Introduction to Digital Logic EECS/CSE 31L

Assignment 4: Register File

Prepared by: Jack Melcher Student ID: 67574625

EECS Department Henry Samueli School of Engineering University of California, Irvine

February 15, 2015

1 Block Description

- Construct a register file that can be parameterized
- Per rising edge of clock, the register file can:
 - Write to up to one register
 - o Read from up to two registers

2 Input/Output Port Description

Generic Name	Data Type	Default Value	Description
NBIT	INTEGER	32	Size of register
NSEL	INTEGER	8	Number of Registers

Port name	Port size	Port Type	Description
Clk	1	IN	Triggers the register process
Rst_s	1	IN	Will reset the register when active and clock rises
We	1	IN	Enables writing to the registers
Raddr_1	NSEL-1 downto 0	IN	The address of the register to be read from
Raddr_2	NSEL-1 downto 0	IN	The address of the register to be read from
Waddr	NSEL-1 downto 0	IN	The address of the register to be written to
Rdata_1	NBIT-1 downto 0	OUT	The data value from the register that was read
Rdata_2	NBIT-1 downto 0	OUT	The data value from the register that was read
Wdata	NBIT-1 downto 0	IN	The data value to be written to the register

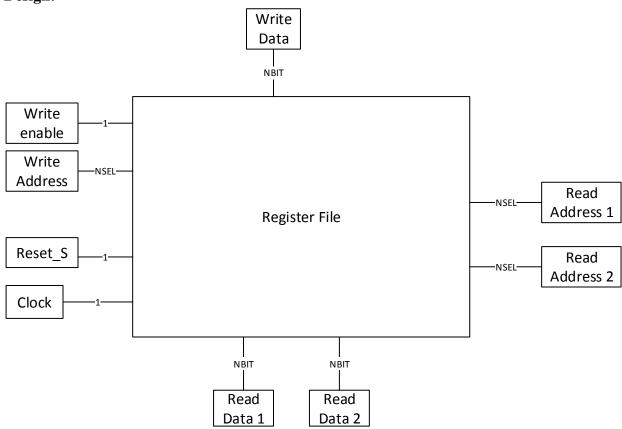
Variable Name	Data Type	Port size	Description
regfile	Array (0 to 2**NSEL)	NBIT-1 downto 0	The registers

3 Design Schematics

Truth Table:

Clk	Rst_s	We	Raddr_1	Raddr_2	Waddr	Function
1	1	X	XXX	XXX	XXX	Synchronous Reset
1	0	0	XXX	XXX	XXX	Prevent writing Also can read from any address
1	0	1	XXX	XXX	XXX	Allow writing to any address Also can read from any address

Design:



