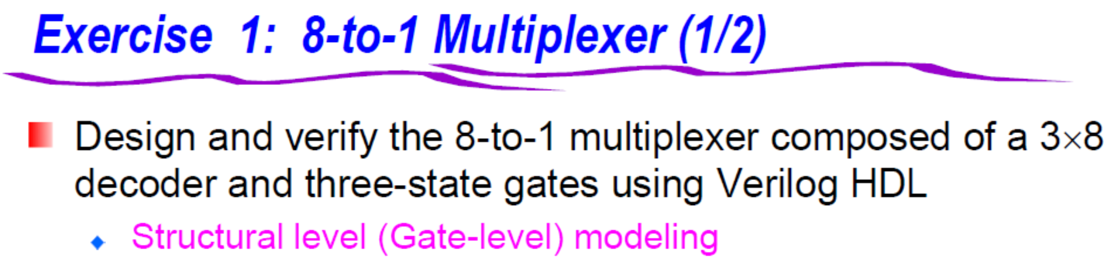
Three-state Gates, Latch, and D Flip-Flop

2016/11/28

范真瑋

Ex1

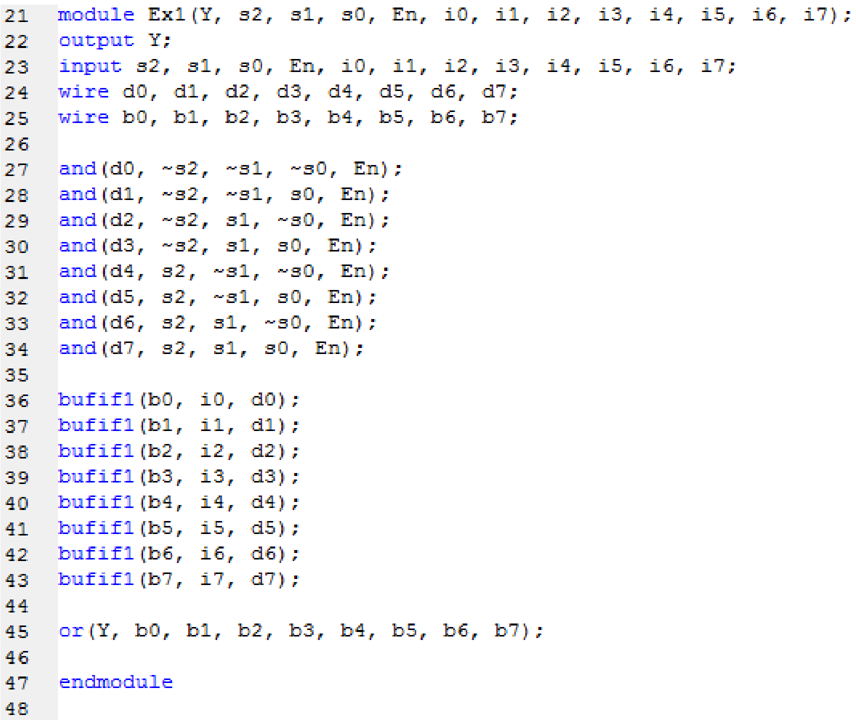
實驗內容：



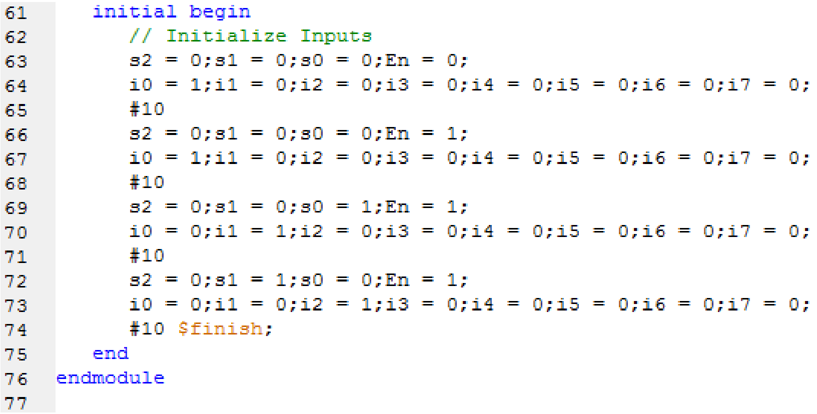
Exercise 1: 8-to-1 Multiplexer (1/2)

