**中国科学技术大学计算机学院**

**《计算机组成原理实验》报告**



实验题目： \_运算器及其应用 \_

学生姓名：\_\_\_\_ 黄瑞轩\_ \_\_\_\_

学生学号：\_\_ PB20111686 \_\_\_

完成日期：\_\_ 2022.3.18 \_\_

计算机实验教学中心制

2020年09月

**实验题目**

运算器及其应用

**实验目的**

* 熟练掌握算术逻辑单元 (ALU) 的功能
* 掌握数据通路和控制器的设计方法
* 掌握组合电路和时序电路，以及参数化和结构化的Verilog描述方法
* 了解查看电路性能和资源使用情况

**实验环境**

* Nexys4-DDR
* Vivado 2019.1

**实验**1: 算术**逻辑单元（ALU）**

* **ALU器件的设计**
* ***要求实现的***功能

32位数据的运算，具体功能为：判断是否相等、不等、有符号数小于、有符号数大于等于、无符号数小于、无符号数大于等于，以及双目减法、加法、与运算、或运算、异或运算、逻辑和算术右移运算、左移运算。

* ALU逻辑设计

器件图和端口含义：

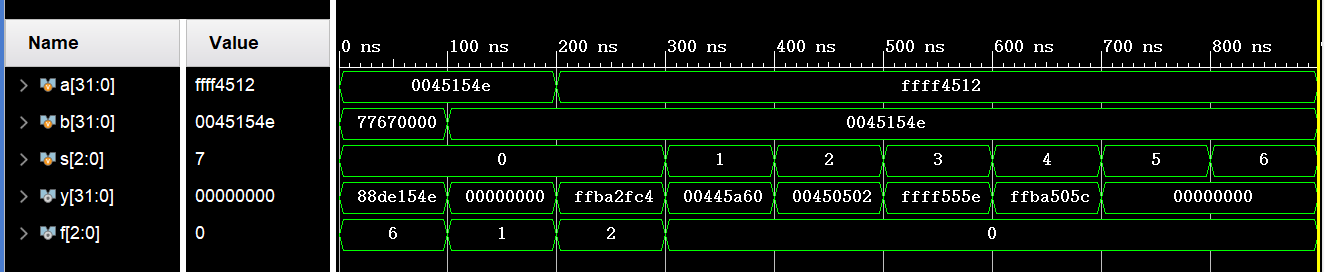
|  |  |
| --- | --- |
|  | s：功能选择，加、减、与、或、异或、逻辑左移、逻辑右移、算术右移等运算  a, b：两个操作数  y：运算结果，和、差 ……  f：标志，相等(eq)，小于(lt, ltu) |
|  | |

* ALU核心代码

见附件alu.v。

* ALU模块的仿真

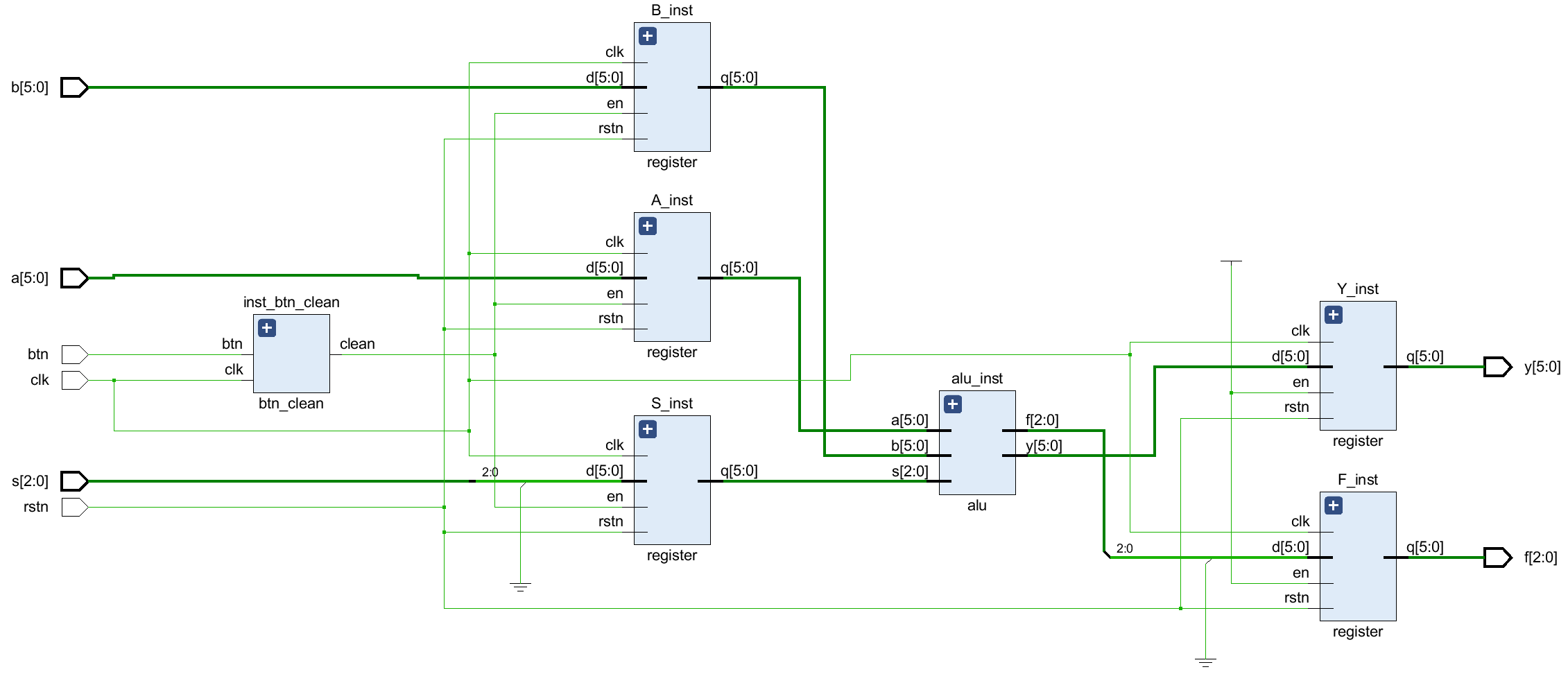
为了验证ALU模块的正确性，编写testbench文件，代码见附件testbench.v，模拟结果见下图，经检验结果正确。



