《计算机组成原理实验》
实验报告

（32 位快速加法器实现）

学院名称： 数据科学与计算机学院

专业（班级）： 18 计算机类 6 班

学生姓名： 宋渝杰

学号： 18340146

时间： 2019 年 11 月 25 日

实验一：32 位快速加法器设计与实现

一. 实验目的：

1. 掌握 4 位、32 位快速加法器的原理和设计方法。

二.实验内容：

1.用 logisim 实现 4 位、32 位加法器

2.将上述加法器用 vivado 实现，并烧写到 basys3 开发板上，需要显示低 16 位的结果，用十六进制显示。从拨号开关拨数输入，再做运算，结果送显示器输出。

三.实验原理

实现 32 位快速加法器可以通过先实现 4 位快速加法器，再将 8 个 4 位加法器整合成 32 位加法器即可。

4 位快速加法器：先实现进位生成函数 G 和进位传递函数 P：

Gn = Xn Yn，Pn = Xn ⊕ Yn

G* = G4 + P4 G3 + P4 P3 G2 + P4 P3 P2 G1

P* = P4 P3 P2 P1

然后根据 G 和 P 生成先行进位输出 C：

Cn = Gn + Pn C(n-1)　其中 C0 = Cin

之后生成和 S：

Sn = Pn ⊕ C(n-1)　其中 C0 = Cin

Cout = C4

这样就形成了一个 4 位快速加法器，为 4 级门电路延迟。

32 位快速加法器：先将 4 个 4 位快速加法器连接成 16 位快速加法器，使用 G* 和 P* 实现先行进位。再将两个 16 位快速加法器串联形成 32 位快速加法器即可。

