



به نام خدا



دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

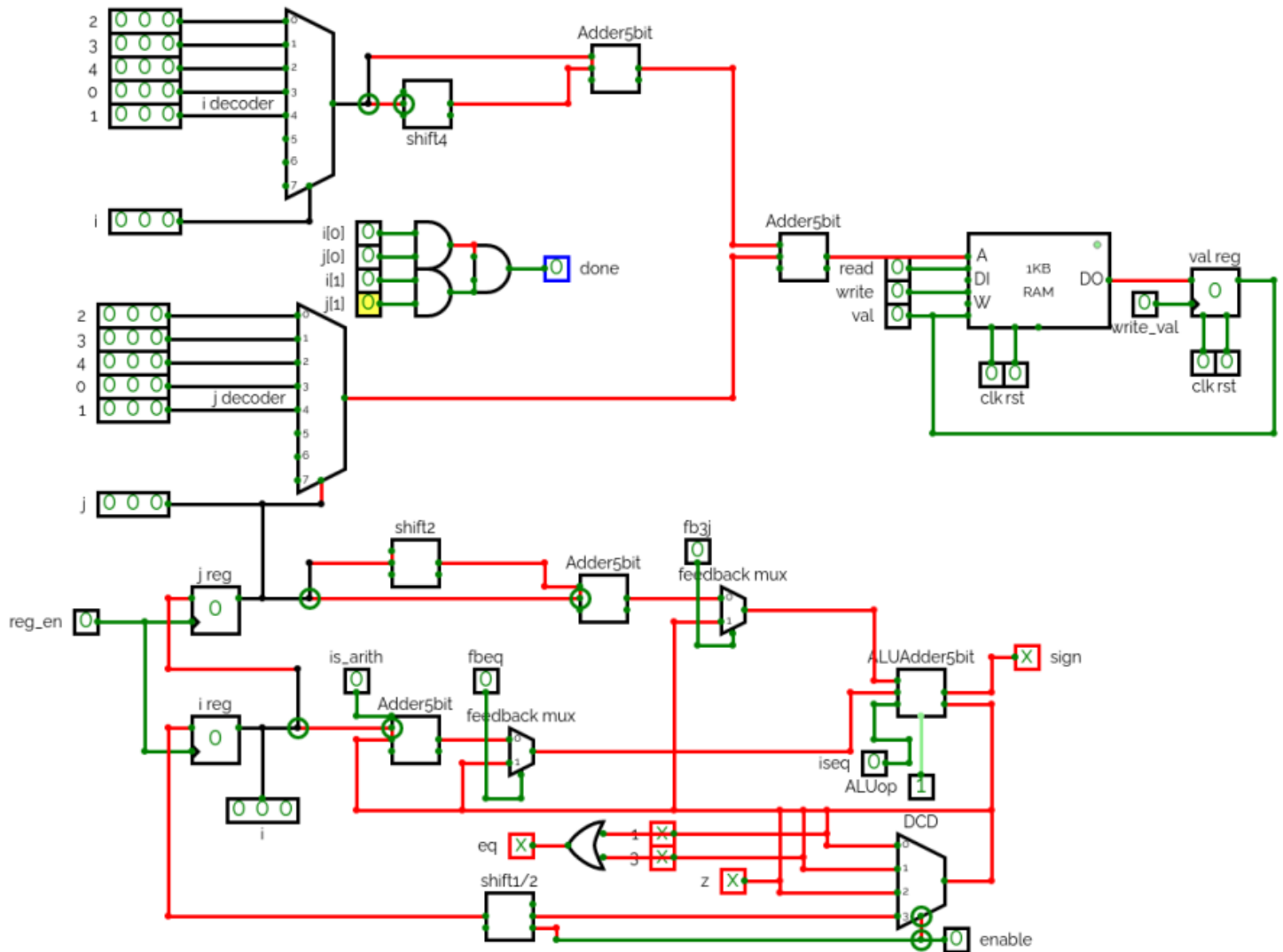
طراحی کامپیوتری سیستم دیجیتال

تمرین دستی 1 فاز 2

نام و نام خانوادگی	آرین سلطانی-ریحانه احمدپور
شماره دانشجویی	810198494-810198558
تاریخ ارسال گزارش	1401/1/18 - پنجشنبه 18 فروردین