* 测试用Top模块编写
* Top核心代码

对实验要求的6位ALU进行测试，代码见附件top.v。对Button按钮去抖动、取边沿的模块见附件btn\_clean.v；寄存器代码见附件register.v；二者不是本实验重点，不做详述。

* Top RTL分析电路



* Top模块与Nexys4开发板的接口对应

xdc约束文件见附件const.xdc。

s[2:0] → sw[15:13]

a[5:0] → sw[11:6]

b[5:0] → sw[5:0]

en → btnc

rstn → cpu\_resetn

clk → clk100mhz

f → led[15:13]

y → led[5:0]

* Top上板下载测试

生成比特流并烧写到开发板上，用如下几组测试数据检验正确性：

**0#**  s = 3’b 000；a = 6’b 101001；b = 6’b 001000；*// 有符号数正负不同*

**1#**  s = 3’b 000；a = 6’b 101001；b = 6’b 101001；*// 相等判断*

**2#**  s = 3’b 000；a = 6’b 101001；b = 6’b 111100；*// 有无符号情况不同*

**3#**  s = 3’b 001；a = 6’b 001001；b = 6’b 001010；*// 加法运算*

**4#**  s = 3’b 010；a = 6’b 101001；b = 6’b 001000；*// 按位与*

**5#**  s = 3’b 011；a = 6’b 101001；b = 6’b 001110；*// 按位或*

**6#**  s = 3’b 100；a = 6’b 101001；b = 6’b 001000；*// 按位异或*

**7#**  s = 3’b 101；a = 6’b 101001；b = 6’b 000100；*// 逻辑右移*

**8#**  s = 3’b 110；a = 6’b 101001；b = 6’b 000100；*// 左移*

**9#**  s = 3’b 111；a = 6’b 101001；b = 6’b 000100；*// 算数右移*

预期结果：

**0#** f[0:2] = 3’b 010；y = 6’b 100001；

**1#** f[0:2] = 3’b 001；y = 6’b 000000；

**2#** f[0:2] = 3’b 100；y = 6’b 101101；

**3#** f[0:2] = 3’b 000；y = 6’b 010011；

**4#** f[0:2] = 3’b 000；y = 6’b 001000；

**5#** f[0:2] = 3’b 000；y = 6’b 101111；

**6#** f[0:2] = 3’b 000；y = 6’b 100001；

**7#** f[0:2] = 3’b 000；y = 6’b 000010；

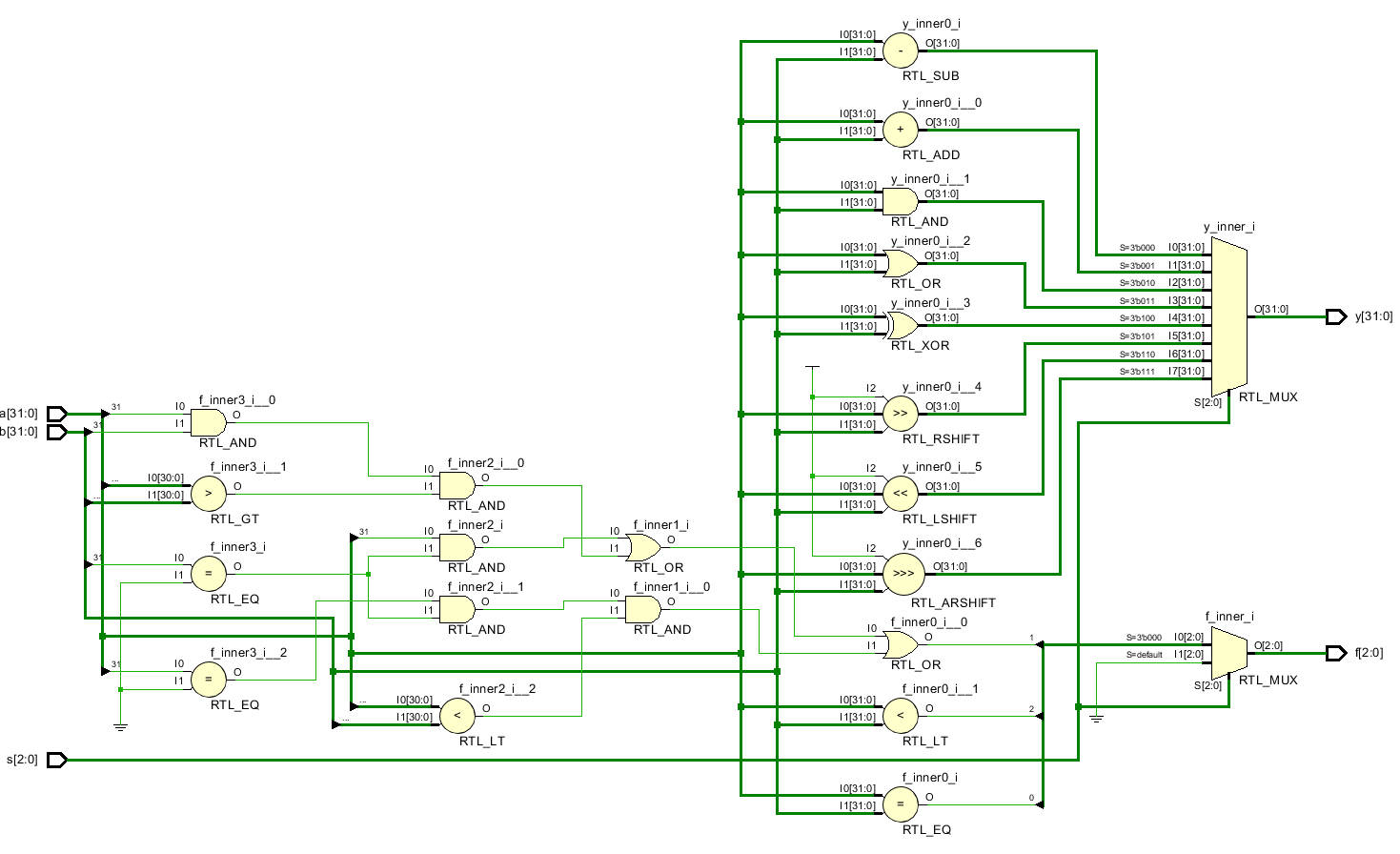
**8#** f[0:2] = 3’b 000；y = 6’b 010000；

**9#** f[0:2] = 3’b 000；y = 6’b 111110；

实验测试结果：

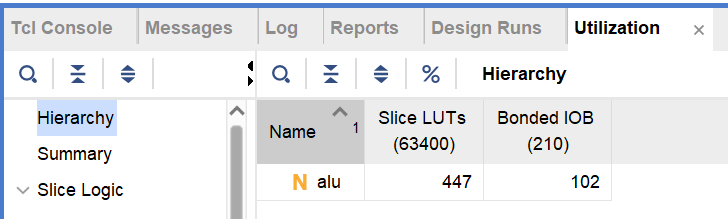
|  |  |  |
| --- | --- | --- |
|  |  |  |
| **0#** 正确 | **1#** 正确 | **2#** 正确 |
|  |  |  |
| **3#** 正确 | **4#** 正确 | **5#** 正确 |
|  |  |  |
| **6#** 正确 | **7#** 正确 | **8#** 正确 |
|  |  |  |
| **9#** 正确 |  |  |

