

第 9 講 函數(Function)

1. 參考文獻:

[1]Online Verilog Simulator: edaplayground 教學:

<https://www.youtube.com/watch?v=f9uwtAax4v0&t=249s>

[2]金門大學陳鍾誠老師電子書: <http://ccckmit.wikidot.com/ve:main>

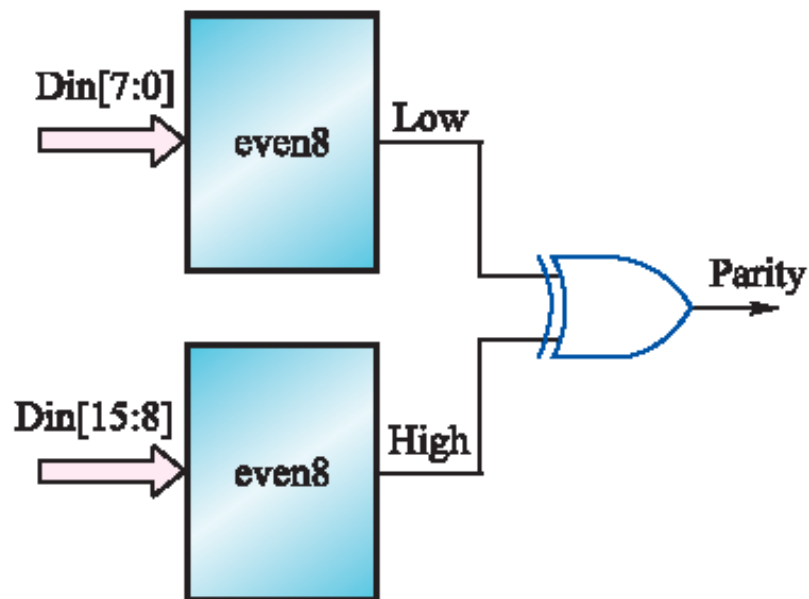
[3]林灶生，verilog 晶片設計第 6 章，全華書局

2. 基本觀念[3]

2.1 函數

```
function [位元長度] <函數名稱>
<函數參數宣告>
-----
<敘述區>
-----
endfunction
```

例題 1: 以 8 位偶同位產生器完成 16 位元偶同位產生器



```

testbench.v
1 module testbench();
2   reg [15:0] Din;
3   wire Pout;
4   even_parity_16 U1(Din, Pout);
5
6   initial
7     $monitor($time, " ", Din=%b, Pout=%b " ", Din, Pout);
8
9   initial
10    begin
11      Din = 16'b0000000000000000;
12      #10 Din = 16'b0000000000000001;
13      #10 Din = 16'b0000100000000001;
14      #10 Din = 16'b0000100001000001;
15      #10 Din = 16'b1000100001000001;
16    end
17
18   initial #60 $finish;
19 endmodule

design.v
1 //Code your design here
2 module even_parity_16(Din, Pout);
3   input [15:0] Din;
4   output Pout;
5   reg [7:0] High_byte, Low_byte;
6   reg High, Low, Pout;
7
8   always@(Din)
9     begin
10      High_byte = Din[15:8];
11      Low_byte = Din[7:0];
12      High = even8(High_byte);
13      Low = even8(Low_byte);
14      Pout = High^Low; //bitwise xor
15    end
16
17   function even8;
18     input [7:0] I;
19     begin
20       even8 = ^I; //reduction xor operation
21     end
22   endfunction
23 endmodule

Log Share
[2024-04-11 01:09:36 UTC] iverilog -wall design.v testbench.v && unbuffer vvp a.out
0, Din=0000000000000000, Pout=0
10, Din=0000000000000001, Pout=1
20, Din=0000100000000001, Pout=0
30, Din=0000100001000001, Pout=1
40, Din=1000100001000001, Pout=0
Done

```

- design.v 程式

```
module even_parity_16(Din, Pout);
```

```
  input [15:0] Din;
```

```
  output Pout;
```

```
  reg [7:0] High_byte, Low_byte;
```

```
  reg High, Low, Pout;
```

```
  always@(Din)
```

```
    begin
```

```
      High_byte = Din[15:8];
```

```
      Low_byte = Din[7:0];
```

```
      High = even8(High_byte);
```

```
      Low = even8(Low_byte);
```

```
      Pout = High^Low; //bitwise xor
```

```
    end
```

```
  function even8;
```

```
    input [7:0] I;
```

```
    begin
```

```
      even8 = ^I; //reduction xor operation
```

```
    end
```

```
endfunction
endmodule
```