- Design and verify the 8-to-1 multiplexer composed of a 3×8 decoder and three-state gates using Verilog HDL
  - Structural level (Gate-level) modeling

程式碼：

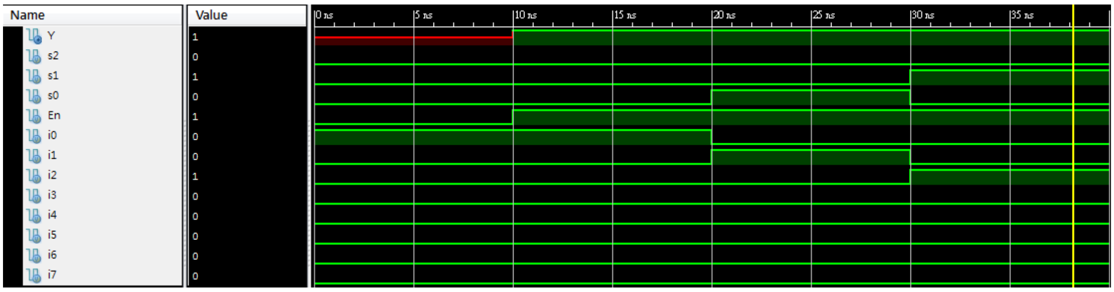


```verilog
module Ex1(Y, s2, s1, s0, En, i0, i1, i2, i3, i4, i5, i6, i7);
output Y;
input s2, s1, s0, En, i0, i1, i2, i3, i4, i5, i6, i7;
wire d0, d1, d2, d3, d4, d5, d6, d7;
wire b0, b1, b2, b3, b4, b5, b6, b7;

and(d0, ~s2, ~s1, ~s0, En);
and(d1, ~s2, ~s1, s0, En);
and(d2, ~s2, s1, ~s0, En);
and(d3, ~s2, s1, s0, En);
and(d4, s2, ~s1, ~s0, En);
and(d5, s2, ~s1, s0, En);
and(d6, s2, s1, ~s0, En);
and(d7, s2, s1, s0, En);

bufif1(b0, i0, d0);
bufif1(b1, i1, d1);
bufif1(b2, i2, d2);
bufif1(b3, i3, d3);
bufif1(b4, i4, d4);
bufif1(b5, i5, d5);
bufif1(b6, i6, d6);
bufif1(b7, i7, d7);

or(Y, b0, b1, b2, b3, b4, b5, b6, b7);

endmodule

```



```verilog
   initial begin
      // Initialize Inputs
      s2 = 0;s1 = 0;s0 = 0;En = 0;
      i0 = 1;i1 = 0;i2 = 0;i3 = 0;i4 = 0;i5 = 0;i6 = 0;i7 = 0;
      #10
      s2 = 0;s1 = 0;s0 = 0;En = 1;
      i0 = 1;i1 = 0;i2 = 0;i3 = 0;i4 = 0;i5 = 0;i6 = 0;i7 = 0;
      #10
      s2 = 0;s1 = 0;s0 = 1;En = 1;
      i0 = 0;i1 = 1;i2 = 0;i3 = 0;i4 = 0;i5 = 0;i6 = 0;i7 = 0;
      #10
      s2 = 0;s1 = 1;s0 = 0;En = 1;
      i0 = 0;i1 = 0;i2 = 1;i3 = 0;i4 = 0;i5 = 0;i6 = 0;i7 = 0;
      #10 $finish;
   end
endmodule

