

《 计算机组成原理 》

实验报告本

|  |  |
| --- | --- |
| 系 别： | **计算机科学与技术** |
| 班 级： | **计213** |
| 学 号： | **21013134** |
| 姓 名： | **徐昊博** |
| 指导教师： | **李冬冬** |

信息科学与工程学院

2022-2023学年 第2学期

### 实验一 **8位算术逻辑运算实验**

**实验一成绩： 教师签名： 日期：**

**一、实验目的**

1、掌握简单运算器的数据传送通路组成原理。

2、验证算术逻辑运算功能发生器 74LS181的组合功能。

**二、实验装置**

DVCC系列计算机组成原理系统

**三、实验内容**

1. 8位算术逻辑运算实验

1）实验原理

实验中所用的运算器数据通路如图所示。其中运算器由两片74LS181以并/串形成8位字长的ALU构成。运算器的输出经过一个三态门74LS245(U33)到内部数据总线BUSD0～D7插座BUS1～2中的任一个（跳线器JA3为高阻时为不接通），内部数据总线通过LZD0～LZD7显示灯显示；运算器的两个数据输入端分别由二个锁存器74LS273（U29、U30）锁存，两个锁存器的输入并联后连至内部总线BUS，实验时通过8芯排线连至外部数据总线EXD0～D7插座EXJ1～EXJ3中的任一个；参与运算的数据来自于8位数据开并KD0～KD7，并经过一三态门74LS245（U51）直接连至外部数据总线EXD0～EXD7，通过数据开关输入的数据由LD0～LD7显示。

图中算术逻辑运算功能发生器 74LS181（U31、U32）的功能控制信号S3、S2、S1、S0、CN、M并行相连后连至6位功能开关，以手动方式用二进制开关S3、S2、S1、S0、CN、M来模拟74LS181（U31、U32）的功能控制信号S3、S2、S1、S0、CN、M；其它电平控制信号LDDR1、LDDR2、ALUB`、SWB`以手动方式用二进制开关LDDR1、LDDR2、ALUB、SWB来模拟，这几个信号有自动和手动两种方式产生，通过跳线器切换，其中ALUB`、SWB`为低电平有效，LDDR1、LDDR2为高电平有效。

另有信号T4为脉冲信号，在手动方式下进行实验时，只需将跳线器J23上T4与手动脉冲发生开关的输出端SD相连，按动手动脉冲开关，即可获得实验所需的单脉冲。



1. 实验接线

2.1）文字说明

本实验用到4个主要模块：⑴低8位运算器模块，⑵数据输入并显示模块，⑶数据总线显示模块，⑷功能开关模块（借用微地址输入模块）。

根据实验原理详细接线如下：

1、J20,J21,J22,接上短路片，

2、J24，J25，J26接左边；

3、J27,J28 右边；

4、J23 置右边T4选“ SD”

5、JA5 置“接通”；

6、JA6 置“手动”；

7、JA3 置“接通”；

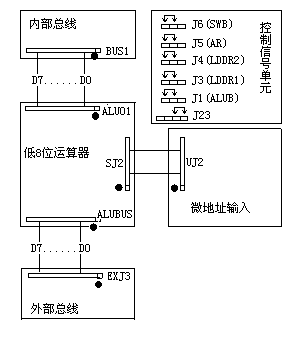
8、JA1,JA2,JA4置“高阻”；

9、JA8 置上面“微地址”

10、EXJ1接BUS3

11、开关CE 、AR 置1

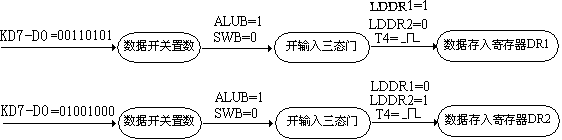
2.2）接线图



3）实验步骤

⑴ 连接线路，仔细查线无误后，接通电源。

⑵ 用二进制数码开关KD0～KD7向DR1和DR2寄存器置数。方法：关闭ALU输出三态门（ALUB`=1），开启输入三态门（SWB`=0），输入脉冲T4按手动脉冲发生按钮产生。设置数据开关具体操作步骤图示如下：



说明：LDDR1、LDDR2、ALUB`、SWB`四个信号电平由对应的开关LDDR1、LDDR2、ALUB、SWB给出，拨在上面为“1”，拨在下面为“0”，电平值由对应的显示灯显示，T4由手动脉冲开关给出。

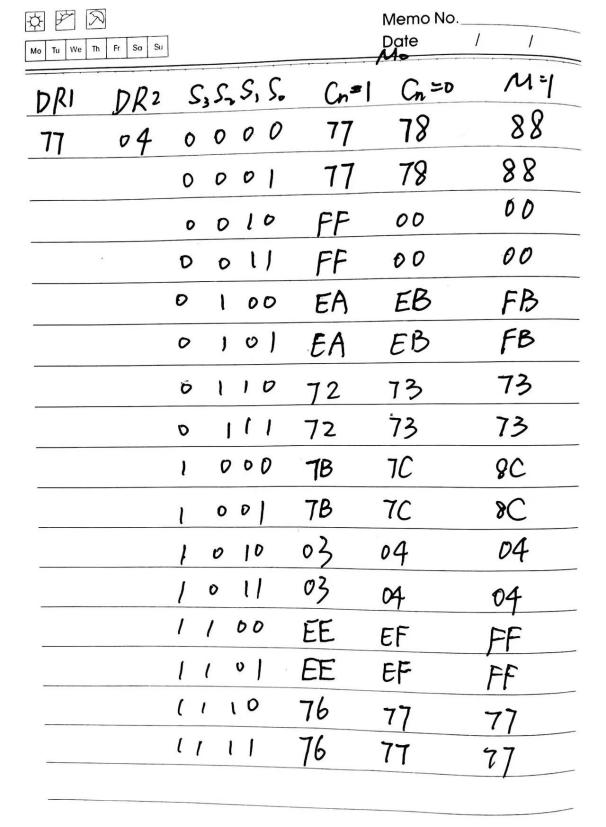
⑶ 检验DR1和DR2中存入的数据是否正确，利用算术逻辑运算功能发生器 74LS181的逻辑功能，即M=1。具体操作为：关闭数据输入三态门SWB`＝1，打开ALU输出三态门ALUB`＝0，当置S3、S2、S1、S0、M为1 1 1 1 1时，总线指示灯显示DR1中的数，而置成1 0 1 0 1时总线指示灯显示DR2中的数。

⑷ 验证74LS181的算术运算和逻辑运算功能（采用正逻辑）

在给定DR1=35、DR2=48的情况下，改变算术逻辑运算功能发生器的功能设置，观察运算器的输出，填入表3—1中，并和理论分析进行比较、验证。

4）实验结果分析

我们小组进行系统测试，选取的DR1与DR2分别为77和04，实验结果记录如下：



2．带进位控制8位算术逻辑运算实验

1）实验原理

带进位控制运算器的实验原理如图3－2所示，在实验⑴的基础上增加进位控制部分，其中高位74LS181（U31）的进位CN4通过门UN4E、UN2C、UN3B进入UN5B的输入端D，其写入脉冲由T4和AR信号控制，T4是脉冲信号，在手动方式下进行实验时，只需将跳线器J23上T4与手动脉冲发生开关的输出端SD相连，按动手动脉冲开关，即可获得实验所需的单脉冲。AR是电平控制信号（低电平有效），可用于实现带进位控制实验。从图中可以看出，AR必须为“0”电平，D型触发器74LS74（UN5B）的时钟端CLK才有脉冲信号输入。才可以将本次运算的进位结果CY锁存到进位锁存器74LS74（UN5B）中。

2）实验接线

2.1）文字说明

1、J20,J21,J22,接上短路片，

2、J24，J25，J26接左边；

3、J27,J28 右边；

4、J23 置右边T4选“ SD”

5、JA5 置“接通”；

6、JA6 置“手动”；

7、JA3 置“接通”；

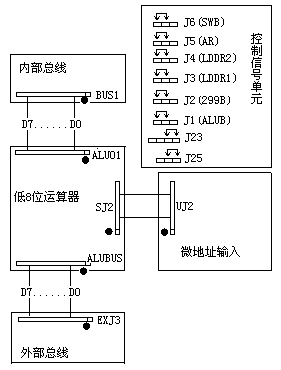
8、JA1,JA2,JA4置“高阻”；

9、JA8 置上面“微地址”