* ALU模块电路及资源与性能
* ALU RTL分析电路

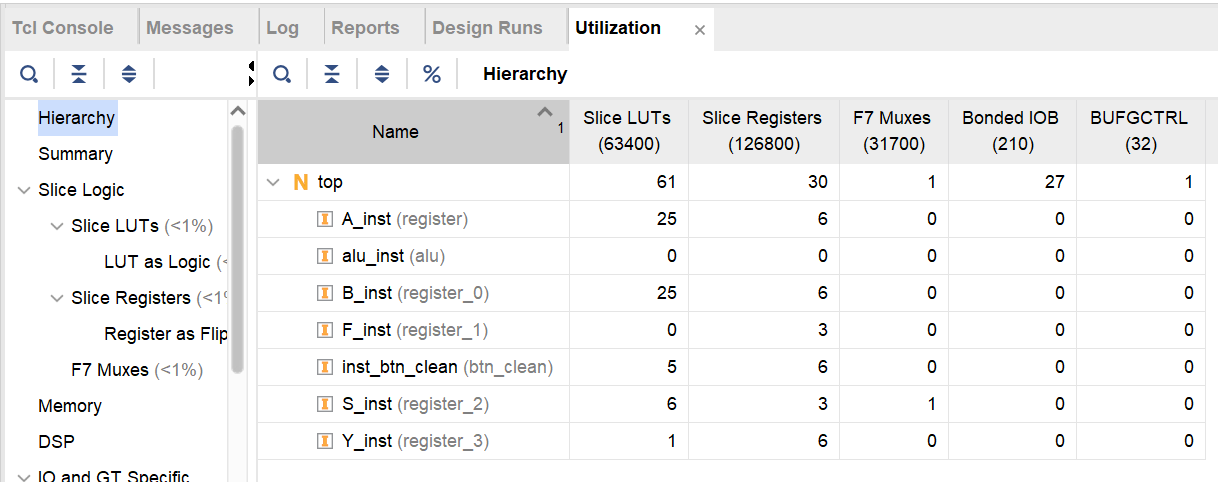


* ALU电路资源

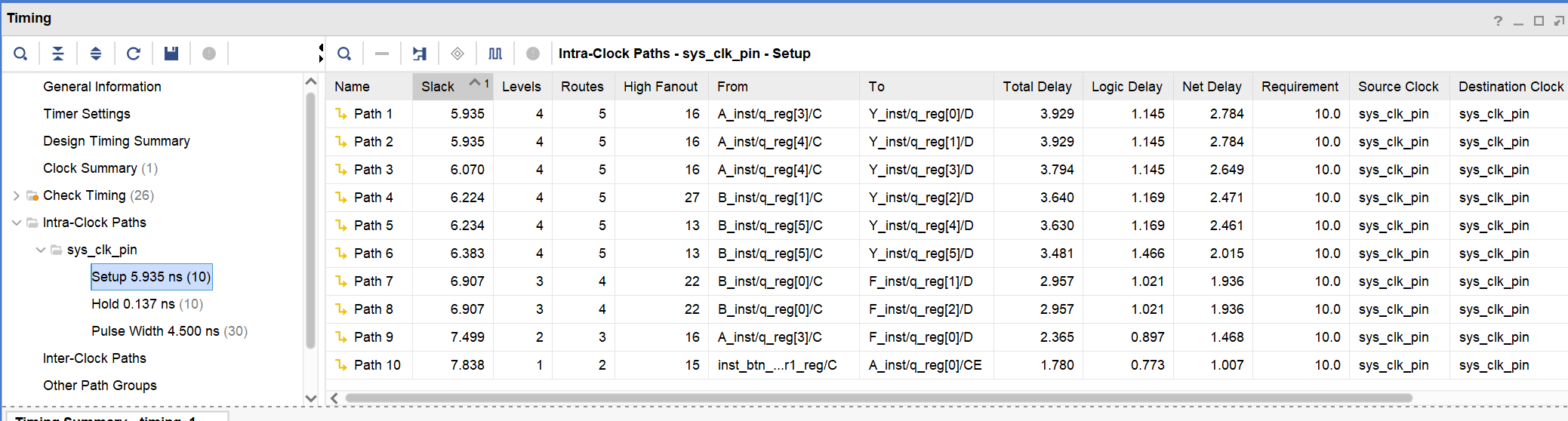
ALU单独模块电路资源



Top模块电路资源



* ALU电路性能

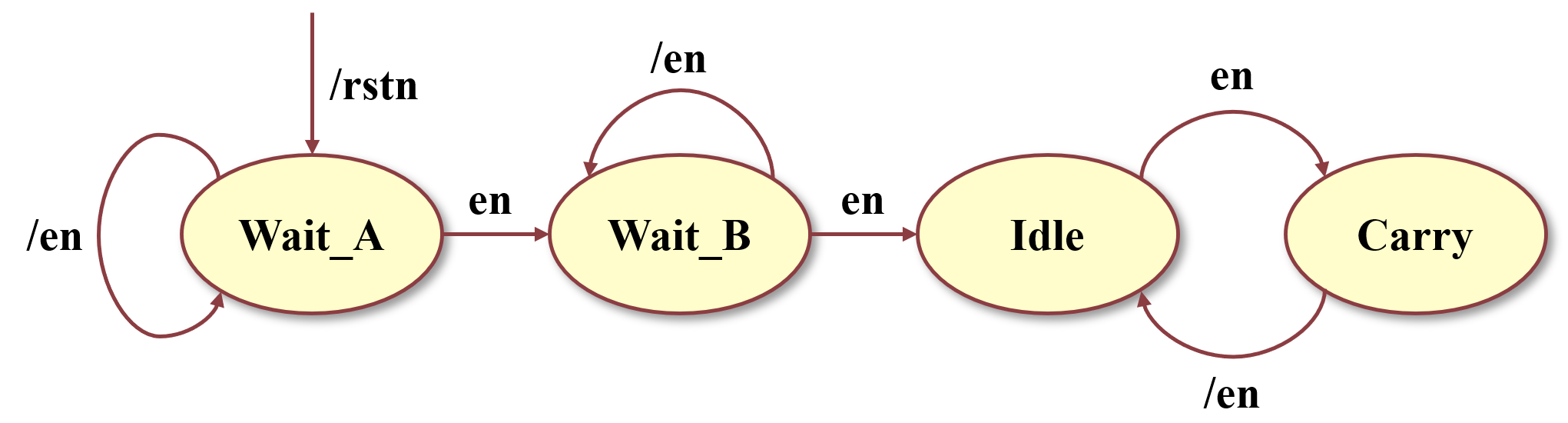


实验2: FLS模块

* FLS模块设计
* 要求实现的功能

自主输入两个基数a和b，此后每一次按下Button（en信号出现高电平脉冲），都计算a + b，并将原来a和b的寄存器内容更新为b和a + b。

* FLS时序逻辑状态图设计



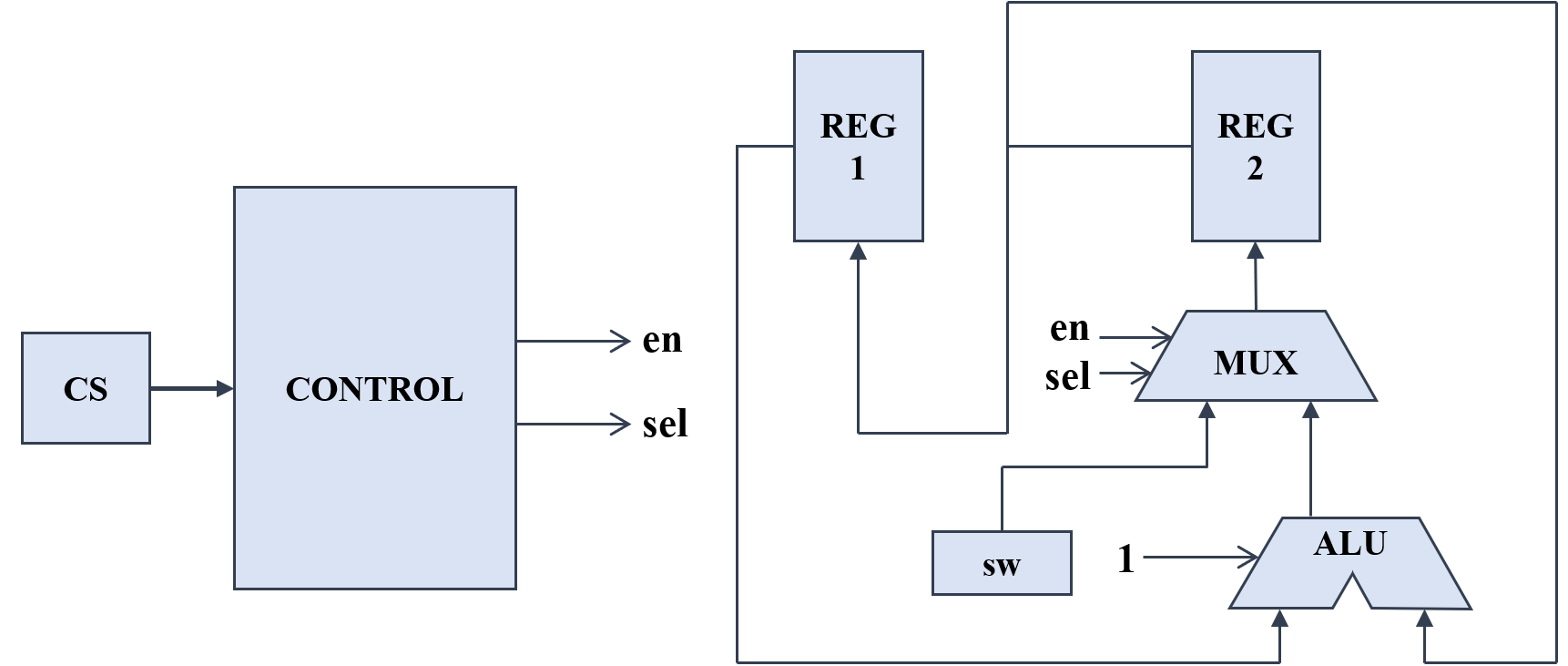
状态编码：Wait\_A[00]，Wait\_B[01]，Idle[11]，Carry[10]。上电之后的默认状态为Wait\_A，当通过开关连续输入a、b后，到待机状态Idle，一旦有使能信号出现，则计算和移动一次，返回待机状态。

* FLS封装设计

器件图及端口解释：

|  |  |
