CSE433
Proje 0
Bit Sequence Detector In Verilog

Lecturer: Alp Arslan Bayrakçı

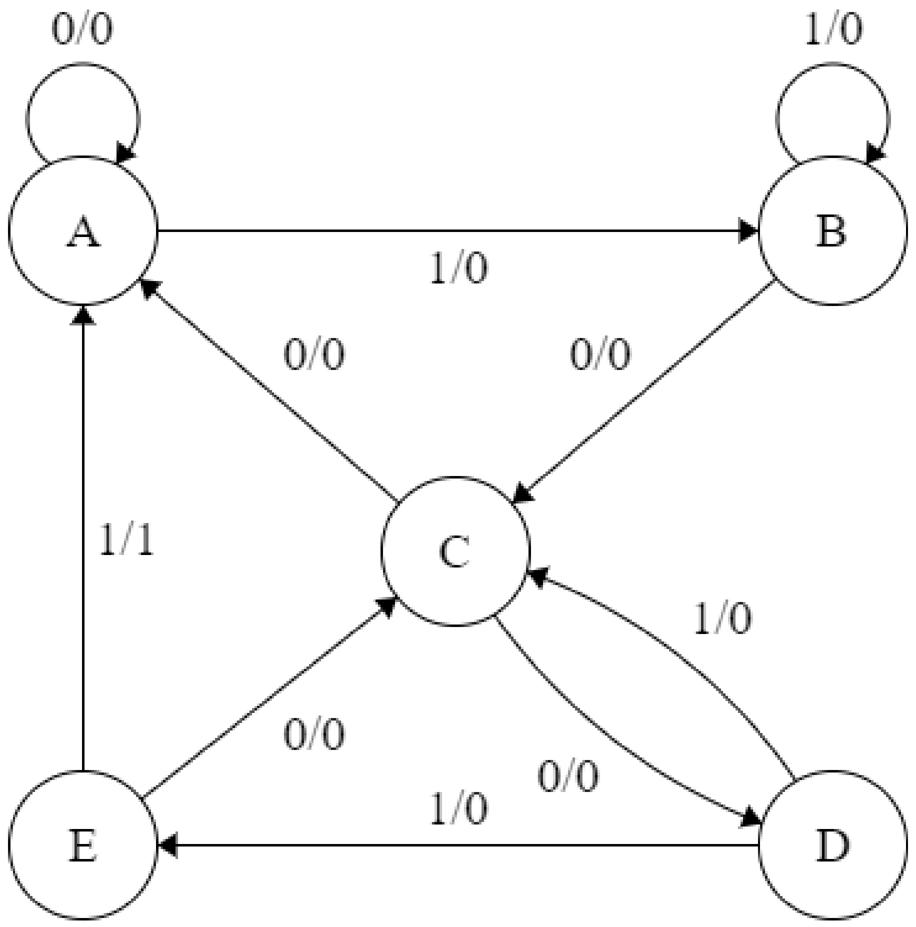
Assistant: Fatma Nur Esirci

Muhammed Okumuş
1510440017
mokumus@gtu.edu.tr

Table Of Contents

State Diagram

Where:
A → ‘-’
B → ‘1’
C → ‘10’
D → ‘101’
E → ‘1011’



Truth Table

Where:
State A → 3’b000
State B → 3’b001
State C → 3’b010
State D → 3’b011
State E → 3’b100

	Previous State				Input	Next State				Output
	Symbol	Q2	Q1	Q0	X	Symbol	Q2+	Q1+	Q0+	Z
0	A	0	0	0	0	A	0	0	0	0
1	A	0	0	0	1	B	0	0	1	0
2	B	0	0	1	0	C	0	1	0	0
3	B	0	0	1	1	B	0	0	1	0
4	C	0	1	0	0	A	0	0	0	0
5	C	0	1	0	1	D	0	1	1	0
6	D	0	1	1	0	C	0	1	0	0
7	D	0	1	1	1	E	1	0	0	0
8	E	1	0	0	0	C	0	1	0	0
9	**E**	**1**	**0**	**0**	**1**	**A**	**0**	**0**	**0**	**1**
10	-	1	0	1	0	-	X	X	X	X
11	-	1	0	1	1	-	X	X	X	X
12	-	1	1	0	0	-	X	X	X	X
13	-	1	1	0	1	-	X	X	X	X
14	-	1	1	1	0	-	X	X	X	X
15	-	1	1	1	1	-	X	X	X	X

There is no need to add D-Flip Flops to the truth table since they will be the same as the ‘Next State’.