10、EXJ1接BUS3

11、CE、299B 置“1”，AR置“0”；

2.2）接线图



3）实验步骤

⑴ 仔细查线无误后，接通电源。

⑵ 用二进制数码开关KD0～KD7向DR1和DR2寄存器置数，方法：关闭ALU输出三态门ALUB=1，开启输入三态门SWB=0，输入脉冲T4按手动脉冲发生按钮产生。如果选择参与操作的两个数据分别为55H、AAH，将这两个数存入DR1和DR2的具体操作步骤如下：



⑶ 开关ALUB=0，开启输出三态门，开关SWB=1，关闭输入三态门，同时让LDDR1=0，LDDR2=0。

⑷ 如果原来有进位，即CY=1，进位灯亮，但需要清零进位标志时，具体操作方法如下：

·S3、 S2、 S1、S0、 M 的状态置为0 0 0 0 0，AR信号置为“0”电平（清零操作时DR1寄存 器中的数应不等于FF）。

·按动手动脉冲发生开关，CY=0，即清进位标志。

注：进位标志指示灯CY亮时表示进位标志为“1”，有进位；进位标志指示灯CY灭时，表示进位位为“0”，无进位。

⑷ 验证带进位运算及进位锁存功能

这里有两种情况：

·进位标志已清零，即CY=0，进位灯灭，此时，使开关CN=0，再来进行带进位算术运算。例如步骤⑵参与运算的两个数为55H和AAH，当S3、 S2、 S1、 S0、 M 状态为1 0 0 1 0 ，此时输出数据总线显示灯上显示的数据为DR1加DR2再加初始进位位“1”（因CN=0），相加的结果应为ALU=00，并且产生进位，此时按动手动脉冲开关，则进位标志灯亮，表示有进位。如果开关CN=1，则相加的结果ALU=FFH，并且不产生进位。

·原来有进位，即CY=1，进位灯亮，此时不考虑CN的状态，再来进行带进位算术运算。同样步骤⑵参与运算的两个数为55H和AAH，当S3、 S2、 S1、 S0、 M 状态为1 0 0 1 0 ，此时输出数据总线显示灯上显示的数据为DR1加DR2再加当前进位标志CY，相加的结果同样为ALU=00，并且产生进位，此时按动手动脉冲开关，则进位标志灯亮，表示有进位。

4）实验结果分析

我所在小组的实验结果如下：

输入数 A: 00101100 (44)

输入数 B: 00011010 (26)

进位控制信号 Cin: 1 (允许进位)

选择进位加法操作：S0-S3 = 0001

输出结果 F: 01000110 (70)，Cout: 1 (有进位)

**四、分析思考题**

**1、**简述8位算术逻辑运算实验的数据通路。

1. 输入：实验的数据通路需要接收两个8位的输入数 A 和 B，以及运算控制信号 Select，用于选择执行不同的算术和逻辑运算。

2. 寄存器：输入数 A 和 B 可以通过寄存器进行存储。寄存器用于暂时保存输入数据，并提供数据的稳定性。

3. 运算单元：数据通路包含一个运算单元，用于执行各种算术和逻辑运算，如加法、减法、与、或、异或等。运算单元可以根据 Select 控制信号选择执行不同的运算。

4. 运算结果：运算单元会产生一个8位的运算结果，表示运算的输出。

5. 进位和溢出：在执行加法或减法时，可能会有进位和溢出的情况。数据通路中会包含相关的电路来检测进位和溢出，并生成相应的进位和溢出标志。

6. 选择器：根据 Select 控制信号，选择运算单元的输入和输出。选择器可以根据需要将输入连接到寄存器或直接连接到运算单元，将运算结果连接到输出寄存器或其他模块。

7. 输出：运算结果可以被存储到输出寄存器中，以便进一步处理或显示。

数据通路的具体设计取决于实验的要求和目标。它可以包含多个模块和电路，如寄存器、运算单元、选择器、进位和溢出检测电路等。这些模块相互连接，形成一个数据通路，用于执行8位算术和逻辑运算。

**2、**简述74LS181的组合功能

74LS181主要功能如下：

1. 加法：74LS181可以执行四位二进制数的加法。它具有两个四位的数据输入端（A和B），一个进位输入端（Cin），以及一个选择输入端（S0-S3）用于选择加法操作。通过将S0-S3设置为二进制编码的"0001"，74LS181将执行加法操作，并在四位输出端（F0-F3）提供加法结果。如果加法操作产生进位，则会在进位输出端（Cout）上输出高电平。

2. 减法：74LS181也可以执行四位二进制数的减法。通过将S0-S3设置为二进制编码的"0010"，74LS181将执行减法操作。在减法操作中，输入B被取反并加1（即使用二进制补码表示），然后与输入A执行加法运算。减法结果将在四位输出端（F0-F3）上提供，进位输出端（Cout）将指示是否产生借位。

3. 逻辑功能：除了加法和减法，74LS181还具有逻辑功能。通过将S0-S3设置为不同的二进制编码，可以选择多种逻辑操作，如与、或、异或、或非等。例如，将S0-S3设置为二进制编码的"0110"，将执行A与B的与运算，并将结果提供在四位输出端（F0-F3）上。

4. 逻辑操作的补充功能：74LS181还提供了逻辑操作的补充功能，如逻辑非和逻辑与非。通过将S0-S3设置为不同的二进制编码，可以选择逻辑非（NOT）和逻辑与非（NAND）操作。

除了上述提到的功能，74LS181还有其他的功能，一并总结在以下表格当中：



总的来说，74LS181是一个功能强大的ALU芯片，可以执行多种组合功能，包括加法、减法和逻辑操作。它适用于计算、控制和处理器等数字电路设计中。、**3、**简述8位算术逻辑运算实验与带进位控制8位算术逻辑运算实验的主要不同点。

8位算术逻辑运算实验和带进位控制8位算术逻辑运算实验之间的主要不同点在于进位控制的存在。

1. 8位算术逻辑运算实验：

该实验主要关注执行8位数的算术和逻辑运算，如加法、减法、与、或、异或等。没有明确的进位控制信号，进位是根据运算结果自动发生的。

2. 带进位控制的8位算术逻辑运算实验：

该实验在8位算术逻辑运算的基础上加入了进位控制功能。引入了额外的进位控制信号，用于控制进位的发生或禁止。

主要区别在于带进位控制的实验中，引入了对进位进行控制的信号，使得可以手动控制进位的发生或禁止。这可以用于某些特定的应用场景，例如在特定条件下实现加法的快速运算或实现一些特定的算法。

需要注意的是，具体实验的差异可能还取决于所使用的电路元件、设计方法和目标要求。上述描述的不同点仅是一般情况下的概括，实际实验中可能还会有其他的差异和细节。

**4、**自己设置几个数，在实验机上测试带进位控制8位算术逻辑运算实验运算结果。

我所在小组的实验结果如下：

输入数 A: 00101100 (44)

输入数 B: 00011010 (26)

进位控制信号 Cin: 1 (允许进位)

选择进位加法操作：S0-S3 = 0001

输出结果 F: 01000110 (70)，Cout: 1 (有进位)

