**组成原理课程第 1 次实验报告**

**实验名称：32位减法器**

学号： 2312141 姓名： 张德民 班次： 李涛老师班

## 实验目的

1. 熟悉 LS-CPU-EXB-002 实验箱和软件平台。

2. 掌握利用该实验箱各项功能开发组成原理和体系结构实验的方法。

3. 理解并掌握加法器的原理和设计。

4. 熟悉并运用 verilog 语言进行电路设计。

5. 为后续设计 cpu 的实验打下基础。

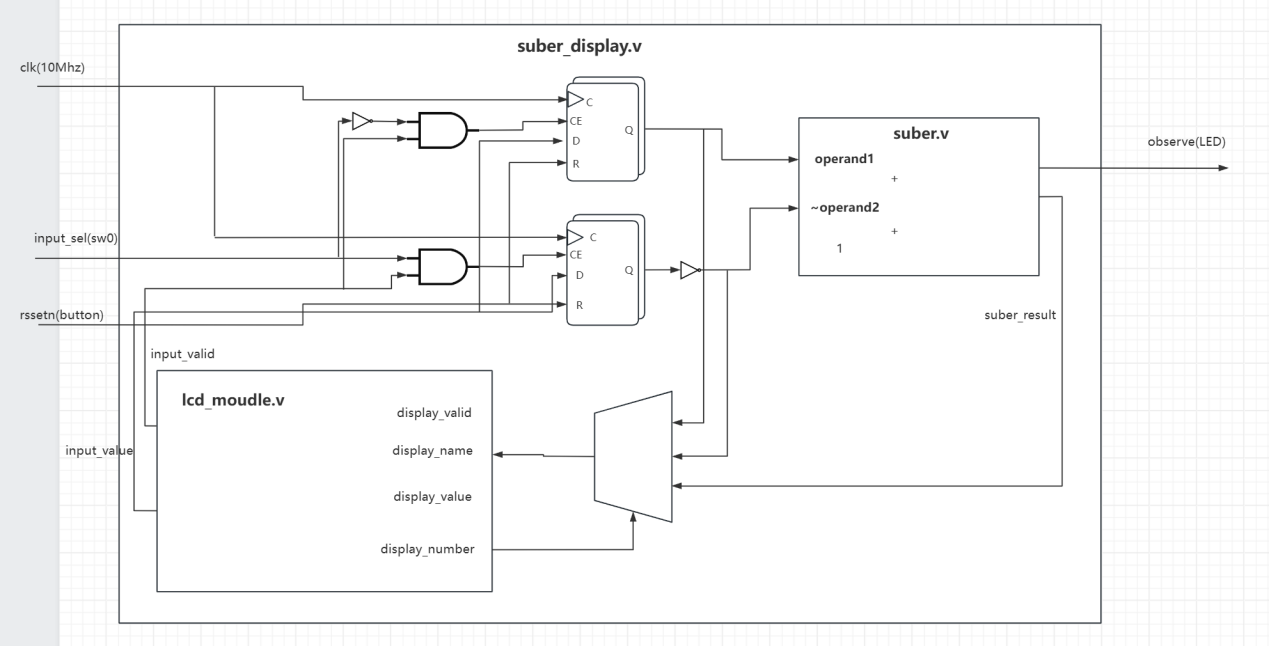
## 实验内容说明

本次实验内容主要为根据第一次定点加法器实验进行学习改进，实现一个减法器，该减法器要求如下：

减法器拿到的32位输入都应该理解成数据的补码形式，要进行补码运算，最终得到差的补码进行输出。最后得到仿真结果。

在实验箱上进行测试，验证结果的正确性。

## 实验原理图

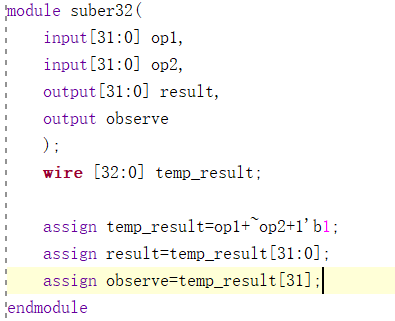


## 实验步骤

#### 模块suber32的实现

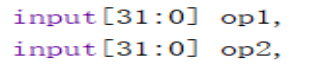
**代码思路：** 首先，我们需要实现减法器的代码，原代码是32位的加法，而减法不能简单的把 加法改为减法，因此我们考虑在数字逻辑课程里学到的知识，使用补码重新把减法改为 加法。并且原代码里有cin和cout分别代表上一位的进位和向下一位的进位，在减法中 应该改为上一位的借位和向下一位的借位。不过考虑到在有符号计算中（也就是答案可 能是负数），向下一位的借位似乎没有什么意义，因此我的减法器模块中删除了原来的 cin和cout，加入了一个检测位observe，这个的作用就是检测最后结果是正数还是负数。

