**实验二 运算器组成实验**

# 一、 实验目的

* 熟悉 Logisim 软件平台。
* 掌握运算器基本工作原理
* 掌握运算溢出检测的原理和实现方法；
* 理解有符号数和无符号数运算的区别；
* 理解基于补码的加/减运算实现原理；
* 熟悉运算器的数据传输通路。

# 二、 实验环境

Logisim 是一款数字电路模拟的教育软件，用户都可以通过它来学习如何创建逻辑电路，方便简单。 它是一款基于 Java 的应用程序，可运行在任何支持 JAVA 环境的平台，方便学生来学习设计和模仿数字逻辑电路。Logisim 中的主要组成部分之一就在于设计并以图示来显示 CPU。当然 Logisim 中还有其他多种组合分析模型来对你进行帮助，如转换电路，表达式，布尔型和真值表等等。同时还可以重新利用小规模的电路来作为大型电路的一部分。

<http://www.cburch.com/logisim/docs.html>

三、 实验内容

### **1.** 八位串行可控加减法电路设计

利用已经封装好的全加器（封装 1）设计 8 位串行可控加减法电路，其引脚电路如下图所示。

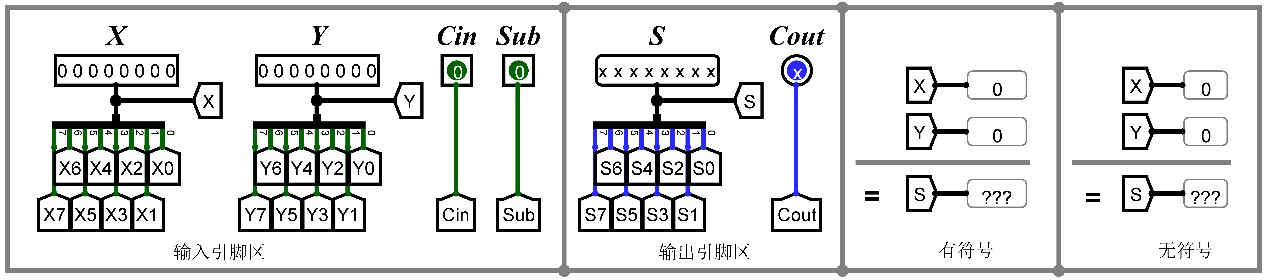
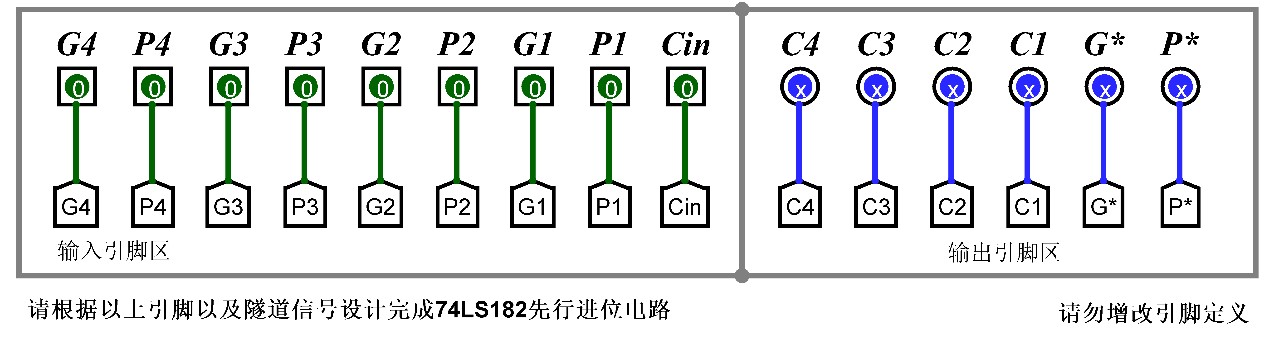


