Digital Design and Computer Organisation Laboratory

UE23CS251A

3rd Semester, Academic Year 2024-25

Date: 16-10-2024

Name: ABHISHEK P	SRN: PES2UG23AM002	Section: A

Week#____6_______ Program Number: ___1___

WRITE A VERILOG PROGRAM TO CONSTRUCT A REGISTER FILE, FROM WHICH TWO 16-BIT VALUES CAN BE READ, AND TO WHICH ONE 16-BIT VALUE WRITTEN, EVERY CLOCK CYCLE. GENERATE THE VVP OUTPUT AND SIMULATION WAVEFORM USING GTKWAVE. VERIFY THE OUTPUT AND WAVEFORM WITH THE TRUTH TABLE.

I. Verilog Code and Test Bench Code
II. Verilog VVP Output Screen Shot
III. GTKWAVE Screenshot

Library file code:

```
module invert (input wire i, output wire o);
   assign o = !i;
endmodule

module and2 (input wire i0, i1, output wire o);
  assign o = i0 & i1;
endmodule

module or2 (input wire i0, i1, output wire o);
  assign o = i0 | i1;
endmodule

module xor2 (input wire i0, i1, output wire o);
  assign o = i0 ^ i1;
endmodule

module nand2 (input wire i0, i1, output wire o);
   wire t;
   and2 and2_0 (i0, i1, t);
   invert invert_0 (t, o);
endmodule

module nor2 (input wire i0, i1, output wire o);
   wire t;
   or2 or2_0 (i0, i1, t);
   invert invert_0 (t, o);
endmodule

module xnor2 (input wire i0, i1, output wire o);
   wire t;
   xor2 xor2_0 (i0, i1, t);
   invert invert_0 (t, o);
endmodule

module and3 (input wire i0, i1, i2, output wire o);
```

```
  wire t;
  and2 and2_0 (i0, i1, t);
  and2 and2_1 (i2, t, o);
endmodule

module or3 (input wire i0, i1, i2, output wire o);
  wire t;
  or2 or2_0 (i0, i1, t);
  or2 or2_1 (i2, t, o);
endmodule

module nor3 (input wire i0, i1, i2, output wire o);
  wire t;
  or2 or2_0 (i0, i1, t);
  nor2 nor2_0 (i2, t, o);
endmodule

module nand3 (input wire i0, i1, i2, output wire o);
  wire t;
  and2 and2_0 (i0, i1, t);
  nand2 nand2_1 (i2, t, o);
endmodule

module xor3 (input wire i0, i1, i2, output wire o);
  wire t;
  xor2 xor2_0 (i0, i1, t);
  xor2 xor2_1 (i2, t, o);
endmodule

module xnor3 (input wire i0, i1, i2, output wire o);
  wire t;
  xor2 xor2_0 (i0, i1, t);
  xnor2 xnor2_0 (i2, t, o);
endmodule

module mux2 (input wire i0, i1, j, output wire o);
 assign o = (j==0)?i0:i1;
endmodule
```

```
module mux4 (input wire [0:3] i, input wire j1, j0, output wire o);
 wire t0, t1;
 mux2 mux2_0 (i[0], i[1], j1, t0);
 mux2 mux2_1 (i[2], i[3], j1, t1);
mux2 mux2_2 (t0, t1, j0, o);
endmodule

module mux8 (input wire [0:7] i, input wire j2, j1, j0, output wire o);
 wire t0, t1;
 mux4 mux4_0 (i[0:3], j2, j1, t0);
 mux4 mux4_1 (i[4:7], j2, j1, t1);
mux2 mux2_0 (t0, t1, j0, o);
endmodule

module demux2 (input wire i, j, output wire o0, o1);
 assign o0 = (j==0)?i:1'b0;
 assign o1 = (j==1)?i:1'b0;
endmodule

module demux4 (input wire i, j1, j0, output wire [0:3] o);
 wire t0, t1;
 demux2 demux2_0 (i, j1, t0, t1);
 demux2 demux2_1 (t0, j0, o[0], o[1]);
 demux2 demux2_2 (t1, j0, o[2], o[3]);
endmodule

module demux8 (input wire i, j2, j1, j0, output wire [0:7] o);
 wire t0, t1;
 demux2 demux2_0 (i, j2, t0, t1);
demux4 demux4_0 (t0, j1, j0, o[0:3]);
demux4 demux4_1 (t1, j1, j0, o[4:7]);
endmodule

module df (input wire clk, in, output wire out);
 reg df_out;
 always@(posedge clk) df_out <= in;
 assign out = df_out;
endmodule
```

```
module dfr (input wire clk, reset, in, output wire out);
  wire reset_, df_in;
  invert invert_0 (reset, reset_);
and2 and2_0 (in, reset_, df_in);
df df_0 (clk, df_in, out);
endmodule

module dfrl (input wire clk, reset, load, in, output wire out);
  wire _in;
  mux2 mux2_0(out, in, load, _in);
dfr dfr_1(clk, reset, _in, out);
endmodule
```