**实验二**  微处理器运算实验

**实验二成绩： 教师签名： 日期：**

**一、实验目的**

1、掌握时序信号发生电路组成原理。

2、掌握微程序控制器的设计思想和组成原理。

3、掌握微程序的编制、写入，观察微程序的运行。

**二、实验装置**

DVCC系列计算机组成原理系统

**三、实验内容**

1.微控制器实验

1）实验原理

实验所用的时序电路原理如图所示,可产生4个等间隔的时序信号TS1～TS4 ，其中SP为时钟信号，由实验机上时钟源提供，可产生频率及脉宽可调的方波信号。学生可根据实验要求自行选择方波信号的频率及脉宽。为了便于控制程序的运行，时序电路发生器设计了一个启停控制触发器UN1B，使TS1～TS4信号输出可控。图中“运行方式”、“运行控制”、“启动运行”三个信号分别是来自实验机上三个开关。当“运行控制”开关置为“运行”，“运行方式”开关置为“连续”时，一旦按下“启动运行”开关，运行触发器UN1B的输出QT一直处于“1”状态，因此时序信号TS1～TS4将周而复始地发送出去；当“运行控制”开关置为“运行”，“运行方式”开关置为“单步”时，一旦按下“启动运行”开关，机器便处于单步运行状态，即此时只发送一个CPU周期的时序信号就停机。利用单步方式，每次只运行一条微指令，停机后可以观察微指令的代码和当前微指令的执行结果。另外，当实验机连续运行时，如果“运行方式”开关置“单步”位置，也会使实验机停机。



2）实验接线

2.1）文字说明

本实验用到4个主要模块：⑴低8位运算器模块，⑵数据输入并显示模块，⑶数据总线显示模块，⑷功能开关模块（借用微地址输入模块）。

根据实验原理详细接线如下：

1、J20,J21,J22,接上短路片，

2、J24，J25，J26接左边；

3、J27,J28 右边；

4、J23 置右边T4选“ SD”

5、JA5 置“接通”；

6、JA6 置“手动”；

7、JA3 置“接通”；

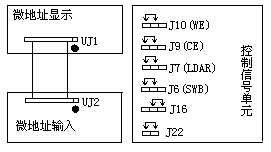
8、JA1,JA2,JA4置“高阻”；

9、JA8 置上面“微地址”

10、EXJ1接BUS3

11、开关CE 、AR置1

2.2）接线图

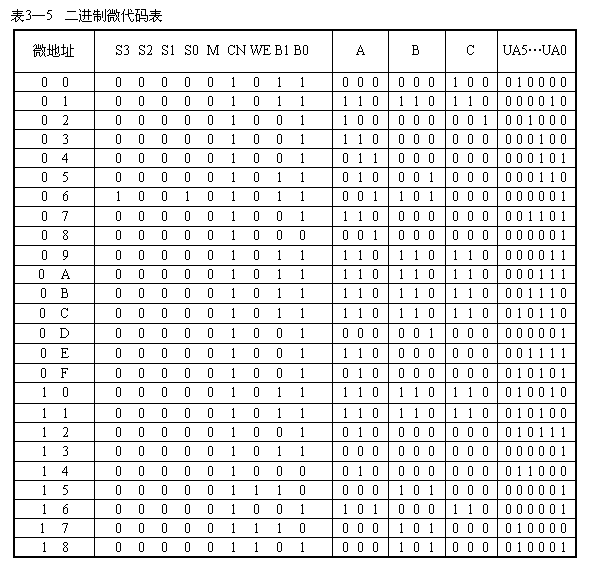


1. 实验步骤

⑴ 根据机器指令画出对应的微程序流程图，如图所示；

⑵ 根据微程序流程图设计微指令，并按微指令格式转换成二进制代码，如表所示。





⑶ 实验接线

1、J20，J21，J22，ZI2接上短路片，

2、JJ23，J24，J25，J26接左边；

3、J27,J28 右边；

4、J29不接；

5、JA1，JA2，JA3，JA4置“高阻”；

6、JA5置“接通”；

7、JA8置“微程序”

8、JA6 置“微控”

⑷仔细查线无误后接通电源

⑸进一步了解微程序控制器的工作原理

① 写微程序

A. “编程开关”置为“写入”状态。

B. “运行控制”开关置为“运行”，“运行方式”开关置为“单步”状态。

C. 用二进制模拟开关UA0～UA5置6位微地址，UA0～UA5的电平由LK0～LK5显示，高电平亮，低电平灭。

D. 用二进制模拟开关MK1～MK24置24位微代码，24位微代码由LMD1～LMD24显示灯显示，高电平亮，低电平灭。

E. 按动“启动运行”开关，启动时序电路，即可将微代码写入到E2PROM 2816的相应地址单元中。

F . 重复C－E步骤，将表3－5的微代码全部写入E2PROM 2816中。

② 读微程序

A. 将“编程开关”设置为“读”状态。

B. “运行控制”开关置为“运行”，“运行方式”开关置为“单步”状态。

C. 用二进制模拟开关UA0～UA5置6位微地址。

D. 按动“启动运行”开关，启动时序电路，读出微代码，观察显示灯LMD1～LMD24的状态，检查读出的微代码是否与写入的相同，如果不同，则将“编程开关”置为“写入”状态。重新执行①即可。

③ 单步运行

A. “编程开关”置于“运行”状态。

B. “运行控制”开关置为“运行”，“运行方式”开关置为“单步”状态。

C. 系统总清，即“总清”开关拨0→1。使微地址寄存器U14～U16清零，从而明确本机的运行入口微地址为000000（二进制）。

D. 按动“启动运行”开关，启动时序电路，则每按动一次，运行一条微指令后停机，此时实验机上的微地址显示灯和微程序显示灯将显示所读出的一条指令。

④　连续运行

A. 将“编程开关”置为“运行”状态。

B. “运行控制”开关置为“运行”，“运行方式”开关置为“连续”状态。

C. 系统总清，即“总清”开关拨0→1。使微地址寄存器U14～U16清零，从而明确本机的运行入口微地址为000000（二进制）。

D. 按动“启动运行”开关，启动时序电路，则连续运行。

4）实验结果分析

通过按照实验步骤，将所有的微指令存入微地址当中，按动“启动运行”开关，启动时序电路，每按动一次，运行一条微指令后停机，我们小组输入的微指令为02，经过多次运行可以见到微指令在01，02，08之间不断循环，符合微程序流程图所示的循环规律。经过检验，其余的微程序也符合要求。在打开连续状态的指令显示后指令最终也会按照图中的流程快速来回闪烁循环，表面实验指令设置没有问题。

**四、分析思考题**

1. 时序信号在这个实验中的作用。

时序信号的作用主要在以下几个方面：

1. 时钟信号：时钟信号是微处理器中最基本的时序信号。它确定了微处理器内部操作的节奏和同步，使得各个部件按照同步的时序进行工作。时钟信号的频率决定了微处理器的工作速度。

2. 读写时序信号：读写时序信号指示了内存或外设与微处理器进行数据读取或写入的时间窗口。它定义了数据有效性、地址稳定性和控制信号的时机，确保正确的数据交换和通信。

3. 存储器时序信号：存储器时序信号涉及到微处理器与存储器之间的数据传输和存取操作。它包括读周期（读取数据的时间）、写周期（写入数据的时间）、地址稳定时间和存储器响应时间等，以确保正确的数据存储和检索。

4. 控制信号时序：控制信号时序包括微处理器内部的各种控制信号，如指令执行、操作码译码、寄存器读写、算术逻辑运算等。这些控制信号的时序决定了微处理器内部各个功能模块的操作顺序和时机，确保指令的正确执行。

5. 中断信号时序：中断信号是微处理器中一种特殊的时序信号。它用于通知微处理器发生了某种特定的事件，需要中断当前的执行流程，并跳转到中断处理程序。中断信号的时序关系到中断请求的检测、中断响应的时机以及中断处理程序的执行。

时序信号的正确配置和控制对于微处理器的正常运行至关重要。微处理器设计者必须仔细定义和管理时序信号，以确保数据的准确传输、操作的正确执行以及系统的稳定性。在微处理器运算实验中，时序信号的正确性和稳定性会直接影响实验结果的准确性和可重复性。

2. 微程序是如何写入微程序控制器的？

1. 设计微指令：根据微处理器的需求和指令集架构，设计出微指令的操作和控制逻辑。微指令是微程序的最小执行单元，它们定义了微处理器的各个操作步骤和控制信号。

2. 编写微程序：根据微指令的设计，将微程序编写成一系列微指令的指令序列。编写微程序通常采用类似汇编语言的符号表示方法，将每个微指令按照特定的格式编写。

3. 确定微指令编码：为了将微程序写入微程序控制器的存储器中，需要为每个微指令分配一个唯一的编码。编码通常采用二进制形式表示，并根据微程序控制器的特性进行编排和分配。

