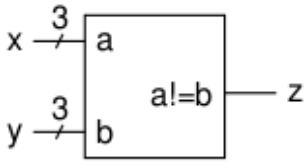
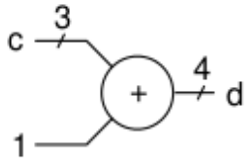
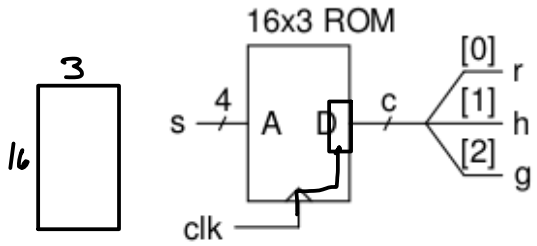


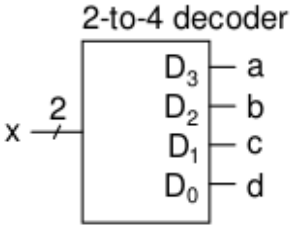

```
assign  z = x != y;
```




```
logic [3:0] d
      [2:0] c
d = c + 1'd1;
```




```
always_ff @ (posedge clk)
  c <= rom[s];

r = c[0];
h = c[1];
g = c[2];
```



logic [3:0] D

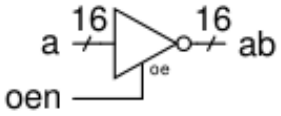
D	D_3			D_0
00	0	0	0	1
01	0	0	1	0
10	0	1	0	0
11	1	0	0	0

```
a='0; d='0;
b='0
c='0
case ( x )
2'b00: d='1 ;
2'b01 : c='1 ;
2'b10 : b='1 ;
2'b11 : a='1 ;
endcase
```

D = 4'b0001;

a = D[3]
b = D[2]

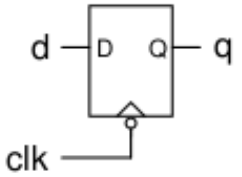
```
a=(b=(c=(d=0)));

{a,b,c,d} = 4'b0001;

{a,b,c,d} = 1'b1<<x;

logic [3:0] D [1:0] = '{4'b0001, ...}

if(x==2'b00)
  {a,b      }

{a,b,c,d} = D[x];
```




```
ab = 16'bz;
```

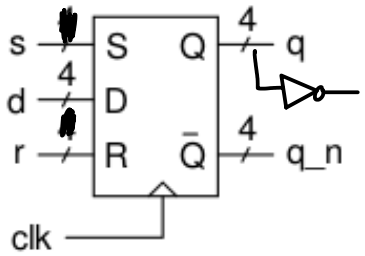
0 1 z X ! ~

```
ab = oen ? ~a : 'z ;
```




```
always_ff @(negedge clk)
    q <= d;
```

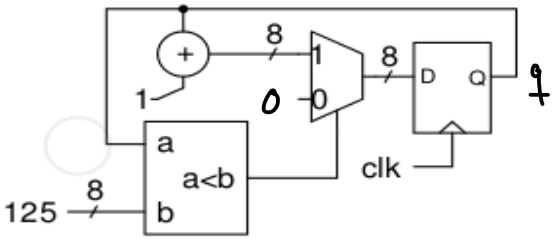
'1 '0 'z 'x



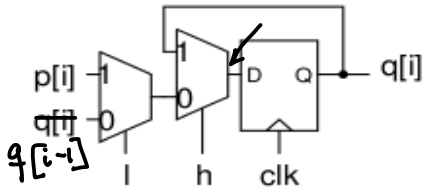

```
always_ff @ (posedge clk)
    if ( s )
        q <= '1;
    else if (r)
        q <= '0;
    else
        q <= d;

assign q_n = ~q; ???
```

s

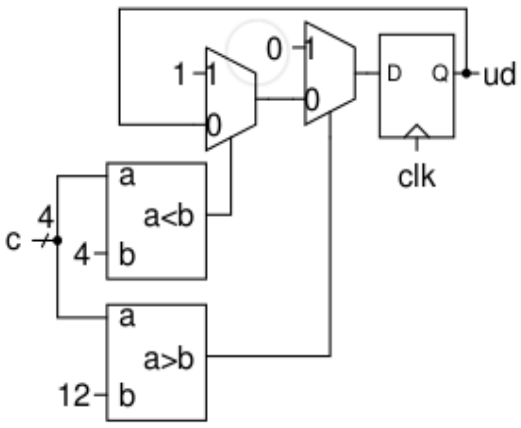
```
always_comb begin
    q_next = s ? '1 : r ? '0 : d;
    q_n_next = ~q_next;
end

always_ff @(posedge clk)
    q <= q_next;
    q_n <= q_n_next;
end
```



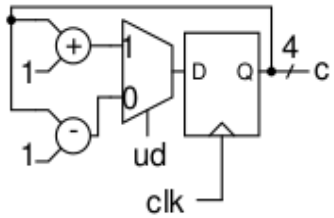

```
always_comb
  q_next = q<125 ? q+1'b1 : '0;

always_ff(@posedge clk)
  q <= q_next;
```

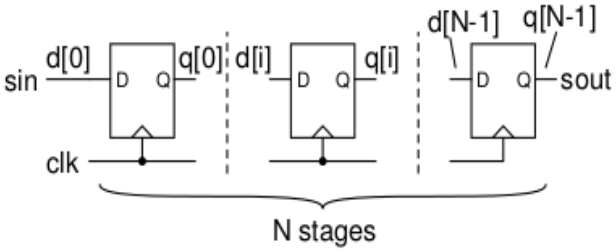



```
if (h)
    d[i] = q[i];
else if (l)
    d[i] = p[i];
else
    d[i] = q[i-1];

always_ff(@          )
    q[i] <= d[i]
```



(this is [almost?] the example from lecture 1)






```
module test (input logic sin, clk, output logic sout ) ;
parameter N = 31 ;
logic q[N-1:0] ;

always_ff@(posedge clk) begin
        q[0] <= sin ;
        for ( int i=1 ; i<N ; i++ ) begin
                q[i] <= q[i-1] ;
        end
end

assign sout = q[N-1] ;

endmodule
```