Main file code:

```
module dfrl_16(input wire clk,reset,load,input wire [0:15]in,output wire
[0:15]out);
dfrl dfrl_0(clk,reset,load,in[0],out[0]);
dfrl dfrl_1(clk,reset,load,in[1],out[1]);
dfrl dfrl_2(clk,reset,load,in[2],out[2]);
dfrl dfrl_3(clk,reset,load,in[3],out[3]);
dfrl dfrl_4(clk,reset,load,in[4],out[4]);
dfrl dfrl_5(clk,reset,load,in[5],out[5]);
dfrl dfrl_6(clk,reset,load,in[6],out[6]);
dfrl dfrl_7(clk,reset,load,in[7],out[7]);
dfrl dfrl_8(clk,reset,load,in[8],out[8]);
dfrl dfrl_9(clk,reset,load,in[9],out[9]);
dfrl dfrl_10(clk,reset,load,in[10],out[10]);
dfrl dfrl_11(clk,reset,load,in[11],out[11]);
dfrl dfrl_12(clk,reset,load,in[12],out[12]);
dfrl dfrl_13(clk,reset,load,in[13],out[13]);
dfrl dfrl_14(clk,reset,load,in[14],out[14]);
dfrl dfrl_15(clk,reset,load,in[15],out[15]);
endmodule

module mux2_16(input wire [15:0]i0,i1,input wire j,output wire [15:0]o);
mux2 mux2_0(i0[0],i1[0],j,o[0]);
mux2 mux2_1(i0[1],i1[1],j,o[1]);
mux2 mux2_2(i0[2],i1[2],j,o[2]);
```

```
mux2 mux2_3(i0[3],i1[3],j,o[3]);
mux2 mux2_4(i0[4],i1[4],j,o[4]);
mux2 mux2_5(i0[5],i1[5],j,o[5]);
mux2 mux2_6(i0[6],i1[6],j,o[6]);
mux2 mux2_7(i0[7],i1[7],j,o[7]);
mux2 mux2_8(i0[8],i1[8],j,o[8]);
mux2 mux2_9(i0[9],i1[9],j,o[9]);
mux2 mux2_10(i0[10],i1[10],j,o[10]);
mux2 mux2_11(i0[11],i1[11],j,o[11]);
mux2 mux2_12(i0[12],i1[12],j,o[12]);
mux2 mux2_13(i0[13],i1[13],j,o[13]);
mux2 mux2_14(i0[14],i1[14],j,o[14]);
mux2 mux2_15(i0[15],i1[15],j,o[15]);
endmodule

module mux8_16(input wire [0:15]i0,i1,i2,i3,i4,i5,i6,i7,input wire [0:2]j,output
wire [0:15]o);
mux8 mux8_0({i0[0],i1[0],i2[0],i3[0],i4[0],i5[0],i6[0],i7[0]},j[0],j[1],j[2],o[0]);
mux8 mux8_1({i0[1],i1[1],i2[1],i3[1],i4[1],i5[1],i6[1],i7[1]},j[0],j[1],j[2],o[1]);
mux8 mux8_2({i0[2],i1[2],i2[2],i3[2],i4[2],i5[2],i6[2],i7[2]},j[0],j[1],j[2],o[2]);
mux8 mux8_3({i0[3],i1[3],i2[3],i3[3],i4[3],i5[3],i6[3],i7[3]},j[0],j[1],j[2],o[3]);
mux8 mux8_4({i0[4],i1[4],i2[4],i3[4],i4[4],i5[4],i6[4],i7[4]},j[0],j[1],j[2],o[4]);
mux8 mux8_5({i0[5],i1[5],i2[5],i3[5],i4[5],i5[5],i6[5],i7[5]},j[0],j[1],j[2],o[5]);
mux8 mux8_6({i0[6],i1[6],i2[6],i3[6],i4[6],i5[6],i6[6],i7[6]},j[0],j[1],j[2],o[6]);
mux8 mux8_7({i0[7],i1[7],i2[7],i3[7],i4[7],i5[7],i6[7],i7[7]},j[0],j[1],j[2],o[7]);
mux8 mux8_8({i0[8],i1[8],i2[8],i3[8],i4[8],i5[8],i6[8],i7[8]},j[0],j[1],j[2],o[8]);
mux8 mux8_9({i0[9],i1[9],i2[9],i3[9],i4[9],i5[9],i6[9],i7[9]},j[0],j[1],j[2],o[9]);
mux8
mux8_10({i0[10],i1[10],i2[10],i3[10],i4[10],i5[10],i6[10],i7[10]},j[0],j[1],j[2],o[10]);
mux8
mux8_11({i0[11],i1[11],i2[11],i3[11],i4[11],i5[11],i6[11],i7[11]},j[0],j[1],j[2],o[11]);
mux8
mux8_12({i0[12],i1[12],i2[12],i3[12],i4[12],i5[12],i6[12],i7[12]},j[0],j[1],j[2],o[12]);
mux8
mux8_13({i0[13],i1[13],i2[13],i3[13],i4[13],i5[13],i6[13],i7[13]},j[0],j[1],j[2],o[13]);
mux8
mux8_14({i0[14],i1[14],i2[14],i3[14],i4[14],i5[14],i6[14],i7[14]},j[0],j[1],j[2],o[14]);
mux8
mux8_15({i0[15],i1[15],i2[15],i3[15],i4[15],i5[15],i6[15],i7[15]},j[0],j[1],j[2],o[15]);
```

```
endmodule

module reg_file(input wire clk,reset,wr,input wire
[0:2]rd_addr_a,rd_addr_b,wr_addr,input wire [0:15]d_in,output wire
[0:15]d_out_a,d_out_b);
wire [0:7]load;
wire [0:15]dout_0,dout_1,dout_2,dout_3,dout_4,dout_5,dout_6,dout_7;

dfrl_16 dfrl_16_0(clk,reset,load[0],d_in,dout_0);
dfrl_16 dfrl_16_1(clk,reset,load[1],d_in,dout_1);
dfrl_16 dfrl_16_2(clk,reset,load[2],d_in,dout_2);
dfrl_16 dfrl_16_3(clk,reset,load[3],d_in,dout_3);
dfrl_16 dfrl_16_4(clk,reset,load[4],d_in,dout_4);
dfrl_16 dfrl_16_5(clk,reset,load[5],d_in,dout_5);
dfrl_16 dfrl_16_6(clk,reset,load[6],d_in,dout_6);
dfrl_16 dfrl_16_7(clk,reset,load[7],d_in,dout_7);
demux8 demux8_0(wr,wr_addr[2],wr_addr[1],wr_addr[0],load);
mux8_16
mux8_16_9(dout_0,dout_1,dout_2,dout_3,dout_4,dout_5,dout_6,dout_7,rd_addr
_a,d_out_a);
mux8_16
mux8_16_10(dout_0,dout_1,dout_2,dout_3,dout_4,dout_5,dout_6,dout_7,rd_ad
dr_b,d_out_b);
endmodule
```

