附3：verilog function函数用法

语法

```
function <返回值的类型或范围>(函数名);
    <端口说明语句>          // input XXX
    <变量类型说明语句>       // reg YYY
    ......
begin
    <语句>
    ......
    函数名 = ZZZ;           // 函数名就相当于输出变量;
end
endfunction
```

函数用关键词 `function` 声明，并用 `endfunction` 结束，不允许输出端口声明（包括输出和双向端口），但可以有多个输入端口。函数只返回一个值到函数被调用的位置，并且在函数中返回值与函数名同名。

函数的定义如下所示：

```
function [range] function_id;
  input_declaration
  other_declarations
  procedural_statement
endfunction
```

在使用函数时有以下几点需要注意：

1. 函数定义只能在模块中完成，不能出现在过程块中；
2. 函数至少要有一个输入端口；不能包含输出端口和双向端口；
3. 在函数结构中， 不能使用任何形式的时间控制语句 （`#`、`wait` 等） ， 也不能使用 `disable` 中止语句；
4. 函数定义结构体中不能出现过程块语句（`always` 语句）；
5. 函数内部可以调用函数，但不能调用任务。

一个简单的例子：

```
module comb15 (A, B, CIN, S, COUT);
  input [3:0] A, B;
  input CIN;
  output [3:0] S;
  output COUT;

  wire [1:0] S0, S1, S2, S3;

  function signed [1:0] ADD;

    input A, B, CIN;

    reg S, COUT;
```

```
14
15      begin
16        S = A ^ B ^ CIN;
17        COUT = (A&B) | (A&CIN) | (B&CIN);
18        ADD = {COUT, S};
19      end
20    endfunction
21
22    assign S0 = ADD (A[0], B[0], CIN),
23    S1 = ADD (A[1], B[1], S0[1]),
24    S2 = ADD (A[2], B[2], S1[1]),
25    S3 = ADD (A[3], B[3], S2[1]),
26    S = {S3[0], S2[0], S1[0], S0[0]},
27    COUT = S3[1];
28  endmodule
```

在函数调用中，有下列几点需要注意：

1. 函数调用可以在过程块中完成，也可以在 `assign` 这样的连续赋值语句中出现。
2. 函数调用语句不能单独作为一条语句出现，只能作为赋值语句的右端操作数。

例子

P1_L9_Comparator

文件上传

题目编号 1109-202

提交要求

使用Verilog搭建一个四位有符号比较器并提交。具体模块端口定义如下：

信号名	方向	描述
A[3:0]	I	第一个四位有符号数
B[3:0]	I	第二个四位有符号数
Out	O	一位输出 1:A小于B 0:A不小于B

- 必须严格按照模块的端口定义
- 文件内模块名: comparator
- 注意是小于!
- 输入的A和B为四位数，不是五位数，且为补码的形式输入，最高位（符号位）+其余三位构成一个输入
- **写代码时不能直接用>或者<(为了避免误判，非阻塞赋值符号"<="也敬请避免使用。)**

分析

由于不能使用`>`，可以使用函数封装一位比较器`bitCmp`

```
function [1:0] bitCmp;//1:小于;2:等于;3:大于
    input a,b;

    begin
        if(a==b) bitCmp = 2;
        else if(a==1 && b==0) bitCmp=3;
        else if(a==0 && b==1) bitCmp=1;
    end

endfunction
```

然后仿照理论课`Comparator`的电路设计即可。

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    22:54:04 10/12/2023
// Design Name:
// Module Name:    P1_L9_Comparator
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
module comparator(
    input [3:0] A,
    input [3:0] B,
    output reg Out
    );


    function [1:0] bitCmp;//1:小于;2:等于;3:大于
        input a,b;

        begin
            if(a==b) bitCmp = 2;
            else if(a==1 && b==0) bitCmp=3;
            else if(a==0 && b==1) bitCmp=1;
        end
```

```
36
37      endfunction
38
39
40      always @(*) begin
41
42          if (A[3]==0 && B[3]==1) Out = 0;
43          else if(A[3]==1 && B[3]==0) Out = 1;
44          else begin
45              if ((bitCmp(A[2],B[2])==1)
46              || (bitCmp(A[2],B[2])==2 && bitCmp(A[1],B[1])==1)
47              || (bitCmp(A[2],B[2])==2 && bitCmp(A[1],B[1])==2 &&
bitCmp(A[0],B[0])==1))
48                  Out = 1;
49              else
50                  Out = 0;
51          end
52
53      end
54
55  endmodule
```