Karnaugh Maps For D-Flip Flops

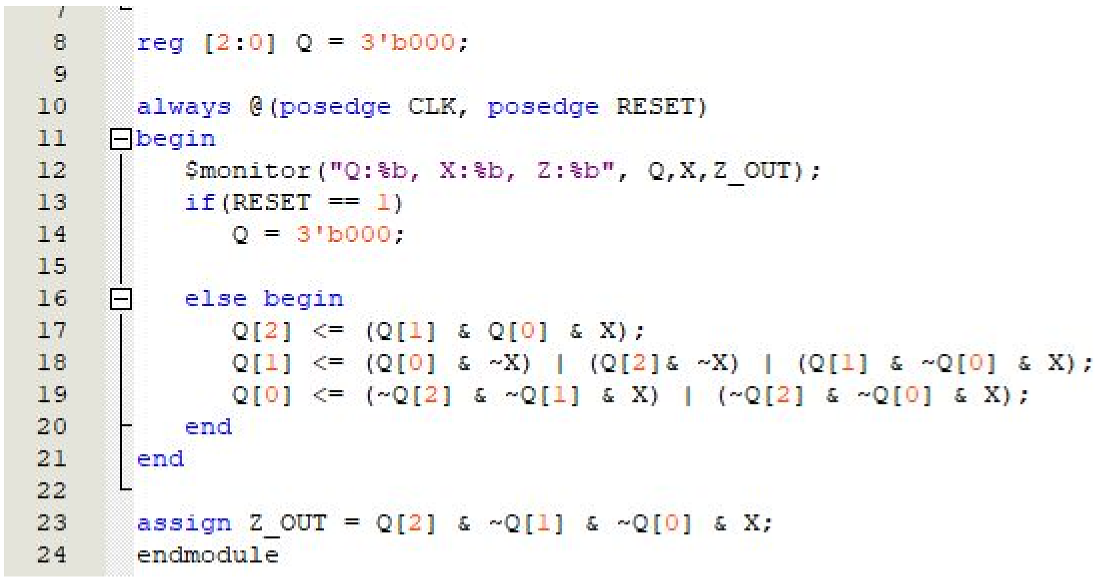
Where:
A → Q2
B → Q1
C → Q0
D → X

D2 : B.C.D → map
= Q1.Q0.X
D1 : CD' + AD' + BC'D → map
= Q0.X’ +Q2.X’ + Q1.Q0’.X
D0 : A'B'D + A'C'D → map
= Q2’.Q1’.X + Q2’.Q0.’X

Links are from an online tool to solve karnaugh maps called http://www.32x8.com.