فهرست گزارش سوالات

- سوال 1 - دیتابث 2
- سوال 2 - کنترلر 5
- سوال 3 - نتایج خروجی 10
- سوال 4 - 10



```
`timescale 1ns/1ns
module Datapath (
    clk,
    rst,
    IJen,
    ALUOp,
    read,
    write,
    initLine,
    line,
    writeVal,
    IJregen,
    fb3j,
    fbeq,
    isArith,
    enable,
    waitCalNexti,
```

```

writeMemReg,
ldTillPositive,
update,//enable for updating i,j after being checked and update "i"

sign3j,
signeq,
done,
sign,
eq,
mem,
firstread,
ok
);
parameter size = 5;
parameter memsize = 25;

parameter initValIJ = 3;

input clk, rst, firstread;
input IJen, ALUop, read, write, initLine, writeMemReg;
input writeVal, IJregen, fbeq, fb3j, isArith, enable, update, waitCalNexti,
ldTillPositive, ok;
input [memsize-1:0]line;
output [24:0]mem;

output sign3j, signeq, done, sign, eq;

wire [2:0]i;
wire [2:0]j;
wire [4:0]iMult4;
wire [2:0]iReg;
wire [2:0]jReg;
wire [4:0]iMult5;
wire [4:0]memIdx;
wire [4:0]iMult2;
wire [4:0]iMult3;
wire [4:0]lastIndex;
wire [4:0]iNextMult2;
wire [4:0]iNextPos;
wire [4:0]iNextPosAdd5;
wire [2:0]iAtLast;
wire [2:0]convertedI;
wire [2:0]convertedJ;
wire [4:0]memIdxOut;
wire [4:0]memInp;

wire newVal;

```

```

wire regVal;

IJMux newI(jReg, convertedI);
IJMux newJ(iReg, convertedJ);

Shifter #(5) multiplyI4(.data({2'b00, convertedI}), .coefficient({2'b01}),
.shifted(iMult4));

Adder #(5) multiplyI5(.i1({2'b00, convertedI}), .i2(iMult4), .a(iMult5));

Adder #(5) indexAdder(.i1(iMult5), .i2({2'b00, convertedJ}), .a(memIdx));

Register #(5) indexMemReg(.clk(clk), .rst(rst), .ld(writeMemReg), .inputData(memIdx),
.outputData(memIdxOut));

assign memInp = write ? memIdxOut: memIdx;

MemoryBlock #(5,25) MB(.clk(clk), .rst(rst), .init(initLine), .line(line),
.index(memInp), .val(regVal), .write(write), .read(read), .out(newVal), .mem(mem),
.firstread(firstread), .ok(ok));

Register #(1) valRegister(.clk(clk), .rst(rst), .ld(writeVal), .inputData(newVal),
.outputData(regVal));

Register #(3) JRegister(.clk(clk), .rst(rst), .ld(IJregen),
.inputData(j), .outputData(jReg));

Shifter #(5) multiplyI2(.data({2'b00, iReg}), .coefficient({2'b00}), .shifted(iMult2));

Adder #(5) multiplyI3(.i1(iMult2), .i2({2'b00, iReg}), .a(iMult3));

Register #(5) regTillPositive(.clk(clk), .rst(rst), .ld(ldTillPositive),
.inputData(iNextPosAdd5), .outputData(iNextPos));

assign sign = iNextPosAdd5[4];

assign iNextPosAdd5 = (waitCalNexti) ? (iNextPos + 5'b00101): iNextMult2;

Register #(5) registerLastIndex(.clk(clk), .rst(rst), .ld(ld_index), .inputData(memIdx),
.outputData(lastIndex));

Register #(3) IRegister(.clk(clk), .rst(rst), .ld(IJregen),
.inputData(i), .outputData(iReg));

assign iAtLast = (iNextPos == 5'b00000) ? 3'b000: (iNextPos == 5'b00001) ?
3'b011: (iNextPos == 5'b00010) ? 3'b001: (iNextPos == 5'b00011) ? 3'b100:
3'b010;

```

```

    Subtractor #(5) twiceNextI(.i1({2'b00, jReg}), .i2(iMult3), .en(isArith),
.out(iNextMult2));

