电 子 设 计 自 动 化

作 业 二

姓　　名	**黄智越**
学　　号	**23S136049**

实验一 测试 define 增强语句的编译结果

一、 实验目的及方案

实验目的：

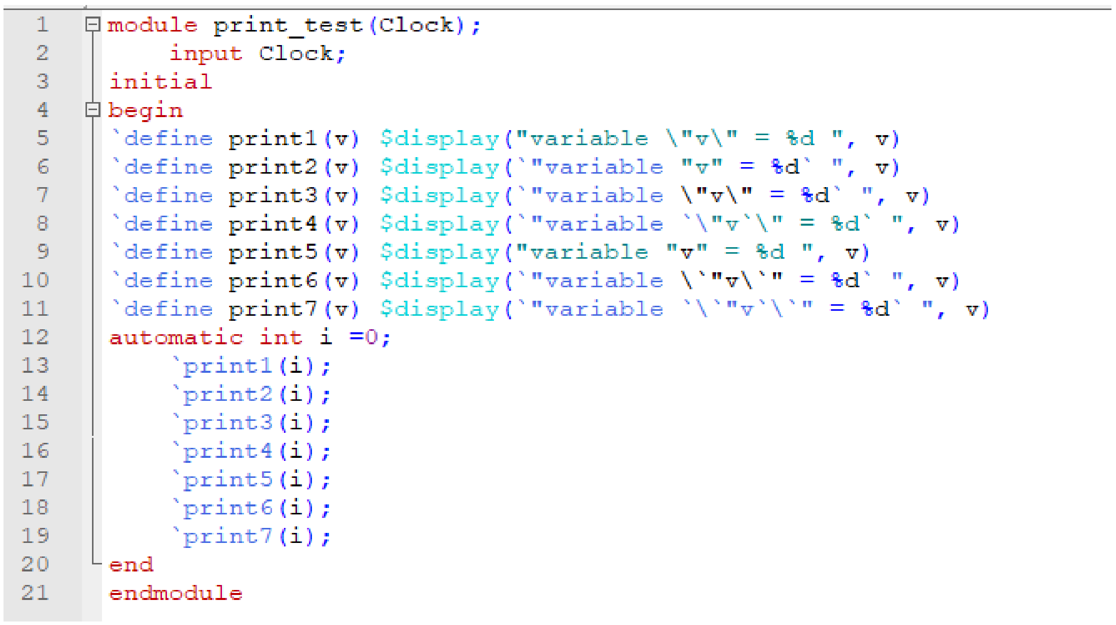
（1）　了解 define 中宏常量如何在输出的引号中灵活变化，而不是保持字符串面量

验证方案：

　　编译如下的代码

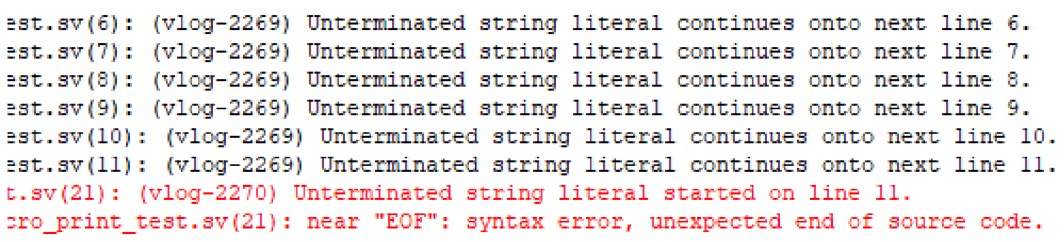
二、 实验内容

1、 书写原代码



```
module print_test(Clock);
     input Clock;
initial
begin
`define print1(v) $display("variable \"v\" = %d ", v)
`define print2(v) $display(`"variable "v" = %d` ", v)
`define print3(v) $display(`"variable \"v\" = %d` ", v)
`define print4(v) $display(`"variable `\"v`\" = %d` ", v)
`define print5(v) $display("variable "v" = %d ", v)
`define print6(v) $display(`"variable \`"v\`" = %d` ", v)
`define print7(v) $display(`"variable `\`"v`\`" = %d` ", v)
automatic int i =0;
     `print1(i);
     `print2(i);
     `print3(i);
     `print4(i);
     `print5(i);
     `print6(i);
     `print7(i);
end
endmodule
```