4 Compilation

Register File

```
■ ISE Project Navigator (P.20131013) - C\Users\Jack\Desktop\EECS 31L\hw4\assignment4_67574625\assignment4_67574625.xise - [assignment4_67574625_registerfile.vhc
File Edit View Project Source Process Tools Window Layout Help
 Design

View: A Implementation A Simulation

Behavioral

Litierarchy
                                                        -- Array for register file
TYPE regfile IS ARRAY (0 TO 2**NSEL-1) OF STD_LOGIC_VECTOR (NBIT-1 downto 0);
       d assignment4_67574625

☐ xc3s700a-4fq484

     44 begin
                                                             gin
PROCESS(clk, rst_s, we)
VARIABLE regfile temp: regfile;
VARIABLE temp_rdata_1, temp_rdata_2 : STD_LOGIC_VECTOR (NBIT-1 downto 0);
      testbench_register - behavior uut - assignment4_6757462
                                                                 GIN

-- Register file functions

IF (clk'EVENT AND clk = '1') THEN

-- Synchronous Reset Register file

IF (rst s = '1') THEN

FOR I IN 0 TO 2**MSEL-1 LOOP

regifile temp(i) := (OTHERS => '0');

END LOOP;
                                                    49
50
51
52
                                                    53
54
   ∢ | | | | |
                                                    55
56
57
58
59
60
61
                                                                      END LOOP;

-- Write to registers

ELSIF (we = '1') THEN

regfile_temp(to_integer(unsigned(waddr))) := wdata;

END IF;

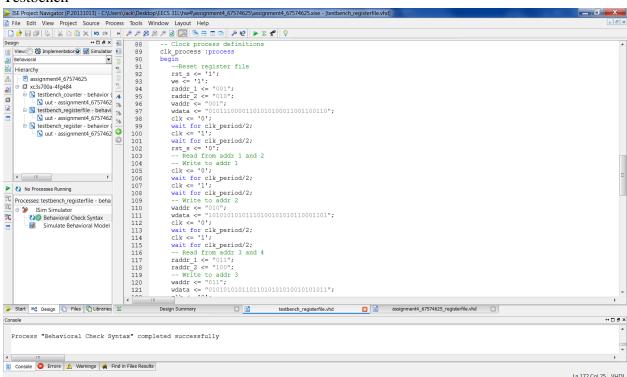
-- Read address 1

temp rdata 1 := regfile_temp(to_integer(unsigned(raddr_1)));

-- Read address 2

-- refile_temp(to_integer(unsigned(raddr_2)));
No Processes Running
Processes: uut - assignment4_67574625
    i⇒ Simulator
Behavioral Check Syntax
Simulate Behavioral Model
                                                    62
63
                                                                      temp_rdata_2 := regfile_temp(to_integer(unsigned(raddr_2)));
                                                              END 1F;
-- Output the value of read address rdata_1 <= temp_rdata_1;
rdata_2 <= temp_rdata_2;
END PROCESS;
                                                    65
66
67
68
69
70
                                                        end Behavioral;
                                                                                 ×
                                                                                                            testbench_registerfile.vhd
> Start  Design  Files Libraries
                                                             Design Summary
  Process "Behavioral Check Syntax" completed successfully
Console Errors & Warnings & Find in Files Results
                                                                                                                                                                                                                                   Ln 63 Col 60 VHDL
```

Testbench



5 Elaboration

Assumptions:

- I didn't need to use the register block to construct the register file.
- Instead, I implemented an array of std_logic_vectors that is basically a set of registers
- Utilizing the array and process, I was able to construct the register file

Regarding how I wrote my vhdl code

- I used a process with variables
- Used sequential if and elsif statements

Errors and Challenges:

- Misusing assignment operators; tried using signal assignment operator on a variable
- Understanding how to access the array properly. I realized that I needed to type convert

Simulation Log:

Started: "Simulate Behavioral Model".

Determining files marked for global include in the design...

Running fuse...

Command Line: fuse -intstyle ise -incremental -o {C:/Users/Jack/Desktop/EECS

31L/hw4/assignment4_67574625/testbench_registerfile_isim_beh.exe} -pri

{C:/Users/Jack/Desktop/EECS 31L/hw4/assignment4_67574625/testbench_registerfile_beh.prj} testbench_registerfile {}

Running: C:\Xilinx\14.7\ISE_DS\ISE\bin\nt64\unwrapped\fuse.exe -intstyle ise -incremental -o C:/Users/Jack/Desktop/EECS

31L/hw4/assignment4_67574625/testbench_registerfile_isim_beh.exe -prj

C:/Users/Jack/Desktop/EECS 31L/hw4/assignment4_67574625/testbench_registerfile_beh.prj testbench_registerfile

ISim P.20131013 (signature 0x7708f090)

Number of CPUs detected in this system: 4

Turning on mult-threading, number of parallel sub-compilation jobs: 8

Determining compilation order of HDL files

Parsing VHDL file "C:/Users/Jack/Desktop/EECS

31L/hw4/assignment4_67574625/assignment4_67574625_registerfile.vhd" into library work

Parsing VHDL file "C:/Users/Jack/Desktop/EECS

31L/hw4/assignment4_67574625/testbench_registerfile.vhd" into library work

Starting static elaboration

Completed static elaboration

Compiling package standard

Compiling package std_logic_1164

Compiling package numeric_std

Compiling architecture behavioral of entity assignment4_67574625_registerfile

[\assignment4_67574625_registerfi...]

Compiling architecture behavior of entity testbench_registerfile

Time Resolution for simulation is 1ps.

Waiting for 1 sub-compilation(s) to finish...

Compiled 6 VHDL Units

Built simulation executable C:/Users/Jack/Desktop/EECS

31L/hw4/assignment4_67574625/testbench_registerfile_isim_beh.exe

Fuse Memory Usage: 35932 KB

Fuse CPU Usage: 498 ms

Launching ISim simulation engine GUI...