```

```

    assign i = IJen ? 3'b011: update ? iAtLast : i;

```

```

    assign j = IJen ? 3'b011: update ? iReg : j;

```

```

    assign done = iReg[0] & iReg[1] & jReg[0] & jReg[1];

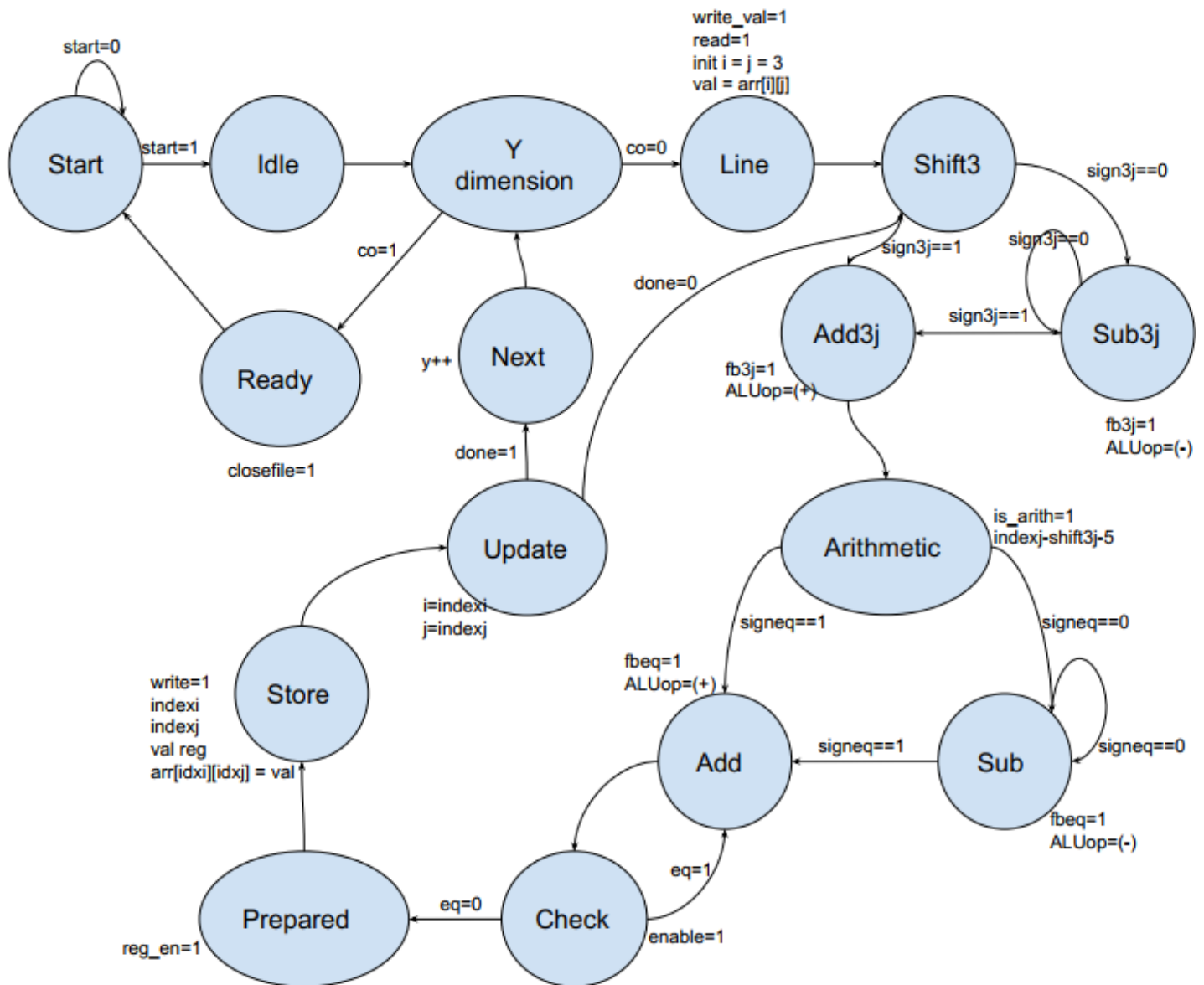
```

```

endmodule