```

實驗結果及分析：

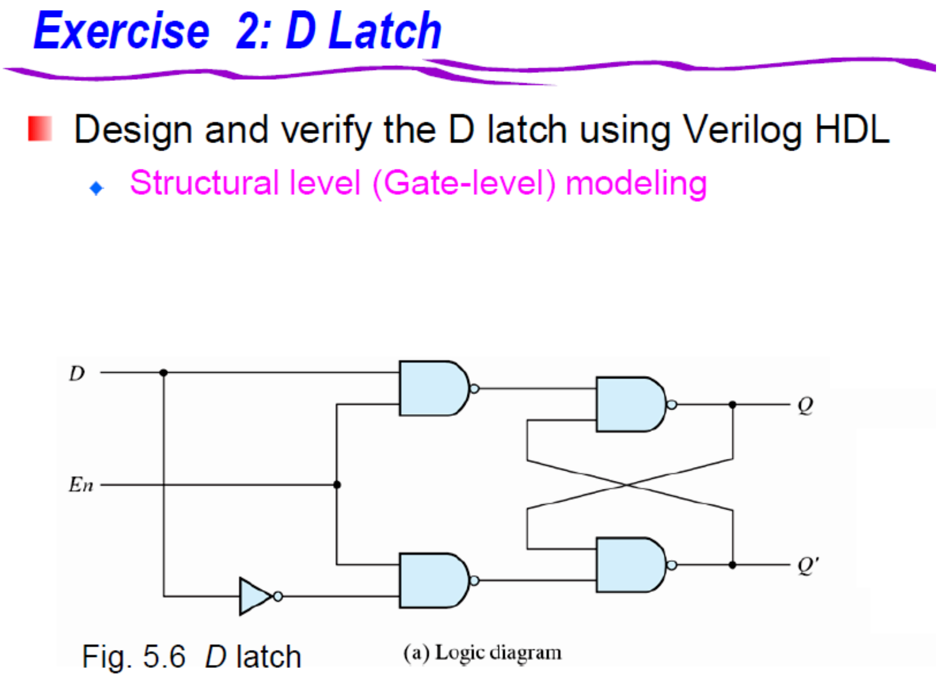


Name	Value
Y	1
s2	0
s1	1
s0	0
En	1
i0	0
i1	0
i2	1
i3	0
i4	0
i5	0
i6	0
i7	0

s2	s1	s0	En	Y
X	X	X	0	X
0	0	0	1	i0
0	0	1	1	i1
0	1	0	1	i2

Ex2

實驗內容：



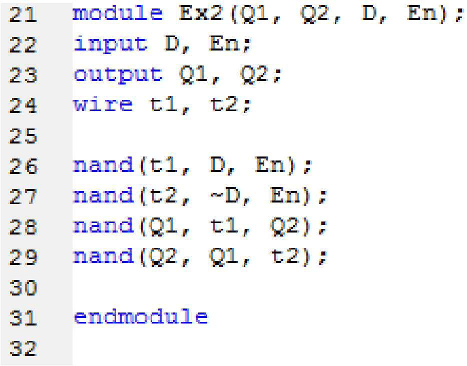
Exercise 2: D Latch

- Design and verify the D latch using Verilog HDL
  - Structural level (Gate-level) modeling

