# 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 3x8 decoder and three-state gates using Verilog HDL
  - Structural level (Gate-level) modeling

```
21 module Ex1(Y, s2, s1, s0, En, i0, i1, i2, i3, i4, i5, i6, i7);
22 output Y;
23 input s2, s1, s0, En, i0, i1, i2, i3, i4, i5, i6, i7;
24 wire d0, d1, d2, d3, d4, d5, d6, d7;
25 wire b0, b1, b2, b3, b4, b5, b6, b7;
26
27
    and(d0, ~s2, ~s1, ~s0, En);
28 and(d1, ~s2, ~s1, s0, En);
29 and (d2, ~s2, s1, ~s0, En);
30 and (d3, ~s2, s1, s0, En);
31 and (d4, s2, ~s1, ~s0, En);
32 and (d5, s2, ~s1, s0, En);
33 and(d6, s2, s1, ~s0, En);
34
    and (d7, s2, s1, s0, En);
35
36 bufif1(b0, i0, d0);
37 bufif1(b1, i1, d1);
38 bufif1(b2, i2, d2);
39 bufif1(b3, i3, d3);
40 bufif1(b4, i4, d4);
41
    bufif1(b5, i5, d5);
42 bufif1(b6, i6, d6);
43 bufif1(b7, i7, d7);
44
    or(Y, b0, b1, b2, b3, b4, b5, b6, b7);
45
46
    endmodule
47
48
61
     initial begin
         // Initialize Inputs
62
          s2 = 0; s1 = 0; s0 = 0; En = 0;
63
          i0 = 1;i1 = 0;i2 = 0;i3 = 0;i4 = 0;i5 = 0;i6 = 0;i7 = 0;
64
          #10
          s2 = 0; s1 = 0; s0 = 0; En = 1;
66
          i0 = 1;i1 = 0;i2 = 0;i3 = 0;i4 = 0;i5 = 0;i6 = 0;i7 = 0;
67
68
          #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;
70
71
          #10
72
          s2 = 0; s1 = 1; s0 = 0; En = 1;
73
          i0 = 0;i1 = 0;i2 = 1;i3 = 0;i4 = 0;i5 = 0;i6 = 0;i7 = 0;
74
          #10 $finish;
75
       end
   endmodule
```

Name	Value	Ons	5 ns	10 ns	15 ns	20 ns	25 ns	30 ns	35 ns	
<b>₩</b> Y	1									
1 <mark>₀ s2</mark>	0									
1⊌ s1	1									
1 <mark>₀</mark> s0	0									
1⅓ En	1									
₹ io	0									
1⅓ i1	0									
1⅓ i2	1									
<b>1⅓</b> i3	0									
1⅓ i4	0									
1/ <sub>6</sub> i5	0									
1 <u>6</u> i6	0									
1 <u>B</u> 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

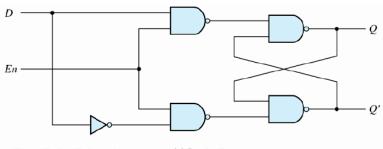


Fig. 5.6 D latch

#### (a) Logic diagram

```
21 module Ex2(Q1, Q2, D, En);
   input D, En;
   output Q1, Q2;
23
24
   wire t1, t2;
25
26 nand(t1, D, En);
27 nand(t2, ~D, En);
28 nand(Q1, t1, Q2);
29
   nand(Q2, Q1, t2);
30
   endmodule
31
32
```

```
25 module Ex2_tb;
26
27
       // Inputs
28
      reg D;
29
       reg En;
30
      // Outputs
31
32
       wire Q1;
       wire Q2;
33
34
      // Instantiate the Unit Under Test (UUT)
35
36
       Ex2 uut (
          .Q1(Q1),
37
          .Q2(Q2),
38
          .D(D),
39
          .En(En)
40
      );
41
42
43
      initial begin
      En = 0;
44
      #10 D = 0;
#10 D = 1;
En = 1;
45
46
47
       #10 D = 0;
48
       #10 D = 1;
49
50
       #10 $finish;
51
      end
52 endmodule
53
```

Name	Value	0 ns	10 ns	20 ns	30 ns	40 ns
16 Q1	1					
16 Q2	0					
16 D	1					
16 En	1					