```

کنترلر:



```

`timescale 1ns/1ns
module Controller (

```

```

    clk,
    rst,
    start,

    sign3j,
    signeq,
    done,
    sign,
    eq,

    waitCalNexti,
    writeMemReg,

    IJen,
    ALUop,
    read,
    write,
    initLine,
    line,
    writeVal,
    IJregen,
    fb3j,
    fbeq,
    isArith,
    enable,
    update,
    readLine,
    ldTillPositive,
    count,
    firstread,
    ok
);
parameter size = 5;
parameter memsize = 25;

input clk, rst;
input start, sign3j, signeq, done, sign, eq;
reg [5:0] count2;
input [5:0] count;

output reg IJen, ALUop, read, write, initLine, firstread;
output reg writeVal, IJregen, fbeq, fb3j, isArith, enable, update, waitCalNexti,
writeMemReg, ldTillPositive;
input [memsize-1:0] line;
output reg readLine, ok;

parameter [3:0]
    Start = 4'd0,

```

```

Idle = 4'd1,
Ydimension = 4'd2,
Line = 4'd3,
Shift3 = 4'd4,
Sub3j = 4'd5,
Add3j = 4'd6,
Arithmetic = 4'd7,
Sub = 4'd8,
Add = 4'd9,
Check = 4'd10,
Prepared = 4'd11,
Store = 4'd12,
Updater = 4'd13,
Next = 4'd14,
Ready = 4'd15;

reg enCount=0, loadCount=0, first = 0;
reg [5:0]loadInit = 0; // n times count = 5
wire coutCount;

reg [3:0] ps, ns;

Counter #6 cc(.clk(clk), .rst(rst), .en(enCount), .ld(loadCount), .initld(loadInit),
.co(coutCount));

always @(posedge clk, posedge rst) begin
    if(rst)begin
        ps <= Start;
        count2 <= 6'b000000;
    end
    else
        ps <= ns;
end

wire [5:0] sig;
wire tmp;
assign sig = ~(count + ~count2 + 6'b000001);
assign tmp = sig[5] & sig[4] & sig[3] & sig[2] & sig[1] & sig[0];

always @(ps, start, sign3j, signeq, done, sign, eq, tmp) begin
    case (ps)
        Start:      ns = start ? Idle : Start;
        Idle:       ns = Ydimension;
        Ydimension: ns = coutCount ? Ready : Line;
        Line:       ns = Store;
        Store:      ns = Shift3;
        Shift3:     ns = Sub3j;
    endcase
end

```

```

Sub3j:      ns = ~sign ? Add3j : Sub3j;
Add3j:      ns = Arithmetic;
Arithmetic: ns = Sub;
Sub:        ns = Add;
Add:        ns = Check;
Check:      ns = Updater;
Prepared:   ns = Next;
Updater:    ns = ~done ? Shift3 : Prepared;
Next:       ns = (~tmp) ? Next: Ydimension;
Ready:      ns = Start;
default: ns = Start;
endcase
end

always @(ps) begin
    {ok, firstread, ldTillPositive, writeMemReg, first, waitCalNexti, IJen, ALUop, read,
write, initline, writeVal, IJregen, fbeq, fb3j, isArith, enable, update, readLine, loadCount,
enCount} = 0;
    case (ps)
        Start:      begin
                        //nothing
                    end
        Idle:        begin
                        loadCount = 1'd1;
                    end
        Ydimension: begin
                        //nothing
                        readLine = 1'b1;
                    end
        Line:        begin
                        IJen = 1'b1;
                        initLine = 1'b1;
                        IJregen = 1'b1;
                        count2 = count2 + 1;
                    end
        Store:       begin
                        writeVal = 1'b1;
                        firstread = 1'b1;
                    end
        Shift3:      begin
                        writeMemReg = 1'b1;
                        ldTillPositive = 1'b1;
                    end
        Sub3j:       begin
                        waitCalNexti = 1'b1;
                        ldTillPositive = sign;
                        update = 1'b1;
                    end
    end
end