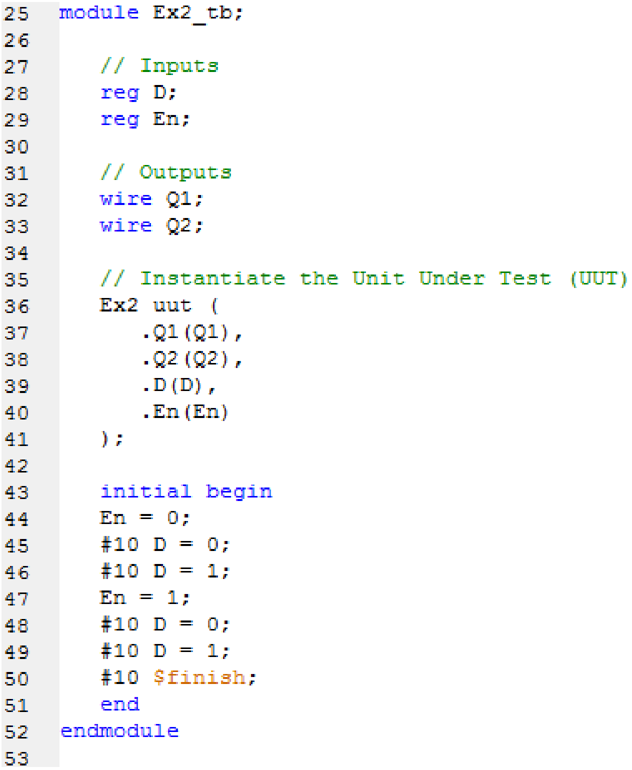
Fig. 5.6 *D* latch

(a) Logic diagram

程式碼：

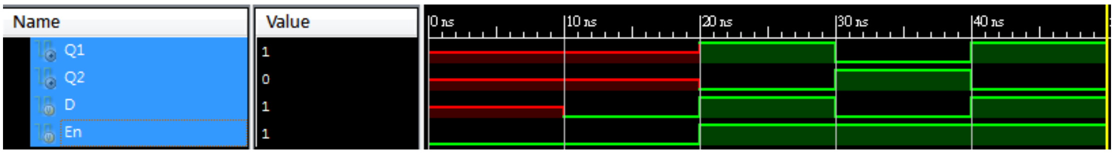


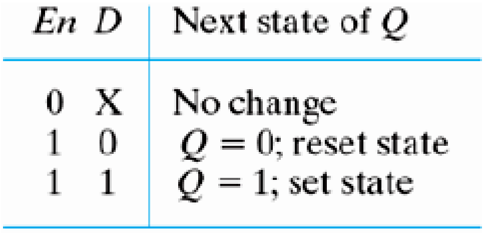
```verilog
module Ex2(Q1, Q2, D, En);
input D, En;
output Q1, Q2;
wire t1, t2;

nand(t1, D, En);
nand(t2, ~D, En);
nand(Q1, t1, Q2);
nand(Q2, Q1, t2);

endmodule

```



```
module Ex2_tb;

   // Inputs
   reg D;
   reg En;

   // Outputs
   wire Q1;
   wire Q2;

   // Instantiate the Unit Under Test (UUT)
   Ex2 uut (
      .Q1(Q1),
      .Q2(Q2),
      .D(D),
      .En(En)
   );

   initial begin
   En = 0;
   #10 D = 0;
   #10 D = 1;
   En = 1;
   #10 D = 0;
   #10 D = 1;
   #10 $finish;
   end
endmodule

```

實驗結果及分析：





En	D	Next state of Q
0	X	No change
1	0	$Q = 0$; reset state
1	1	$Q = 1$; set state

Ex3

實驗內容：

Exercise 3: Master-slave D Flip-Flop

- Design and verify the positive-edge-triggered master-slave D flip-flop using Verilog HDL
  - Structural level (Gate-level) modeling