2、 进行编译查看结果



```
est.sv(6): (vlog-2269) Unterminated string literal continues onto next line 6.
est.sv(7): (vlog-2269) Unterminated string literal continues onto next line 7.
est.sv(8): (vlog-2269) Unterminated string literal continues onto next line 8.
est.sv(9): (vlog-2269) Unterminated string literal continues onto next line 9.
est.sv(10): (vlog-2269) Unterminated string literal continues onto next line 10.
est.sv(11): (vlog-2269) Unterminated string literal continues onto next line 11.
t.sv(21): (vlog-2270) Unterminated string literal started on line 11.
cro_print_test.sv(21): near "EOF": syntax error, unexpected end of source code.
```

三、 实验结果及分析

编译结果显示：从第六行到 11 行代码里出现了未终止的字符串面量，全都是因为反向单引号和双引号打的位置不合理，经过注释其他代码发现其中 print1()宏定义才是正确的写法。

实验二 删除前缀 Logic

一、 实验目的及方案

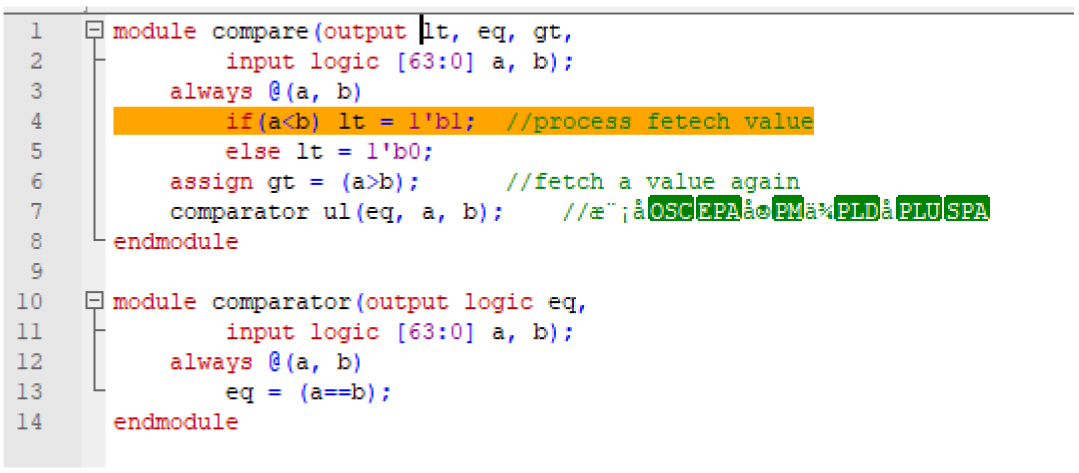
实验目的：

（1）了解模块参数中删除 logic 会发生什么

验证方案：

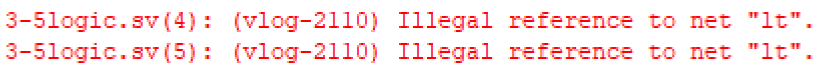
书写下列代码并编译

二、 实验内容



```
module compare(output lt, eq, gt,
        input logic [63:0] a, b);
    always @(a, b)
        if(a<b) lt = 1'b1;  //process fetech value
        else lt = 1'b0;
    assign gt = (a>b);      //fetch a value again
    comparator u1(eq, a, b);    //æ¨¡åOSCEPAå®PMä¾PLDåPLUSPA
endmodule

module comparator(output logic eq,
        input logic [63:0] a, b);
    always @(a, b)
        eq = (a==b);
endmodule
```