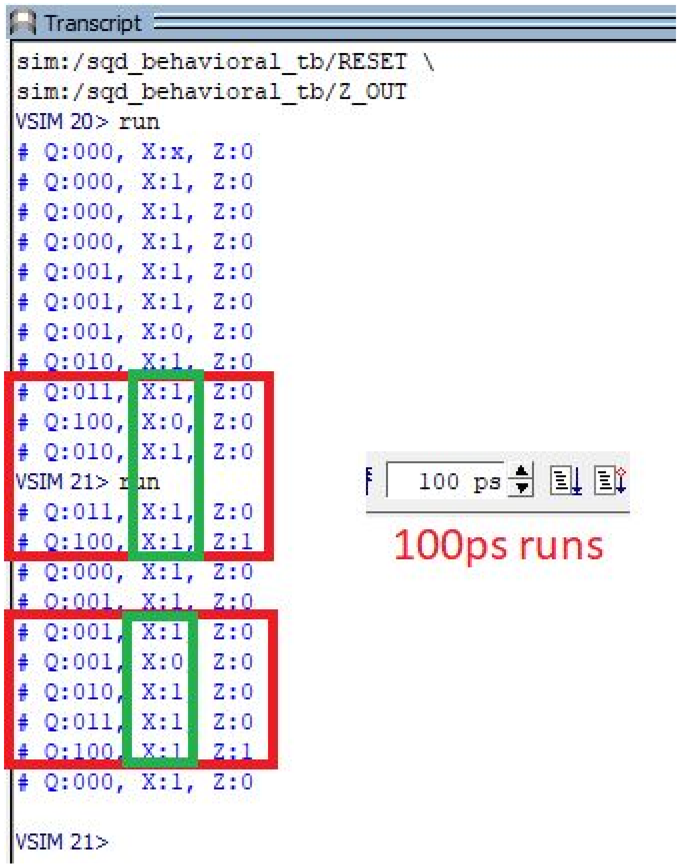
Verifying Design With Behavioral Verilog

sqd_behavioral Module

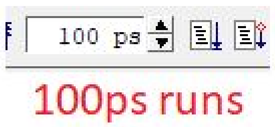


```
reg [2:0] Q = 3'b000;

always @(posedge CLK, posedge RESET)
begin
    $monitor("Q:%b, X:%b, Z:%b", Q,X,Z_OUT);
    if(RESET == 1)
        Q = 3'b000;

    else begin
        Q[2] <= (Q[1] & Q[0] & X);
        Q[1] <= (Q[0] & ~X) | (Q[2]& ~X) | (Q[1] & ~Q[0] & X);
        Q[0] <= (~Q[2] & ~Q[1] & X) | (~Q[2] & ~Q[0] & X);
    end
end

assign Z_OUT = Q[2] & ~Q[1] & ~Q[0] & X;
endmodule
```

