Counter & Memory

2016/12/26

范真瑋

Ex1

實驗內容：

Exercise 1: Four-bit up-down binary counter (1/2)

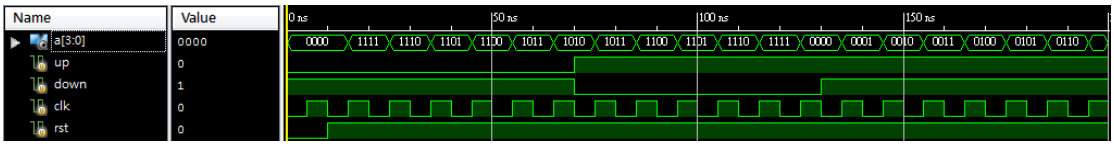
- Design and verify the four-bit up-down binary counter using T flip-flops with asynchronous reset and Verilog HDL
  - structural (gate-level) modeling

程式碼：

```
21  module ex1(output [3:0]a, input up,down,clk,rst);
22  wire t1,t2,t3,t4,t5,t6,t7,t8,t9,t10,t11;
23
24  and(t1,~up,down);
25  or(t2,up,t1);
26
27  and(t3,t1,~a[0]);
28  and(t4,up,a[0]);
29  or(t5,t4,t3);
30
31  and(t6,t3,~a[1]);
32  and(t7,t4,a[1]);
33  or(t8,t6,t7);
34
35  and(t9,t6,~a[2]);
36  and(t10,t7,a[2]);
37  or(t11,t9,t10);
38
39  tff tff1(a[0],t2,clk,rst);
40  tff tff2(a[1],t5,clk,rst);
41  tff tff3(a[2],t8,clk,rst);
42  tff tff4(a[3],t11,clk,rst);
43
44  endmodule
45
46  module tff(output Q, input t,clk,reset);
47  wire t1,s,r,t2,q,d;
48
49  xor(d,Q,t);
50  nand(t1,t2,s);
51  nand(s,t1,clk,reset);
52  nand(r,s,clk,t2);
53  nand(t2,r,d,reset);
54  nand(Q,s,q);
55  nand(q,Q,r,reset);
56
57  endmodule
```

```
25  module ex1_tb;
26
27      // Inputs
28      reg up;
29      reg down;
30      reg clk;
31      reg rst;
32
33      // Outputs
34      wire [3:0] a;
35
36      // Instantiate the Unit Under Test (UUT)
37      ex1 uut (
38          .a(a),
39          .up(up),
40          .down(down),
41          .clk(clk),
42          .rst(rst)
43      );
44
45  initial begin clk = 0; forever #5 clk = ~clk; end
46  initial begin
47      rst = 0;
48      up = 0;
49      down = 1;
50      #10 rst = 1;
51      #10 down = 1;
52      #10 down = 1;
53      #10 down = 1;
54      #10 down = 1;
55      #10 down = 1;
56      #10 down = 0; up = 1;
57      #10 down = 0; up = 1;
58      #10 down = 0; up = 1;
59      #10 down = 0; up = 1;
60      #10 down = 0; up = 1;
61      #10 down = 0; up = 1;
62      #10 down = 1;
63
64      end
65      initial #200 $finish;
66  endmodule
```

實驗結果及分析：



Up	*Down*	Function
0	0	No Change
0	1	Down
1	0	Up
1	1	Up

一開始 reset 為 0，之後從 1111 開始減 5 次變成 1010(up=0,down=1)，然後再加 6 次變成 0000(up=1,down=0)，後面因為 up 和 down 同時為 1，所以一直加到結束

Ex2

實驗內容：

Exercise 2: 4 x 4 RAM (1/2)