4. 存储器编程：将编写好的微程序按照确定的微指令编码写入微程序控制器的存储器中。存储器可以是可编程的ROM（只读存储器）或EEPROM（可擦写可编程存储器）。

5. 配置微程序控制器：将存储器中的微程序加载到微程序控制器中。这通常涉及将存储器芯片与微程序控制器的相应引脚连接，并进行适当的配置和编程操作。

需要注意的是，微程序的写入过程可能因微处理器的架构和微程序控制器的设计而有所不同。一些微程序控制器可能提供专门的工具或软件来帮助编写和编程微程序。此外，某些微处理器架构也支持动态微指令的生成和加载，以便在运行时修改微程序的行为。

总而言之，微程序的写入是一个将设计好的微指令转化为二进制编码，并将其存储到微程序控制器中的过程。这样，微程序控制器可以按照微程序的指令序列来控制微处理器的操作和执行。

### 实验三 运算器设计

**实验三成绩： 教师签名： 日期：**

### 第1关 8位可控加减法电路设计

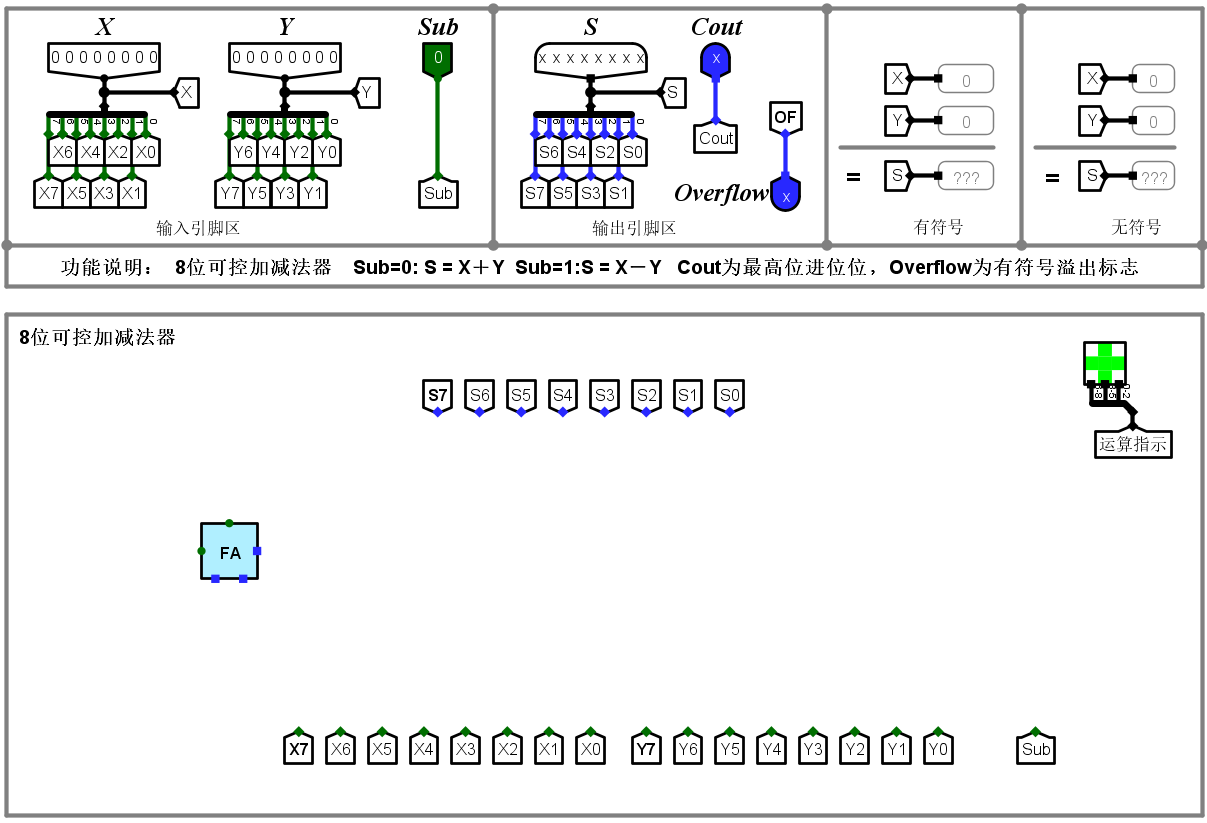
**一、实验目的**

帮助学生掌握一位全加器的实现逻辑，掌握多位可控加减法电路的实现逻辑，熟悉 Logisim 平台基本功能，能在 logisim 中实现多位可控加减法电路。

**二、实验环境**

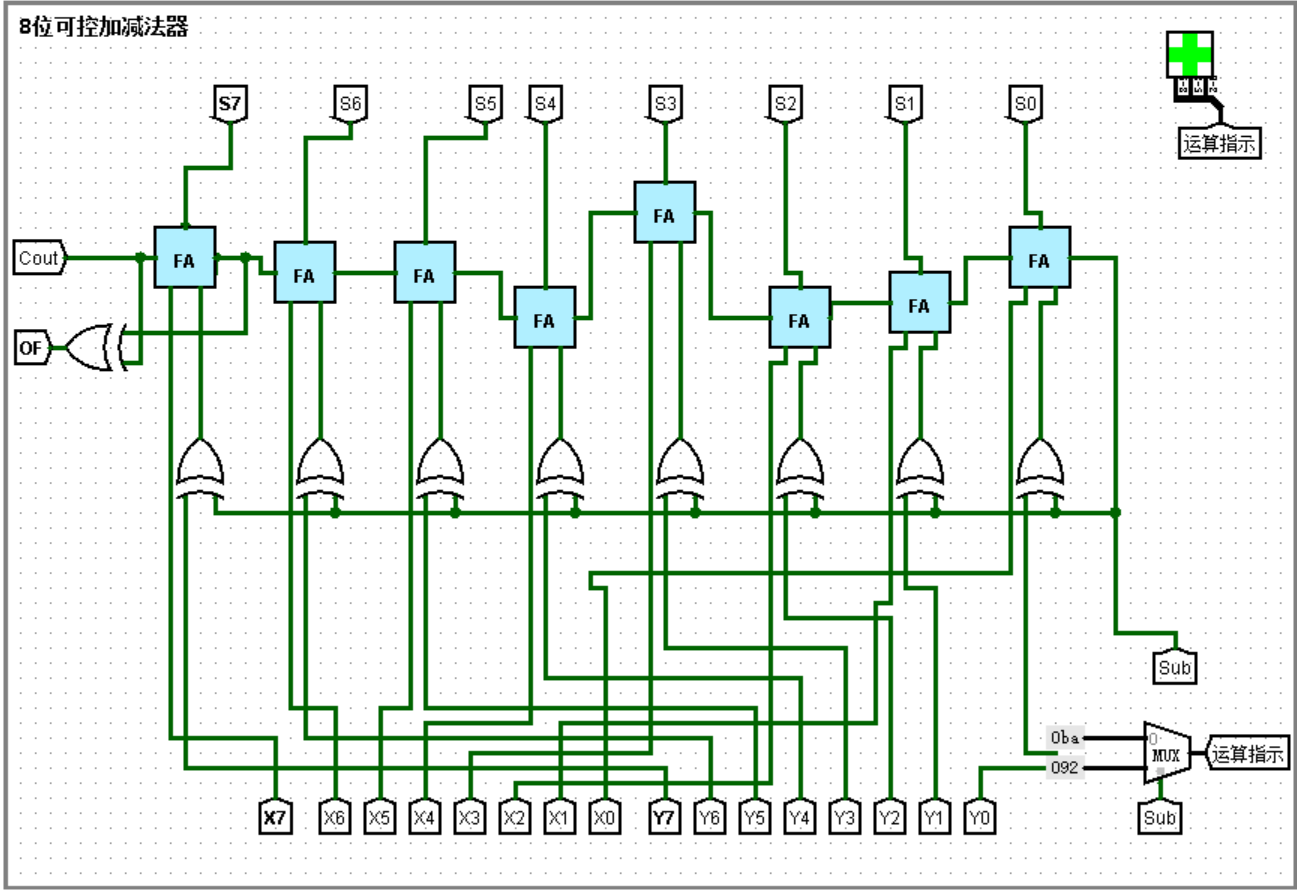
1. 实验所使用的工具软件：Logisim
2. 实验平台：[https://www.educoder.net/classrooms/17032/shixun\_homework](https://www.educoder.net/classrooms/13035/shixun_homework)