四.实验器材

电脑一台，Vivado 软件一套，logisim 软件一套

五.实验过程和结果

（一）设计思想

根据上述的实验原理，一步步设计模块，进行连线即可

（二）设计方法

具体步骤如下：

Logisim：

1.根据表达式设计 Gn 和 Pn；

2.根据表达式设计 Cn、G*和 P*，形成 74182 芯片；

3.根据表达式设计 Sn 和 Cout，形成 4 位快速加法器；

4.使用 4 位快速加法器生成 16 位快速加法器，最后生成 32 位快速加法器。

Vivado：

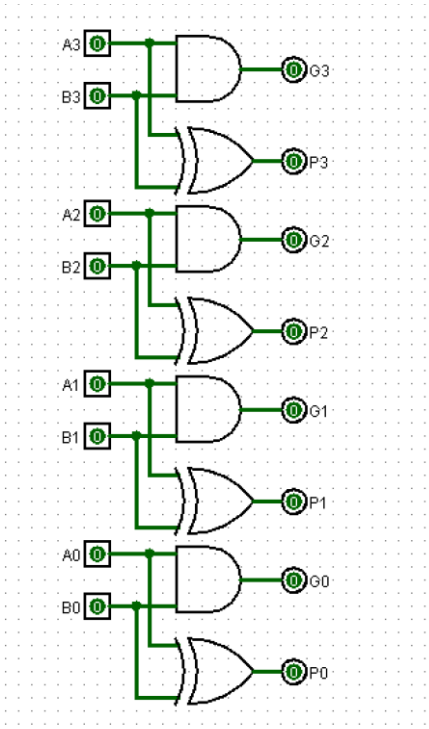
1.根据上述步骤设计 vivado 版 32 位加法器；

2.设计约束文件，将加法器烧入开发板中，并实现相应操作和功能。

（三）设计模块

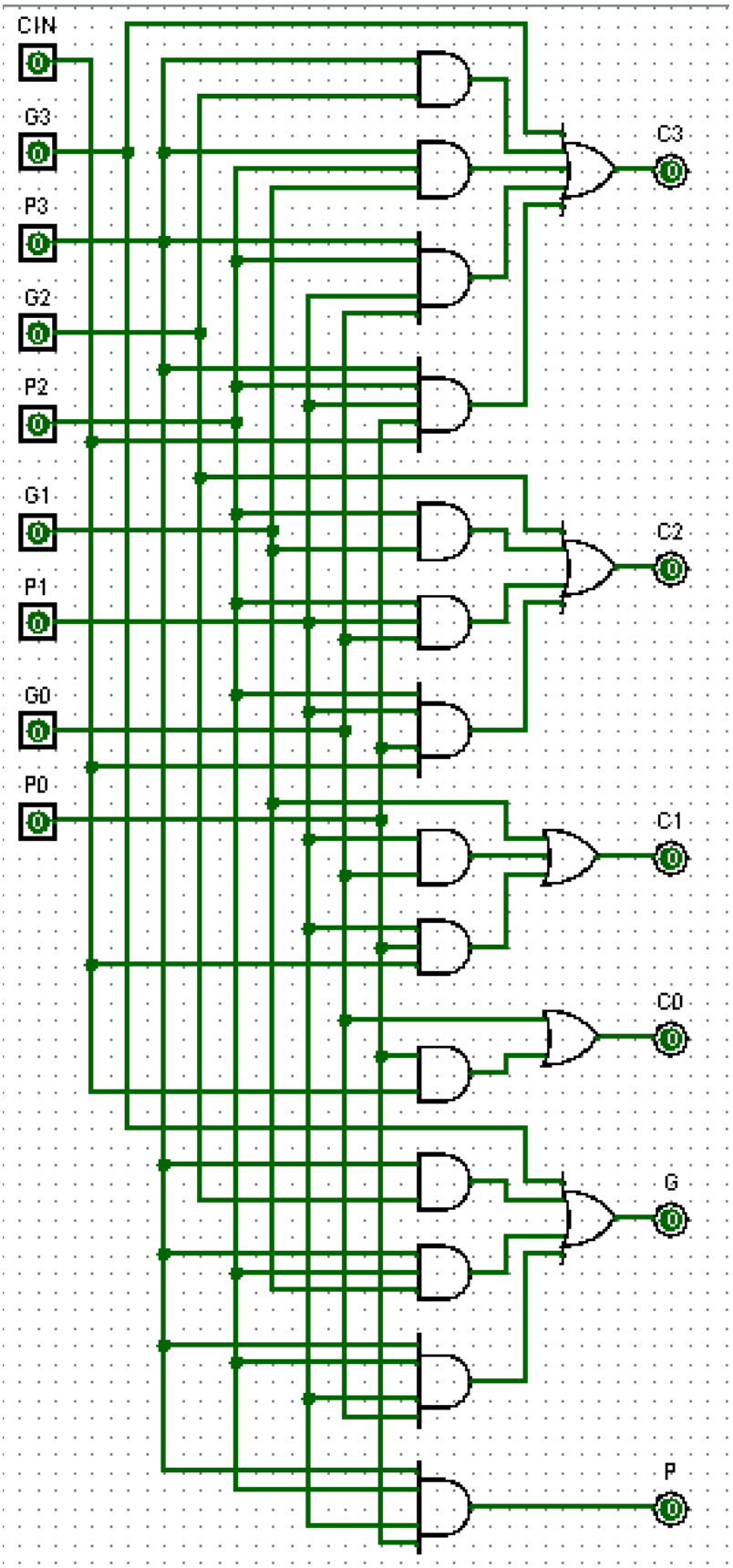
1.设计 Gn 和 Pn

通过表达式 Gn = Xn Yn，Pn = Xn ⊕ Yn，设计出 Gn 和 Pn 的电路。