三、实验结果及分析



```
3-5logic.sv(4): (vlog-2110) Illegal reference to net "lt".
3-5logic.sv(5): (vlog-2110) Illegal reference to net "lt".
```

无法解析 Lt,因为 systemverilog 中变量不仅具有 4 态还具有 2 态的所以 systemverilog 中需要明确写出数据是两态还是四态。

实验三 避免变量的多重驱动

一、 实验目的及方案

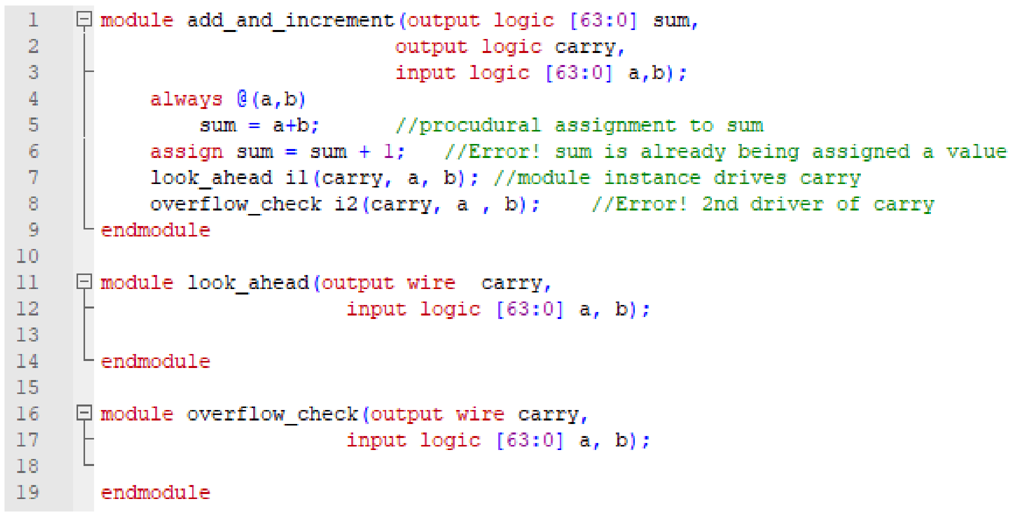
实验目的：

（1） 认识到变量只能有一个驱动源

验证方案:

二、 实验内容

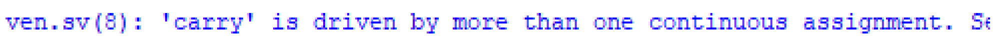
代码如图所示:



```
module add_and_increment(output logic [63:0] sum,
                        output logic carry,
                        input logic [63:0] a,b);
    always @(a,b)
        sum = a+b;      //procudural assignment to sum
    assign sum = sum + 1;   //Error! sum is already being assigned a value
    look_ahead i1(carry, a, b); //module instance drives carry
    overflow_check i2(carry, a , b);    //Error! 2nd driver of carry
endmodule

module look_ahead(output wire  carry,
                    input logic [63:0] a, b);

endmodule

module overflow_check(output wire carry,
                    input logic [63:0] a, b);

endmodule
```

三、 实验结果及分析

Sum 编译报错，因为它既被过程赋值，又被 sum=sum+1 赋值一次，一个变量只能有一个源，所以编译器报错。
而 carry 默认为变量类型，所以在运行时才会报错有多个驱动源，具体如下图。