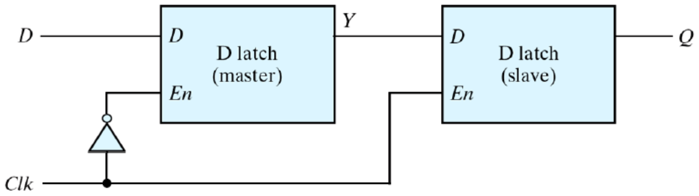
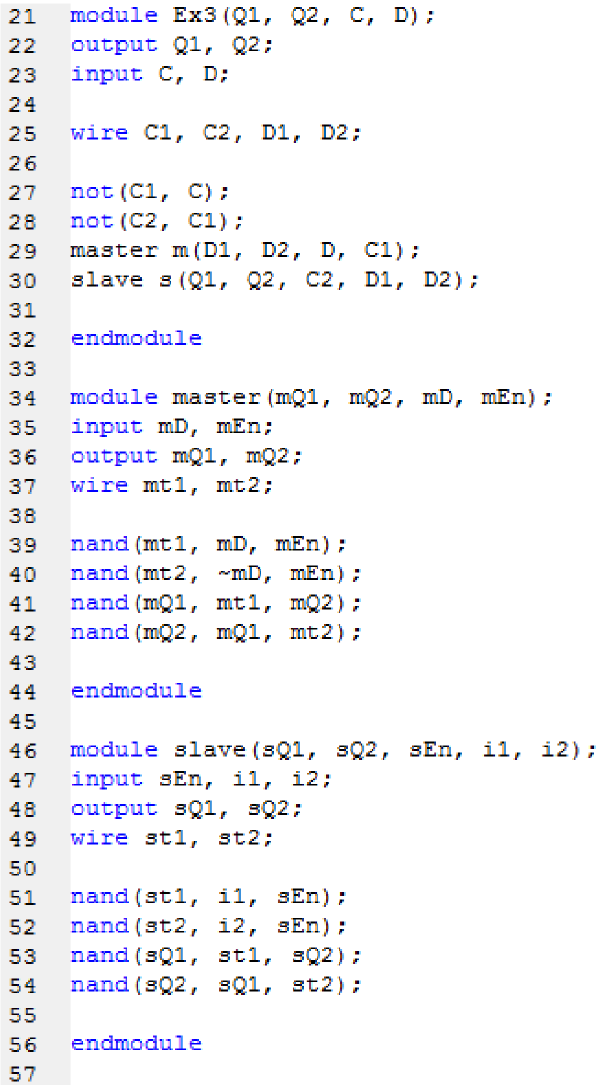
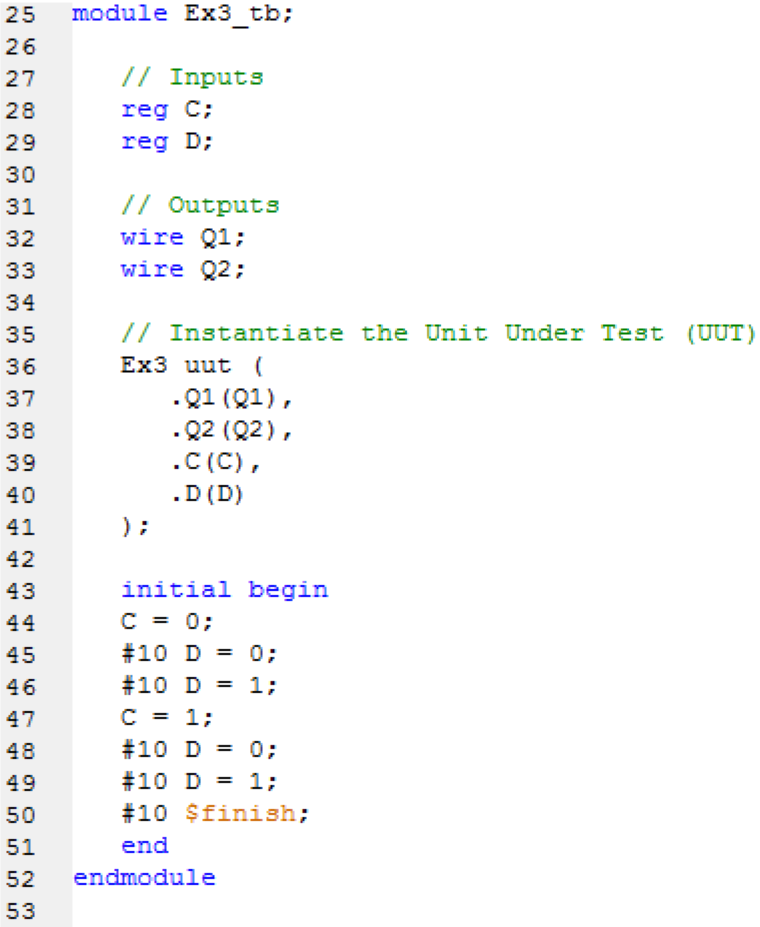