**三、实验内容**

在 Logisim 模拟器中打开 alu.circ 文件，在对应子电路中利用已经封装好的全加器设计8位串行可控加减法电路，其电路引脚定义如图所示，用户可以直接使用在电路中使用对应的隧道标签，其中 X，Y 为两输入数，Sub 为加减控制信号，S 为运算结果输出，Cout 为进位输出，OF 为有符号运算溢出位。



**四、实验方案设计**



说明：SUB是加减控制信号，根据已经封装好的FA加法器的性质，将X与Y的输入与FA连接输出计算结果与S连接，左端输出，被加数再与SUB控制信号进行连接在8位可控加减法电路中，将输入的加数 Y 与 SUB（加减控制信号）使用异或连接的目的是实现减法操作。

当 SUB 为低电平时，异或连接将保持 Y 输入不变，从而实现加法操作。因为异或操作的特性是当两个输入不同时输出为高电平，当两个输入相同时输出为低电平。当 SUB 为高电平时，异或连接将对 Y 输入进行取反操作，从而实现减法操作。因为异或连接的另一个特性是当两个输入相同时输出为低电平，当两个输入不同时输出为高电平。这就实现了对 Y 输入的取反操作。

通过使用异或连接将加数 Y 与 SUB 相连，可以根据 SUB 控制信号的不同值，选择是保持加法（SUB = 0）还是进行减法（SUB = 1）。这样就实现了8位可控加减法电路中的加法和减法操作。

**五、实验步骤**

1、将其中一个加数Y将其每一位与SUB进行XOR逻辑操作，当SUB为0时对应的在FA中就是加法操作，当SUB为1的时候在FA中为减法操作

2、在连接电路时将处理好的Y与X一同输入

3、处理符号溢出时将最高位的计算结果计算前与计算后输出的信号做XOR逻辑操作最终得到OF的隧道信号

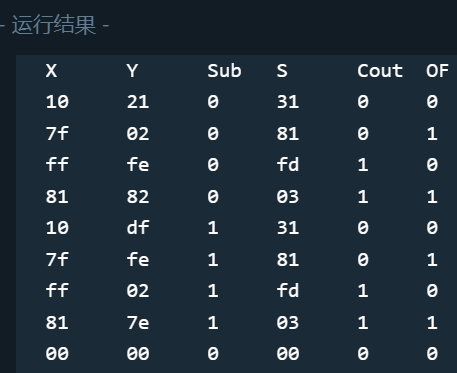
4、将8个计算结果串联输出得到Cout隧道门，至此完成了整个电路设计

5、将本实验用的logisim文件改为记事本下阅读，复制代码到平台上运行即可。

**六、故障与调试**

经过调试通过测试，本次实验没有故障。

1. **实验结果测试与分析**



### 如图按照测试案例，对运行结果来说，X+Y的值用S来表示结果，对于有进位的结果，在Cout处得到显示，而OF显示的是有溢出位数的情况，可见所有结果都正确。

### 第2关 CLA182四位先行进位电路设计

**一、实验目的**

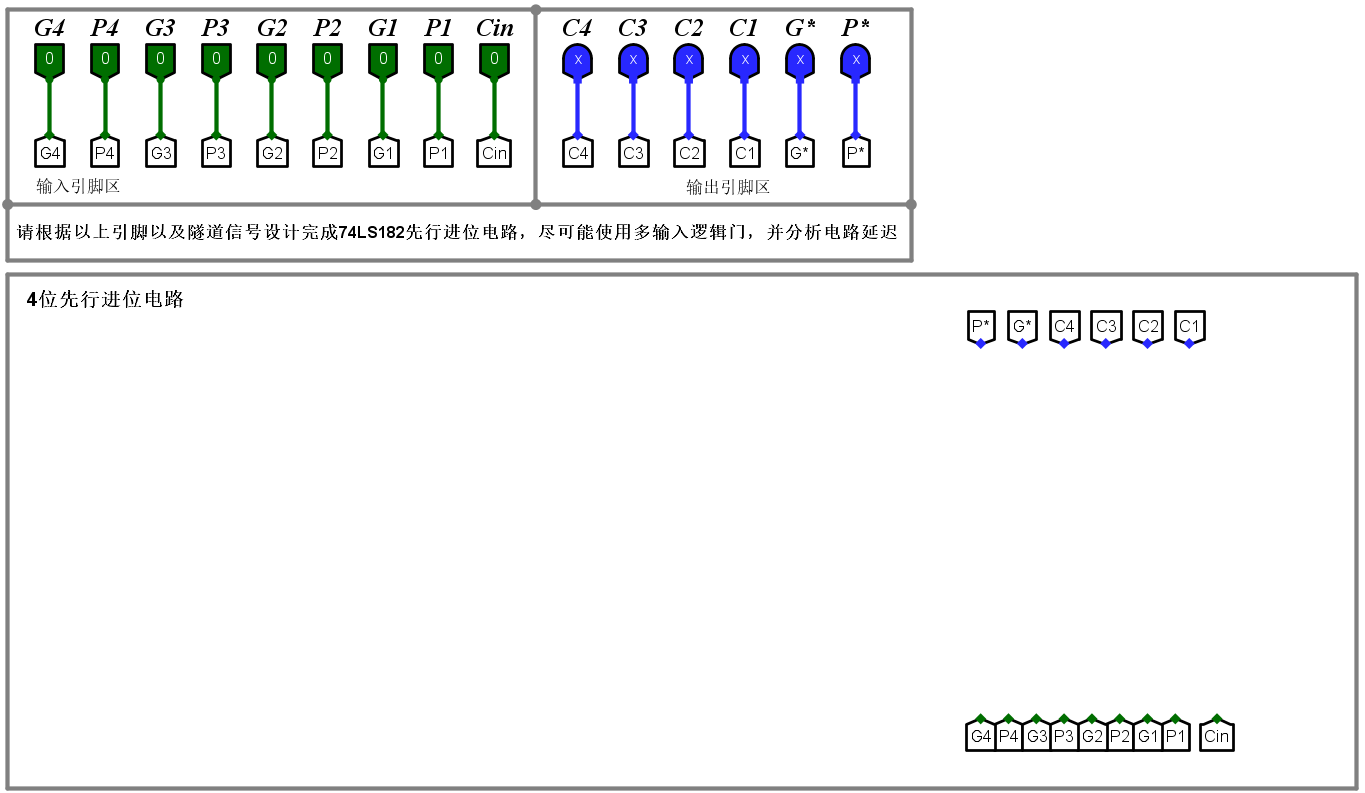
帮助学生掌握快速加法器中先行进位的原理，能利用相关知识设计4位先行进位电路，并利用设计的4位先行进位电路构造4位快速加法器，能分析对应电路的时间延迟。

**二、实验环境**

1. 实验所使用的工具软件：Logisim
2. 实验平台：[https://www.educoder.net/classrooms/17032/shixun\_homework](https://www.educoder.net/classrooms/13035/shixun_homework)