```



```

Add3j:      begin
    IJregen = 1'b1;

end
Arithmetic: begin
    isArith = 1'b1;
end
Sub:        begin
    read = 1'b1;
    fbeq = 1'b1;
    ALUop = 1'b0;
    ok = 1'b1;
end
Add:        begin
    fbeq = 1'b1;
    ALUop = 1'b1;
    write = 1'b1;
    ok = 1'b1;
end
Check:      begin
    enable = 1'b1;
    ok = 1'b1;
end
Prepared:   begin
    ok = 1'b1;
    enCount = 1'b1;
end
Store:      begin // fix algo
end
Updater:    begin

end
Next:       begin
    ok = 1'b1;
end
Ready:      begin
    //nothing
end
endcase
end
endmodule

```

نتایج خروجی:

```

F: > semester6 > CAD > hw > 1 > newsamples > 0.out
1 0000110100111011101110101
2 1101000011110001100101111
3 1001110011100010000100010
4 1000011010010010001010000
5 1001000101111111000010101
6 010111110111010000001010
7 100011101010000100000011
8 1100000001000100110111011
9 0000001110101101000110001
10 0010100100010000110010001
11 0010111010111110011110110
12 0001101110111010101010010
13 0101100100100101001101001
14 0000010011110101100110000
15 111011111010111100101000
16 111000000110011000110001
17 0111101000001000001011001
18 1101110111010010110100011
19 1001110001011001110111110
20 1101010011110110111010000
21 0001011001010000010011011
22 1001101100100110010011010
23 0011001100111111111110010
24 1010100110011011011100011
25 1110010010010100100100100
26 1100011000000011010011110
27 0111000001110000100010111
28 1101001110101001011101100
29 1111101100101111000110000
30 10000111101000011000001001
31 0010101010100001100101001
32 1000001011000110110110110
33 1111101011110110111111101
34 0110010110110100001100000
35 1010010000011110101001001
36 1011001001100011101011100
37 1011111011111100011101110
38 1010110001101001111111100
39 0010010100001110010110011
40 1101101010001010101100100
41 0101000010010011011000110
42 1001100011101111011010100
43 1111111010111110111101111
44 0001100100100010101010101
45 1101011000110000111101110
46 1011100110111100011110100
47 1100100111111111010000011
48 1000111110000011011101010
49 0100000100000101100111011
50 1110100101100010001010110
51 0100011001110111111010101
52 1011000011110011001011010
53 1111011100011110100111110
54 1011101001101101000111111
55 10111010100111010100001
56 1110000100011111100010100
57 1101110101111010011101101
58 1001101000011111000001110
59 0110000111011111000111001
60 0100001010011100101001110
61 1010111111010110001101000
62 0110110111110101000100000
63 0110100000110110101001101
64 0001111010100010000000010
65

1 0000110100111011101110101
2 1101000011110001100101111
3 1001110011100010000100010
4 1000011010010010001010000
5 1001000101111111000010101
6 010111110111010000001010
7 100011101010000100000011
8 1100000001000100110111011
9 0000001110101101000110001
10 0010100100010000110010001
11 0010111010111110011110110
12 0001101110111010101010010
13 0101100100100101001101001
14 0000010011110101100110000
15 111011111010111100101000
16 111000000110011000110001
17 0111101000001000001011001
18 1101110111010010110100011
19 1001110001011001110111110
20 1101010011110110111010000
21 0001011001010000010011011
22 1001101100100110010011010
23 0011001100111111111110010
24 1010100110011011011100011
25 1110010010010100100100100
26 1100011000000011010011110
27 0111000001110000100010111
28 1101001110101001011101100
29 1111101100101111000110000
30 10000111101000011000001001
31 0010101010100001100101001
32 1000001011000110110110110
33 1111101011110110111111101
34 0110010110110100001100000
35 1010010000011110101001001
36 1011001001100011101011100
37 1011111011111100011101110
38 1010110001101001111111100
39 0010010100001110010110011
40 1101101010001010101100100
41 0101000010010011011000110
42 1001100011101111011010100
43 1111111010111110111101111
44 0001100100100010101010101
45 1101011000110000111101110
46 1011100110111100011110100
47 1100100111111111010000011
48 1000111110000011011101010
49 0100000100000101100111011
50 1110100101100010001000110
51 0100011001110111111010101
52 1011000011110011001011010
53 1111011100011110100111110
54 1011101001101101000111111
55 10111010100111010100001
56 1110000100011111100010100
57 1101110101111010011101101
58 1001101000011111000001110
59 0110000111011111000111001
60 0100001010011100101001110
61 1010111111010110001101000
62 0110110111110101000100000
63 0110100000110110101001101
64 0001111010100010000000010
65