图 **1** 八位串行可控加减法电路引脚定义

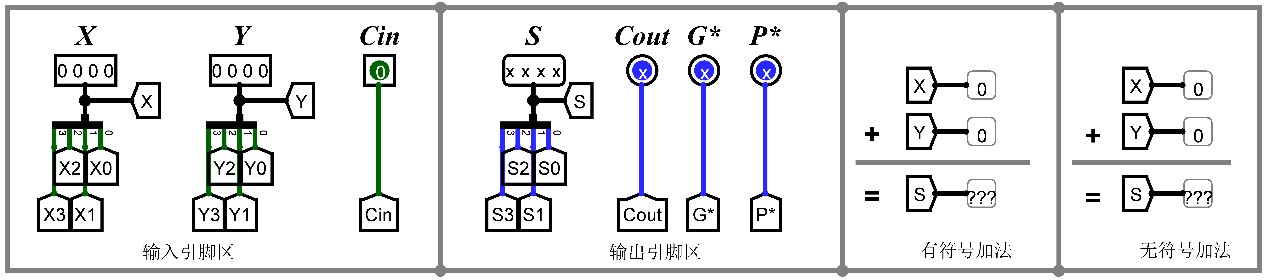
### **2.** 四位先行进位电路

根据下图定义的输入输出引脚完成 4 位先行进位电路。

图 **2** 四位先行进位电路引脚定义

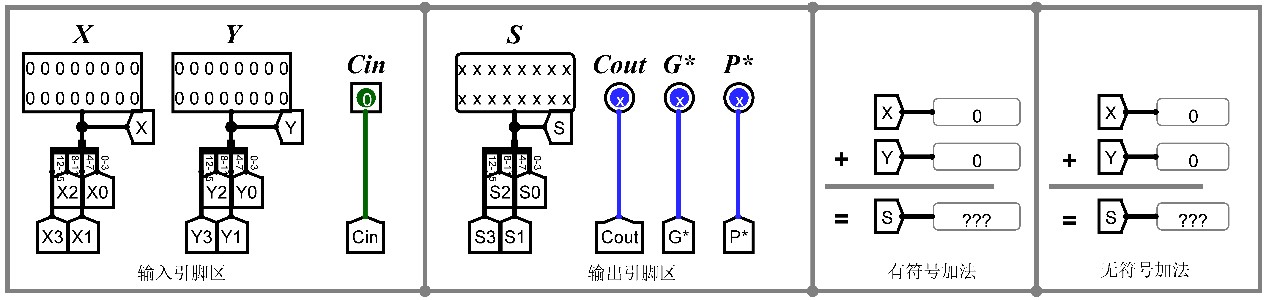
### **3.(1)** 四位快速加法器设计

利用已经前一步设计好的四位先行进位电路构造四位快速加法器，其引脚定义如下图。

图 **3** 四位快速加法器引脚定义

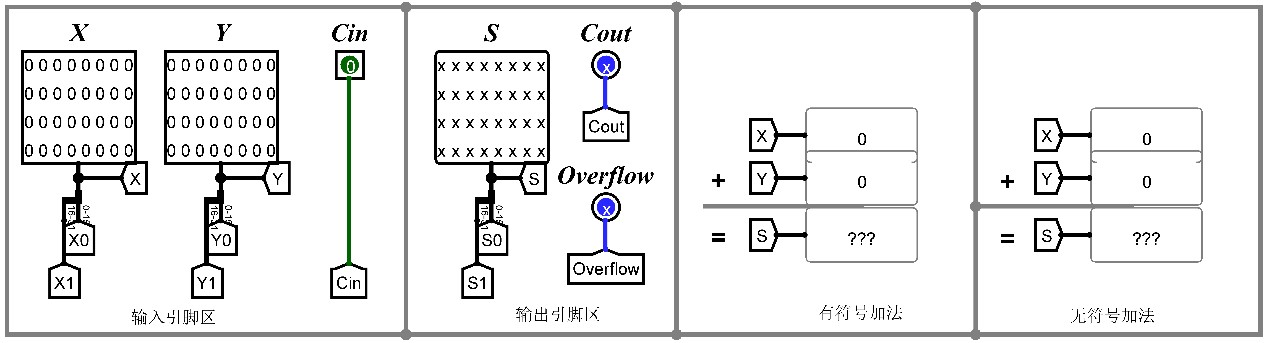
### **(2)** 十六位快速加法器设计

利用四位先行进位电路和四位快速加法器构造十六位组间先行进位，组内先行进位快速加法器，其引脚定义如下图。

图 **4** 十六位快速加法器引脚定义

### **(3)**32位快速加法器设计

利用前面构建的部件完成 32 位快速加法器，并分析其时间延迟，其引脚定义如下。

 图 **5 32** 位快速加法器引脚定义

## 4. 32 位 MIPS 运算器设计

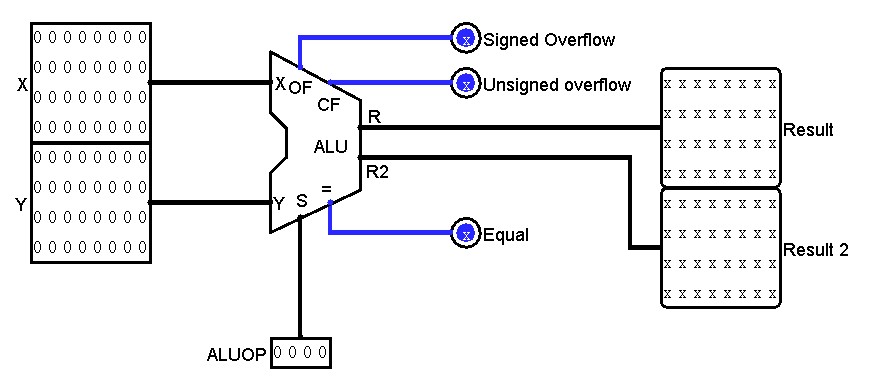
构建 **32** 位运算器。利用封装好的 32 位加法器以及 logisim 平台中现有运算部件（禁用系统自带的加法器，减法器）构建一个 32 位运算器，可支持算术加、减、乘、除，逻辑与、或、非、异或运算、逻辑左移、逻辑右移，算术右移运算，支持常用程序状态标志（有符号溢出 OF、无符号溢出 CF，结果相等 Equal），运算器功能以及输入输出引脚见下表，在主电路中详细测试自己封装的运算器，在报告中分析该运算器的优缺点。

