Part 1:

Objectives of the Lab/Program. (1 point)

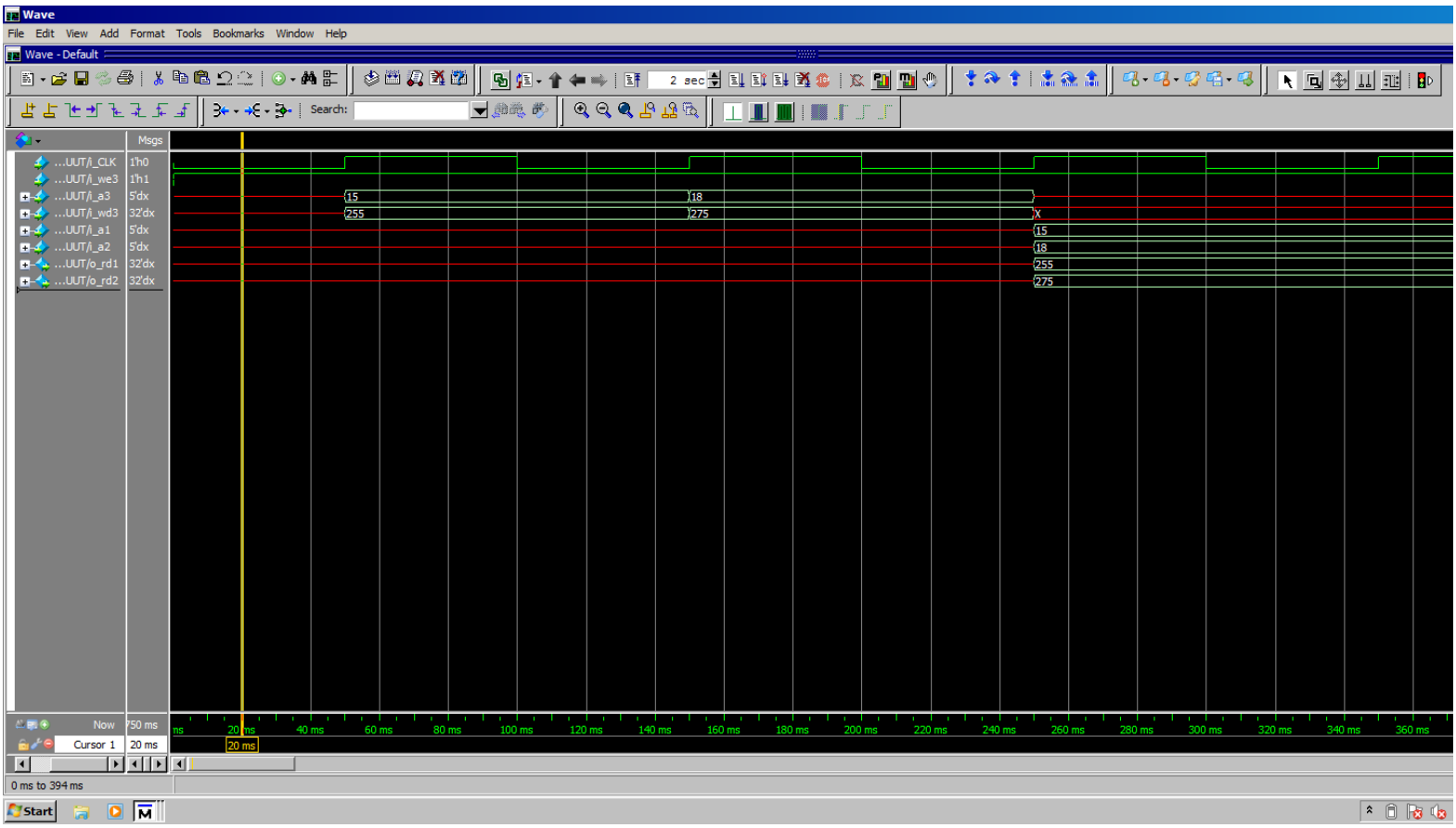
The objectives of the laboratory assignment is to design, implement, test, and simulate a 32-bit register file using a standardized Hardware Description Language.

