

## Verilog 第 5 次实验报告

实验名称	实现不同位加法器				
学生姓名	邢清画	学号	2211999	指导老师	董前琨
实验地点	津南实验楼 A 区 308		实验时间	2023 年 11 月 29 日	

### 一、实验项目名称

统计 32 位 2 进制数 0 和 1 出现的次数

### 二、实验目的

本实验旨在运用 Verilog 语言自行设计一个模块，完成统计 32 位二进制数中 0 和 1 出现的次数。这个任务要求我们可以运用所学的 Verilog 程序设计知识，特别是赋值语句、块语句、条件语句等，来实现一个计数器模块。通过这个实验，能够更好地理解和掌握 Verilog 语言的基本语法和使用方法，并在实际的编程实践中加深对数字逻辑设计的理解

### 三、必修或选修

选修

### 四、实验平台

Vivado

### 五、实验内容及步骤

#### 实验内容：

1. 定义了一个 counter01 模块，有一个 32 位的输入 cin，两个输出 sum0 和 sum1，分别用于输出 0 和 1 出现的次数。
2. 在 always 块中，使用一个循环来遍历 32 位的输入 cin，对每一位进行判断，如果是 0 则 sum0 加 1，如果是 1 则 sum1 加 1。
3. 用 random 得到一个随机的三十二位二进制 cin，带入 counter01 的函数中，获得该数中 0 和 1 的次数。

代码如下：

```
module counter01(input [31:0]cin,
output reg[6:0]sum0,
output reg[6:0]sum1 );
integer i;//定义一个整数 i
always@(cin[31:0])
begin
sum0=0;//令 sum0=0, sum1=0
sum1=0;
```

```

for(i=0;i<32;i=i+1)//循环 32 次，判断 cin 每位上为 0 还是 1
begin
    if(cin[i]==0)
        begin
            sum0=sum0+1;//如果 num 该位上为 0 则 time0+1
        end
    else
        begin
            sum1=sum1+1;//如果 num 该位上为 1 则 time1+1
        end
    end
end
end
endmodule

```

```

module tb01( );
reg [31:0]cin;
wire [6:0]sum0;
wire [6:0]sum1;//输入 cin, 输出 sum0, sum1
counter01 utt(cin,sum0,sum1);
initial begin
cin=32'b0000_0000_0000_0000_0000_0000_0000_0000;
end
always#33cin=$random%33'b1_0000_0000_0000_0000_0000_0000_0000;//获得一个三
十二位二进制随机数
endmodule

```

The screenshot shows a Verilog IDE with two files open: 'counter01.v' and 'tb01.v'. The 'counter01.v' file contains the implementation of a counter module. The code is as follows:

```

// Revision 0.01 - File Created
// Additional Comments:
//
module counter01(input [31:0]cin,
output reg[6:0]sum0,
output reg[6:0]sum1);
integer i;//定义一个整数i
always@(cin[31:0])
begin
sum0=0;//令sum0=0, sum1=0
sum1=0;
for(i=0;i<32;i=i+1)//循环32次，判断cin每位上为0还是1
begin
if(cin[i]==0)
begin
sum0=sum0+1;//如果num该位上为0则time0+1
end
else
begin
sum1=sum1+1;//如果num该位上为1则time1+1
end
end
end
endmodule

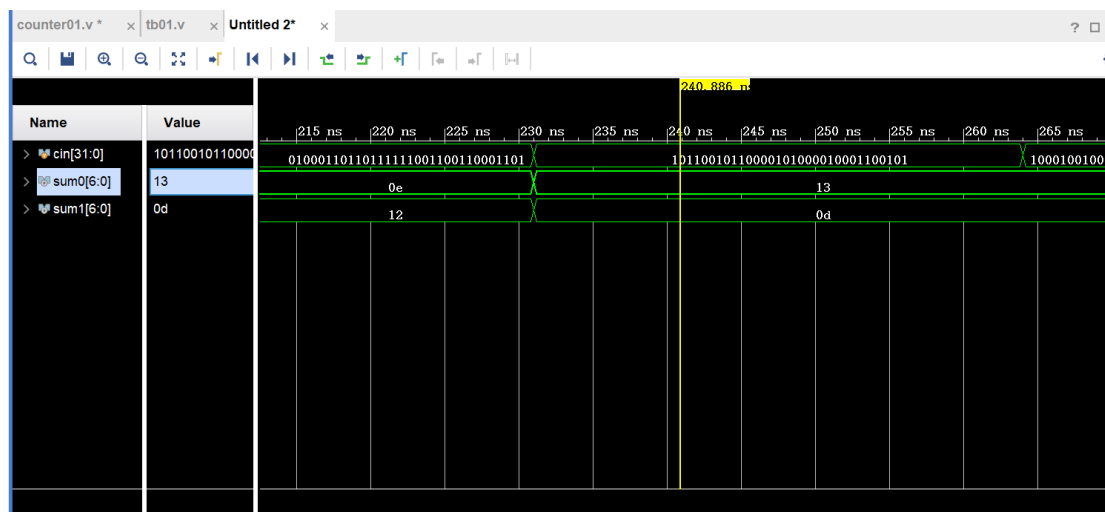
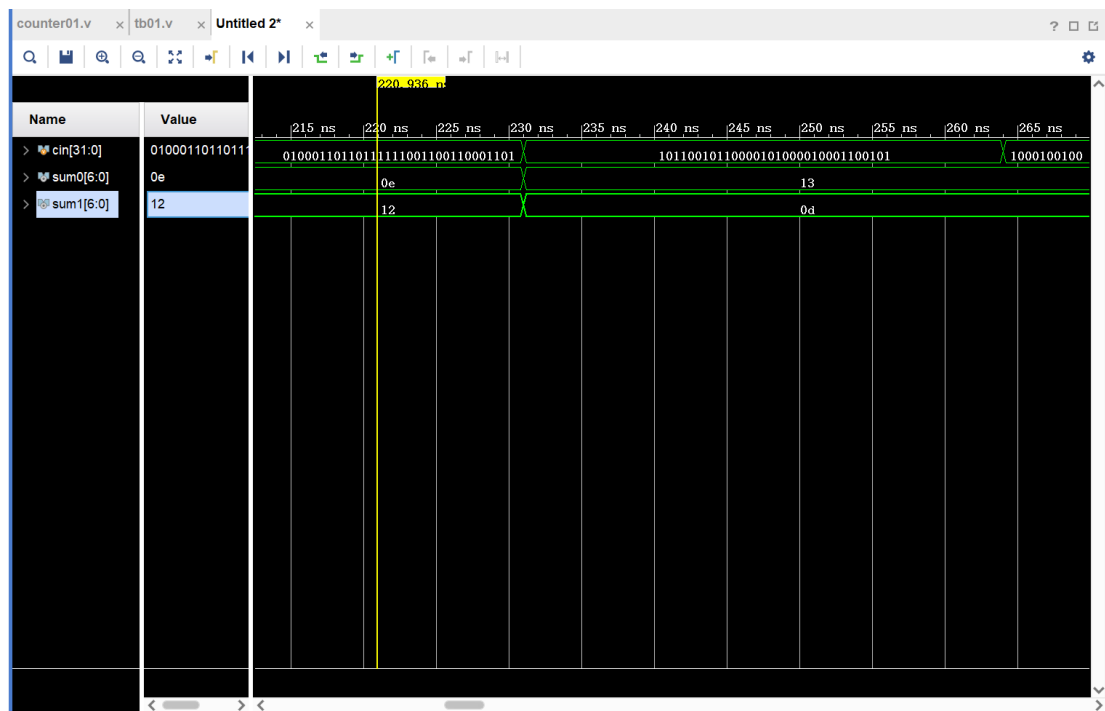
```