sqd_behavioral Module Test Bench Results



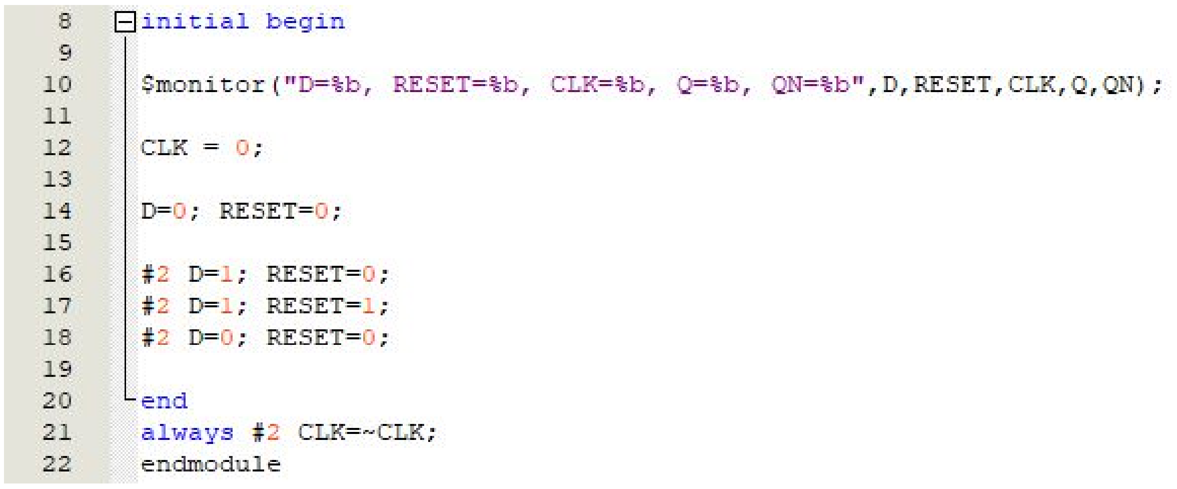
```
Transcript
sim:/sqd_behavioral_tb/RESET \
sim:/sqd_behavioral_tb/Z_OUT
VSIM 20> run
# Q:000, X:x, Z:0
# Q:000, X:1, Z:0
# Q:000, X:1, Z:0
# Q:000, X:1, Z:0
# Q:001, X:1, Z:0
# Q:001, X:1, Z:0
# Q:001, X:0, Z:0
# Q:010, X:1, Z:0
# Q:011, X:1, Z:0
# Q:100, X:0, Z:0
# Q:010, X:1, Z:0
VSIM 21> run
# Q:011, X:1, Z:0
# Q:100, X:1, Z:1
# Q:000, X:1, Z:0
# Q:001, X:1, Z:0
# Q:001, X:1  Z:0
# Q:001, X:0  Z:0
# Q:010, X:1  Z:0
# Q:011, X:1  Z:0
# Q:100, X:1  Z:1
# Q:000, X:1, Z:0

VSIM 21>
```