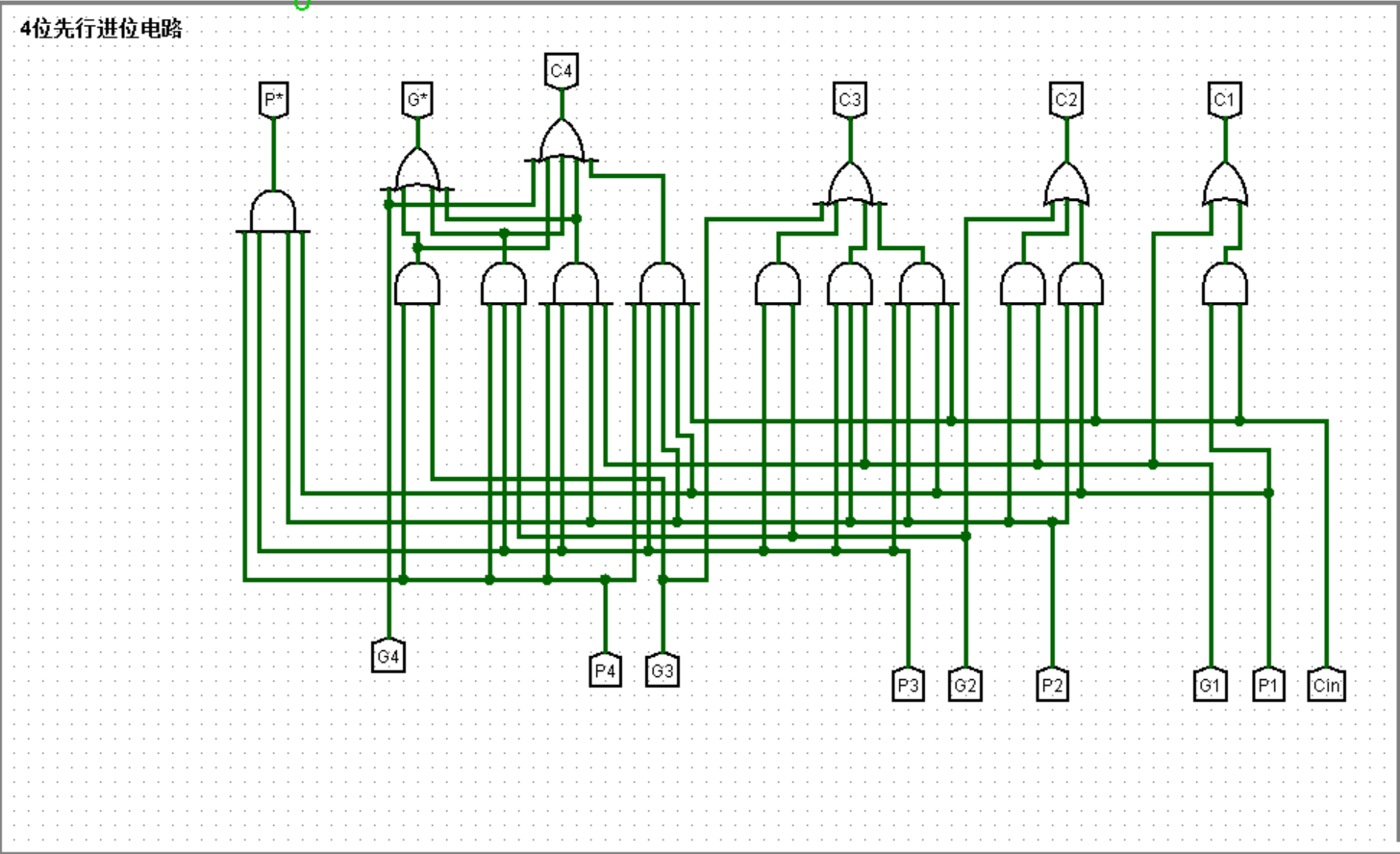
**三、实验内容**

在 Logisim 中打开 alu.circ 文件，按照图中定义的输入输出引脚，在对应子电路中实现可级联的4位先行进位电路。其中 Gi，Pi 为进位生成函数和传递函数，Cin 为进位输入，C1~C4 为进位输出，G，P 为成组进位生成函数和成组进位传递函数。电路框架和引脚如下所示：





**四、实验方案设计**



对于上述门电路，其输出门主要遵循下面的逻辑来进行：最低位的输出门C由地位进位信号Cin与P1进行与运算再与第二个输出G2进行或运算这样就能得到最低位是否有进位的信息，按照同样的道理，当计算C2是否有进位时，不仅要考虑前两位的进位情况，还要考虑刚刚输出的C1是否有进位，所以还需要P1，G1，P2的三个门的与操作，以此类推到第三个门C3，第四个门C4。综合来看步骤如下：

1. 连接第一个CLA Generator（CLA1）：

输入：Gi和Pi（i = 0）。

输出：C1。

Gi和Pi直接作为CLA1的输入。CLA1通过异或门计算Gi，并将结果作为C1的输出。

2. 连接第二个CLA Generator（CLA2）：

输入：Gi、Pi和C1（i = 1）。

输出：C2。

Gi和Pi直接作为CLA2的输入。C1连接到CLA2的与门输入。CLA2通过异或门计算Gi，并将结果作为C2的输出。

3. 连接第三个CLA Generator（CLA3）：

输入：Gi、Pi、C1和C2（i = 2）。

输出：C3。

Gi和Pi直接作为CLA3的输入。C1和C2连接到CLA3的与门输入。CLA3通过异或门计算Gi，并将结果作为C3的输出。

4. 连接第四个CLA Generator（CLA4）：

输入：Gi、Pi、C1、C2和C3（i = 3）。

输出：C4。

Gi和Pi直接作为CLA4的输入。C1、C2和C3连接到CLA4的与门输入。CLA4通过异或门计算Gi，并将结果作为C4的输出。

**五、实验步骤**

1. 设计四个先行进位生成器，用于生成每一位的进位生成信号 Gi 和进位传递信号 Pi。每个先行进位生成器的输入为前一位的进位传递信号 Pi-1 和进位输入 Cin，输出为对应的 Gi 和 Pi。

2. 将 G1、G2、G3 和 G4 信号按照设计分别连接到与门，产生成组进位生成函数 G。

3. 将 P1、P2、P3 和 P4 信号按照设计分别连接到或门，产生成组进位传递函数 P。

4. 将最低位的进位传递信号 P1 直接作为进位输出 C1。

5. 对于其他位，将进位传递信号 Pi 和该位的进位输入 Ci-1 连接到一个与门，产生该位的进位输出 Ci。这样，通过使用修正后的CLA182四位先行进位电路设计，可以实现高速的四位先行进位运算，输出每一位的进位信号 C1~C4。