代码：



**修改部分：**

1. 首先，我将input部分的cin与ouput部分的cout删除，因为在有符号整数减法中，这两个部分并没有什么意义。



1. 对于结果的计算是本代码的核心，我使用补码将减法修改为加法，即op1-op2=op1+~op2+1,对op2按位取反并加一。



1. 然后我添加了检测为observe，这一位等于结果result的最高位，如果结果为正，那么observe就是0，如果为负数，那么就是1。



#### 模块suber\_display的实现

**代码思路：**

这一部分的代码和原代码基本上是一致的，对于一些删除与修改的input和output量还有其名字进行了一些修改。

代码：

module suber\_display(

//时钟与复位信号

input clk,

input resetn, //后缀"n"代表低电平有效

//拨码开关，用于选择输入数和产生cin

input input\_sel, //0:输入为被减数1(add\_operand1);1:输入为减数2(add\_operand2)

input sw\_cin,

//led灯，用于显示cout

output led\_cout,

//触摸屏相关接口，不需要更改

output lcd\_rst,

output lcd\_cs,

output lcd\_rs,

output lcd\_wr,

output lcd\_rd,

inout[15:0] lcd\_data\_io,

output lcd\_bl\_ctr,

inout ct\_int,

inout ct\_sda,

output ct\_scl,

output ct\_rstn

);

//-----{调用减法模块}begin

reg [31:0] suber\_operand1;

reg [31:0] suber\_operand2;

wire [31:0] suber\_result;

wire observe;

suber32 suberer\_module(

.op1(suber\_operand1),

.op2(suber\_operand2),

.result (suber\_result ),

.observe(observe)

);

assign suber\_cin = sw\_cin;

//-----{调用减法模块}end

//---------------------{调用触摸屏模块}begin--------------------//

//-----{实例化触摸屏}begin

//此小节不需要更改

reg display\_valid;

reg [39:0] display\_name;

reg [31:0] display\_value;

wire [5 :0] display\_number;

wire input\_valid;

wire [31:0] input\_value;

lcd\_module lcd\_module(

.clk (clk ), //10Mhz

.resetn (resetn ),

//调用触摸屏的接口

.display\_valid (display\_valid ),

.display\_name (display\_name ),

.display\_value (display\_value ),

.display\_number (display\_number),

.input\_valid (input\_valid ),

.input\_value (input\_value ),

//lcd触摸屏相关接口，不需要更改

.lcd\_rst (lcd\_rst ),

.lcd\_cs (lcd\_cs ),

.lcd\_rs (lcd\_rs ),

.lcd\_wr (lcd\_wr ),

.lcd\_rd (lcd\_rd ),

.lcd\_data\_io (lcd\_data\_io ),

.lcd\_bl\_ctr (lcd\_bl\_ctr ),

.ct\_int (ct\_int ),

.ct\_sda (ct\_sda ),

.ct\_scl (ct\_scl ),

.ct\_rstn (ct\_rstn )

);

//-----{实例化触摸屏}end

//-----{从触摸屏获取输入}begin

//根据实际需要输入的数修改此小节，

//建议对每一个数的输入，编写单独一个always块

//当input\_sel为0时，表示输入数为被减数1，即operand1

always @(posedge clk)

begin

if (!resetn)

begin

suber\_operand1 <= 32'd0;

end

else if (input\_valid && !input\_sel)

begin

suber\_operand1 <= input\_value;

end

end

//当input\_sel为1时，表示输入数为减数2，即operand2

always @(posedge clk)

begin

if (!resetn)

begin

suber\_operand2 <= 32'd0;

end

else if (input\_valid && input\_sel)

begin

suber\_operand2 <= input\_value;

end

end

//-----{从触摸屏获取输入}end

//-----{输出到触摸屏显示}begin

//根据需要显示的数修改此小节，

//触摸屏上共有44块显示区域，可显示44组32位数据

//44块显示区域从1开始编号，编号为1~44，

always @(posedge clk)

begin

case(display\_number)

6'd1 :

begin

display\_valid <= 1'b1;

display\_name <= "SUB\_1";

display\_value <= suber\_operand1;

end

6'd2 :

begin

display\_valid <= 1'b1;

display\_name <= "SUB\_2";

display\_value <= suber\_operand2;

end

6'd3 :

begin

display\_valid <= 1'b1;

display\_name <= "RESUL";

display\_value <= suber\_result;

end  
 6'd4 :

begin

display\_valid <= 1'b1;

display\_name <= "OBSERVE";

display\_value <= observe;

end

default :

begin

display\_valid <= 1'b0;

display\_name <= 40'd0;

display\_value <= 32'd0;

