```
module dp ram (clk,wr addr,rd addr,data in,data out,cs,we,re,oe);
parameter ADDR WIDTH =4;
parameter DATA WIDTH =8;
parameter DEPTH =16:
input clk, cs, we, re, oe;
input [ADDR WIDTH-1:0] rd addr;
input [ADDR WIDTH-1:0] wr addr;
input [DATA WIDTH-1:0] data in;
input [DATA WIDTH-1:0] data out;
reg [DATA WIDTH-1:0] tmp data; T
reg [DATA WIDTH-1:0] mem[DEPTH];
always @(posedge clk) begin
if (cs & we)
   mem[wr addr] <= data in;
end
always @(posedge clk) begin
if(cs & re)
   tmp data <=mem[rd addr];
end
assign data out = cs & oe & re ? tmp data : 'hz;
endmodule
```

```
dp_ram_test.v
  Open *
 include "dp ram.v"
nodule dp ram test;
parameter ADDR WIDTH =4;
parameter DATA WIDTH =8;
parameter DEPTH =16:
reg clk,cs,we,re,oe;
reg [ADDR WIDTH-1:0] wr addr;
reg [ADDR WIDTH-1:0] rd addr;
wire [DATA WIDTH-I:0] data out;
wire [DATA WIDTH-1:0] data in;
reg [DATA WIDTH-1:0] tb data;
ip ram inst(clk,wr addr,rd addr,data in,data out,cs,we,re,oe);
always #10 clk = ~clk:
assign data in= we ? tb data : 'hz;
initial begin
smonitor("Time=%0t data in=%0d mem=[%0d] <==> data out=%0d mem=[%0d]",$time,data in,wr addr,data out,rd addr);
end
initial begin
{clk,cs,we,re,wr addr,rd addr,tb data,oe} <=0;
#18
#20 we =1; cs=1; wr addr =1; tb data=10;
#20 we =1; cs=1; wr addr =2; tb data=20;
#20 we =1; cs=1; wr addr =3; tb data=30;
#20 we =1; cs=1; wr addr =4; tb data=40;
120 we =1; cs=1; wr addr =5; tb data=50;
100 we =1. rs=1. wr addr =6. th data=60.
```

Applications Places Text Editor

```
Open -
                                                                                   dp_ram_test.v
                                                                                     -/verilog/ram
#20 we =1; cs=1; wr addr =3; tb data=30;
#20 we =1; cs=1; wr addr =4; tb data=40;
#20 we =1; cs=1; wr addr =5; tb data=50;
#20 we =1; cs=1; wr addr =6; tb data=60;
#20 we =1; cs=1; wr addr =7; tb data=70;
#28 we =1; cs=1; wr addr =8; tb data=80;
#20 we =1; cs=1; wr addr =9; tb data=90;
#20 we =1; cs=1; wr addr =10; tb data=100;
#20 we =1; cs=1; wr addr =11; tb data=110;
#20 we =1; cs=1; wr addr =12; tb data=120;
#20 we =1; cs=1; wr addr =13; tb data=130;
#20 we =1; cs=1; wr addr =14; tb data=140;
#20 we =1; cs=1; wr addr =15; tb data=150;
#20 we =1; cs=1; wr addr =5; tb data=5;
#20
#200:
end
initial begin
#50
#20 re=1; cs=1; oe=1;
#20 rd addr=1;
#20 rd addr=2;
#20 rd addr=3;
#28 rd addr=4;
#20 rd addr=5;
#20 rd addr=6;
#20 rd addr=7;
#20 rd addr=8;
#20 rd addr=9:
#20 rd addr=10:
470 rd addr-33.
```

Applications Places Text Editor

```
dp_ram_test.v
  Open -
            图
#20 WE -1; CS-1; WI duur -14; LD udld-140;
#20 we =1; cs=1; wr addr =15; tb data=150;
#20 we =1; cs=1; wr addr =5; tb data=5;
#20
#200;
end
initial begin
#50
#20 re=1; cs=1; oe=1;
#20 rd addr=1:
#20 rd addr=2;
#20 rd addr=3;
#20 rd addr=4:
#20 rd addr=5;
#20 rd addr=6;
#20 rd addr=7:
#20 rd addr=8;
#20 rd addr=9;
#20 rd addr=10;
#28 rd addr=11:
#20 rd addr=12;
#20 rd addr=13;
#20 rd addr=14;
#28 rd addr=15;
#40 rd addr=0;
#40
$finish;
end
endmodule
```

