第 6 講 行為描述---方塊程序與非方塊程序指定

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

2.1.1 Initial 區塊[3]

- Initial 區塊: 包含一個敘述 或 begin … end 含括多個敘述。

2.1.2 Always 區塊[3]

```
always @ (事件表示式 1 or 事件表示式 2 or ..... or 事件表示式 n)
    begin
        <敘述區>
    end
```

(1) 事件表示式中特定值之改變

```
always @(CLK)
    q=d;                    //當 CLK 之值改變時，執行 q=d
```

(2) 時脈信號正緣觸發時

```
always @ (posedge CLK)
    q=d;                    //當 CLK 正緣觸發時，執行 q=d
```

(3) 時脈信號負緣觸發時

```
always @ (negedge CLK)
    q=d;                    //當 CLK 負緣觸發時，執行 q=d
```

(4) 時脈信號或一個非同步事件

```
always @ (posedge CLK or negedge CLR)
        begin
            if (!CLR)
              q=l'b0;          //清除 q
            else
              q=d;             //載入 d
        end
```

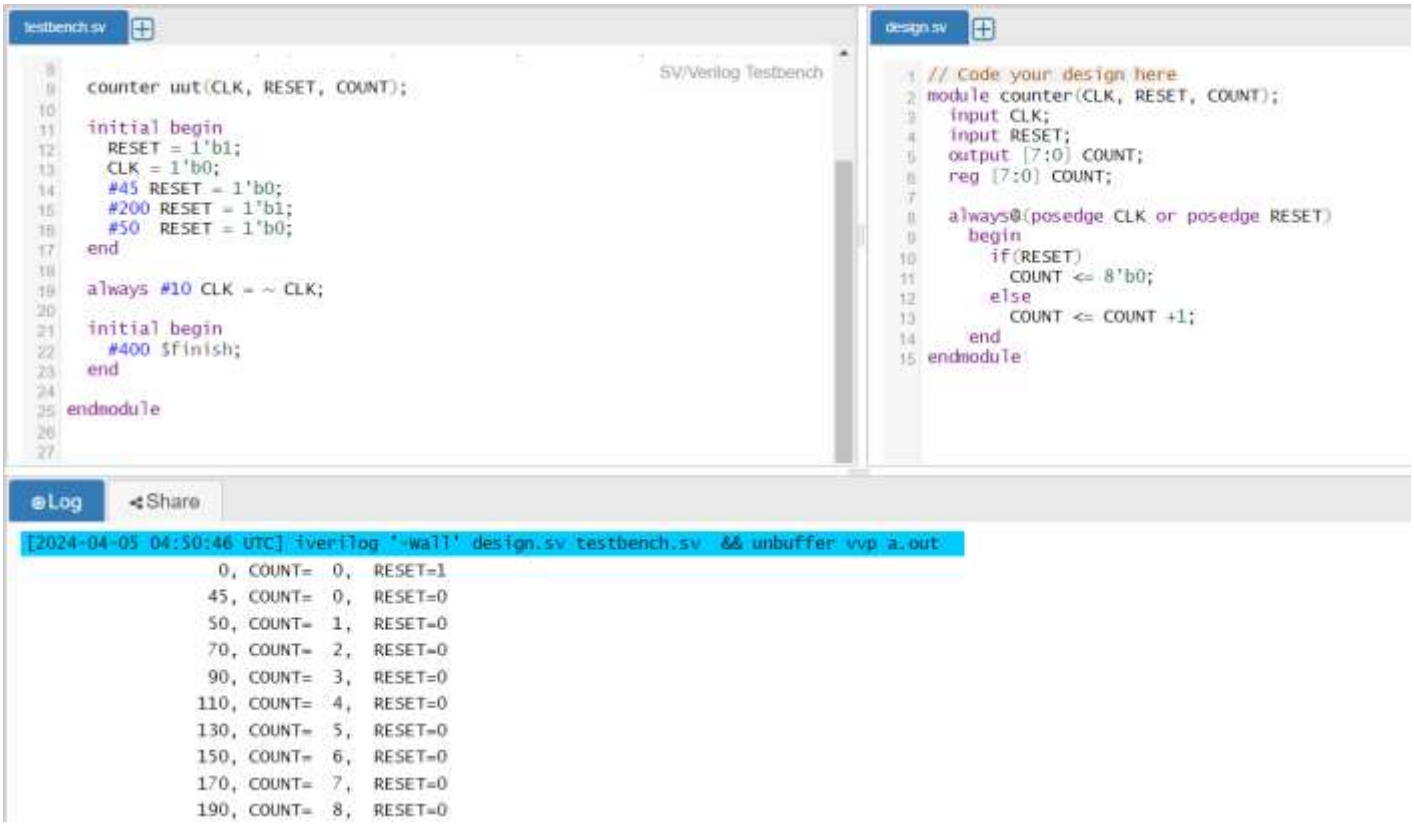
(5) 時脈信號或多個非同步事件

```
always @ (posedge CLK or posedge SET or negedge CLR)
        begin
            if (SET)
              q=l'b1;          //設定 q
        else if (!CLR)
            q=l'b0;            //清除 q
        else
            q=d;               //載入 d
        end
```

```
always @ (b or c)              //a 並無列於事件表示式中
    y = a | b | c;
```

```
always @ (a or b or c)
    y = a | b | c;
```