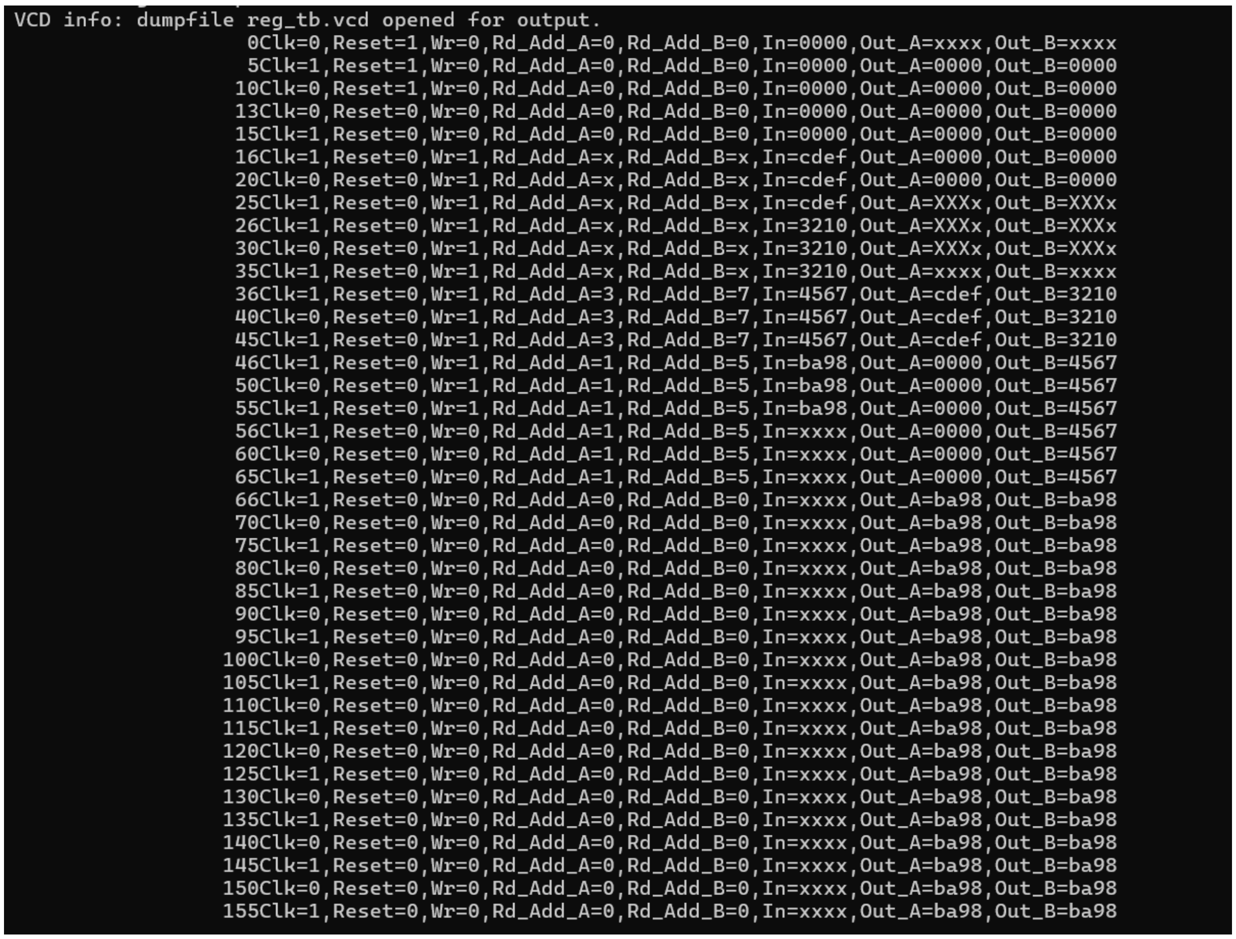
Testbench file code:

```
`timescale 1 ns / 100 ps
`define TESTVECS 6

module reg_tb;
reg clk,reset,wr;
reg [2:0]rd_addr_a,rd_addr_b,wr_addr;
reg [15:0]d_in;
wire [15:0]d_out_a,d_out_b;
reg [25:0]test_vecs [0:(`TESTVECS-1)];
integer i;
initial
begin
```