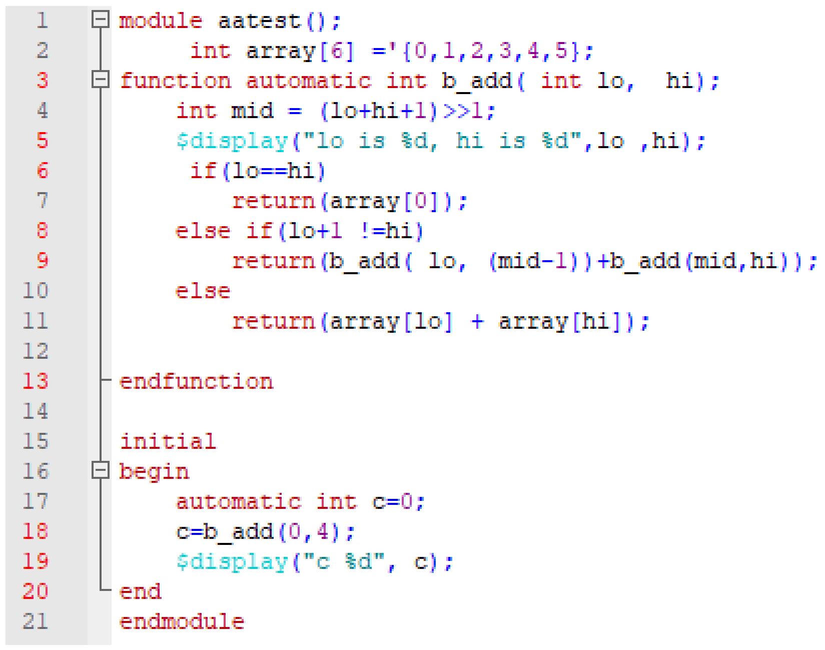
```
ven.sv(8): 'carry' is driven by more than one continuous assignment. Se
```

实验四 跟踪递归实现过程

一、 实验目的及方案

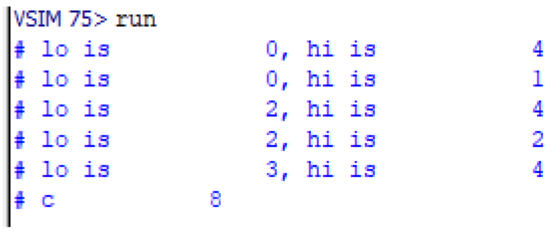
实验目的：
（1）熟悉递归函数的运行流程。
验证方案：
根据代码加入 display 显示各个函数调用

二、 实验内容

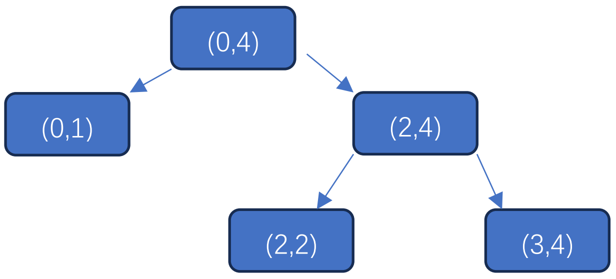


```
module aatest();
     int array[6] ='{0,1,2,3,4,5};
function automatic int b_add( int lo,  hi);
    int mid = (lo+hi+1)>>1;
    $display("lo is %d, hi is %d",lo ,hi);
     if(lo==hi)
        return(array[0]);
    else if(lo+1 !=hi)
        return(b_add( lo, (mid-1))+b_add(mid,hi));
    else
        return(array[lo] + array[hi]);

endfunction

initial
begin
    automatic int c=0;
    c=b_add(0,4);
    $display("c %d", c);
end
endmodule
```

三、实验结果及分析



```
VSIM 75> run
# lo is          0, hi is          4
# lo is          0, hi is          1
# lo is          2, hi is          4
# lo is          2, hi is          2
# lo is          3, hi is          4
# c          8
```