6.将本实验用的logisim文件改为记事本下阅读，复制代码到平台上运行即可。

**六、故障与调试**

1. 故障一
2. 故障现象

输出的部分进位结果当Cin=1时，均与理想结果不同。

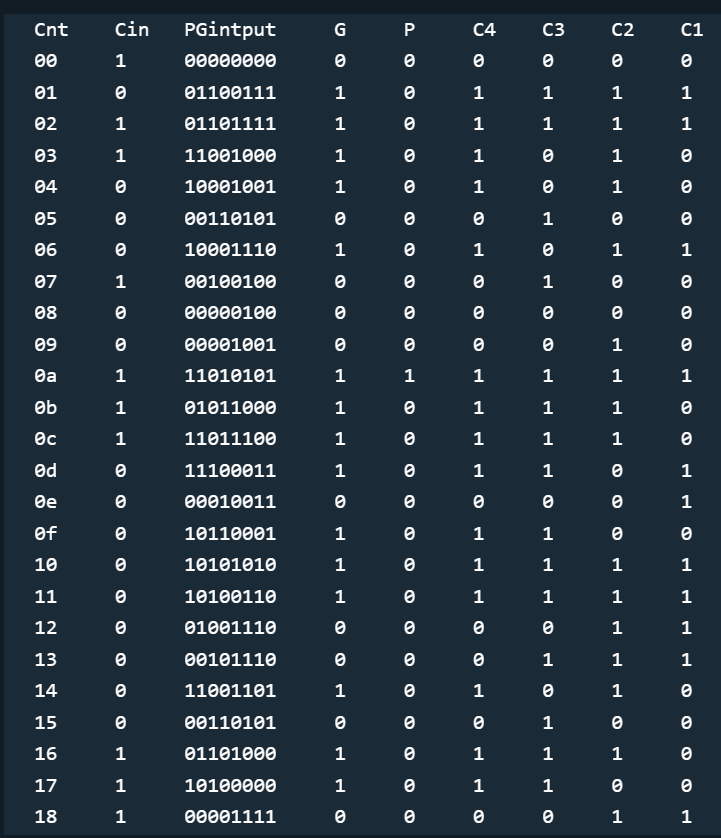
1. 原因分析

经过门电路分析发现前位Cin的进位没有考虑在内导致所有当Cin为1的输出错误

1. 解决方案

将Cin进位信号进行与操作到G，P输入信号当中

**七、实验结果测试与分析**



说明：本次测试结果一共有8x16=144个测试案例，考虑到报告的简洁性，在这里仅仅展示一部分：可以看到前四位为G的输入与后四位P的输入，以第二个例子为例

在这个例子中，我们有：

1. 最低位的加法运算：0 + 1 + 1 = 10。这里的G0为0，P0为1，前一位的进位C0为1。根据加法规则，和S0为0，进位C1为1。

2. 第二位的加法运算：1 + 1 + 1 = 11。这里的G1为1，P1为1，前一位的进位C1为1。根据加法规则，和S1为1，进位C2为1。

3. 第三位的加法运算：1 + 1 + 1 = 11。这里的G2为1，P2为1，前一位的进位C2为1。根据加法规则，和S2为1，进位C3为1。

4. 最高位的加法运算：0 + 1 + 1 = 10。这里的G3为0，P3为1，前一位的进位C3为1。根据加法规则，和S3为0，进位C4为1。因此，最终的输出为C4C3C2C1 = 1111。结果正确。

### 第3关 4位快速加法器设计

**一、实验目的**

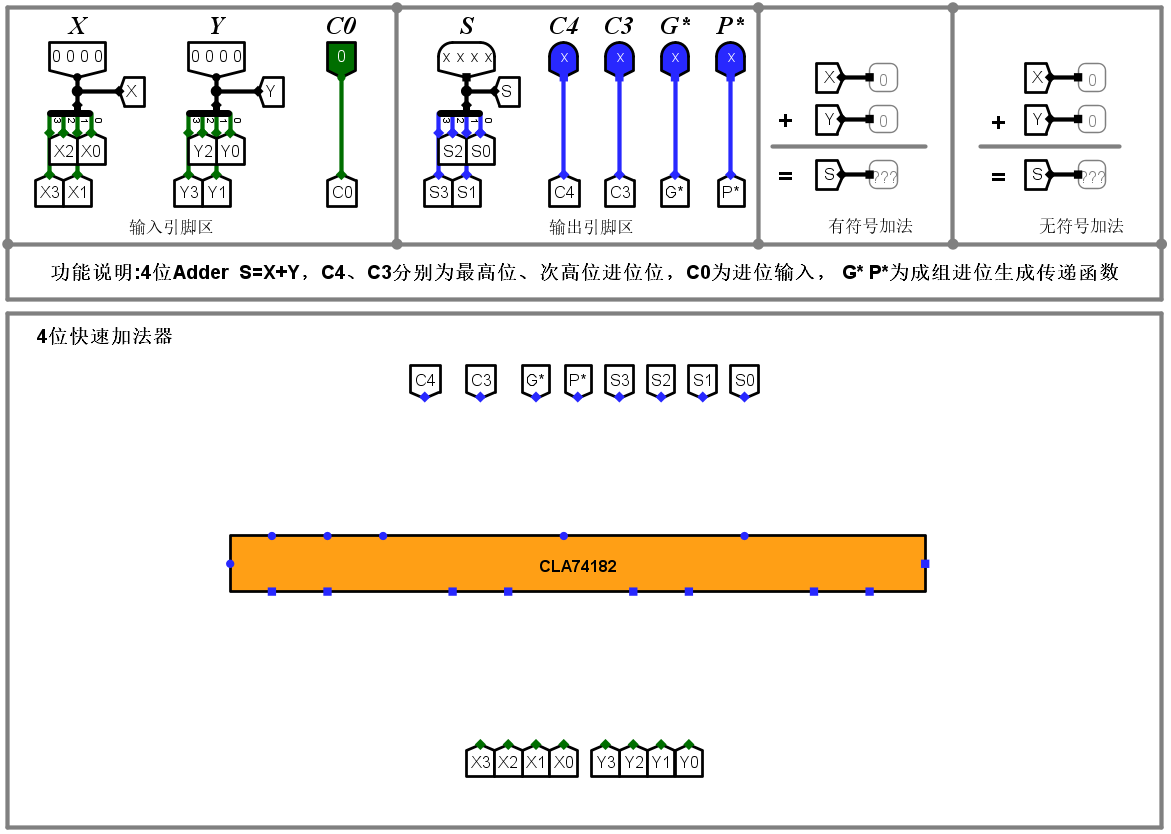
帮助学生掌握快速加法器中先行进位的原理，能利用相关知识设计4位先行进位电路，并利用设计的4位先行进位电路构造4位快速加法器，能分析对应电路的时间延迟。

**二、实验环境**

1. 实验所使用的工具软件：Logisim
2. 实验平台：[https://www.educoder.net/classrooms/17032/shixun\_homework](https://www.educoder.net/classrooms/13035/shixun_homework)

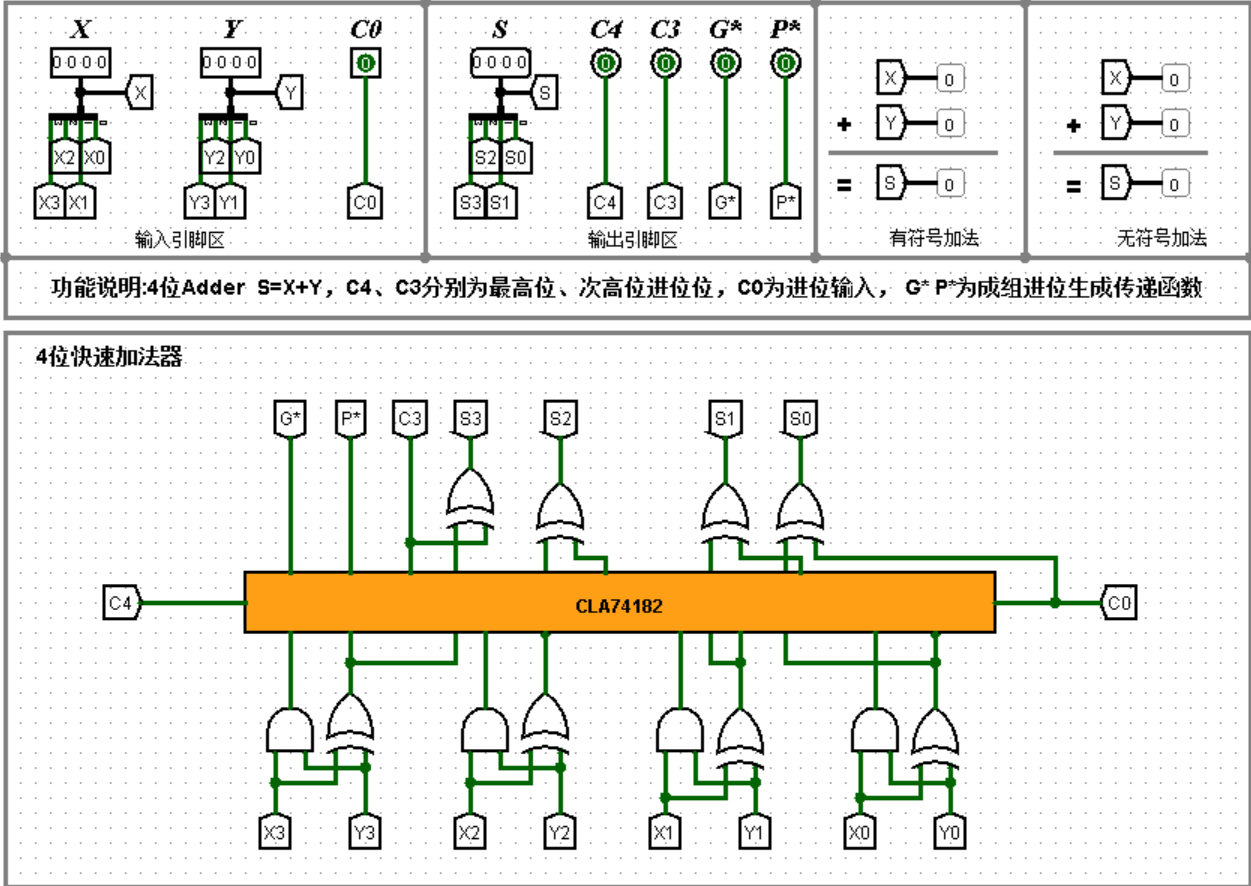
**三、实验内容**

利用前一步设计好的四位先行进位电路构造四位快速加法器，其引脚定义如图所示，其中 X，Y 为四位相加数，Cin 为进位输入，S 为和数输出，Cout 为进位输出，G，P 为 4 位成组进位生成函数和成组进位传递函数。电路框架和引脚如下所示：





