实验报告

实验名称：vivado 实验五

院系：

数据科学与计算机学院

软件工程(移动信息工程)

班级： 1518

姓名： 张镓伟

学号：15352408

指导老师：郭雪梅

一、实验内容：

设计 8 位的加减法器，设计一个可变位宽的加减法器并在 vivado 内进行封装成 ip 核，再新建一个工程调用之前设计的 ip 核。

二、实验目的：

1.熟练使用 vivado 的各种功能。

2.练习 verilog 编程。

三、实验原理：

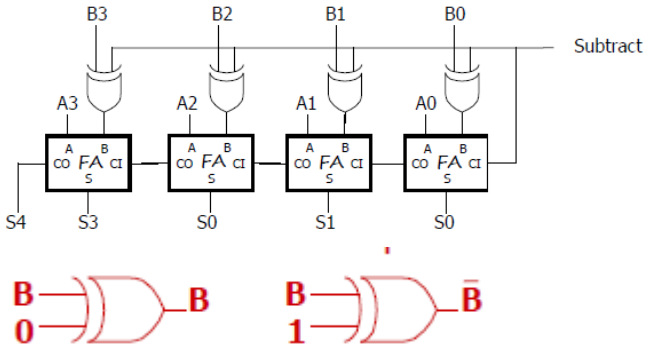
加减法运算器设计依据：

Subtract B from A = add 2’s complement of B to A

In 2’s complement: –B = ~B + 1

即将减法也转变成加法，减数变成其补码再与被减数相加就是原减法的结果。而若是加法则正常相加。

电路图参照：



代码设计：

```
module addsub
#(parameter WIDTH=8)            //指定数据宽度参数，缺省值是8
(
    input [(WIDTH-1):0] a,       // 缺省位数由参数WIDTH决定
    input [(WIDTH-1):0] b,
    input  sub,                 // =1为减法
    output [(WIDTH-1):0] sum,
    output cf,                   // 进位标志
    output ovf,                   // 溢出标志
    output sf,                   // 符号标志
    output zf                    // 为0标志
    );
    wire [(WIDTH-1):0] subb;
    wire [(WIDTH-1):0] subb1;
    wire cf2;
    assign subb1 = b^{WIDTH{sub}};
    assign subb=subb1+sub;
    assign {cf2,sum}=a+subb;
    assign sf =sum[WIDTH-1];
    assign zf=(sum==0)?1:0;
    assign cf=cf2^sub;
    assign ovf=(a[WIDTH-1]^sum[WIDTH-1])&(subb[WIDTH-1]^sum[WIDTH-1]);

endmodule
```