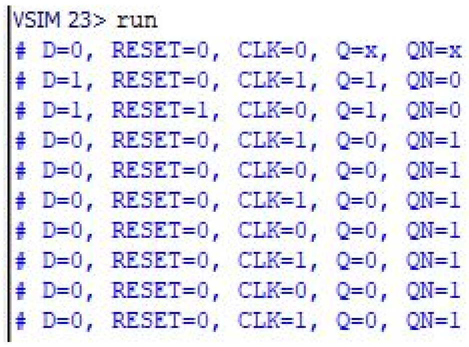
Design is working correctly, the FSM flow is right. We continue with structural implementation next.

Structural Verilog Implementation & Testbenches

d_flipflop Module Testbench



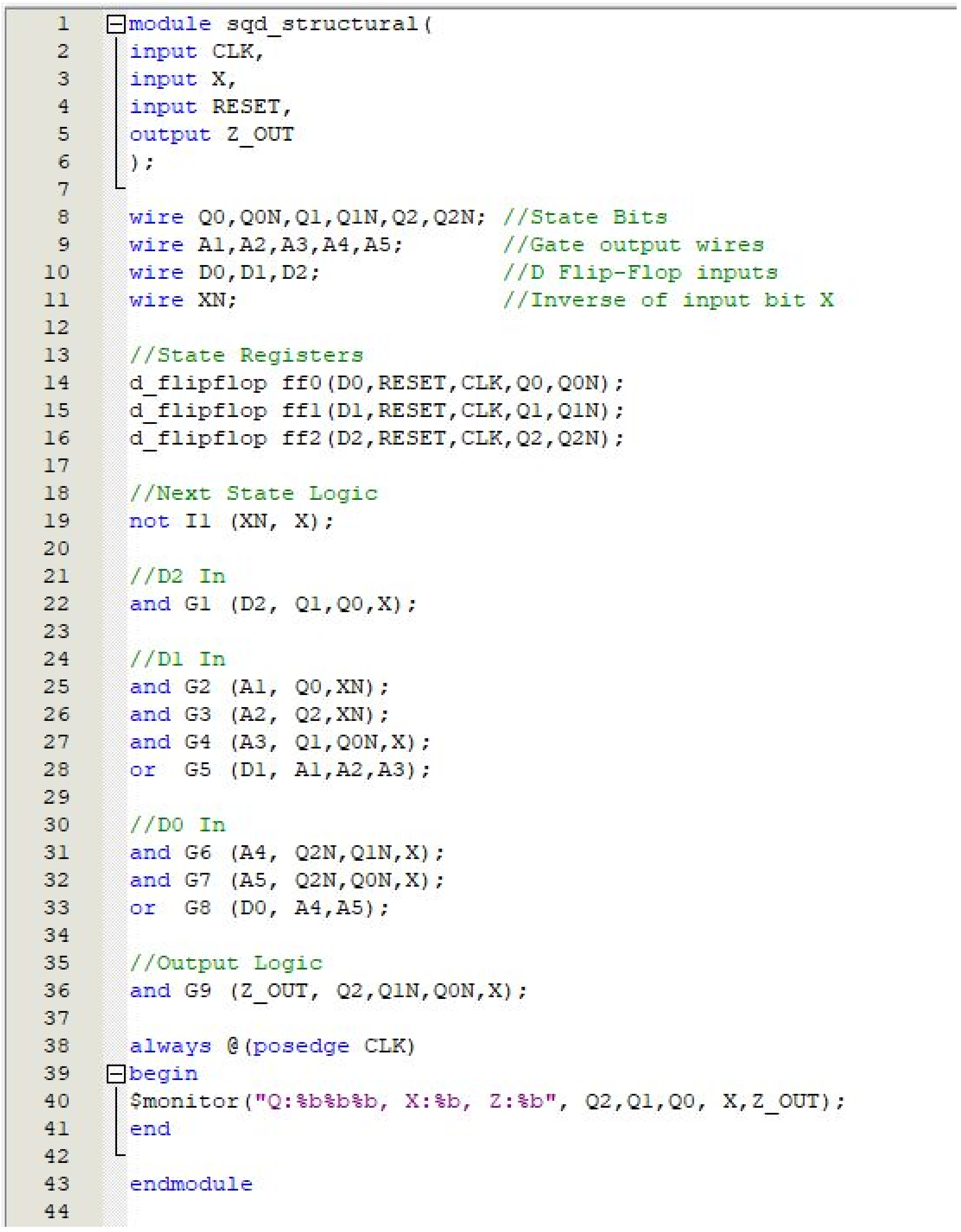
```
initial begin

$monitor("D=%b, RESET=%b, CLK=%b, Q=%b, QN=%b",D,RESET,CLK,Q,QN);

CLK = 0;

D=0; RESET=0;

#2 D=1; RESET=0;
#2 D=1; RESET=1;
#2 D=0; RESET=0;

end
always #2 CLK=~CLK;
endmodule
```



```
VSIM 23> run
# D=0, RESET=0, CLK=0, Q=x, QN=x
# D=1, RESET=0, CLK=1, Q=1, QN=0
# D=1, RESET=1, CLK=0, Q=1, QN=0
# D=0, RESET=0, CLK=1, Q=0, QN=1
# D=0, RESET=0, CLK=0, Q=0, QN=1
# D=0, RESET=0, CLK=1, Q=0, QN=1
# D=0, RESET=0, CLK=0, Q=0, QN=1
# D=0, RESET=0, CLK=1, Q=0, QN=1
# D=0, RESET=0, CLK=0, Q=0, QN=1
# D=0, RESET=0, CLK=1, Q=0, QN=1
```