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

#### 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

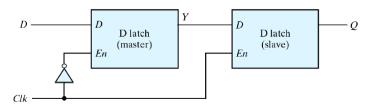
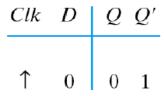


Fig. Positive-edge-triggered master-slave *D* flip-flop

```
21 module Ex3(Q1, Q2, C, D);
   output Q1, Q2;
22
   input C, D;
23
24
   wire C1, C2, D1, D2;
25
26
27
   not (C1, C);
28 not (C2, C1);
29 master m(D1, D2, D, C1);
30
   slave s(Q1, Q2, C2, D1, D2);
31
   endmodule
32
33
34 module master(mQ1, mQ2, mD, mEn);
35 input mD, mEn;
   output mQ1, mQ2;
36
   wire mt1, mt2;
37
38
39
    nand(mt1, mD, mEn);
    nand(mt2, ~mD, mEn);
40
    nand(mQ1, mt1, mQ2);
41
    nand(mQ2, mQ1, mt2);
42
43
    endmodule
44
45
46 module slave(sQ1, sQ2, sEn, i1, i2);
   input sEn, i1, i2;
47
   output sQ1, sQ2;
48
    wire st1, st2;
49
50
51 nand(st1, i1, sEn);
52 nand(st2, i2, sEn);
53 nand(sQ1, st1, sQ2);
   nand(sQ2, sQ1, st2);
56 endmodule
57
```

```
25 module Ex3 tb;
26
       // Inputs
27
28
       reg C;
       reg D;
29
30
       // Outputs
31
       wire Q1;
32
       wire Q2;
33
34
35
       // Instantiate the Unit Under Test (UUT)
       Ex3 uut (
36
          .Q1(Q1),
37
38
           .Q2(Q2),
           .C(C),
39
           .D(D)
40
       );
41
42
43
       initial begin
       C = 0;
44
45
       #10 D = 0;
       #10 D = 1;
46
       c = 1;
47
       #10 D = 0;
48
49
       #10 D = 1;
50
       #10 $finish;
       end
51
52 endmodule
53
```





# 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)

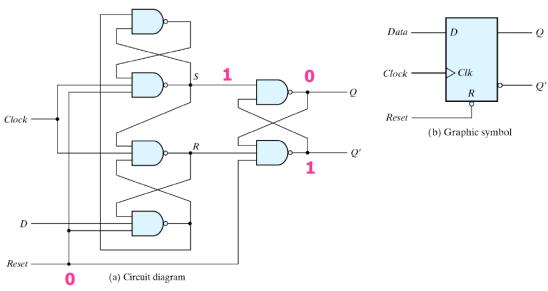
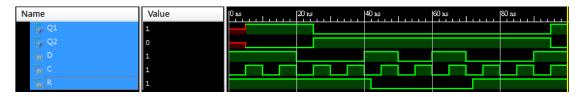


Fig. 5.14 D flip-flop with asynchronous reset

```
21 module Ex4(Q1, Q2, D, C, R);
22 output Q1, Q2;
   input D, C, R;
23
   wire t1, t2, t3, t4;
24
25
   nand(t1, t2, D, R);
26
   nand(t2, t3, C, t1);
27
28
   nand(t3, t4, C, R);
29 nand(t4, t1, t3);
30 nand(Q1, t3, Q2);
   nand(Q2, Q1, t2, R);
32
33
   endmodule
34
```

```
25 module Ex4_tb;
26
       // Inputs
27
       reg D;
28
29
       reg C;
30
       reg R;
31
       // Outputs
32
33
       wire Q1;
       wire Q2;
34
35
       // Instantiate the Unit Under Test (UUT)
36
37
       Ex4 uut (
          .Q1(Q1),
38
39
          .Q2(Q2),
40
          .D(D),
          .C(C),
41
           .R(R)
42
       );
43
44
45
       initial #100 $finish;
       initial begin C = 0; forever #5 C = ~C; end
46
47
       initial fork
48
       D = 1;
       R = 1;
49
       #20 D = 0;
50
51
       #40 D = 1;
       #50 D = 0;
52
       #60 D = 1;
53
54
       #70 D = 0;
55
       #90 D = 1;
       #42 R = 0;
56
       #72 R = 1;
57
58
       join
59
   endmodule
60
```



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