| --- | --- |
|  | 复位后，前2次按下btn时，f分别输出f0和f1 (= sw)。  随后每次按下btn时， f依次输出f*n* = f*n*-2 + f*n*-1, *n* > 1。 |

* FLS数据通路设计

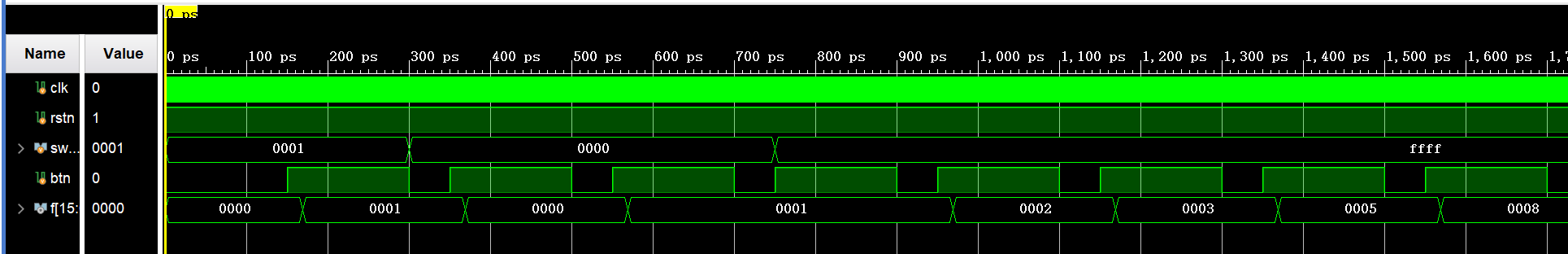


* FLS核心代码

见附件fls.v。此外，为了方便模块复用，额外编写mux和control模块，见附件mux.v和control.v。

* FLS模块仿真及下载测试
* FLS模块的仿真

为了验证FLS模块的正确性，编写testbench文件，代码见附件fls\_tb.v，模拟结果见下图，经检验结果正确。



* FLS模块与Nexys4接口对应关系

由于设计的FLS模块已经高度封装化，不需要再额外进行Top模块编写，直接生成比特流并烧写到开发板上。

FLS模块与Nexys4开发板的接口对应关系如下，xdc约束文件见附件const.xdc。

sw[15:0] → sw[15:0]

rstn → cpu\_resetn

clk → clk100mhz

btn **→** btnc

f → led[15:0]