例題 1: 非同步重置 8 位元計數器



- **design.sv 程式:**

```
module counter(CLK, RESET, COUNT);
    input CLK;
    input RESET;
    output [7:0] COUNT;
    reg [7:0] COUNT;

    always@(posedge CLK or posedge RESET)
        begin
            if(RESET)
                COUNT <= 8'b0;
            else
                COUNT <= COUNT +1;
        end
endmodule
```

- **testbench 程式:**

```
module testbench();
```

```
    reg CLK, RESET; //inputs
    wire [7:0] COUNT; //outputs
    initial
        $monitor($time, '', COUNT=%d,    RESET=%b", COUNT[7:0], RESET);

    counter uut(CLK, RESET, COUNT);

    initial begin
        RESET = 1'b1;
        CLK = 1'b0;
        #45 RESET = 1'b0;
        #200 RESET = 1'b1;
        #50    RESET = 1'b0;
    end

    always #10 CLK = ~ CLK;

    initial begin
        #400 $finish;
    end

endmodule
```

- 執行後結果:

```
[2024-04-07 10:44:18 UTC] iverilog '-Wall' design.sv testbench.sv  &&
unbuffer vvp a.out
                   0, COUNT=  0,  RESET=1
                  45, COUNT=  0,  RESET=0
                  50, COUNT=  1,  RESET=0
                  70, COUNT=  2,  RESET=0
                  90, COUNT=  3,  RESET=0
                 110, COUNT=  4,  RESET=0
                 130, COUNT=  5,  RESET=0
                 150, COUNT=  6,  RESET=0
                 170, COUNT=  7,  RESET=0
                 190, COUNT=  8,  RESET=0
                 210, COUNT=  9,  RESET=0
                 230, COUNT= 10,  RESET=0
```

```
245, COUNT=  0,  RESET=1
295, COUNT=  0,  RESET=0
310, COUNT=  1,  RESET=0
330, COUNT=  2,  RESET=0
350, COUNT=  3,  RESET=0
370, COUNT=  4,  RESET=0
390, COUNT=  5,  RESET=0
```

Done

2.2.1 程序指定(方塊程序)

<識別字> = <時間控制> <表示式>;