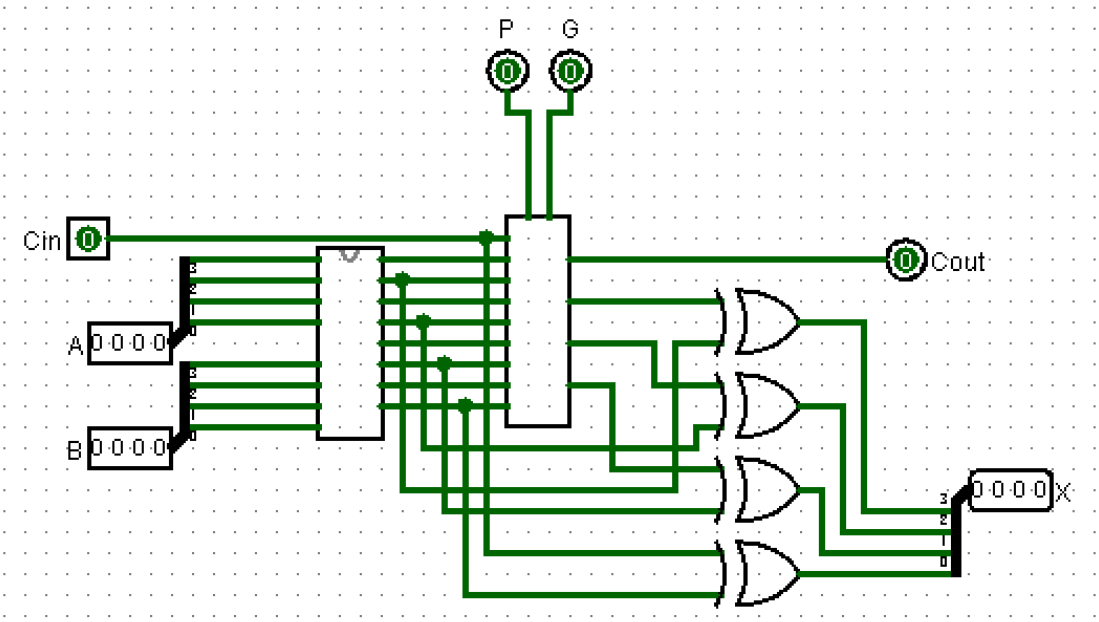
2.设计 74182 芯片

根据 Cn、G*和 P*的表达式，设计 74182 芯片的电路。



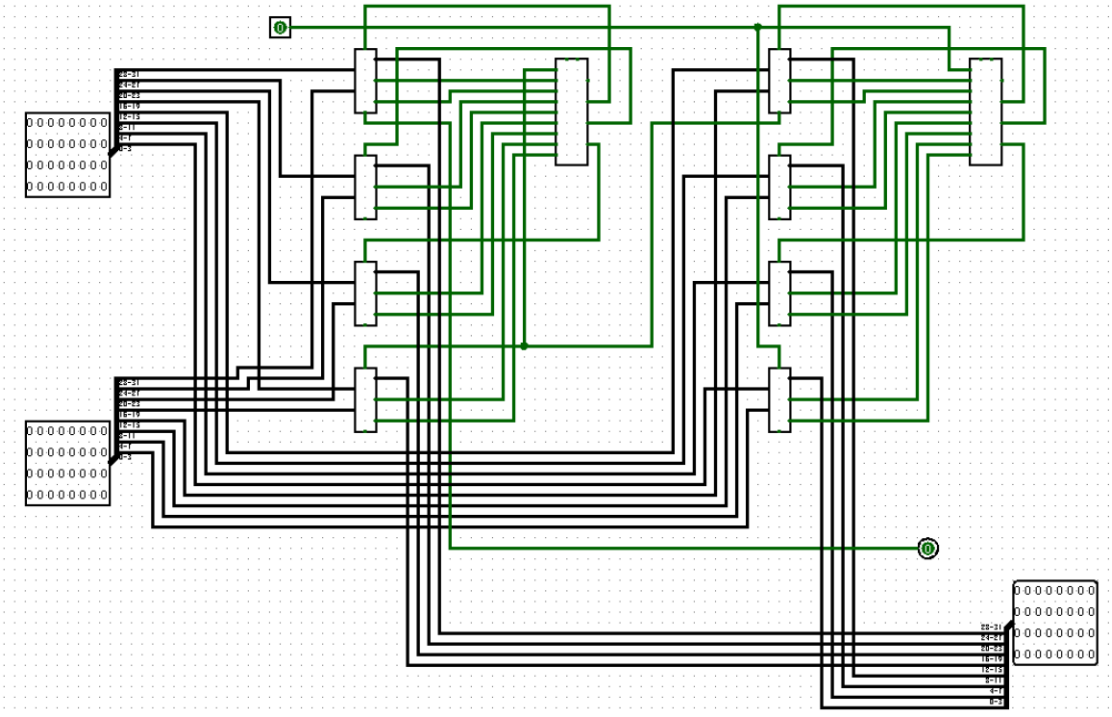
3.设计 4 位快速加法器

根据 Sn 和 Cout 的表达式，设计 4 位快速加法器电路



4.设计 32 位快速加法器

先将 4 个 4 位快速加法器生成 16 位快速加法器，再将两个 16 位快速加法器串联即可



5.vivado 设计开发板按键功能和约束文件

设计的开发板按键功能为左边第一个拨码开关为写使能，第二个为写入的数（0 为 A，1 为 B），第三、四个为控制写入数的位置（00 为低 8 位，01 为 9-16 位，10 为 17-24 位，11 为 25-32 位），右边八个拨码开关为写入的 8 位 2 进制数，左边按钮为清零，当写使能为 1 时按下即将 AB 清零，右边按钮为写入信号，当写使能为 1 时按下即将 8 位 2 进制数写入对应位置。