* FLS模块下载测试

测试样例：a = 4，b = 7。

预期结果：（下方“点”表示熄灭，“圈”表示亮起）

hex: 0004, led: .............O..

hex: 0007, led: .............OOO

hex: 000b, led: ............O.OO

hex: 0012, led: ...........O..O.

hex: 001d, led: ...........OOO.O

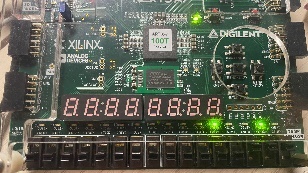
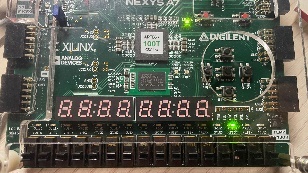
hex: 002f, led: ..........O.OOOO

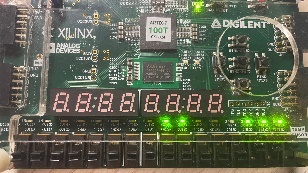
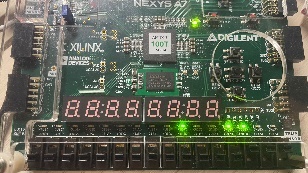
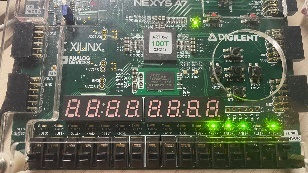
hex: 004c, led: .........O..OO..