end

endcase

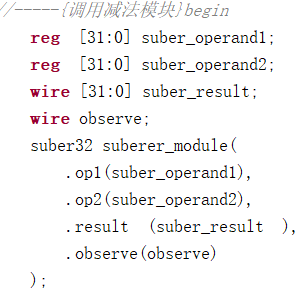
end

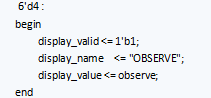
//-----{输出到触摸屏显示}end

//----------------------{调用触摸屏模块}end---------------------//

endmodule

**修改部分：**

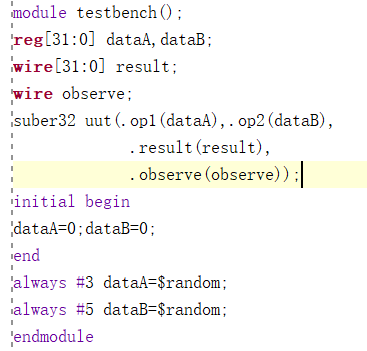
1. 首先我修改了一些名字，把原来的add改为sub，改了参数的量，删除了原来的cin与cout，添加了observe  
   
2. 接下来要改的是触摸屏，我要添加一个新的名字为OBSERVE的模块，用来展示observe检测值，在第四块添加即可，格式与之前类似。还有把之前的ADD\_1和ADD\_2改为SUB\_1和SUB\_2。



3.testbench模块的实现  
**代码思路：**

这一部分只需要修改一下调用的suber32模块的参数名以及参数量就可以。

**代码：**



**修改部分：**

1. 修改调用suber32模块的时候传入的参数，删除cin和cout，加入observe即可。

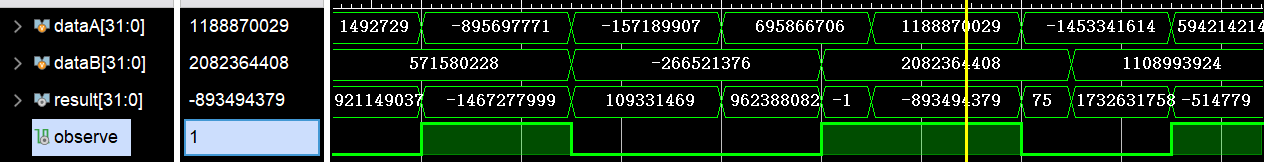
4.constraints模块的实现  
这一部分没有什么变化，使用之前的限制文件即可。

#### 实验结果分析

仿真结果分析：

仿真结果我选取四种，即正减正，正减负，负减正，负减负。换为有符号十进制数，方便观察。

正减正:



这里被减数11\_8887\_0029，减去20\_8236\_4408，结果是负数，-8\_9349\_4379，检测位为1。答案正确。

正减负：



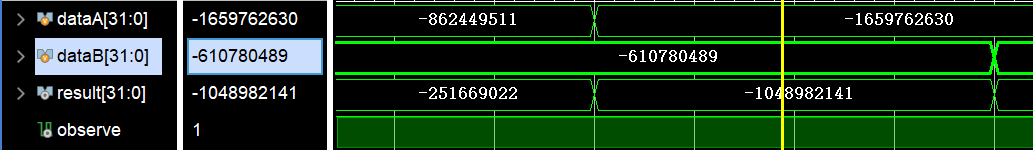
这里14\_7231\_9919减去-3\_0781\_5205，得到17\_8013\_5124，检测位为0，代表正数。答案正确。