Fig. Positive-edge-triggered master-slave *D* flip-flop

程式碼：

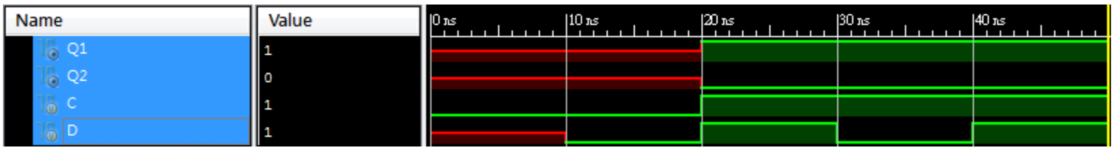


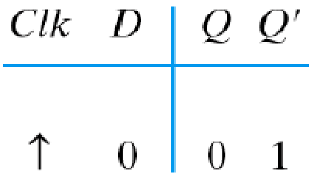
```
module Ex3(Q1, Q2, C, D);
output Q1, Q2;
input C, D;

wire C1, C2, D1, D2;

not(C1, C);
not(C2, C1);
master m(D1, D2, D, C1);
slave s(Q1, Q2, C2, D1, D2);

endmodule

module master(mQ1, mQ2, mD, mEn);
input mD, mEn;
output mQ1, mQ2;
wire mt1, mt2;

nand(mt1, mD, mEn);
nand(mt2, ~mD, mEn);
nand(mQ1, mt1, mQ2);
nand(mQ2, mQ1, mt2);

endmodule

module slave(sQ1, sQ2, sEn, i1, i2);
input sEn, i1, i2;
output sQ1, sQ2;
wire st1, st2;

nand(st1, i1, sEn);
nand(st2, i2, sEn);
nand(sQ1, st1, sQ2);
nand(sQ2, sQ1, st2);

endmodule

```



```verilog
module Ex3_tb;

    // Inputs
    reg C;
    reg D;

    // Outputs
    wire Q1;
    wire Q2;

    // Instantiate the Unit Under Test (UUT)
    Ex3 uut (
        .Q1(Q1),
        .Q2(Q2),
        .C(C),
        .D(D)
    );

    initial begin
    C = 0;
    #10 D = 0;
    #10 D = 1;
    C = 1;
    #10 D = 0;
    #10 D = 1;
    #10 $finish;
    end
endmodule