hex: 007b, led: .........OOOO.OO

hex: 00c7, led: ........OO...OOO

测试结果：





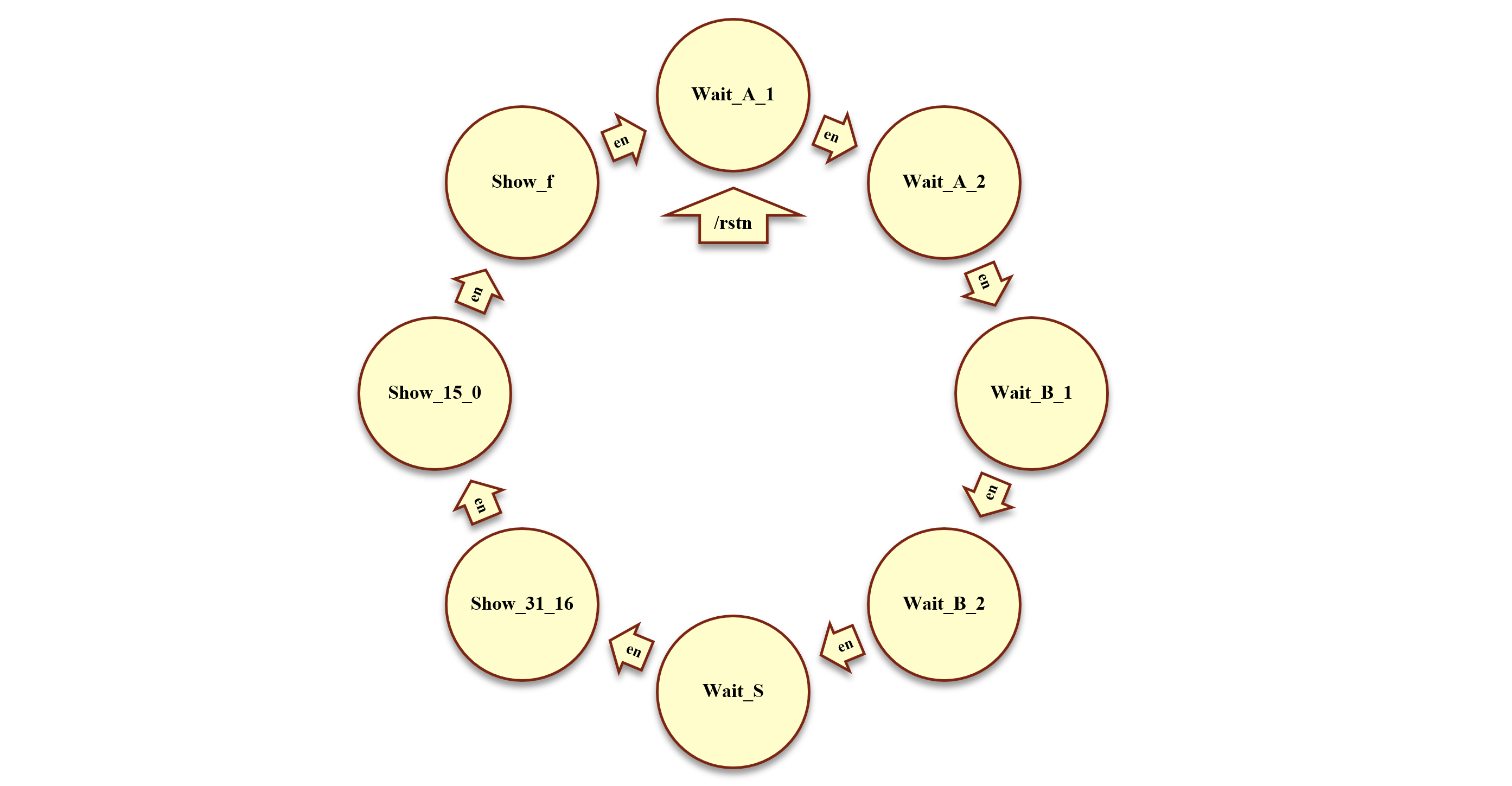
与预期结果完全一致。

实验选项: 32位ALU设计及下载测试

* 32位ALU设计
* 相对于6位ALU下载测试的改进设计

由于Nexys4开发板一次最多输入16位，所以需要设计时序状态，分阶段输入。

* 32位ALU状态图设计



* 32位ALU逻辑设计

上电之后默认状态为Wait\_A\_1，此时拨动开关sw，按下btn后读取sw值为操作数a的高16位，进入下一个状态；此时拨动开关sw，按下btn后读取sw值为操作数a的低16位，进入下一个状态；此时拨动开关sw，按下btn后读取sw值为操作数b的高16位，进入下一个状态；此时拨动开关sw，按下btn后读取sw值为操作数b的低16位，进入下一个状态；此时拨动开关sw，按下btn后读取sw[15:13]为操作数s，进入下一个状态；此时led将展示运算结果的高16位，再按下btn，进入下一个状态；此时led将展示运算结果的低16位，再按下btn，进入下一个状态；此时led[15:13]将展示f[2:0]，一个计算循环到此结束。

* 32位ALU核心代码

见附件alu\_32.v。

* 32位ALU的仿真

仿真文件见附件alu\_32\_tb.v，仿真结果见下图。即进行运算：0x536a1220 – 0x771dff00，结果为0xdc4c1320，经验证正确。f[2:0]给出仅ltu、lt两者为高电平，与事实一致。



* 32位ALU下载测试
* 32位ALU与Nexys4端口对应

sw[15:0] → sw[15:0]

clk → clk100mhz

btn → btnc

led[15:0] → led[15:0]