负减正：



-10\_9299\_1875减去3\_5362\_3338，得到-13\_3661\_5213，检测位为1，代表负数。正确。

负减负：

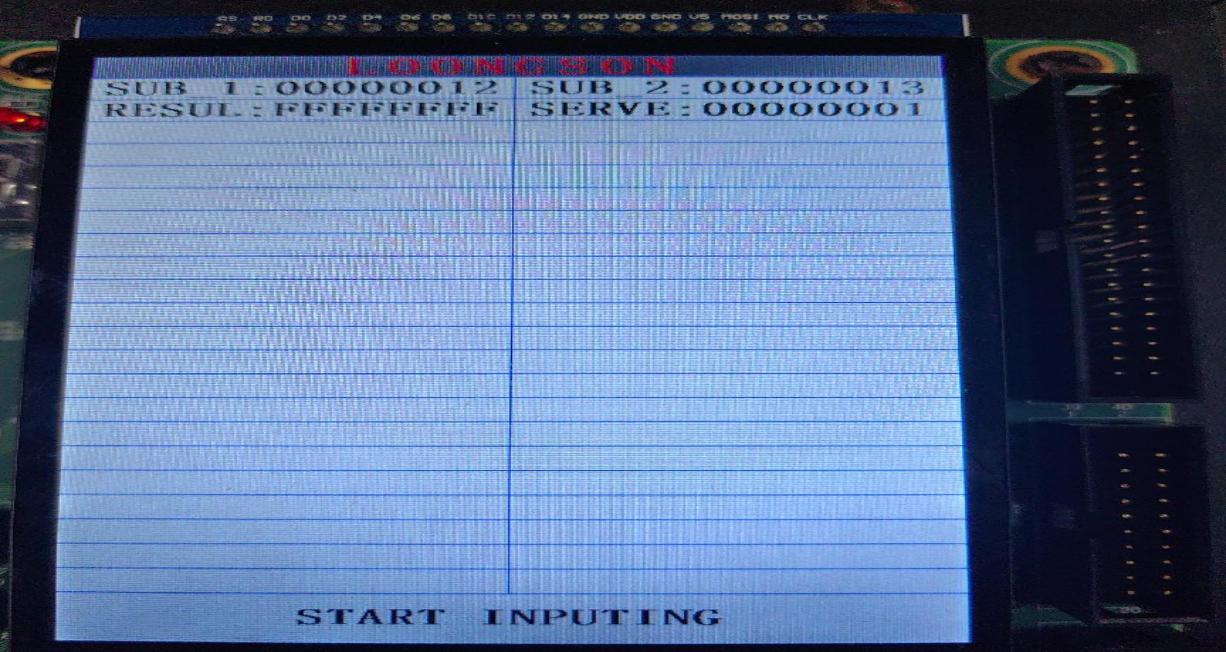


-16\_5976\_2630减去-6\_1078\_0489，得到-10\_4898\_2141,检测位是1，代表负数，正确。

实验箱验证结果：

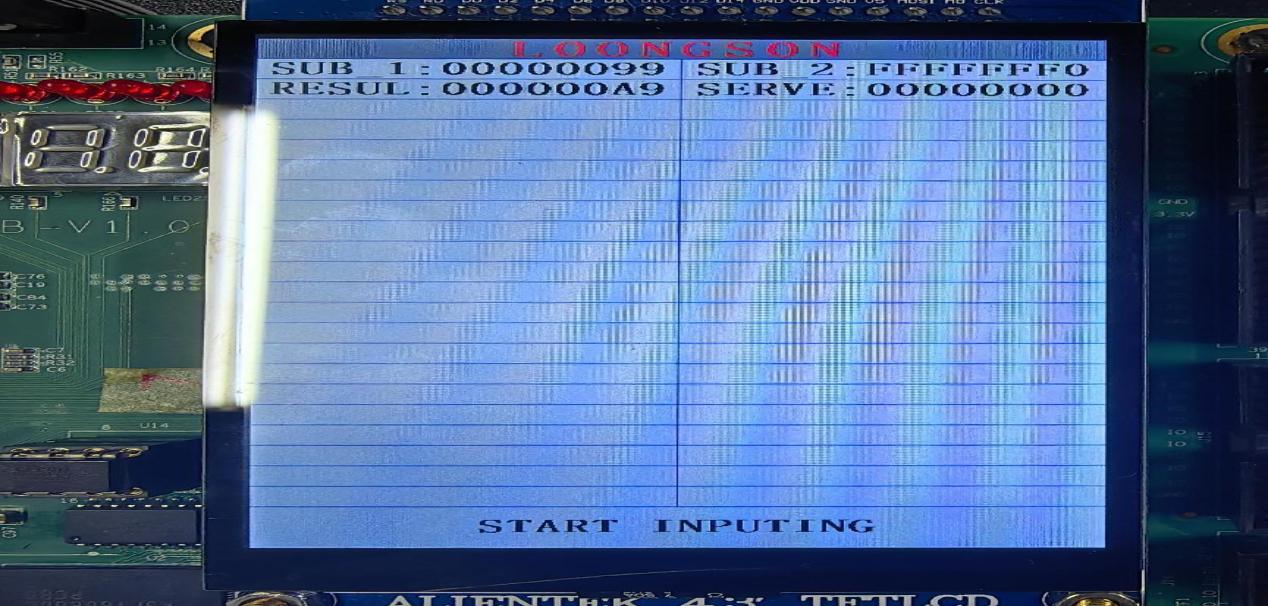
在suber\_display模块中我给检测位命名为OBSERVE,但是触摸屏上只有SERVE，这应为触摸屏长度限制，导致的显示不全。

1



如上图，我输入了被减数12h和减数13h，结果应该为-1，补码格式是FFFFFFFF，检测位1，结果正确。

2



如上图，我输入被减数99h和减数FFFFFFF0h（-16），结果是A9h，检测位是0，结果正确。

#### 总结感想

这是计组第一次实验，这次实验里学习了verilog语言，并且学习使用vivado进行模拟仿真实验，还在实验箱上进行了测试。在改代码的过程里对verilog有了更深入的理解。并且在改写减法器的过程中，对于加法器的理解也更加深入了。