代码中各变量的解释与表达式的推导：

sub: 减法标志，0为加法，1为减法。

subb1: B的反码，加法时为本身,减法时各位取反。真值表如下

B	sub	subb1
0	0	0
1	0	1
0	1	1
1	1	0

由真值表可知subb1=B^sub

subb:B的补码，减法时为反码+1=subb1+sub；加法时为B本身=反码+0=subb1+sub。

sum：为A与B的补码相加的结果。

cf2： 为A与B的补码相加后最高位的溢出结果。

sf: 结果sum的符号位，即其最高位。

zf： sum为0时zf=1，sum为1时zf=0；

cf：加法的进位标志，需分加减法情况考虑，减法中的进位其实是借位，真值表如下：

sub	cf2	cf
0	0	0
0	1	1
1	0	1
1	1	0

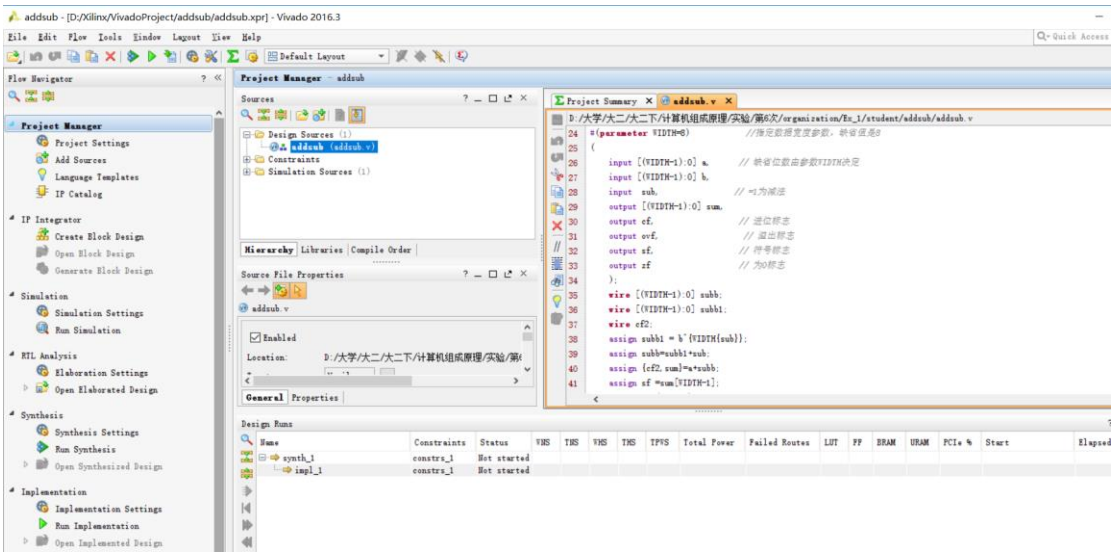
可得cf=cf2^sub。

ovf：溢出标志。如果sum的最高位与a和b的补码的最高位都不一样就说明相加发生了溢出。

四、实验过程：

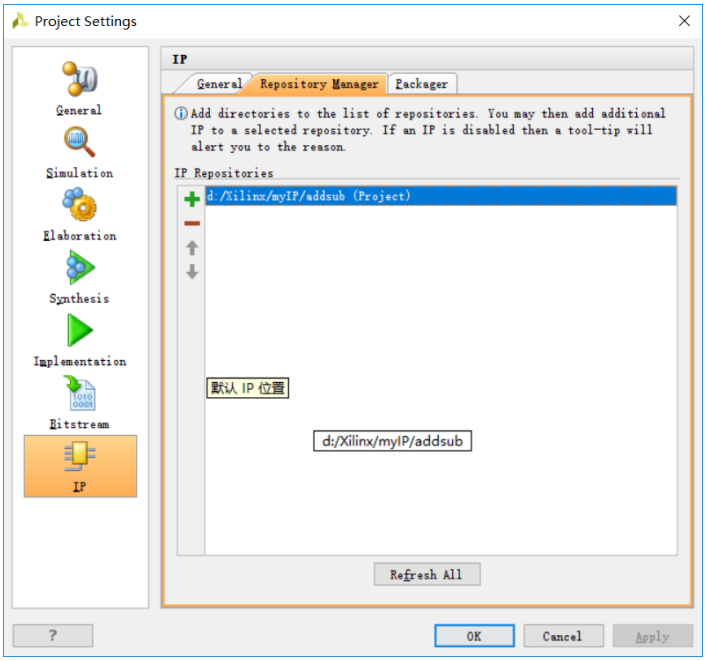
1.打开 vivado，创建一个关于 Artix-7 xc7a35tcpg236-1 型号的 basys3 开发板的 RTL project，project 命名为 addsub。