```

The image shows a Visual Studio Code editor window with two tabs open: `output_1.txt` and `output_2.txt`. Both tabs display a list of 65 lines of binary data, each line consisting of a sequence of 0s and 1s. The editor's interface includes a top menu bar with options like File, Edit, Selection, View, Go, Run, Terminal, and Help. A sidebar on the left contains various icons for file management and development tools. The status bar at the bottom indicates the current position (Ln 1, Col 1), spacing (Spaces: 4), encoding (UTF-8), and line ending (LF).

```

File Edit Selection View Go Run Terminal Help
output_2.txt ↔ 2.out - code - Visual Studio Code

output_0.txt ↔ 0.out  output_2.txt ↔ 2.out  Controller.v  2.out  Generate Simulate ↑ ↓ 🔍 ⌂

F:\semester6 > CAD > hw > 1 > newsamples > 2.out

1 0101100000001111011000010
2 0001100010110110000100110
3 1011001001000001111100111
4 1000101001011110000010100
5 0101111111010101001101110
6 1010010100111110111010000
7 0110010010100001101101010
8 0100000110110010101011000
9 0100110100011010101011110
10 1000000100011010111100100
11 0100010101110010110001111
12 1111110001111001000111000
13 1011110110111110110100101
14 1100110101110111000010010
15 1010101011100011010101100
16 0011101001101011001001101
17 1111011110000001101001100
18 0111100000111011001100001
19 1101110001011100111001110
20 1001000111011000001000000
21 1000001111011101110101110
22 0111010111100100101111111
23 0011000100110001010000110
24 1101100101101110111011100
25 1111110010111001011110100
26 0110001100010100111101100
27 1100111100101110110010100
28 1111000111011000111010010
29 1100101001101001001010110
30 1110101000100110011110110
31 1100010100010101001000000
32 1101010101000111010000010
33 0101001101000010011010111
34 0010101001100000001100101
35 1001000110110010111110101
36 1010110001101001000010000
37 1110011110110011110101101
38 1010100100011010100111110
39 0101011010001110010000010
40 1100111110001100100110000
41 0100011000010000010001110
42 0111100011111010011110000
43 101010110111110100000101
44 0000010100000100100010011
45 1100010000010100010100100
46 0110001110111010001010110
47 1001111111001001011110001
48 0100101010011001010010110
49 0001111110101100100100100
50 1111110010011110100001111
51 0011001000000000011110100
52 0000101010110011001111000
53 0010111111111001111010010
54 0000111011000010000001110
55 0101011001100010001000110
56 0011010100010110101011100
57 0011111000010010001100101
58 0110001000101000000001111
59 0001000001011010011111010
60 1101111000111111001010000
61 1000101011011110110100101
62 1110001110111001011111011
63 1100001101001000101100000
64 11010101110001000010011
65

1 0101100000001111011000010
2 0001100010110110000100110
3 1011001001000001111100111
4 1000101001011110000010100
5 0101111111010101001101110
6 1010010100111110111010000
7 0110010010100001101101010
8 0100000110110010101011000
9 0100110100011010101011110
10 1000000100011010111100100
11 0100010101110010110001111
12 1111110001111001000111000
13 1011110110111110110100101
14 1100110101110111000010010
15 1010101011100011010101100
16 0011101001101011001001101
17 1111011110000001101001100
18 0111100000111011001100001
19 1101110001011100111001110
20 1001000111011000001000000
21 1000001111011101110101110
22 0111010111110010010111111
23 0011000100110001010000110
24 1101100101101110111011100
25 1111110010111001011110100
26 0110001100010100111101100
27 1100111100101110110010100
28 1111000111011000111010010
29 1100101001101001001010110
30 1110101000100110011110110
31 1100010100010101001000000
32 11010101010000111010000010
33 0101001101000010011010111
34 0010101001100000001100101
35 1001000110110010111110101
36 1010110001101001000010000
37 1110011110110011110101101
38 1010100100011010100111110
39 0101011010001110010000010
40 1100111110001100100110000
41 0100011000010000010001110
42 0111100011111010011110000
43 101010110111110100000101
44 0000010100000100100010011
45 1100010000010100010100100
46 0110001110111010001010110
47 1001111111001001011110001
48 0100101010011001010010110
49 0001111110101100100100100
50 1111110010011110100001111
51 0011001000000000011110100
52 0000101010110011001111000
53 0010111111111001111010010
54 0000111011000010000001110
55 0101011001100010001000110
56 0011010100010110101011100
57 0011111000010010001100101
58 0110001000101000000001111
59 0001000001011010011111010
60 1101111000111111001010000
61 1000101011011110110100101
62 1110001110111001011111011
63 1100001101001000101100000
64 11010101110001000010011
65