以下均爲合法之方塊程序指定式：

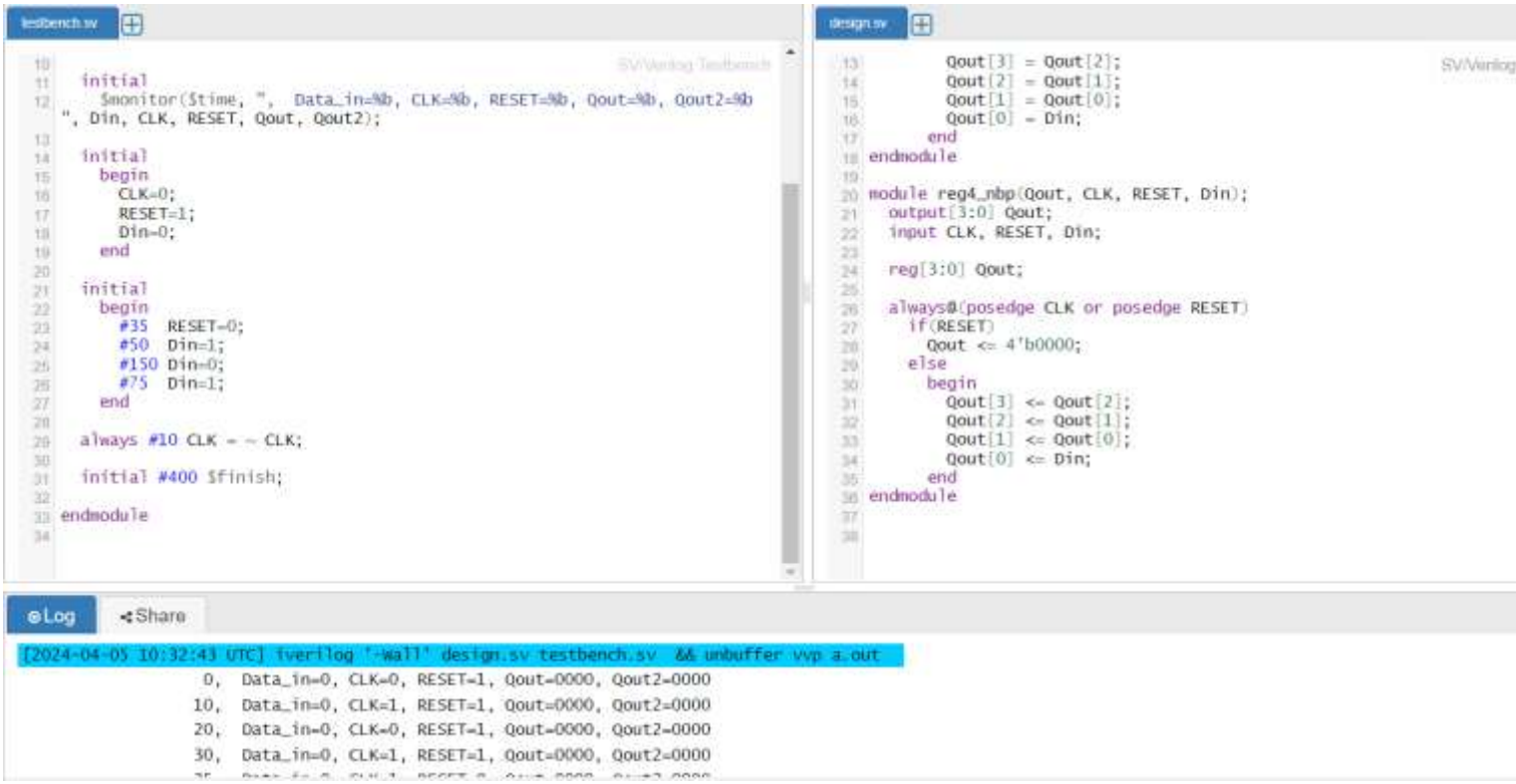
```
a[4] = 1'b1;
a[7:5] = 7;
{count,sum} = a + b + Cin;
```

2.2.2 程序指定(非方塊程序)

<識別字> <= <時間控制><表示式>;

```
a[3] <=1'b0;
a[3:0] <= 12
c <= {a[7:3],b[2:0]};
```

例題 2: 4 位元暫存器



- **design.sv 程式:**

```
module reg4_bp(Qout, CLK, RESET, Din);
   output[3:0] Qout;
   input CLK, RESET, Din;

   reg[3:0] Qout;

   always@(posedge CLK or posedge RESET)
      if(RESET)
         Qout = 4'b0000;
      else
         begin
            Qout[3] = Qout[2];
            Qout[2] = Qout[1];
            Qout[1] = Qout[0];
            Qout[0] = Din;
         end
endmodule

module reg4_nbp(Qout, CLK, RESET, Din);
   output[3:0] Qout;
```

```
    input CLK, RESET, Din;

    reg[3:0] Qout;

    always@(posedge CLK or posedge RESET)
        if(RESET)
            Qout <= 4'b0000;
        else
            begin
                Qout[3] <= Qout[2];
                Qout[2] <= Qout[1];
                Qout[1] <= Qout[0];
                Qout[0] <= Din;
            end
endmodule
```

- **testbench 程式:**

```
module testbench();
    reg CLK, RESET, Din;

    wire [3:0] Qout, Qout2;

    reg4_bp    reg4_1(Qout,    CLK, RESET, Din);
    reg4_nbp reg4_2(Qout2, CLK, RESET, Din);

    initial
        $monitor($time, ",    Data_in=%b, CLK=%b, RESET=%b, Qout=%b, Qout2=%b    ",
Din, CLK, RESET, Qout, Qout2);

    initial
        begin
            CLK=0;
            RESET=1;
            Din=0;
        end

    initial
        begin
```

```
        #35    RESET=0;
        #50    Din=1;
        #150 Din=0;
        #75    Din=1;
      end

    always #10 CLK = ~ CLK;

    initial #400 $finish;

endmodule
```