- testbench.sv

```
module testbench();
  reg [15:0] Din;
  wire Pout;
```

```
  even_parity_16 U1(Din, Pout);
```

```
  initial
```

```
    $monitor($time, ", Din=%b, Pout=%b ", Din, Pout);
```

```
  initial
```

```
    begin
```

```
      Din = 16'b0000000000000000;
```

```
      #10 Din = 16'b0000000000000001;
```

```
      #10 Din = 16'b0000100000000001;
```

```
      #10 Din = 16'b0000100001000001;
```

```
      #10 Din = 16'b1000100001000001;
```

```
    end
```

```
  initial #60 $finish;
```

```
endmodule
```

- 執行後結果

```
[2024-04-13 01:09:36 UTC] iverilog '-wall' design.sv testbench.sv
```

```
&& unbuffer vvp a.out
```

```
0, Din=0000000000000000, Pout=0
```

```
10, Din=0000000000000001, Pout=1
```

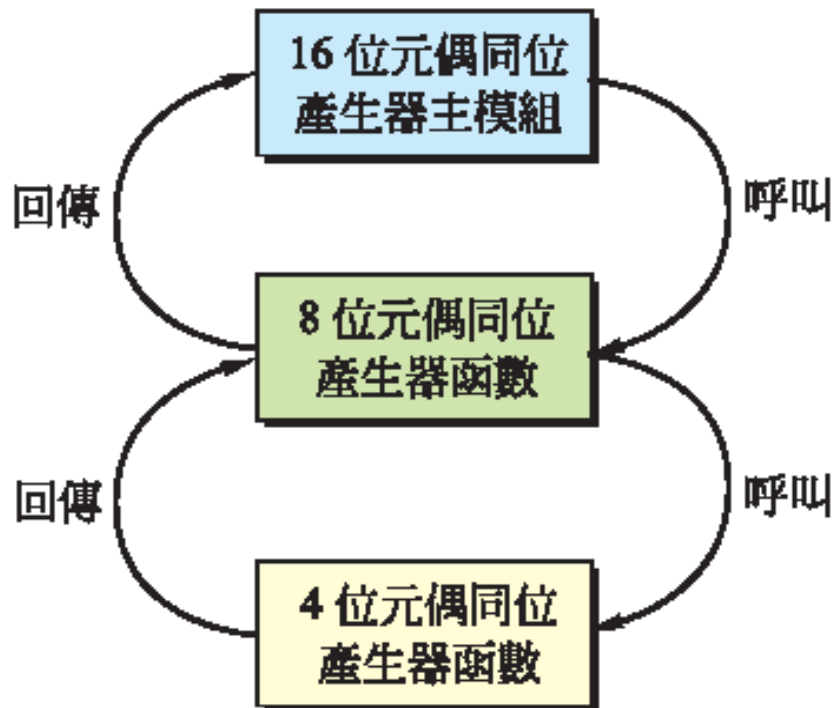
```
20, Din=0000100000000001, Pout=0
```

```
30, Din=0000100001000001, Pout=1
```

```
40, Din=1000100001000001, Pout=0
```

```
Done
```

2.2 函數呼叫函數



例題 2: 函數呼叫函數完成 16 位元偶同位產生器

```
1 // Code your testbench here
2 // or browse Examples
3 module testbench();
4   reg [15:0] Din;
5   wire Pout;
6
7   even16_fun_fun #1(Din, Pout);
8
9   initial
10    $monitor($time, ", Din=%b, Pout=%b ", Din, Pout);
11
12   initial
13   begin
14     Din = 16'b0000000000000000;
15     #10 Din = 16'b0000000000000001;
16     #10 Din = 16'b0000100000000001;
17     #10 Din = 16'b0000100001000001;
18     #10 Din = 16'b1000100001000001;
19   end
20
21   initial #60 $finish;
22
23 endmodule
```

```
1 // Code your design here
2 module even16_fun_fun(Din, Pout);
3   input [15:0] Din;
4   output Pout;
5   reg [7:0] High_byte, Low_byte;
6   reg High, Low;
7   reg Pout;
8
9   always@(Din)
10  begin
11    High_byte = Din[15:8];
12    Low_byte = Din[7:0];
13    High = even8(High_byte);
14    Low = even8(Low_byte);
15    Pout = High^Low; //bitwise xor
16  end
17
18  function even8;
19    input [7:0] I8;
20    begin
21      even8 = even4(I8[7:4])^even4(I8[3:0]); //xor operation
22    end
23  endfunction
24
```