```

```

Transcript
# ** Warning: (vsim-2685) [TFMPC] - Too few port connections for 'IRegister'. Expected 6, found 5.
# Time: 0 ns Iteration: 0 Instance: /TB/dp/IRegister File: ../src/hdl/Datapath.v Line: 103
# ** Warning: (vsim-3015) [PCDPC] - Port size (6) does not match connection size (5) for port 'out'. The port defin
# Time: 0 ns Iteration: 0 Instance: /TB/dp/twiceNextI File: ../src/hdl/Datapath.v Line: 110
# ** Note: (vsim-8716) Reusing existing debug database vsim.dbg.
VSIM 141> run -all
# GetModuleFileName: The specified module could not be found.
#
#
# ** Note: $stop : ./tb/TB.v(116)
# Time: 3849090 ns Iteration: 0 Instance: /TB
# Break in Module TB at ./tb/TB.v line 116
VSIM 142> quit -sim
# End time: 03:25:38 on Apr 08,2022, Elapsed time: 0:11:19
# Errors: 0, Warnings: 10
ModelSim> cd F:/semester6/CAD/hw/1/trunk/sim
# reading modelsim.ini

ModelSim> ]
<No Design Loaded> <No Context>

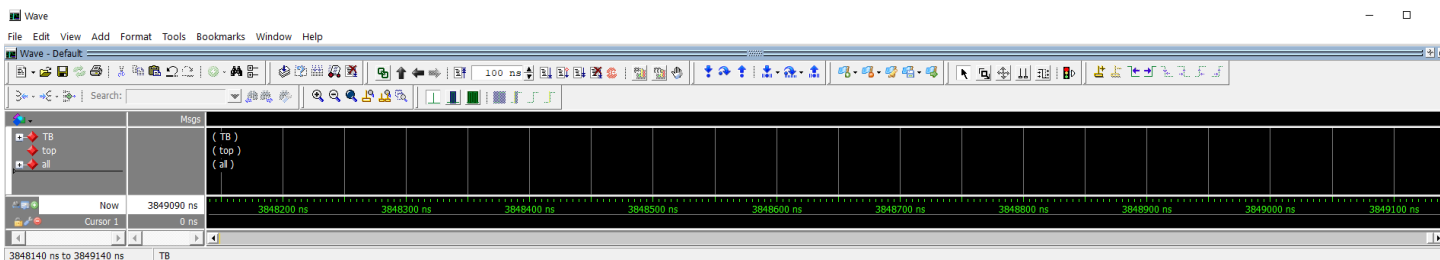
```