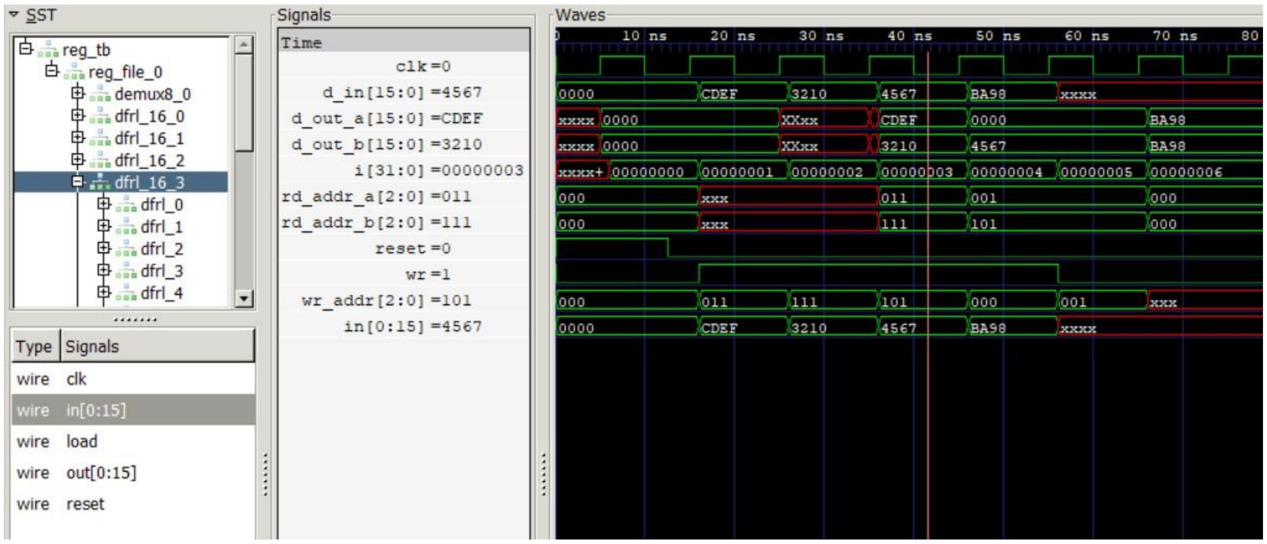
```
$dumpfile("reg_tb.vcd");
$dumpvars(0,reg_tb);
end
initial begin reset = 1'b1; #12.5 reset = 1'b0; end
initial clk = 1'b0; always #5 clk =~ clk;
initial begin
test_vecs[0][25] = 1'b1; test_vecs[0][24:22] = 3'ox; test_vecs[0][21:19] = 3'ox;
test_vecs[0][18:16] = 3'h3; test_vecs[0][15:0] = 16'hcdef;
test_vecs[1][25] = 1'b1; test_vecs[1][24:22] = 3'ox; test_vecs[1][21:19] = 3'ox;
test_vecs[1][18:16] = 3'o7; test_vecs[1][15:0] = 16'h3210;
test_vecs[2][25] = 1'b1; test_vecs[2][24:22] = 3'o3; test_vecs[2][21:19] = 3'o7;
test_vecs[2][18:16] = 3'o5; test_vecs[2][15:0] = 16'h4567;
test_vecs[3][25] = 1'b1; test_vecs[3][24:22] = 3'o1; test_vecs[3][21:19] = 3'o5;
test_vecs[3][18:16] = 3'o0; test_vecs[3][15:0] = 16'hba98;
test_vecs[4][25] = 1'b0; test_vecs[4][24:22] = 3'o1; test_vecs[4][21:19] = 3'o5;
test_vecs[4][18:16] = 3'o1; test_vecs[4][15:0] = 16'hxxxx;
test_vecs[5][25] = 1'b0; test_vecs[5][24:22] = 3'o0; test_vecs[5][21:19] = 3'o0;
test_vecs[5][18:16] = 3'ox; test_vecs[5][15:0] = 16'hxxxx;
end
initial {wr, rd_addr_a, rd_addr_b, wr_addr, d_in} = 0;
reg_file reg_file_0 (clk, reset, wr, rd_addr_a, rd_addr_b, wr_addr, d_in, d_out_a,
d_out_b);
initial
begin
$monitor($time,"Clk=%b,Reset=%b,Wr=%h,Rd_Add_A=%h,Rd_Add_B=%h,In=%h,
Out_A=%h,Out_B=%h",clk,reset,wr,rd_addr_a,rd_addr_b,d_in,d_out_a,d_out_b);
end
initial begin
#6 for(i=0;i<`TESTVECS;i=i+1)
begin #10 {wr, rd_addr_a, rd_addr_b, wr_addr, d_in}=test_vecs[i]; end
#100 $finish;
end
endmodule
```