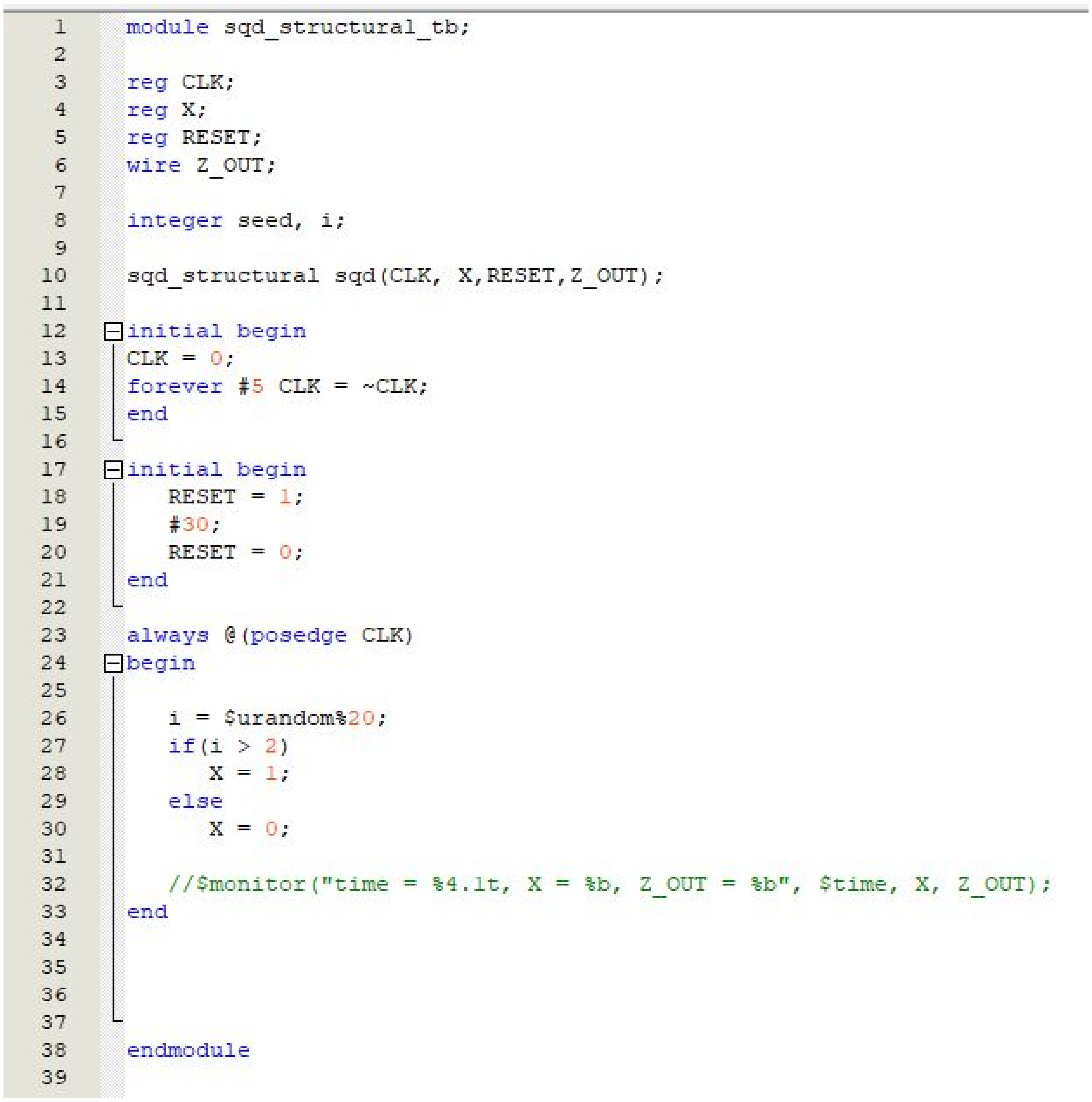
This module is working correctly.