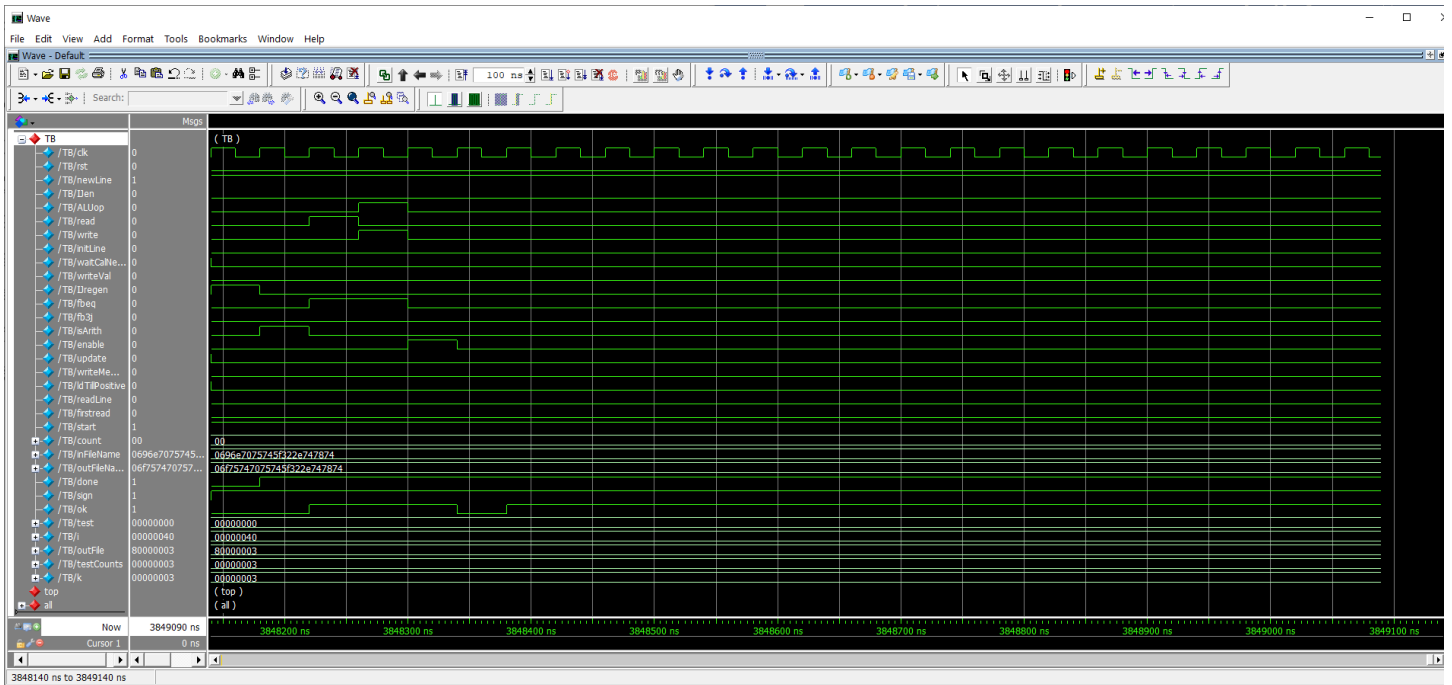
```

Transcript
# ** Warning: (vsim-2685) [TFMPC] - Too few port connections for 'IRegister'. Expected 6, found 5.
# Time: 0 ns Iteration: 0 Instance: /TB/dp/IRegister File: ../src/hdl/Datapath.v Line: 103
# ** Warning: (vsim-3015) [PCDPC] - Port size (6) does not match connection size (5) for port 'out'. The port defin
# Time: 0 ns Iteration: 0 Instance: /TB/dp/twiceNextI File: ../src/hdl/Datapath.v Line: 110
# ** Note: (vsim-8716) Reusing existing debug database vsim.dbg.
VSIM 141> run -all
# GetModuleFileName: The specified module could not be found.
#
#
# ** Note: $stop : ./tb/TB.v(116)
# Time: 3849090 ns Iteration: 0 Instance: /TB
# Break in Module TB at ./tb/TB.v line 116
do sim_top.tcl

ModelSim> do sim_top.tcl
<No Design Loaded> <No Context>

```





تست 0:

تست 1:

Volume (F:) > semester6 > CAD > hw > 1 > trunk > sim >				Search sim	
Name	Date modified	Type	Size		
file	4/8/2022 3:37 AM	File folder			
model	9/30/2018 7:06 PM	File folder			
tb	4/7/2022 3:41 PM	File folder			
work	4/8/2022 3:37 AM	File folder			
modelsim	4/8/2022 3:37 AM	Configuration setti...	11 KB		
sim_top.tcl	4/7/2022 3:43 PM	TCL File	2 KB		
vsim.dbg	4/8/2022 3:35 AM	DBG File	288 KB		
vsim.wlf	4/8/2022 3:37 AM	WLF File	0 KB		