Applications Places

Text Editor

sp\_ram.v

```
include "sp ram.v"
module sp ram test;
parameter ADDR WIDTH =4;
parameter DATA WIDTH =8;
parameter DEPTH =16;
reg clk, cs, we, oe;
reg [ADDR WIDTH-1:0] addr;
wire [DATA WIDTH-1:0] data;
reg [DATA WIDTH-1:0] tb data;
sp ram inst(clk,addr,data,cs,we,oe);
always #10 clk = ~clk;
assign data= !oe ? tb data : 'hz;
initial begin
$monitor("Time=%0t data=%0d mem=[%0d]",$time,data,addr);
{clk,cs,we,addr,tb data,oe} <=0;
#18
$display("Time=>Writing in Memory");
#20 we =1; cs=1; addr =1; tb data=10;
#20 we =1; cs=1; addr =2; tb data=20;
#20 we =1; cs=1; addr =3; tb data=30;
#20 we =1; cs=1; addr =4; tb data=40;
#20 we =1; cs=1; addr =5; tb data=50;
#20 we =0; cs=1; oe=1;
```

Sdisnlav("Timp=>\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*).

```
Applications Places Text Editor
  Open -
sp ram institctk, auur, uata, cs, we, ue);
always #10 clk = ~clk;
assign data= !oe ? tb data : 'hz;
initial begin
$monitor("Time=%0t data=%0d mem=[%0d]",$time,data,addr);
{clk,cs,we,addr,tb data,oe} <=0;
#10
$display("Time=>Writing in Memory");
#20 we =1; cs=1; addr =1; tb data=10;
#20 we =1; cs=1; addr =2; tb data=20;
#20 we =1; cs=1; addr =3; tb data=30;
#20 we =1; cs=1; addr =4; tb data=40;
#20 we =1; cs=1; addr =5; tb data=50;
#20 we =0; cs=1; oe=1;
$display("Time=>********************):
$display("Time=>Reading from Memory");
#20 addr=1;
#20 addr=2;
#20 addr=3;
#20 addr=4;
#20 addr=5;
#40
$finish;
```

end

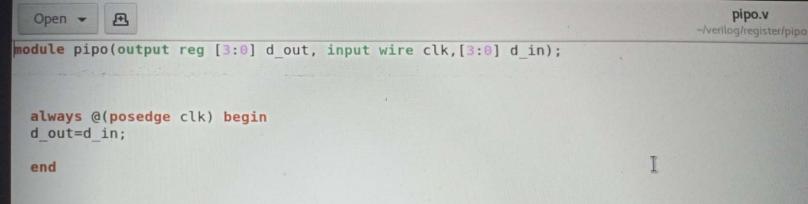
endmodule

sp\_ram\_test.v

end

```
include "mealy practice.v"
module mealy practice test;
reg clk, reset, in;
wire out:
always #10 clk = ~ clk;
mealy practice inst(.clk(clk), .reset(reset), .in(in), .out(out));
initial begin
$monitor("Time=%0t in=%0b out=%0b",$time,in,out);
clk=0;
#10 reset =1;
#20 reset =0;
#20 in=1;
#20 in=1:
#28 in=1;
#20 in=1;
#20 in=0;
#20 in=0;
#20 in=0;
#20 in=1;
#20 in=1;
#20 in=1;
#28 in=1;
#20 in=0:
#20 in=1:
#20 in=1;
#20 in=1;
#20 in=1;
#20 in=0;
#20
Sfinish;
```

```
include "pipo.v"
module pipo test:
reg clk;
reg [3:0] d in;
wire [3:0] d out;
pipo inst(.clk(clk), .d in(d in), .d out(d out));
always #10 clk = ~clk;
initial begin
 $monitor("Time=%0t, d in=%b, d out=%b",$time,d in,d out);
 clk=0;
 #10 d in=4'b1010;
 #20
 #20 d in=4'b1111;
 #20
 #20 d in=4'b0000;
 #20
 #28 d in=4'b1111;
 #28
#28
$finish;
end
endmodule
```



```
module piso(output reg d_out, input wire clk,load,[3:0] d_in);
```

```
reg [3:0] tmp;

always @(posedge clk) begin
  if(load)
   tmp <=d_in;
  else
   tmp <= {1'b0,tmp[3:1]};</pre>
```

```
end
assign d_out=tmp[0];
```

```
Open - 🖭
module piso test;
reg clk;
reg [3:0] d in, load;
wire d out;
piso inst(.clk(clk), .d in(d in), .d out(d out), .load(load));
 always #10 clk = -clk;
initial begin
 Smonitor("Time="0t, d in="b, d out="b ", Stime, d in, d out);
 clk=0;
 #10 load=1'b1;d in=4'b1010;
 #10 load=1'b0;
#98 load=1'b1;d in=4'b1111;
#10 load=1'b0;
#70 load=1'b1;d in=04'b0000;
#10 load=1'b0;
#78 load=1'b1;d in=4'b1111;
#10 load=1'b8;
#80
Sfinish;
endmodule
```