sqd_structural Module



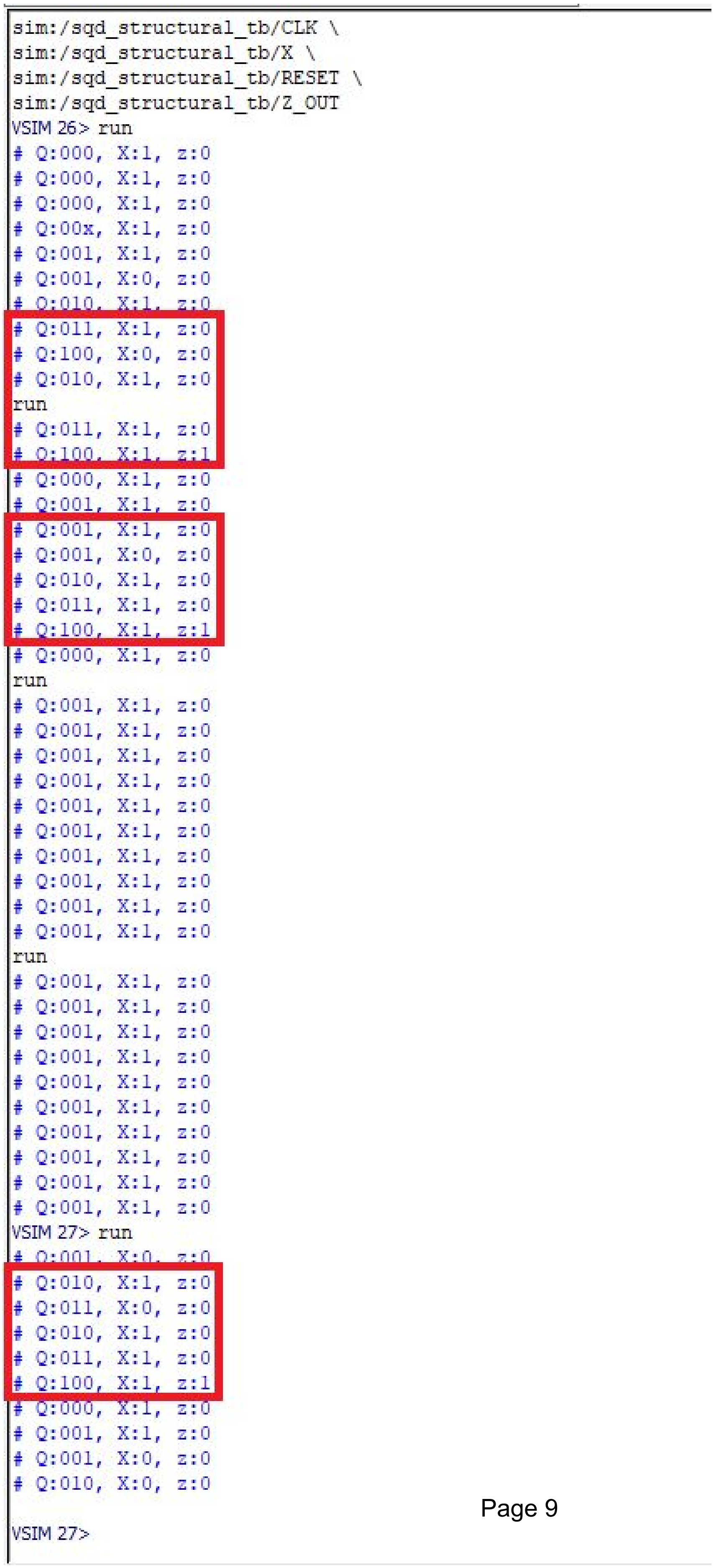
```
module sqd_structural(
input CLK,
input X,
input RESET,
output Z_OUT
);

wire Q0,Q0N,Q1,Q1N,Q2,Q2N; //State Bits
wire A1,A2,A3,A4,A5;       //Gate output wires
wire D0,D1,D2;             //D Flip-Flop inputs
wire XN;                   //Inverse of input bit X

//State Registers
d_flipflop ff0(D0,RESET,CLK,Q0,Q0N);
d_flipflop ff1(D1,RESET,CLK,Q1,Q1N);
d_flipflop ff2(D2,RESET,CLK,Q2,Q2N);

//Next State Logic
not I1 (XN, X);

//D2 In
and G1 (D2, Q1,Q0,X);

//D1 In
and G2 (A1, Q0,XN);
and G3 (A2, Q2,XN);
and G4 (A3, Q1,Q0N,X);
or  G5 (D1, A1,A2,A3);

//D0 In
and G6 (A4, Q2N,Q1N,X);
and G7 (A5, Q2N,Q0N,X);
or  G8 (D0, A4,A5);

//Output Logic
and G9 (Z_OUT, Q2,Q1N,Q0N,X);

always @(posedge CLK)
begin
$monitor("Q:%b%b%b, X:%b, Z:%b", Q2,Q1,Q0, X,Z_OUT);
end

endmodule

```

sqd_structural Module Testbench



```verilog
module sqd_structural_tb;

reg CLK;
reg X;
reg RESET;
wire Z_OUT;

integer seed, i;

sqd_structural sqd(CLK, X,RESET,Z_OUT);

initial begin
CLK = 0;
forever #5 CLK = ~CLK;
end

initial begin
   RESET = 1;
   #30;
   RESET = 0;
end

always @(posedge CLK)
begin

   i = $urandom%20;
   if(i > 2)
      X = 1;
   else
      X = 0;

   //$monitor("time = %4.1t, X = %b, Z_OUT = %b", $time, X, Z_OUT);
end




endmodule