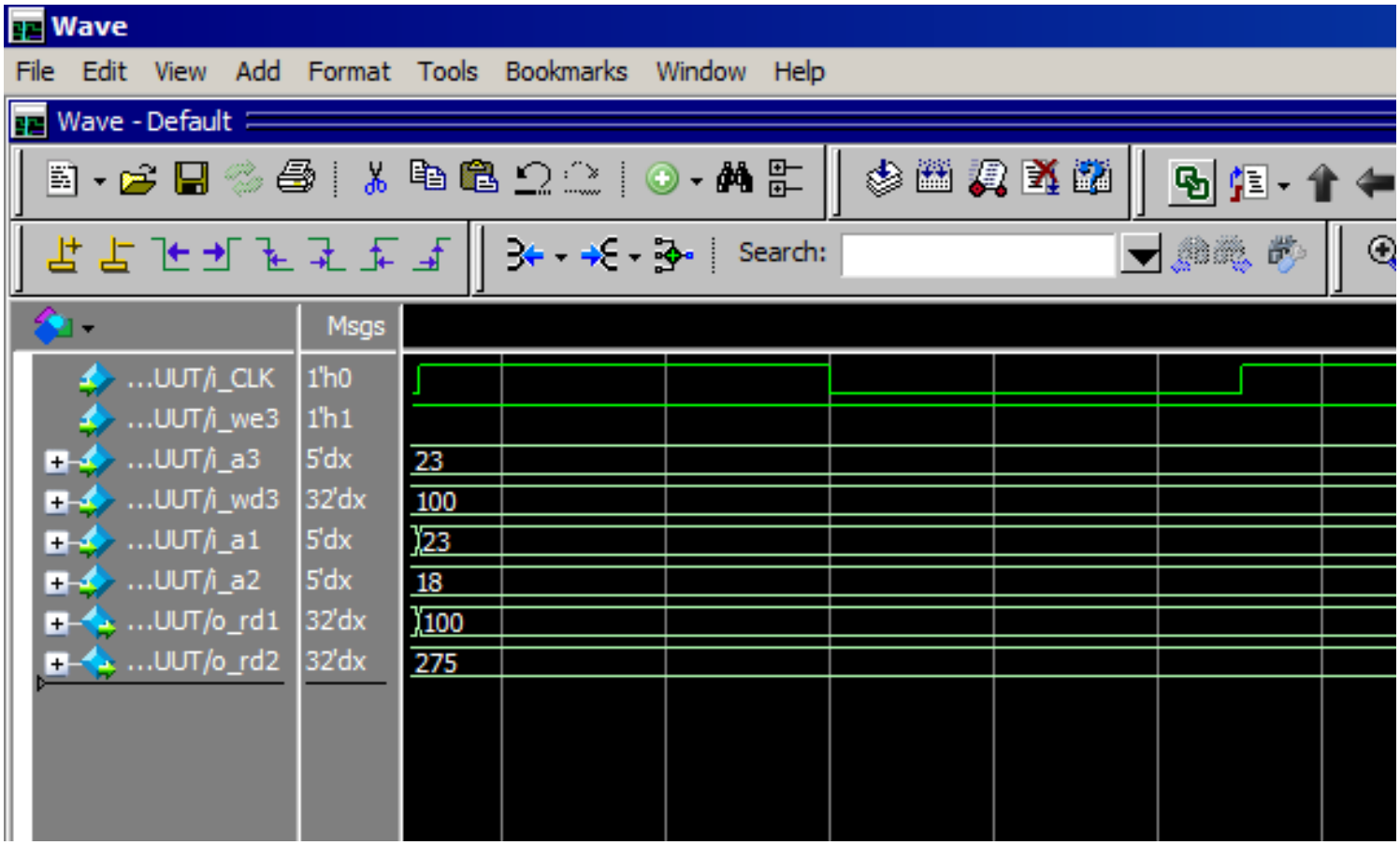
Verilog/VHDL Source Codes (4 points)

```
//Lab 6:
//Implement a register file. See pdf for details
//Author: Peter Dranishnikov    //Partner: NOT APPLICABLE
//Due 07/24/18: Demo: 08:00; Report: 23:59
module regfile
    (
        i_a1, i_a2, i_a3, i_wd3, i_CLK, i_we3,
        o_rd1, o_rd2
    );

    input [4:0] i_a1, i_a2, i_a3;
    input [31:0] i_wd3;
    input i_CLK, i_we3;
    output [31:0] o_rd1, o_rd2;

    /* RegFile structure
     * There are 2^5 == 32 PIPO registers with write enable
for:
     * READING:
     * bus a1 and a2 are select lines for a 32x5 output mux
for each o_rd bus
     * WRITING:
     * at a low level, a decoder on the a3 line AND w/ we3 at
the individual register
     * enables the writing of the register
     * Verilog abstracts this as writing a 1d array to a 2d
array
     */
    //Register block definition
    reg [31:0] r_register_set [31:0];
    //Read code
    assign o_rd1 = r_register_set[i_a1];
    assign o_rd2 = r_register_set[i_a2];
    //Write code
    always begin
```

```
            @(posedge i_CLK)
                if (i_we3)
                    r_register_set[i_a3] <= i_wd3;
        end
    endmodule
    `timescale 1ms/1us
    module t_regfile;
        reg[4:0] t_i_a1, t_i_a2, t_i_a3;
        reg[31:0] t_i_wd3;
        reg t_i_CLK = 1'b0;
        reg t_i_we3 = 1'b0;
        wire[31:0] t_o_rd1, t_o_rd2;

        regfile UUT
        (
            .i_a1(t_i_a1),
            .i_a2(t_i_a2),
            .i_a3(t_i_a3),
            .i_wd3(t_i_wd3),
            .i_CLK(t_i_CLK),
            .i_we3(t_i_we3),
            .o_rd1(t_o_rd1),
            .o_rd2(t_o_rd2)
        );
        always #50 t_i_CLK <= !t_i_CLK;

        initial begin
            t_i_we3 <= 1'b1;
            #50
            t_i_a3 <= 5'd15;
            t_i_wd3 <= 32'd255;
            #100
            t_i_a3 <= 5'd18;
            t_i_wd3 <= 32'd275;
            #100
            t_i_a3 <= 5'dx;
            t_i_wd3 <= 1'bx;
            t_i_a1 <= 5'd15;
            t_i_a2 <= 5'd18;
            #500
            $stop;
        end
    endmodule
```

Screen shot of the simulation (waveforms). (4 points)





Note: Images are cropped to fit page.

Conclusion and References. (1 point)

To verify the design of the register file, two registers are written to with different decimal values. The values are then read simultaneously. Afterwards, an input and output register number of 23, input value 100 are forced using the simulation. The register file behaves as described in the assignment file (not included). Therefore, it is possible to implement both a combinational and sequential logic and storage in a single block in a standardized HDL.

Patterson, David A. & Hennessy, John L. *Computer Organization and Design* 3rd edition
Harris, David Money & Harris, Sarah L. *Digital Design and Computer Architecture* 2nd edition