当写使能为 0 时，为显示数字功能，第二、三个拨码开关为显示的数（00 为显示和 SUM，10 为显示 A，01 为显示 B），第四个拨码开关为显示的位数（0 为显示低 16 位，1 为显示高 16 位）。

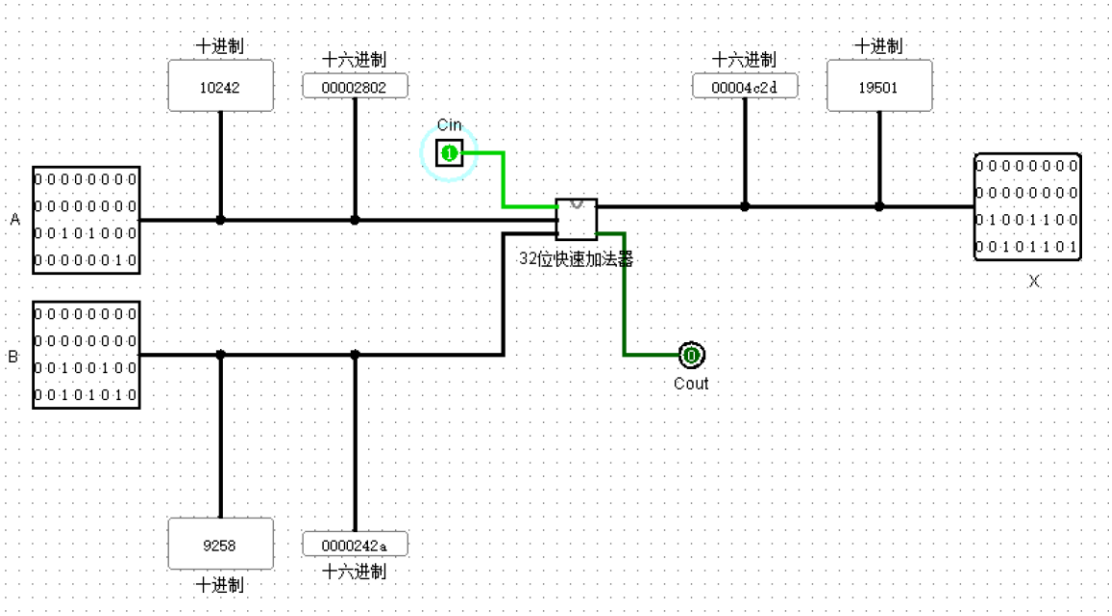
```
always@(posedge clk )begin
    write_old<=write;
    write_posedg<=~write_old & write; //write上升沿
    clear_old<=clear;
    clear_posedg<=~clear_old & clear; //clear上升沿
    if(code[3])begin // 写操作
        if(clear_posedg)begin
            A<=0;//ff7da5c9 + 6f2d3901 = 6eaadeca
            B<=0;
        end
        if(!code[2])begin   // 写入 A
            if(code[1] & code[0])begin// 31-24
                A[31:24]<=data;
                num<=A[31:16];
            end
            else if(code[1] & ~code[0])begin//23-16
                A[23:16]<=data;
                num<=A[31:16];
            end
            else if(~code[1] & code[0])begin//15-8
                A[15:8]<=data;
                num<=A[15:0];
            end
            else if(~code[1] & ~code[0])begin//7-0
                A[7:0]<=data;
                num<=A[15:0];
            end
        end
```

```
    else begin  // 写入 B
        if(code[1] & code[0])begin// 31-24
            B[31:24]<=data;
            num<=B[31:16];
        end
        else if(code[1] & ~code[0])begin//23-16
            B[23:16]<=data;
            num<=B[31:16];
        end
        else if(~code[1] & code[0])begin//15-8
            B[15:8]<=data;
            num<=B[15:0];
        end
        else if(~code[1] & ~code[0])begin//7-0
            B[7:0]<=data;
            num<=B[15:0];
        end
    end
end
  if(code[2] & ~code[1])begin   // 显示 A
      if(code[0])begin // 高 16 位
          num<=A[31:16];
      end
      else begin  //低16位
          num<=A[15:0];
      end
  end
  else if(~code[2] & code[1])begin  //显示B
      if(code[0])begin // 高 16 位
          num<=B[31:16];
      end
      else begin  //低16位
          num<=B[15:0];
      end
  end
  else if(code[2] & code[1])begin  //显示0
      num<=0;
  end
  else begin  //显示结果C
      if(code[0])begin // 高 16 位
          num<=C[31:16];
      end
      else begin  //低16位
          num<=C[15:0];
      end
  end
```