```

實驗結果及分析：

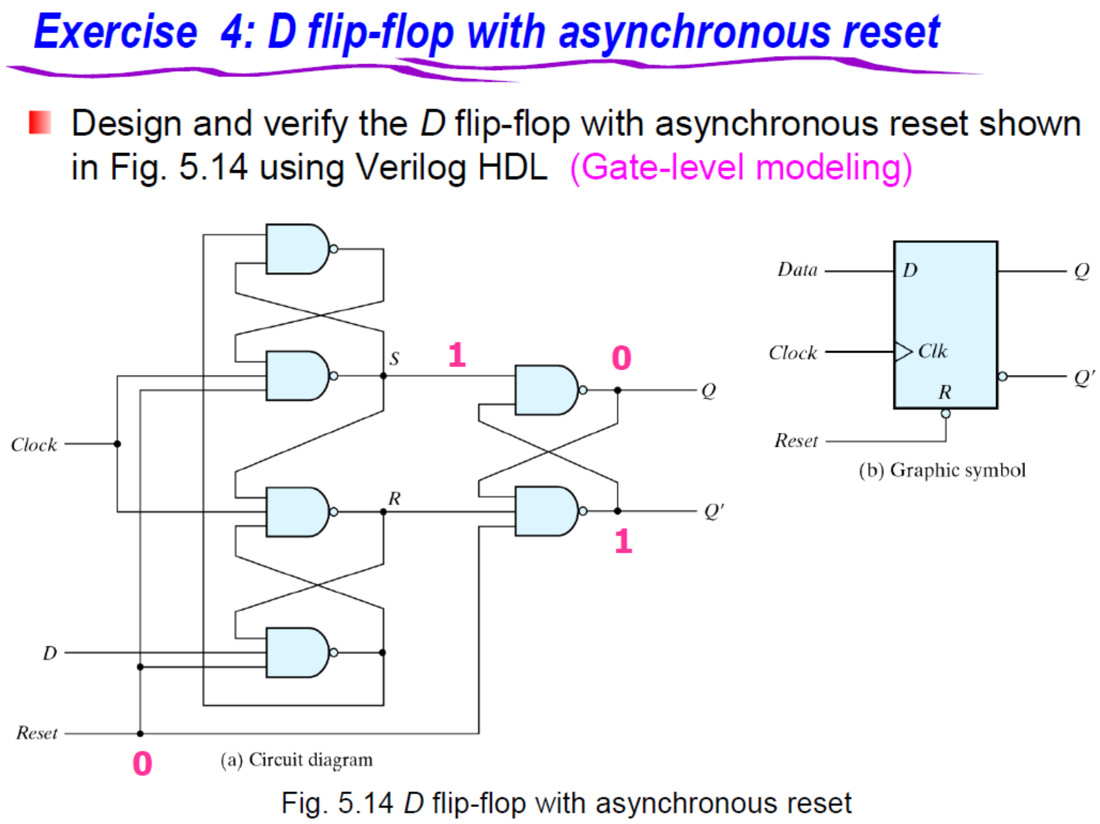




Clk	D	Q	Q'
↑	0	0	1

Ex4

實驗內容：

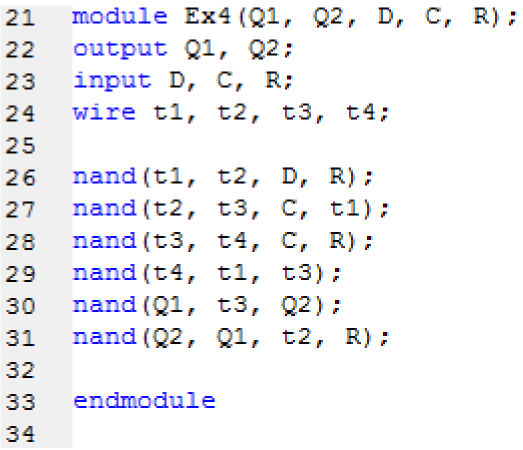


Exercise 4: D flip-flop with asynchronous reset

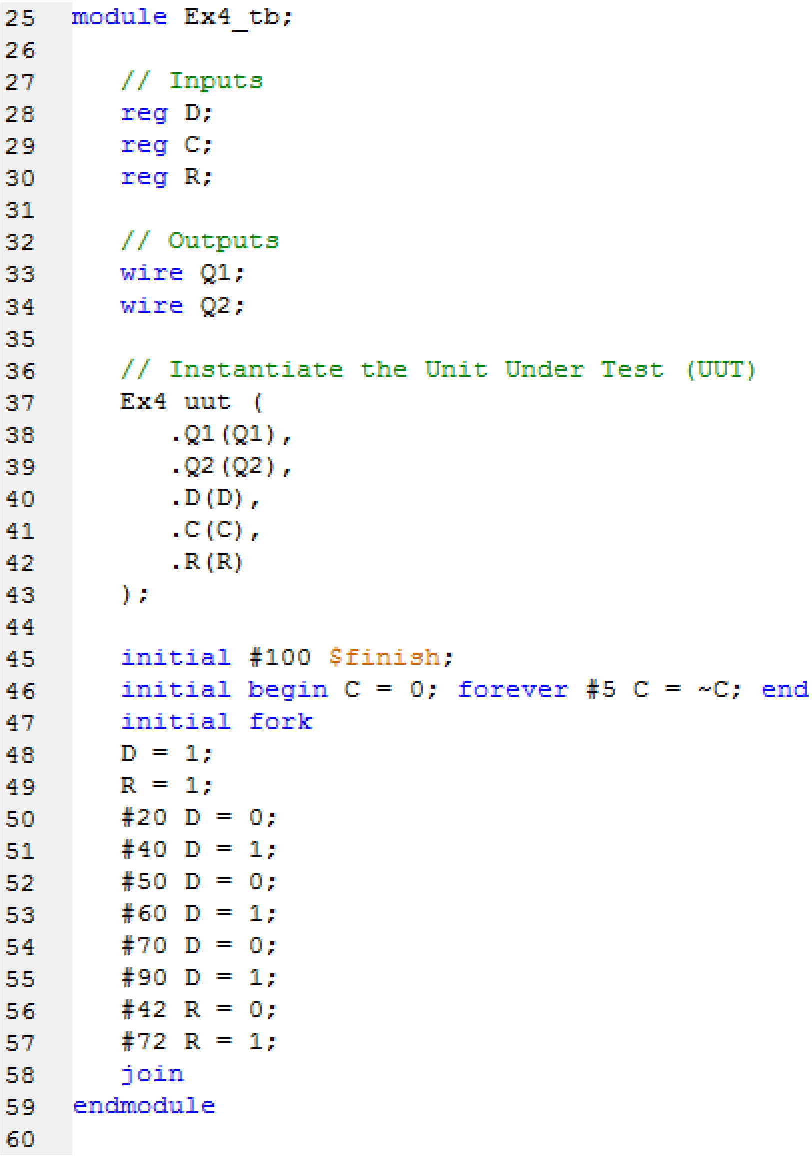
- Design and verify the *D* flip-flop with asynchronous reset shown in Fig. 5.14 using Verilog HDL (Gate-level modeling)