* 32位ALU下载测试

测试用例：

**0#**  s = 3’b 000；a = 32’h ff8710cd；b = 32’h 896a557f；

**1#**  s = 3’b 000；a = 32’h 4543ffff；b = 32’h abcd1234；

**2#**  s = 3’b 001；a = 32’h ff8710cd；b = 32’h 896a557f；

**3#**  s = 3’b 111；a = 32’h ff8710cd；b = 32’h 0000000f；

预期结果：

**0#** y = 32’h 761cbb4e；f = 3’b 010；

**1#** y = 32’h 9976edcb；f = 3’b 100；

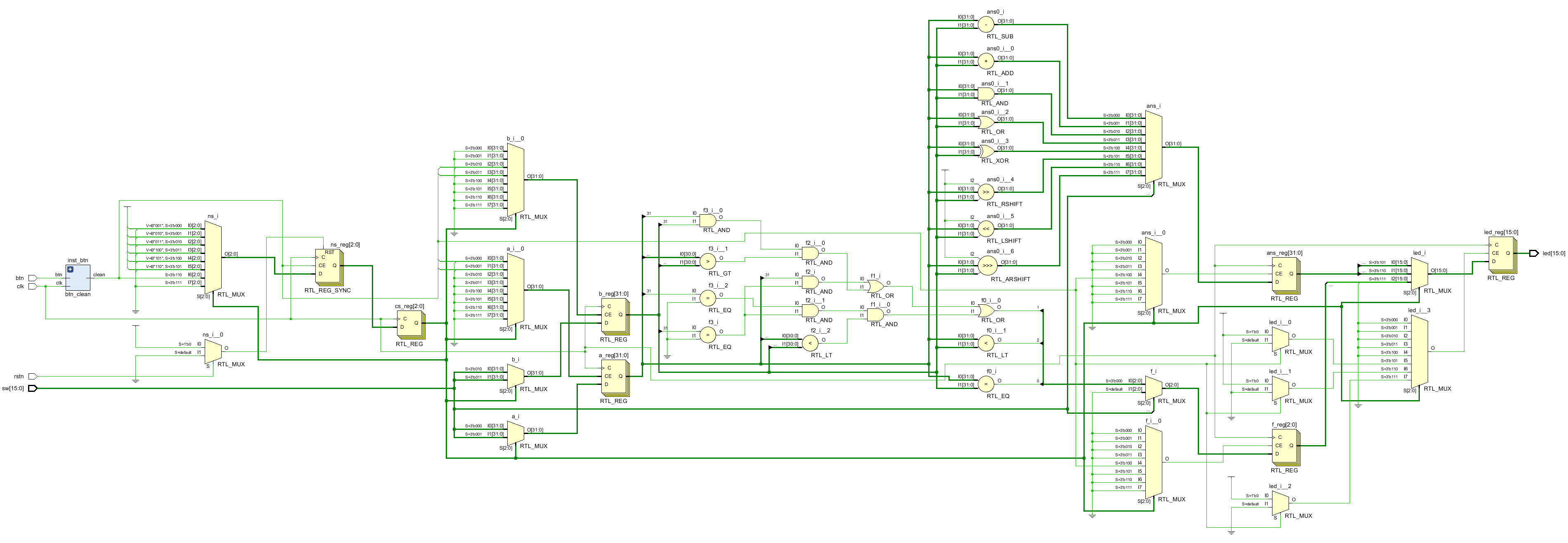
**2#** y = 32’h 88f1664c；f = 3’b 000；

**3#** y = 32’h ffffff0e；f = 3’b 000；

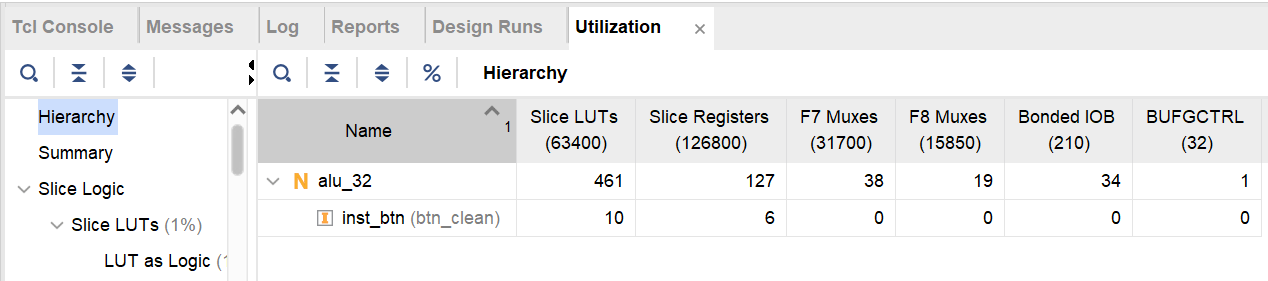
测试结果：

|  |  |  |
| --- | --- | --- |
| 结果高位 |  |  |
| 结果低位 |  |  |
| f[2:0] |  |  |
|  | **0#** 正确 | **1#** 正确 |
| 结果高位 |  |  |