- 執行後結果:

```
[2024-04-07 10:46:25 UTC] iverilog '-Wall' design.sv testbench.sv  &&
unbuffer vvp a.out
                0,  Data_in=0, CLK=0, RESET=1, Qout=0000, Qout2=0000
               10,  Data_in=0, CLK=1, RESET=1, Qout=0000, Qout2=0000
               20,  Data_in=0, CLK=0, RESET=1, Qout=0000, Qout2=0000
               30,  Data_in=0, CLK=1, RESET=1, Qout=0000, Qout2=0000
               35,  Data_in=0, CLK=1, RESET=0, Qout=0000, Qout2=0000
               40,  Data_in=0, CLK=0, RESET=0, Qout=0000, Qout2=0000
               50,  Data_in=0, CLK=1, RESET=0, Qout=0000, Qout2=0000
               60,  Data_in=0, CLK=0, RESET=0, Qout=0000, Qout2=0000
               70,  Data_in=0, CLK=1, RESET=0, Qout=0000, Qout2=0000
               80,  Data_in=0, CLK=0, RESET=0, Qout=0000, Qout2=0000
               85,  Data_in=1, CLK=0, RESET=0, Qout=0000, Qout2=0000
               90,  Data_in=1, CLK=1, RESET=0, Qout=0001, Qout2=0001
              100,  Data_in=1, CLK=0, RESET=0, Qout=0001, Qout2=0001
              110,  Data_in=1, CLK=1, RESET=0, Qout=0011, Qout2=0011
              120,  Data_in=1, CLK=0, RESET=0, Qout=0011, Qout2=0011
              130,  Data_in=1, CLK=1, RESET=0, Qout=0111, Qout2=0111
              140,  Data_in=1, CLK=0, RESET=0, Qout=0111, Qout2=0111
              150,  Data_in=1, CLK=1, RESET=0, Qout=1111, Qout2=1111
              160,  Data_in=1, CLK=0, RESET=0, Qout=1111, Qout2=1111
              170,  Data_in=1, CLK=1, RESET=0, Qout=1111, Qout2=1111
              180,  Data_in=1, CLK=0, RESET=0, Qout=1111, Qout2=1111
              190,  Data_in=1, CLK=1, RESET=0, Qout=1111, Qout2=1111
              200,  Data_in=1, CLK=0, RESET=0, Qout=1111, Qout2=1111
```

```
210,  Data_in=1, CLK=1, RESET=0, Qout=1111, Qout2=1111
220,  Data_in=1, CLK=0, RESET=0, Qout=1111, Qout2=1111
230,  Data_in=1, CLK=1, RESET=0, Qout=1111, Qout2=1111
235,  Data_in=0, CLK=1, RESET=0, Qout=1111, Qout2=1111
240,  Data_in=0, CLK=0, RESET=0, Qout=1111, Qout2=1111
250,  Data_in=0, CLK=1, RESET=0, Qout=1110, Qout2=1110
260,  Data_in=0, CLK=0, RESET=0, Qout=1110, Qout2=1110
270,  Data_in=0, CLK=1, RESET=0, Qout=1100, Qout2=1100
280,  Data_in=0, CLK=0, RESET=0, Qout=1100, Qout2=1100
290,  Data_in=0, CLK=1, RESET=0, Qout=1000, Qout2=1000
300,  Data_in=0, CLK=0, RESET=0, Qout=1000, Qout2=1000
310,  Data_in=1, CLK=1, RESET=0, Qout=0001, Qout2=0001
320,  Data_in=1, CLK=0, RESET=0, Qout=0001, Qout2=0001
330,  Data_in=1, CLK=1, RESET=0, Qout=0011, Qout2=0011
340,  Data_in=1, CLK=0, RESET=0, Qout=0011, Qout2=0011
350,  Data_in=1, CLK=1, RESET=0, Qout=0111, Qout2=0111
360,  Data_in=1, CLK=0, RESET=0, Qout=0111, Qout2=0111
370,  Data_in=1, CLK=1, RESET=0, Qout=1111, Qout2=1111
380,  Data_in=1, CLK=0, RESET=0, Qout=1111, Qout2=1111
390,  Data_in=1, CLK=1, RESET=0, Qout=1111, Qout2=1111
400,  Data_in=1, CLK=0, RESET=0, Qout=1111, Qout2=1111
```

Done

3. 作業 6-1: 重做例題 2，但將 **design.sv** 程式修改(如紅字所示)如下:

```
module reg4_bp(Qout, CLK, RESET, Din);
.....
.....
.....
        Begin
           Qout[0] = Din;
           Qout[1] = Qout[0];
           Qout[2] = Qout[1];
           Qout[3] = Qout[2];
        end

module reg4_nbp(Qout, CLK, RESET, Din);
....
....
....
        Begin
           Qout[0] <= Din;
           Qout[1] <= Qout[0];
           Qout[2] <= Qout[1];
           Qout[3] <= Qout[2];
        end
```

(1)請執行上述程式，將執行結果貼到作業報告中，

(2)請問其執行結果與例題 2 有何不同 ？ 為何不同，請說明理由。