```

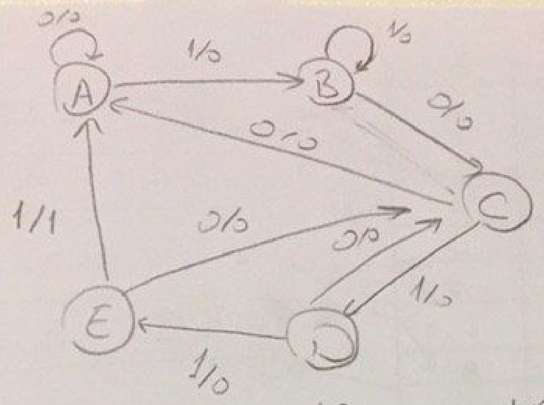
This testbench generates random bits. Approximately %10 of bits are ‘0’, %90 of bits are ‘1’.



```
sim:/sqd_structural_tb/CLK \
sim:/sqd_structural_tb/X \
sim:/sqd_structural_tb/RESET \
sim:/sqd_structural_tb/Z_OUT
VSIM 26> run
# Q:000, X:1, z:0
# Q:000, X:1, z:0
# Q:000, X:1, z:0
# Q:00x, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:0, z:0
# Q:010, X:1, z:0
# Q:011, X:1, z:0
# Q:100, X:0, z:0
# Q:010, X:1, z:0
run
# Q:011, X:1, z:0
# Q:100, X:1, z:1
# Q:000, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:0, z:0
# Q:010, X:1, z:0
# Q:011, X:1, z:0
# Q:100, X:1, z:1
# Q:000, X:1, z:0
run
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
run
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:1, z:0
VSIM 27> run
# Q:001, X:0, z:0
# Q:010, X:1, z:0
# Q:011, X:0, z:0
# Q:010, X:1, z:0
# Q:011, X:1, z:0
# Q:100, X:1, z:1
# Q:000, X:1, z:0
# Q:001, X:1, z:0
# Q:001, X:0, z:0
# Q:010, X:0, z:0

VSIM 27>
```

On Paper Design

In case there is a typo in the tables or FSM in the report, here is the original design.



$$\begin{tikzcd}
A \arrow[loop above, "0/0"] \arrow[rr, "1/0"] & & B \arrow[loop above, "1/0"] \arrow[dr, "0/0"] & \\
 & & & C \arrow[ulll, "0/0"] \arrow[dl, bend left, "1/0"] \\
E \arrow[uu, "1/1"] \arrow[urrr, "0/0"] & & D \arrow[ll, "1/0"] \arrow[ur, "0/0"] &
\end{tikzcd}$$

A → 000 '-'
B → 001 '1'
C → 010 '10'
D → 011 '101'
E → 100 '1011'

P.S.				Input	N.S.				Output
Symbol	Q_2	Q_1	Q_0	X	Symbol	Q_2^+	Q_1^+	Q_0^+	Z
A	0	0	0	0	A	0	0	0	0
A	0	0	0	1	B	0	0	1	0
B	0	0	1	0	C	0	1	0	0
B	0	0	1	1	B	0	0	1	0
C	0	1	0	0	A	0	0	0	0
C	0	1	0	1	D	0	1	1	0
D	0	1	1	0	C	0	1	0	0
D	0	1	1	1	E	1	0	0	0
E	1	0	0	0	C	0	1	0	0
E	1	0	0	1	A	0	0	0	1
-	1	0	1	0					
-	1	0	1	1					
-	1	1	0	0					
-	1	1	0	1					
-	1	1	1	0					
-	1	1	1	1					

$$D_2 = BCD$$

$$\boxed{= Q_1 Q_0 x}$$

$$D_1 = C\bar{D} + A\bar{D} + B\bar{C}D$$

$$\boxed{= Q_0 \bar{x} + Q_2 \bar{x} + Q_1 \bar{Q}_0 x}$$

$$D_0 = \bar{A}\bar{B}D + \bar{A}\bar{C}D$$

$$\boxed{= \bar{Q}_2 \bar{Q}_1 x + \bar{Q}_2 \bar{Q}_0 x}$$