| 结果低位 |  |  |
| f[2:0] |  |  |
|  | **2#** 正确 | **3#** 正确 |

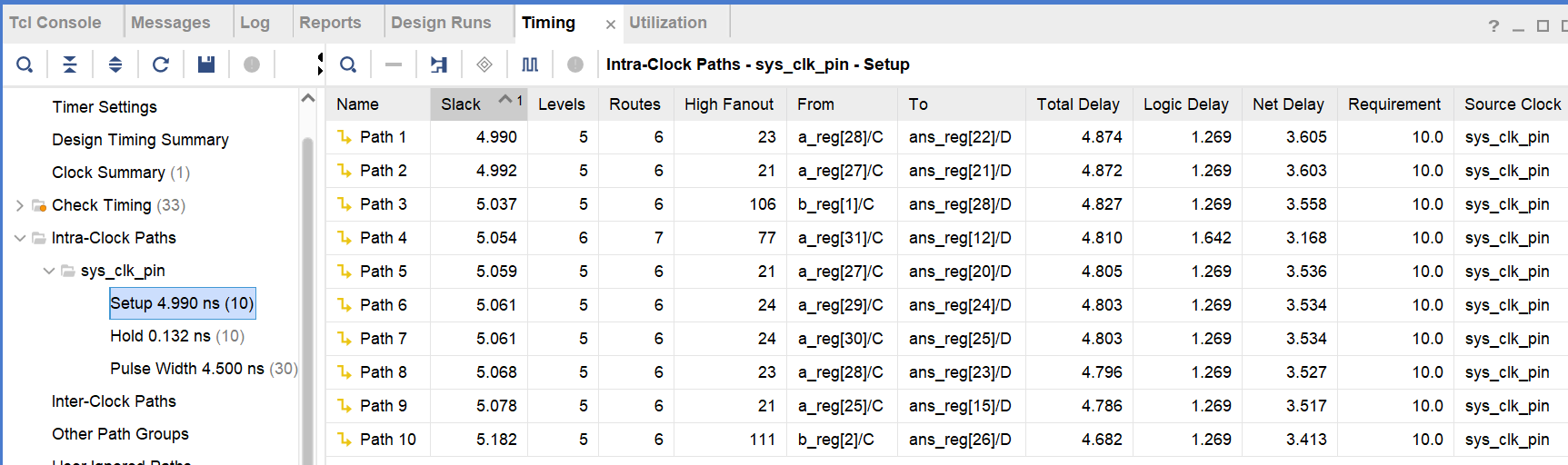
* 32位ALU电路性能
* 32位ALU RTL分析电路



* 32位ALU电路资源



* 32位ALU电路性能



总结篇

* 收获

经过这次实验，我重温了Verilog各种语法和状态机的设计实现，为今后的实验做好了准备。这次实验是我第一次拿到FPGA电路板Nexys4并在其上进行烧写测试，感觉很新奇，也很独特，这令枯燥的实验增添了几分色彩。此外，在进行这次实验的过程中，我多次尝试编写testbench来调试各个小模块，这在以前的数电实验中是欠缺的，这次实验补足了我这方面的能力，我也因此解决了很多编写上的问题，可以算是达到了实验的目的。

* 建议

1. 这次实验中，我一开始发现算术右移和逻辑右移的结果是完全一模一样的，经过搜索才知道右移时Verilog默认左操作数按无符号数解释，并且可以用$signed()函数强制按有符号数解释；但在编写有符号数小于的功能时，单用$signed()函数又不起作用。希望老师在讲Verilog基础语法之外，也可以适当补充一些Verilog的特性和内置函数。
2. 本次实验的要求之一是查看电路资源和电路性能，但是老师并没有给出分析这两项报告的标准，使得这两项任务变得较为鸡肋。建议删去，或者补充。