"C:/Users/Jack/Desktop/EECS

31L/hw4/assignment4_67574625/testbench_registerfile_isim_beh.exe" -intstyle ise -gui -tclbatch isim.cmd -wdb "C:/Users/Jack/Desktop/EECS

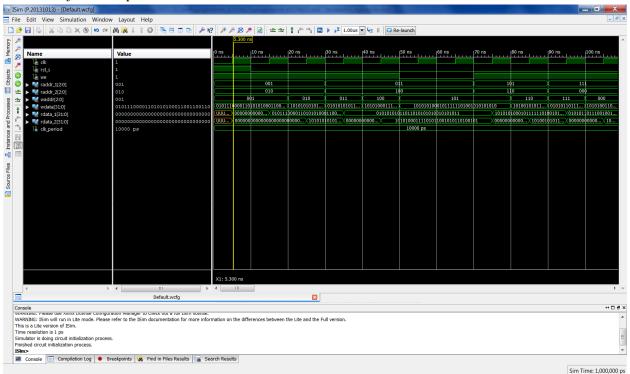
31L/hw4/assignment4_67574625/testbench_registerfile_isim_beh.wdb"

ISim simulation engine GUI launched successfully

Process "Simulate Behavioral Model" completed successfully

6 Waveforms

Full Waveform Snapshot



Snapshots of Waveform

Reset

Name	Value	324 ns	326 ns	328 ns	330 ns	332 ns	334 ns		336 ns	338 ns	340 ns	342 ns
Ū₀ clk	1											
V₀ rst_s	1											
14 we	1											
▶ ■ raddr_1[2:0]	001		111							001		
▶ 😽 raddr_2[2:0]	010		000							010		
▶ ■ waddr[2:0]	001		000							001		
▶ ■ wdata[31:0]	01011100001101010100011001100110	1010100110	10101001010101	10101100					010111000	01101010100011	001100110	
	000000000000000000000000000000000000000		010101	10111001001010	100001101010		X		00	00000000000000	0000000000000000	00
▶ ➡ rdata_2[31:0]	000000000000000000000000000000000000000	0000 X	10	10100110101010	01010101101011	00	X			00000	000000000000000000000000000000000000000	0000000000000
□ clk_period	10000 ps							1000	0 ps			

Writing and Reading from various Addresses

Name	Value		352 ns	354 ns		356 ns	358 ns	360 ns	362 ns	364 ns	366 ns	368 ns	370 ns	372 ns	374 ns
Ūei clk	1	1													
Ve rst_s	0														
Ve we	1														
▶ ■ raddr_1[2:0]	001				001						011				
▶ ■ raddr_2[2:0]	010				010						100				
▶ ■ waddr[2:0]	010				010					011			X	100	
▶ ■ wdata[31:0]	101010101011101001010101110001101		10101	01010111	10010	010110001101			01010101	11011010101010	0010101011		10101000111	0101100101011	0100101
	01011100001101010100011001100110				0:	01110000110101	01000110011001	10		X		101010101110110	10101010010101	011	
▶ ₹ rdata_2[31:0]	1010101010111010010101010110001101	00000	00000000000000000	0000		10	101010101111010	01010101100011	01	X	0	000000000000000000000000000000000000000	0000000000000000	00	
le clk_period	10000 ps								10000 ps						

Hold (write enable off)

Name	Value	1	378 ns	380 ns	382 ns	384 ns	 386 ns	388 ns	390 ns	392 ns	394 ns	396 ns	398 ns
Ūe clk	1												
la clk la rst_s	0												
le we	0												
▶ ™ raddr_1[2:0]	011							011					X
▶ 😽 raddr_2[2:0]								100					
▶ ■ waddr[2:0]	101		100						101				
▶ ₩ wdata[31:0]	10101010001011111101001010101010	10101000	11110101100					101010100010	1111101001010	01010			
▶ ➡ rdata_1[31:0]	0101010101101101010101010010101011						0101010101	10110101010100	10101011				
# rdata_2[31:0]	10101000111101011001010110100101						1010100011	11010110010101	10100101				
le clk_period	10000 ps							10000 ps					