2.添加空白代码文件 addsub.v，将我们实验原理中的代码写在该文件中并保存。如下图：

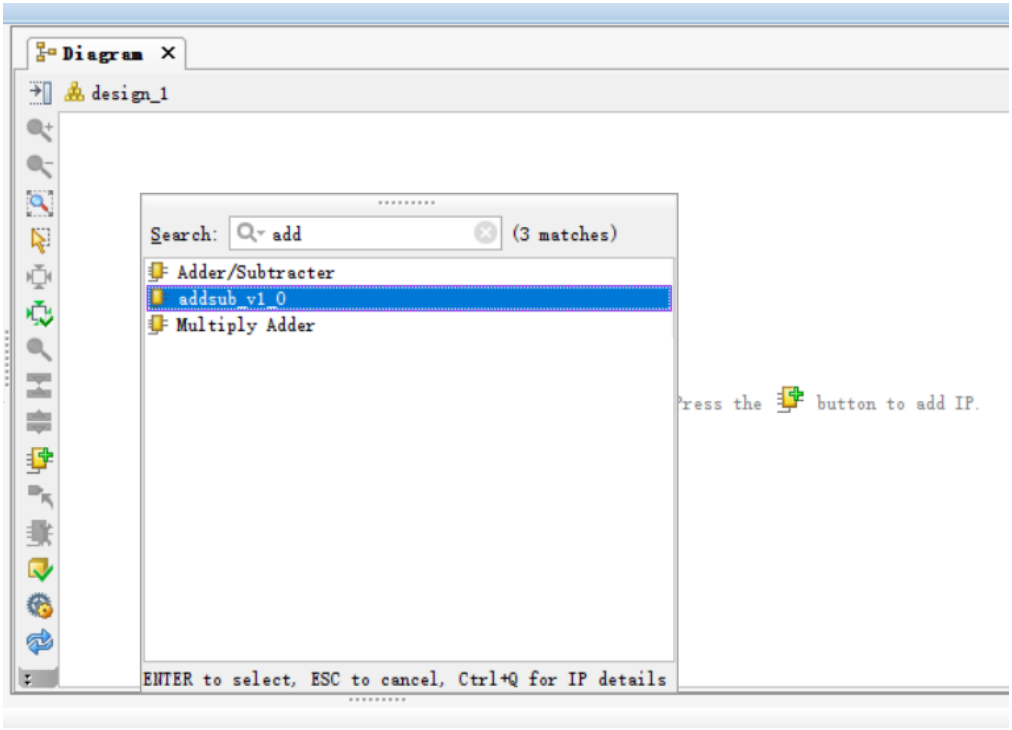


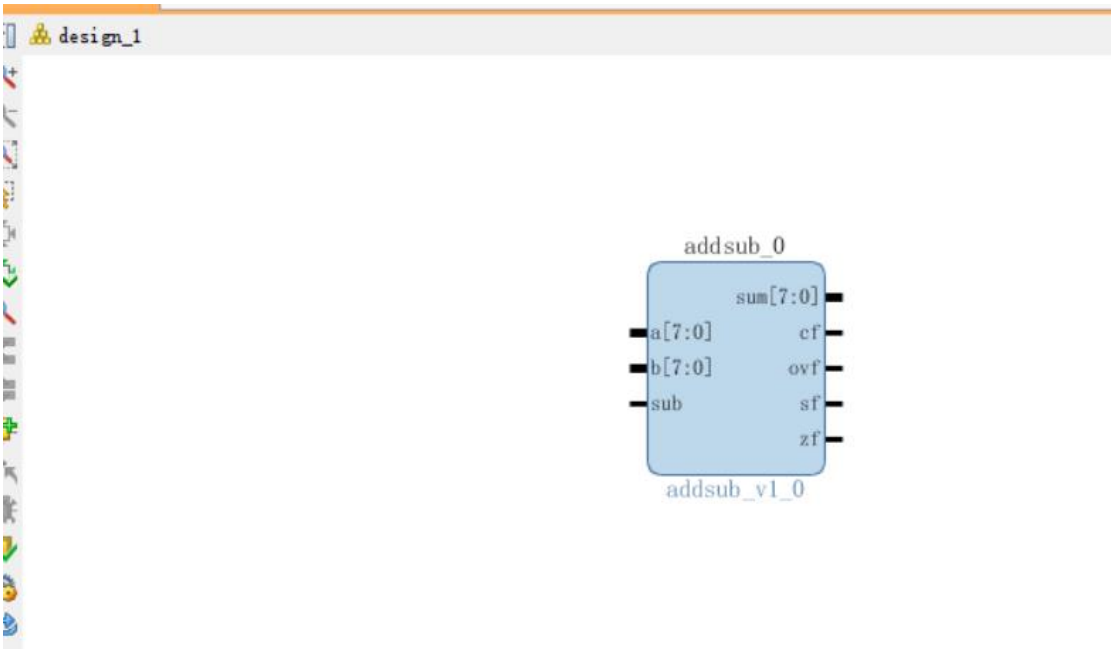
3. 点击project setting->IP,在package界面上可以设置库名和目录。我定义库名为addsub。然后点击菜单栏tool->create and package ip选项默认，一直点击next，在最后点finish。在新界面Review and package 一项中点击package ip完成ip的封装。

4.创建一个新的调用该 ip 的工程文件，我起名为 addsub8。点点击 project setting->IP,在该界面将之前设计的 ip 核所在工程路径添加到“ip repositories”中，点击 ok。

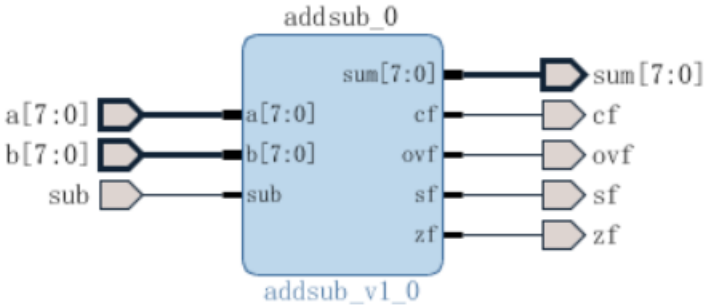


5. 在ip integrator选项中，点击create block design，创建之后在新出现的界面中点击add ip。在新出现的界面中在搜索栏里输入之前设计的ip的名字，双击将ip核添加进去。





6.接下来将引脚引出，右击a-> make external,则将a引出，其他端口类似。



7. 在source窗口下，右击->选择create HDL wrapper, 点击ok，则生成文件内调用了译码器ip核。

8.增加仿真文件。在source窗口下，右击Simulation Sources，选择Add Sources..添加写好的仿真文件addsub_sim.v。仿真代码如下：

