

- ❑ All VHDL source code.
- ❑ Test bench source code.

- ❑ Simulation waveforms from your functional simulation
- ❑ Gate-level schematic of your register file printed from the Design Analyzer.
- ❑ Signed TA verification for the functional simulation.
- ❑ From the various Xilinx report files, record the number of flip-flops and CLBs used in your design. Print the summary of FPGA resource usage from the appropriate report file and indicate which report file you used.
- ❑ Answer the following question:

  1. Compare the expected synthesis results in using the following three VHDL constructs to multiplex 4 or more inputs to an output:
     - Case statement
     - If then elsif statement
     - Conditional signal assignment

     Which construct would you expect to generate the fastest multiplexer? Why? If time permits, you could test your answer by coding multiplexers with each of the constructs and analyzing and elaborating the code with the Synopsys Design Analyzer, although this is not required.