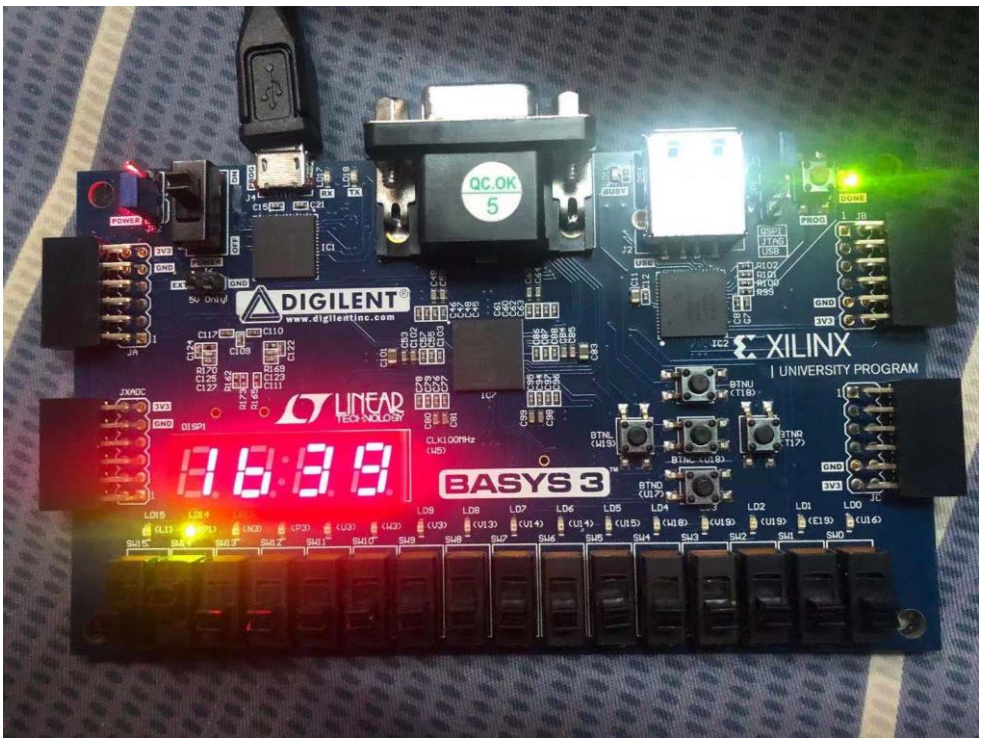
R2	0		code[3]	LVCMOS33*	Input	3.300
T2	1					
R3	1					
T3	1					
T1	0		code[2]	LVCMOS33*	Input	3.300
U1	0		code[1]	LVCMOS33*	Input	3.300
V2	1					
W2	0		code[0]	LVCMOS33*	Input	3.300