递归过程如上。

实验五 count_ones 跟踪

一、 实验目的及方案

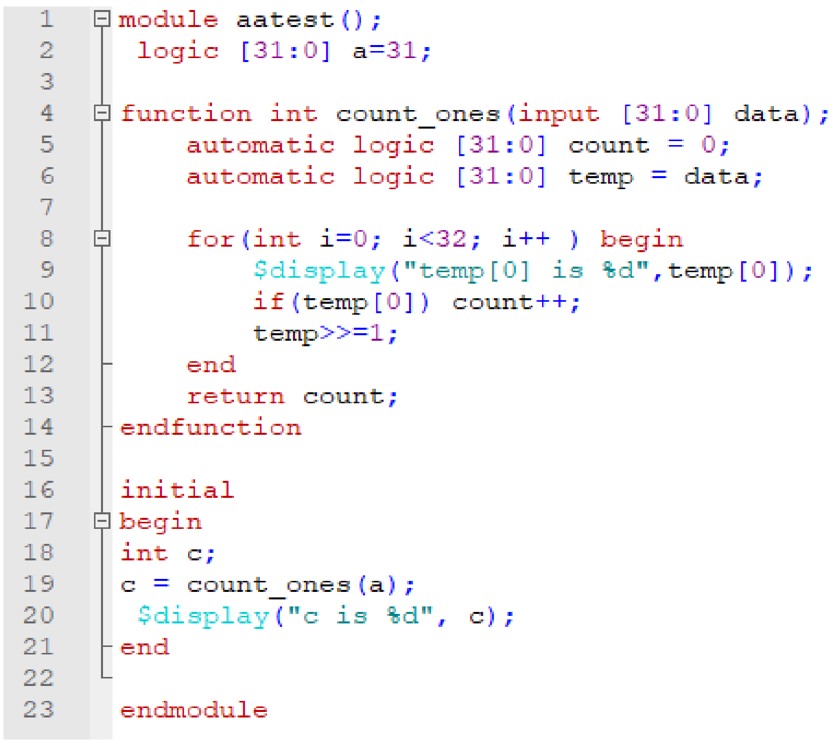
实验目的：

（1）学习如何单步执行代码

验证方案:

通过在 run 之前设置断点，从而当程序执行到指定断点时停下来

二、 实验内容

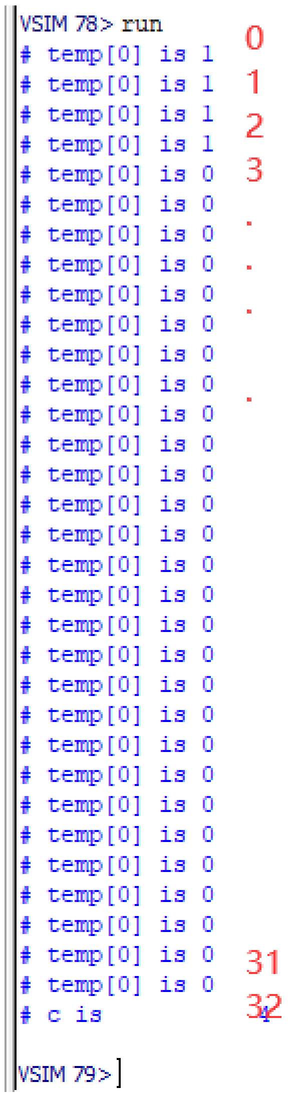


```
module aatest();
 logic [31:0] a=31;

function int count_ones(input [31:0] data);
    automatic logic [31:0] count = 0;
    automatic logic [31:0] temp = data;

    for(int i=0; i<32; i++ ) begin
        $display("temp[0] is %d",temp[0]);
        if(temp[0]) count++;
        temp>>=1;
    end
    return count;
endfunction

initial
begin
int c;
c = count_ones(a);
 $display("c is %d", c);
end

endmodule
```

三、实验结果及分析

其结果如下图所示，由上到下表示低位到高位。

选取 a = 15 ,二进制为 0000 0000 0000 0000 0000 0000 0000 1111,所以计数为 4



```
VSIM 78> run
# temp[0] is 1
# temp[0] is 1
# temp[0] is 1
# temp[0] is 1
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# temp[0] is 0
# c is                  4

VSIM 79>
```