Verilog VVP Output Screenshot:



```
VCD info: dumpfile reg_tb.vcd opened for output.
                   0Clk=0,Reset=1,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=0000,Out_A=xxxx,Out_B=xxxx
                   5Clk=1,Reset=1,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=0000,Out_A=0000,Out_B=0000
                  10Clk=0,Reset=1,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=0000,Out_A=0000,Out_B=0000
                  13Clk=0,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=0000,Out_A=0000,Out_B=0000
                  15Clk=1,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=0000,Out_A=0000,Out_B=0000
                  16Clk=1,Reset=0,Wr=1,Rd_Add_A=x,Rd_Add_B=x,In=cdef,Out_A=0000,Out_B=0000
                  20Clk=0,Reset=0,Wr=1,Rd_Add_A=x,Rd_Add_B=x,In=cdef,Out_A=0000,Out_B=0000
                  25Clk=1,Reset=0,Wr=1,Rd_Add_A=x,Rd_Add_B=x,In=cdef,Out_A=XXXx,Out_B=XXXx
                  26Clk=1,Reset=0,Wr=1,Rd_Add_A=x,Rd_Add_B=x,In=3210,Out_A=XXXx,Out_B=XXXx
                  30Clk=0,Reset=0,Wr=1,Rd_Add_A=x,Rd_Add_B=x,In=3210,Out_A=XXXx,Out_B=XXXx
                  35Clk=1,Reset=0,Wr=1,Rd_Add_A=x,Rd_Add_B=x,In=3210,Out_A=xxxx,Out_B=xxxx
                  36Clk=1,Reset=0,Wr=1,Rd_Add_A=3,Rd_Add_B=7,In=4567,Out_A=cdef,Out_B=3210
                  40Clk=0,Reset=0,Wr=1,Rd_Add_A=3,Rd_Add_B=7,In=4567,Out_A=cdef,Out_B=3210
                  45Clk=1,Reset=0,Wr=1,Rd_Add_A=3,Rd_Add_B=7,In=4567,Out_A=cdef,Out_B=3210
                  46Clk=1,Reset=0,Wr=1,Rd_Add_A=1,Rd_Add_B=5,In=ba98,Out_A=0000,Out_B=4567
                  50Clk=0,Reset=0,Wr=1,Rd_Add_A=1,Rd_Add_B=5,In=ba98,Out_A=0000,Out_B=4567
                  55Clk=1,Reset=0,Wr=1,Rd_Add_A=1,Rd_Add_B=5,In=ba98,Out_A=0000,Out_B=4567
                  56Clk=1,Reset=0,Wr=0,Rd_Add_A=1,Rd_Add_B=5,In=xxxx,Out_A=0000,Out_B=4567
                  60Clk=0,Reset=0,Wr=0,Rd_Add_A=1,Rd_Add_B=5,In=xxxx,Out_A=0000,Out_B=4567
                  65Clk=1,Reset=0,Wr=0,Rd_Add_A=1,Rd_Add_B=5,In=xxxx,Out_A=0000,Out_B=4567
                  66Clk=1,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                  70Clk=0,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                  75Clk=1,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                  80Clk=0,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                  85Clk=1,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                  90Clk=0,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                  95Clk=1,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                 100Clk=0,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                 105Clk=1,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                 110Clk=0,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                 115Clk=1,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                 120Clk=0,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                 125Clk=1,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                 130Clk=0,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                 135Clk=1,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                 140Clk=0,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                 145Clk=1,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                 150Clk=0,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
                 155Clk=1,Reset=0,Wr=0,Rd_Add_A=0,Rd_Add_B=0,In=xxxx,Out_A=ba98,Out_B=ba98
```