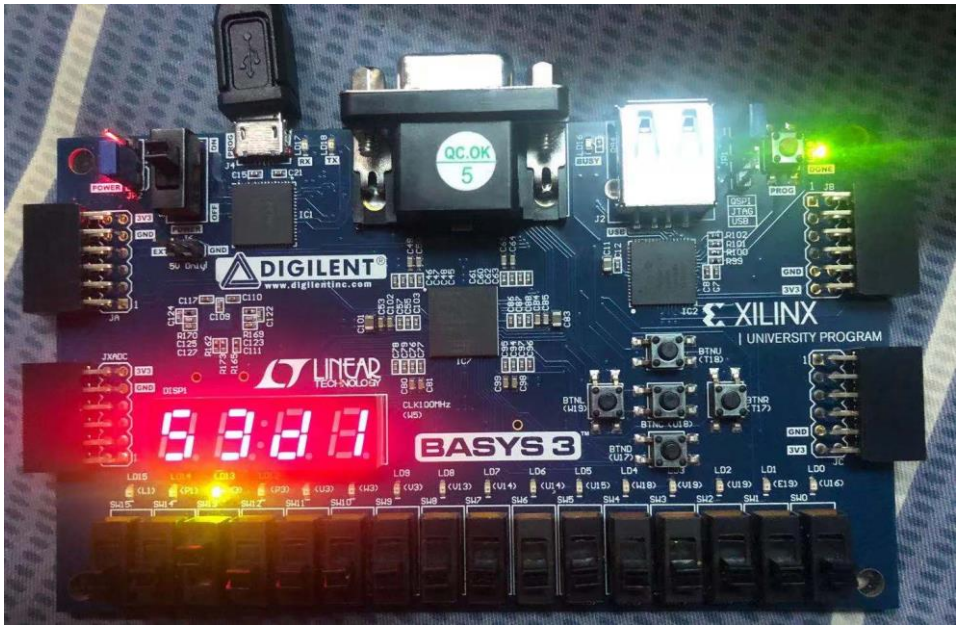
（四）进行仿真测试

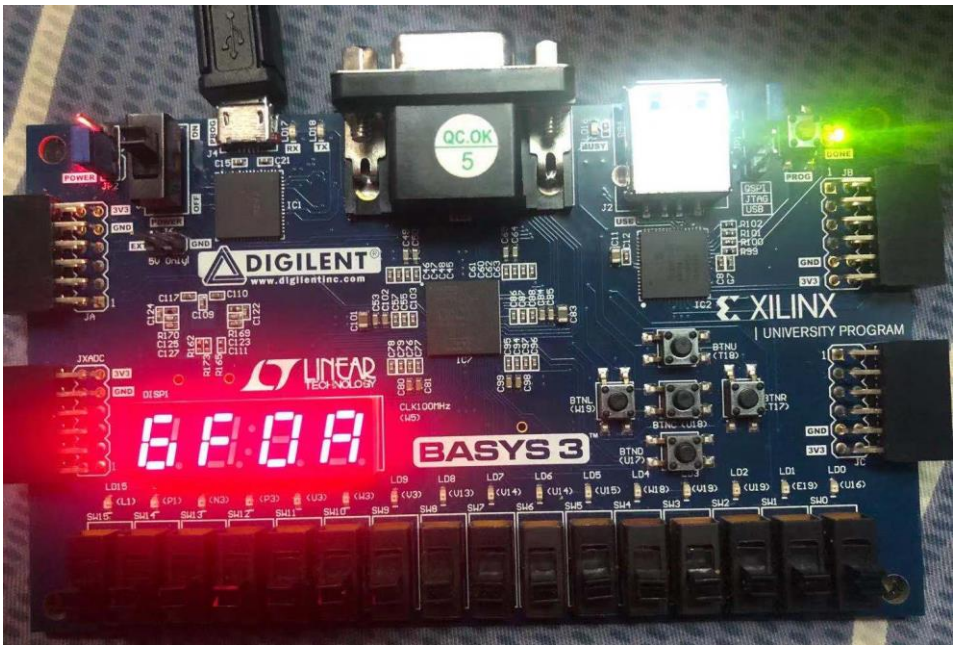
Logisim：（10242+9258+1 = 19501）



Vivado：（1b39h + 53d1h = 6f0ah）







六.实验心得

本次加法器设计实验中，具体实现过程比较明确，遵循步骤设计即可。

遇到的困难：

1.快速加法器的设计原理图：设计从 G、P 到 C 再到 S，减少门电路延迟时间的操作需要一定时间来理解正确性；

2.设计开发板的功能以及约束文件，也需要一段时间分析思考。

解决方式：

1.翻阅课件 ppt，自己推导一遍函数表达式的正确性，也在 logisim 实践测试正确性；

2.参考了多位同学的开发板功能设计思路，整合生成比较适合自己的设计。

实验收获：

1.完成了 4 位、32 位加法器的设计和烧板功能的实现；

2.对 vivado 和 logisim 软件的运用更加熟练。