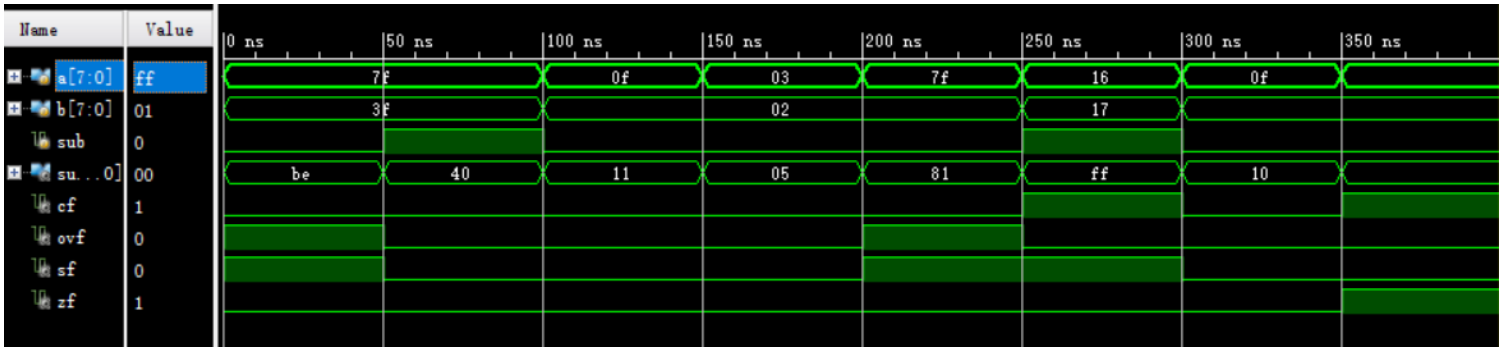
```
module addsub_sim(        );
    // input
    reg [7:0] a = 8'd01111111;
    reg [7:0] b = 8'd00111111;
    reg sub = 0;
    //output
    wire [7:0] sum;
    wire cf;
    wire ovf;
    wire sf;
    wire zf;
   addsub8_wrapper addsub8
                (.a(a),
```

```
        .b(b),
        .cf(cf),
        .ovf(ovf),
        .sf(sf),
        .sub(sub),
        .sum(sum),
        .zf(zf));
    // initial
    addsub #(8) U (a,b,sub,sum,cf,ovf,sf,zf);
    initial begin
    #50 sub = 1;
    #50 begin a = 8'd00001111; b = 8'd00000010; sub = 0; end
    #50 begin a = 8'd00000011; b = 8'd00000010; sub = 0; end
    #50 begin a = 8'd01111111; b = 8'd00000010; sub = 0; end
    #50 begin a = 8'd00010110; b = 8'd00010111; sub = 1; end
    #50 begin a = 8'd00001111; b = 8'd00000001; sub = 0; end
    #50 begin a = 8'd11111111; b = 8'd00000001; sub = 0; end
    end
  endmodule
```

9.左边菜单栏下Simulation一栏中点击Run Simulation 选择Run behavioral Simulation 完成仿真。

四、实验结果

注意图中波形图里的数字是以十六进制标注的。



分析：前 50ns：a= 0111_1111b=7fh，b= 0011_1111b=3fh；

a+b=beh，正确。但 beh=10111110b 符号位变了，

两个正数相加改变了符号位说明发生了一处，

ovf=1，sf=1，正确。cf=zf=0，正确。

50-100ns：a-b=7fh-3fh=40h，正常相减，无溢出，无借位。

100-150ns：a+b=fh+2h=11h，正常相加，无溢出，无进位。

150-200ns：a+b=3h+2h=5h，正常相加，无溢出，无进位。

200-250ns：a+b=7fh+2h=81h，符号位从0变成1，发生溢出。

250-300ns: a+b=16h-17h=ffh，结果是-1，-1的补码就是所有位为1，结果正确。相减为负数，说明发生了借位，cf=1。

300-350ns: a+b=fh+1h=10h。正常相加，无溢出，无进位。

350-400ns: a+b=ffh+1h。即-1+1=0.无溢出，有进位cf=1，sum=0则zf=1.

以上各结果均正确，说明加减法器设计正确。

五、实验感想

这次实验主要学会了如何使用vivado编写verilog代码，打包IP以及调用ip和仿真功能。前三者其实上学期我就已经做得很熟练了，所以这次我新掌握的是如何使用仿真功能。要仿真首先要写仿真代码，代码中通过“#数字”的组合来给电路的某段波形做延迟以便于我们观看每一步的结果。仿真结果中我们可以使用ctrl+滚轮的方式改变时间间隔，从而将图像调整到合适的大小以便于观察。