GTKWAVE Screenshot:



```
reg_tb
  reg_file_0
    demux8_0
    dfrl_16_0
    dfrl_16_1
    dfrl_16_2
    dfrl_16_3
      dfrl_0
      dfrl_1
      dfrl_2
      dfrl_3
      dfrl_4
```

Type	Signals
wire	clk
wire	in[0:15]
wire	load
wire	out[0:15]
wire	reset

```
Time
clk =0
d_in[15:0] =4567
d_out_a[15:0] =CDEF
d_out_b[15:0] =3210
i[31:0] =00000003
rd_addr_a[2:0] =011
rd_addr_b[2:0] =111
reset =0
wr =1
wr_addr[2:0] =101
in[0:15] =4567
```

SST
Signals
Waves
10 ns
20 ns
30 ns
40 ns
50 ns
60 ns
70 ns
80
0000 CDEF 3210 4567 BA98 xxxx
xxxx 0000 XXxx CDEF 0000 BA98
xxxx 0000 XXxx 3210 4567 BA98
xxxx+ 00000000 00000001 00000002 00000003 00000004 00000005 00000006
000 xxx 011 001 000
000 xxx 111 101 000
000 011 111 101 000 001 xxx
0000 CDEF 3210 4567 BA98 xxxx

Disclaimer:

- The programs and output submitted is duly written, verified and executed by me.
- I have not copied from any of my peers nor from the external resource such as internet.
- If found plagiarized, I will abide with the disciplinary action of the University.

Signature: ABHISHEK

P SRN:

PES2UG23AM002

Section: A