- Design and verify the 4 x 4 RAM using Verilog HDL
  - Structural modeling

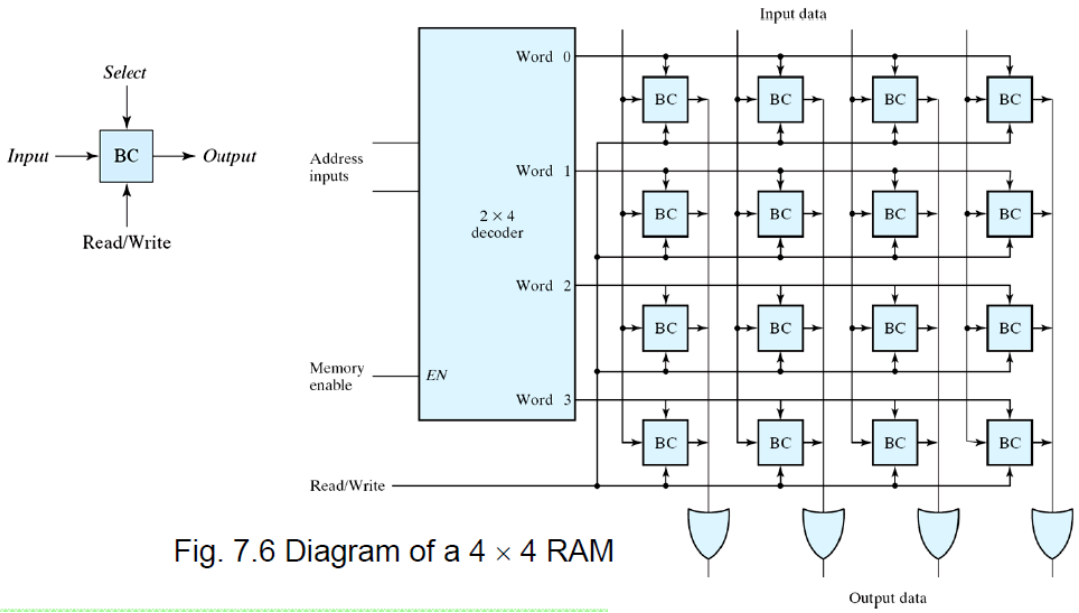


Fig. 7.6 Diagram of a 4 × 4 RAM

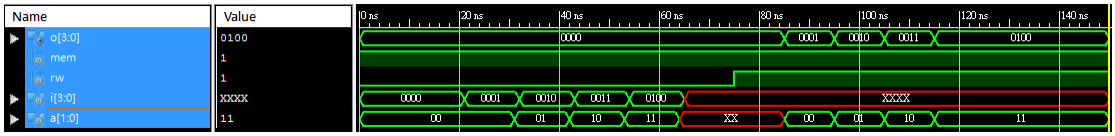
程式碼：

```
21  module EX2(output [3:0]o, input mem,rw,input[3:0]i,input[1:0]a);
22  wire [3:0]d;
23  wire [15:0]t;
24
25  decoder_case twotofour(a, d , mem);
26
27  bc bc1(t[0],i[0],d[0],rw);
28  bc bc2(t[1],i[0],d[1],rw);
29  bc bc3(t[2],i[0],d[2],rw);
30  bc bc4(t[3],i[0],d[3],rw);
31  or(o[0],t[0],t[1],t[2],t[3]);
32
33  bc bc5(t[4],i[1],d[0],rw);
34  bc bc6(t[5],i[1],d[1],rw);
35  bc bc7(t[6],i[1],d[2],rw);
36  bc bc8(t[7],i[1],d[3],rw);
37  or(o[1],t[4],t[5],t[6],t[7]);
38
39  bc bc9(t[8],i[2],d[0],rw);
40  bc bc10(t[9],i[2],d[1],rw);
41  bc bc11(t[10],i[2],d[2],rw);
42  bc bc12(t[11],i[2],d[3],rw);
43  or(o[2],t[8],t[9],t[10],t[11]);
44
45  bc bc13(t[12],i[3],d[0],rw);
46  bc bc14(t[13],i[3],d[1],rw);
47  bc bc15(t[14],i[3],d[2],rw);
48  bc bc16(t[15],i[3],d[3],rw);
49  or(o[3],t[12],t[13],t[14],t[15]);
50
51  endmodule
```

```
53  module decoder_case(binary_in, decoder_out , enable);
54     input [1:0] binary_in ; // 2 bit binary input
55     input enable ; // Enable for the decoder
56     output [3:0] decoder_out ; // 4-bit out
57
58     reg [3:0] decoder_out ;
59
60   always @ (enable or binary_in)
61   begin
62     decoder_out = 0;
63     if (enable) begin
64        case (binary_in)
65           2'b00 : decoder_out = 4'b0001;
66           2'b01 : decoder_out = 4'b0010;
67           2'b10 : decoder_out = 4'b0100;
68           2'b11 : decoder_out = 4'b1000;
69        endcase
70     end
71   end
72  endmodule
73
74  module bc(output o, input i,s,rw);
75  reg q,o;
76
77  always @ (i or s or rw)
78    if(s)
79      if(rw) o=q;
80      else q=i;
81    else o=0;
82  endmodule
```

```
25  module ex2_tb;
26
27     // Inputs
28     reg mem;
29     reg rw;
30     reg [3:0] i;
31     reg [1:0] a;
32
33     // Outputs
34     wire [3:0] o;
35
36     // Instantiate the Unit Under Test (UUT)
37     EX2 uut (
38        .o(o),
39        .mem(mem),
40        .rw(rw),
41        .i(i),
42        .a(a)
43     );
44   initial #150 $finish;
45
46   initial begin
47   mem = 1;
48   rw = 0;
49   a = 0;
50   i = 0;
51   #10 rw=1'b0;
52   #10 a=2'b00;#1 i=4'b0001;
53   #10 a=2'b01;#1 i=4'b0010;
54   #10 a=2'b10;#1 i=4'b0011;
55   #10 a=2'b11;#1 i=4'b0100;
56   #10 a=2'bxx;#1 i=4'bxxxx;
57   #10 rw=1'b1;
58   #10 a=2'b00;
59   #10 a=2'b01;
60   #10 a=2'b10;
61   #10 a=2'b11;
62   end
63  endmodule
```

實驗結果及分析：



前半部 rw=0 為 write 操作，將 input 寫入記憶元件中；後半部 rw=1 為 read 操作，讀取前面儲存的資料；當 address(a)為 00，經過 decoder 後，第一列的 BC 做寫入或讀取