Log

```
[2024-04-11 07:46:50 UTC] [verilog -wall] design.sv testbench.sv && unbuffer vvp a.out
0, Din=0000000000000000, Pout=0
10, Din=0000000000000001, Pout=1
20, Din=0000100000000001, Pout=0
30, Din=0000100001000001, Pout=1
40, Din=1000100001000001, Pout=0
```

- design.sv 程式

```
module even16_fun_fun(Din, Pout);
    input [15:0] Din;
    output Pout;
    reg [7:0] High_byte, Low_byte;
    reg High, Low;
    reg Pout;

    always@(Din)
        begin
            High_byte = Din[15:8];
            Low_byte = Din[7:0];
            High = even8(High_byte);
            Low = even8(Low_byte);
            Pout = High^Low; //Bitwise xor
        end

    function even8;
        input [7:0] I8;
        begin
            even8 = even4(I8[7:4])^even4(I8[3:0]); //xor operation
        end
    endfunction

    function even4;
        input [3:0] I4;
        begin
            even4 = ^I4; //reduction xor operation
        end
    endfunction
endmodule
```

- testbench.sv 程式

```
module testbench();
    reg [15:0] Din;
    wire Pout;
```

```

even16_fun_fun U1(Din, Pout);

initial
    $monitor($time, ", Din=%b, Pout=%b ", Din, Pout);

initial
    begin
        Din = 16'b0000000000000000;
        #10 Din = 16'b0000000000000001;
        #10 Din = 16'b0000100000000001;
        #10 Din = 16'b0000100001000001;
        #10 Din = 16'b1000100001000001;
    end

    initial #60 $finish;

endmodule

```

- 執行後之結果

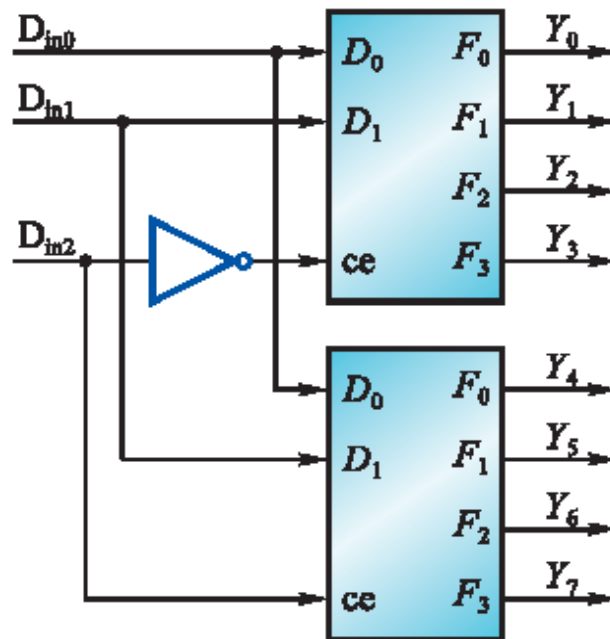
```

[2024-04-13 07:46:50 UTC] iverilog '-wall' design.sv testbench.sv
&& unbuffer vvp a.out
      0, Din=0000000000000000, Pout=0
     10, Din=0000000000000001, Pout=1
     20, Din=0000100000000001, Pout=0
     30, Din=0000100001000001, Pout=1
     40, Din=1000100001000001, Pout=0

Done

```

3. 作業 9-1: 以 2 對 4 解碼器產生 3 對 8 解碼器



```

● design.sv
module decod_fcn(Din, Y);
    input [2:0] Din;
    output [7:0] Y;
    reg [7:0] Ytmp;

    always@(Din)
        begin
            Ytmp[7:4] = decod4(Din[1:0], Din[2]);
            Ytmp[3:0] = decod4(Din[1:0], ~Din[2]);
        end
    assign Y = Ytmp;

    function [3:0] decod4;
        input [1:0] D;
        input ce;

        if(ce == 1'b1)
            begin
                case(D)
                    2'b00:decod4=4'b0001;
                    2'b01:decod4=4'b0010;
                endcase
            end
    endfunction

```

```
        2'b10:decod4=4'b0100;
        2'b11:decod4=4'b1000;
        default: decod4 = 4'b0000;
    endcase
end
else
    decod4 = 4'b0000;
endfunction

endmodule
```

(1) 請撰寫 testbench 執行上述程式，將結果貼到作業報告中，