表 **1.** 芯片引脚与功能描述

|  |  |  |  |
| --- | --- | --- | --- |
| 引脚 | 输入/输出 | 位宽 | 功能描述 |
| X | 输入 | 32 | 操作数 X |
| Y | 输入 | 32 | 操作数 Y |
| ALU\_OP | 输入 | 4 | 运算器功能码，具体功能见下表 |
| Result | 输出 | 32 | ALU 运算结果 |
| Result2 | 输出 | 32 | ALU 结果第二部分，用于乘法指令结果高位或除法指令的余数位，其他操作为零 |
| OF | 输出 | 1 | 有符号加减溢出标记，其他操作为零 |
| UOF | 输出 | 1 | 无符号加减溢出标记，其他操作为零  溢出条件（加法和小于加数，减法差大于被减数） |
| Equal | 输出 | 1 | Equal=(x==y)?1:0, 对所有操作有效 |

表 **2.** 运算符功能

|  |  |  |
| --- | --- | --- |
| ALU OP | 十进制 | 运算功能 |
| 0000 | 0 | Result = X << Y 逻辑左移 （Y 取低五位）  Result2=0 |
| 0001 | 1 | Result = X >>>Y 算术右移 （Y 取低五位）  Result2=0 |
| 0010 | 2 | Result = X >> Y 逻辑右移 （Y 取低五位）  Result2=0 |
| 0011 | 3 | Result = (X \* Y)[31:0]; Result2 = (X \* Y)[63:32] 有符号 |
| 0100 | 4 | Result = X/Y; Result2 = X%Y 无符号 |
| 0101 | 5 | Result = X + Y (Set OF/UOF) |
| 0110 | 6 | Result = X - Y (Set OF/UOF) |
| 0111 | 7 | Result = X & Y 按位与 |
| 1000 | 8 | Result = X | Y 按位或 |
| 1001 | 9 | Result = X⊕Y 按位异或 |
| 1010 | 10 | Result = ~(X |Y) 按位或非 |
| 1011 | 11 | Result = (X < Y) ? 1 : 0 符号比较 |
| 1100 | 12 | Result = (X < Y) ? 1 : 0 无符号比较 |

 图 **6** 运算器封装示意图

（请直接在实验包中的 **alu.circ** 中构建，构建后的电路应可以直接在 **alutest.circ** 中使用，相应封装应正好连接对应电路），最终结果将在 **alutest.circ** 文件中由教师进行详细测试并自动评分，请各位同学在 **alutest.circ** 中详细测试自己的电路，可以申请多次测试，但每多一次测试总分扣减 **5%**。

# 四、 实验步骤

## 1、实验准备

1. 复习有关运算器的内容，对数据通路的构成、数据在数据通路中的流动及控制方法有基本的了解。
2. 熟悉电路中各部分的关系及信号间的逻辑关系
3. 设计实验电路，画出各模块的图，注意各引脚的标注，节省实验的时间。

## 2、实验步骤

实验可按照自己设计的电路或参考电路按照搭积木的方式进行。先完成运算器的数据通路部分，在运算器部分能够正确完成各类运算的基础上，再增加累加器等其他部件。

# 五、 结果提交

请将完成后的 alu.circ 文件作为实验结果提交给班级指导教师当场检查并归档。

六、 实验报告要求

1. 实验目的；
2. 各模块的设计电路和系统的整体电路,对设计要进行详细的分析与说明；
3. 实验结果的记录与分析；
4. 列出操作步骤及顺序,标出重要的开关控制端；
5. 实验收获和体会；
6. 实验中碰到的问题和解决的方法。

# 七、 注意事项

* 不要对时钟信号进行门级操作，在实际电路中这是非常糟糕的设计，会导致一系列严重的故障，如险象。
* 大区域拷贝粘贴移动电路可能会导致logisim崩溃，请随时ctrl+s保存电路。
* Logisim工具栏器件可以改变其默认属性，可以根据需要修

改。

* 红色信号线肯定是明显的错误，通常在复杂电路中会出现，调试的时候应注意是否出现以下情况引起：