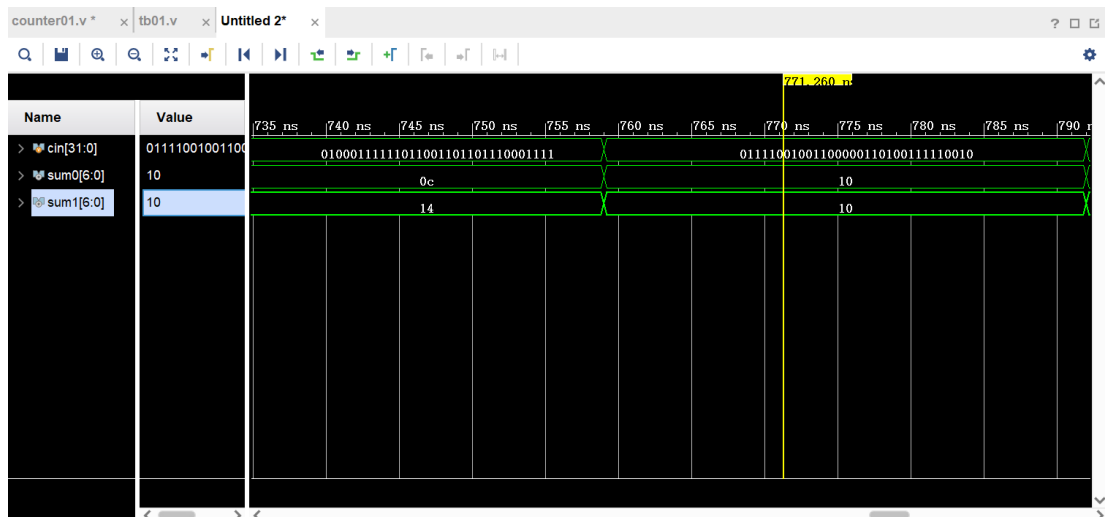
The 'tb01.v' file contains the testbench code, which is partially visible in the previous block. The IDE interface includes a toolbar with various icons for file operations, editing, and simulation. The file explorer on the right shows the project structure.

```
counter01.v * x tb01.v x Untitled 2* x
D:/verilog/project_5/srcs/sim_1/new/tb01.v

15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 ///////////////////////////////////////////////////////////////////
21
22
23 module tb01();
24 reg [31:0] cin;
25 wire [6:0] sum0;
26 wire [6:0] sum1; //输入cin, 输出sum0, sum1
27 counter01 uut(cin, sum0, sum1);
28 initial begin
29     cin=32'b0000_0000_0000_0000_0000_0000_0000_0000;
30 end
31 always#33cin=$random%33'b1_0000_0000_0000_0000_0000_0000_0000_0000; //获得一个三十二位二进制随机数
32 endmodule
33
```

## 六、关键问题讨论





如上波形图所示：在这个模块中

图 1: cin= 00011110100011011100110100111101, 其中 sum0=0e (16 进制)=14, sum1=12 (16 进制)=18, 加和结果为 32, 结果正确。

图 2: cin= 10001001001100101101011000010010, 其中 sum0=13 (16 进制)=19, sum1=0d (16 进制)=13, 加和结果为 32, 结果正确

图 3: cin= 01111001001100000110100111110010, 其中 sum0=10 (16 进制)=16, sum1=10 (16 进制)=10, 加和结果为 32, 结果正确

## 七、总结

更加深入理解 Verilog 硬件描述语言在设计和验证数字电路中的应用。学习了 for 循环语句, 并使用在对 32 位输入遍历和判断中, 实现对于 0 和 1 个数的统计。reg 通常用于存储状态或中间计算结果, wire 只能用于连接各个逻辑元素而不能用于存储状态。