Fig. 5.14 *D* flip-flop with asynchronous reset

程式碼：

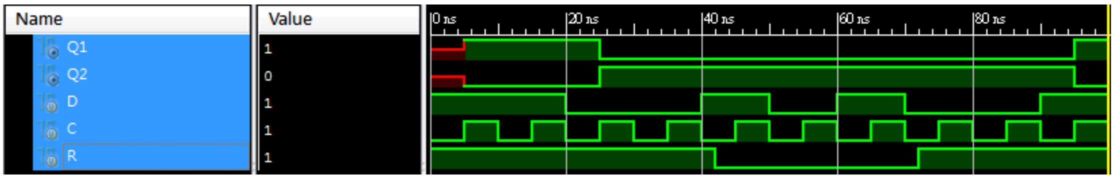


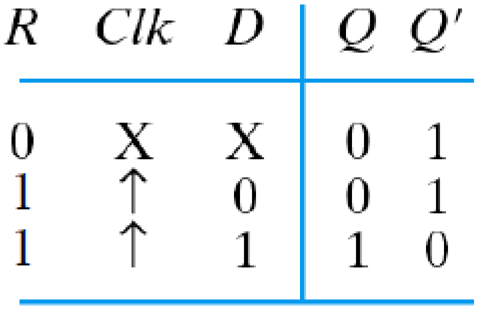
```
module Ex4(Q1, Q2, D, C, R);
output Q1, Q2;
input D, C, R;
wire t1, t2, t3, t4;

nand(t1, t2, D, R);
nand(t2, t3, C, t1);
nand(t3, t4, C, R);
nand(t4, t1, t3);
nand(Q1, t3, Q2);
nand(Q2, Q1, t2, R);

endmodule

```



```
module Ex4_tb;

    // Inputs
    reg D;
    reg C;
    reg R;

    // Outputs
    wire Q1;
    wire Q2;

    // Instantiate the Unit Under Test (UUT)
    Ex4 uut (
        .Q1(Q1),
        .Q2(Q2),
        .D(D),
        .C(C),
        .R(R)
    );

    initial #100 $finish;
    initial begin C = 0; forever #5 C = ~C; end
    initial fork
    D = 1;
    R = 1;
    #20 D = 0;
    #40 D = 1;
    #50 D = 0;
    #60 D = 1;
    #70 D = 0;
    #90 D = 1;
    #42 R = 0;
    #72 R = 1;
    join
endmodule

```

實驗結果及分析：





R	Clk	D	Q	Q'
0	X	X	0	1
1	↑	0	0	1
1	↑	1	1	0