piso\_test.v

Applications Places Text Editor

```
Open -
module sipo(output reg [3:0] d out, input wire clk,d in);
  always @(posedge clk) begin
  // d out= {d out[2:0], d in}; //left shift
 d out= {d in,d out[3:1]}; //right shift
 end
endmodule
```

```
d_out= {d_in,d_out[3:1]}; //right shift
end
endmodule
```

```
include "sipo.v"
module sipo test;
reg clk,d in;
wire [3:0]d out;
sipo inst(.clk(clk), .d in(d in), .d out(d out));
always #10 clk = ~clk;
initial begin
$monitor("Time=%0t, d in=%b d out=%b" ,$time,d in,d out,);
 clk=0;
 #10 d in=1'b1;
 #20 d in=1'b0;
 #20 d in=1'bl;
 #20 d in=1'b0;
 #20
```

\$finish;

end

```
Open -
nodule siso(output reg d out, input wire clk,d in);
  reg [3:0] tmp;
always @(posedge clk) begin
 tmp= {tmp[2:0],d in}; //right shift
// tmp= {d in,tmp[3:1]}; //left shift
d out=tmp[3];
```

siso.v

~/verilog/register

end endmodule

siso\_test.v

\$finish; end

nodule comparator(input[3:0]a,b, output lt,gt,eq);

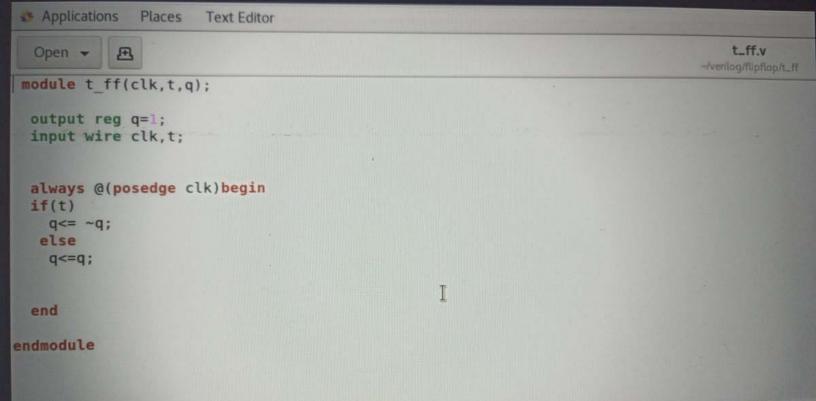
```
always @* begin

if(a>b) begin

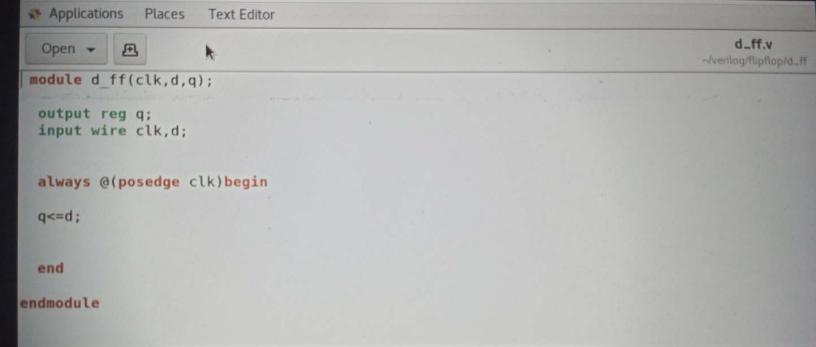
gt=1; lt=
end else if (a<b) begin
 a_lt_b = 1;
end else begin
a_eq_b = 1;</pre>
```

I

end end



t\_ff\_test.v /venlog/flipflop/t\_ff



d\_ff\_test.v

endmodule

end

\$finish;

```
module jk_ff(clk,j,k,q);
```

```
output reg q;
input wire clk,j,k;
```

## always @(posedge clk)begin

```
case({j,k})
  2'b00: q<=q;
  2'b01: q<=1'b0;
  2'b10: q<=1'b1;
  2'b11: q<= ~q;
  default : q<=q;</pre>
```

endcase

end

endmodule

9

```
include "jk ff.v"
module jk ff test;
reg clk, j, k;
wire q:
jk ff inst(.clk(clk), .j(j), .k(k), .q(q));
always #10 clk = ~clk;
initial begin
  $monitor("Time=%0t, j=%b k=%b q=%b" ,$time,j,k,q);
 clk=0;
  #10 j=0; k=0;
 #20 j=0; k=1;
 #20 j=1; k=0;
 #20 j=1; k=1;
 #20 j=0; k=0;
 #20
 $finish;
 end
 endmodule
```