实验六 count_ones 改正

一、 实验目的及方案

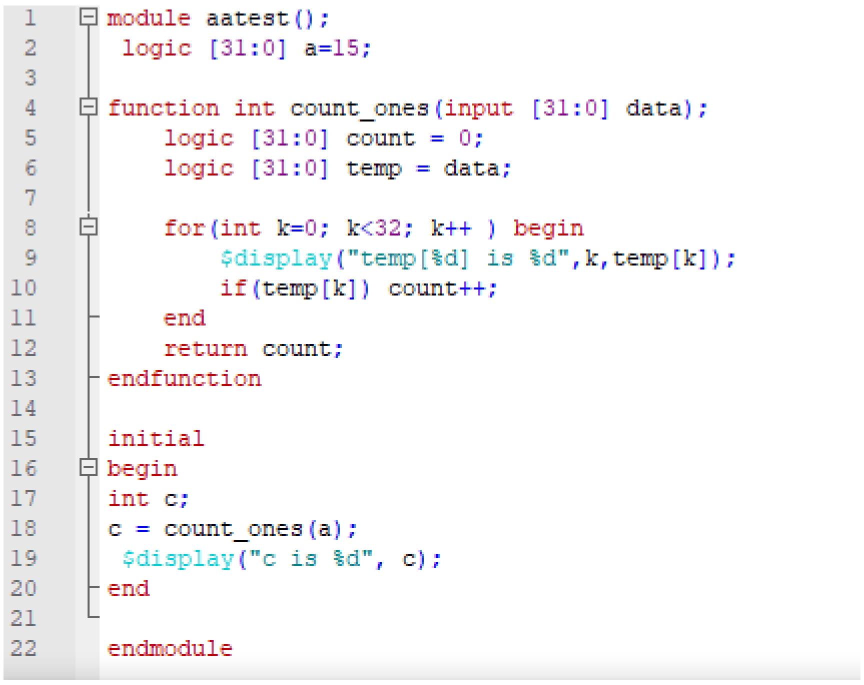
实验目的：

（1）体会静态变量和动态变量在程序执行过程中的不同

验证方案:

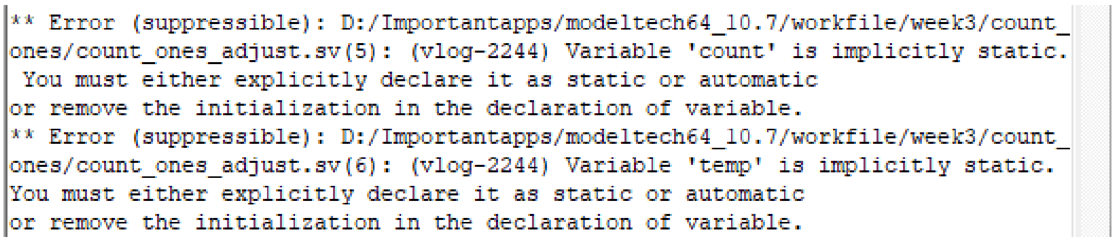
实验五已经验证设置为动态变量的情况;

二、 实验内容



```
module aatest();
 logic [31:0] a=15;

function int count_ones(input [31:0] data);
    logic [31:0] count = 0;
    logic [31:0] temp = data;

    for(int k=0; k<32; k++ ) begin
        $display("temp[%d] is %d",k,temp[k]);
        if(temp[k]) count++;
    end
    return count;
endfunction

initial
begin
int c;
c = count_ones(a);
 $display("c is %d", c);
end

endmodule
```

三、实验结果及分析



```
** Error (suppressible): D:/Importantapps/modeltech64_10.7/workfile/week3/count_
ones/count_ones_adjust.sv(5): (vlog-2244) Variable 'count' is implicitly static.
 You must either explicitly declare it as static or automatic
or remove the initialization in the declaration of variable.
** Error (suppressible): D:/Importantapps/modeltech64_10.7/workfile/week3/count_
ones/count_ones_adjust.sv(6): (vlog-2244) Variable 'temp' is implicitly static.
You must either explicitly declare it as static or automatic
or remove the initialization in the declaration of variable.
```

编译结果显示 count 和 temp 必须显示的申明为静态或动态。如果函数只被调用一次，用静态和动态申明对结果都无影响，但是如果函数被调用多次，就必须要申明为动态，因为静态变量只会被初始化一次，第二次及以后 count 不会被初始化为 0