**四、实验方案设计**



### 本次实验设计的原理是基于上一个CLA182四位先行进位电路设计的基础上进行了4为快速加法器的设计，首先要解决的是原先的四位先行进位电路中传递依赖函数的设计，对于原始的输入Xi与Yi，将X与Y进行异或操作得到传递函数P，将X与Y与操作得到了生成函数G，输出的信号需要将其与前一位的进位进行加和也就是异或操作得到最终输出门S。

1. **实验步骤**

1. 确定输入和输出：

两个四位相加数，命名为X3X2X1X0和Y3Y2Y1Y0，进位输入，命名为Cin，四位和数输出，命名为S3S2S1S0，进位输出，命名为Cout。

2. 连接电路：

将Cin连接到CLA74182的Cin输入，连接CLA74182的S输出作为和数输出，命名为S3S2S1S0，连接CLA74182的Cout输出作为进位输出，命名为Cout。

3. 设计成组进位生成函数G和成组进位传递函数P：

对于每个位（i=0到3），根据以下规则计算G和P。

Gi = Xi · Yi，表示位i的成组进位生成函数。

Pi = Xi + Yi，表示位i的成组进位传递函数。

连接成组进位生成函数和成组进位传递函数。

将组装好的P与G按顺序接入即可

4.将本实验用的logisim文件改为记事本下阅读，复制代码到平台上运行即可。

**六、故障与调试**

1. 故障一
2. 故障现象

部分导线出现红色冲突。

1. 原因分析

连接导线时，电子模块背面具有不透明性，导致导线没能连接下端接口而是连到了上端的接口。

1. 解决方案

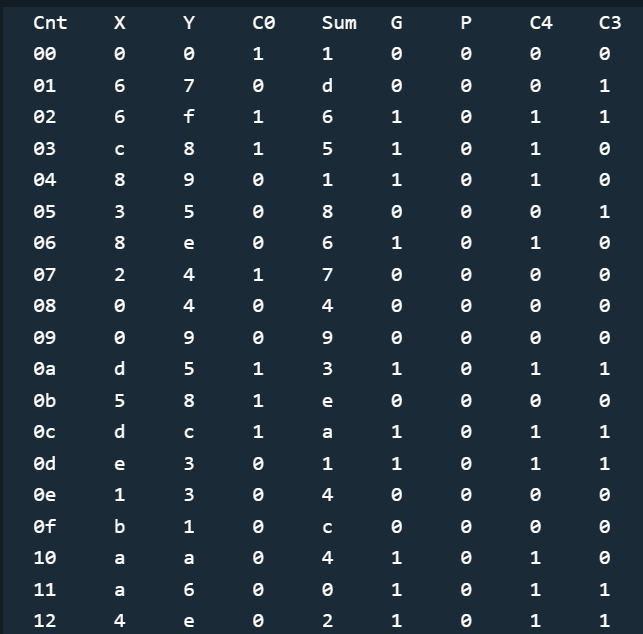
移动大型的电子元件进行检查，反复检查逻辑与测试调整。

1. **实验结果测试与分析**

说明：本次测试结果一共有8x16=144个测试案例，考虑到报告的简洁性，在这里仅仅展示一部分。

提交平台后以第二个例子为例：

在这个例子当中，输入的X为6，Y为7，前位的进位为0，得到的最终结果为d也就是十进制中的13，而最低位的进位C3显示为1也就是在最低位置有进位，符合实际情况，因而得到的是正确的结果。平台提供的测试结果如下：



### 第4关 16位快速加法器设计

**一、实验目的**

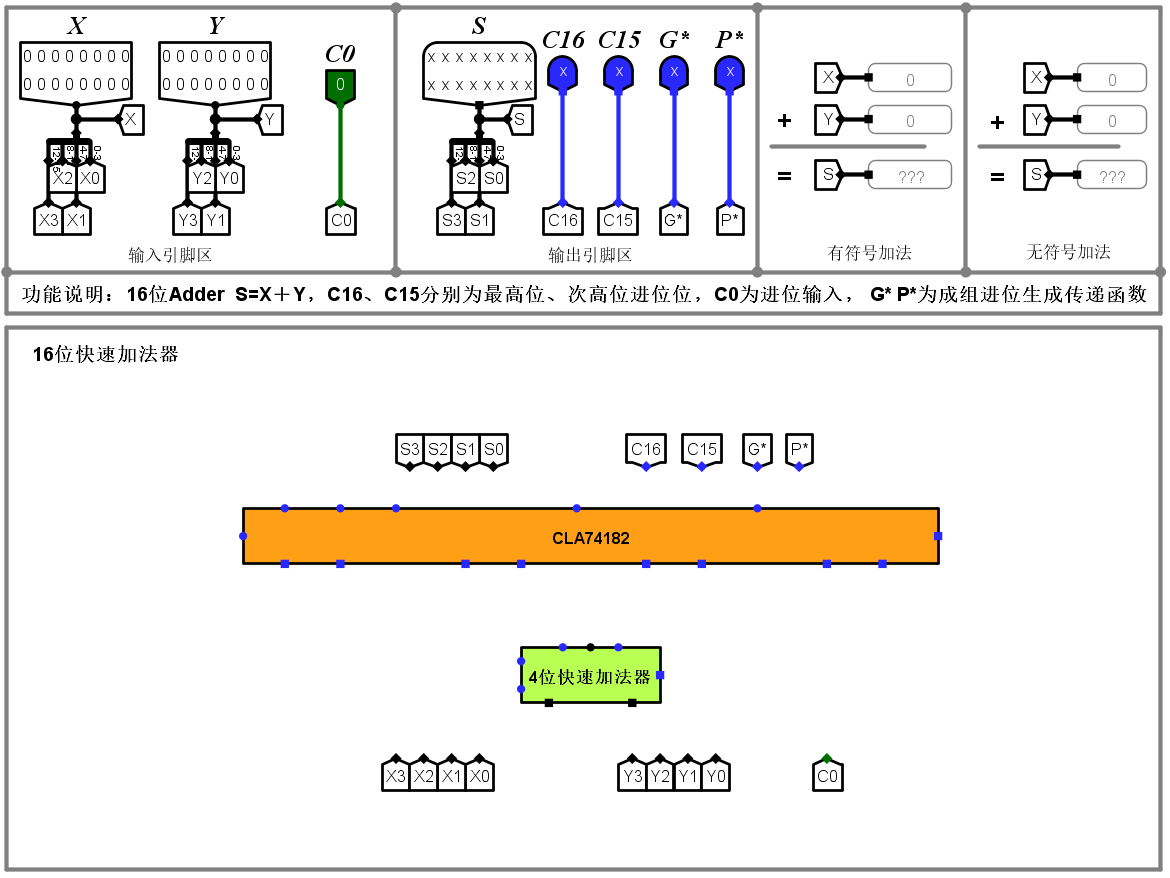
帮助学生理解成组进位产生函数，成组进位传递函数的概念，熟悉 Logisim 平台子电路的概念，能利用前述实验封装好的4位先行进位子电路以及4位快速加法器子电路构建16位快速加法器，并能利用相关知识分析对应电路的时间延迟，理解电路并行的概念。

二、实验环境

1. 实验所使用的工具软件：Logisim
2. 实验平台：[https://www.educoder.net/classrooms/17032/shixun\_homework](https://www.educoder.net/classrooms/13035/shixun_homework)

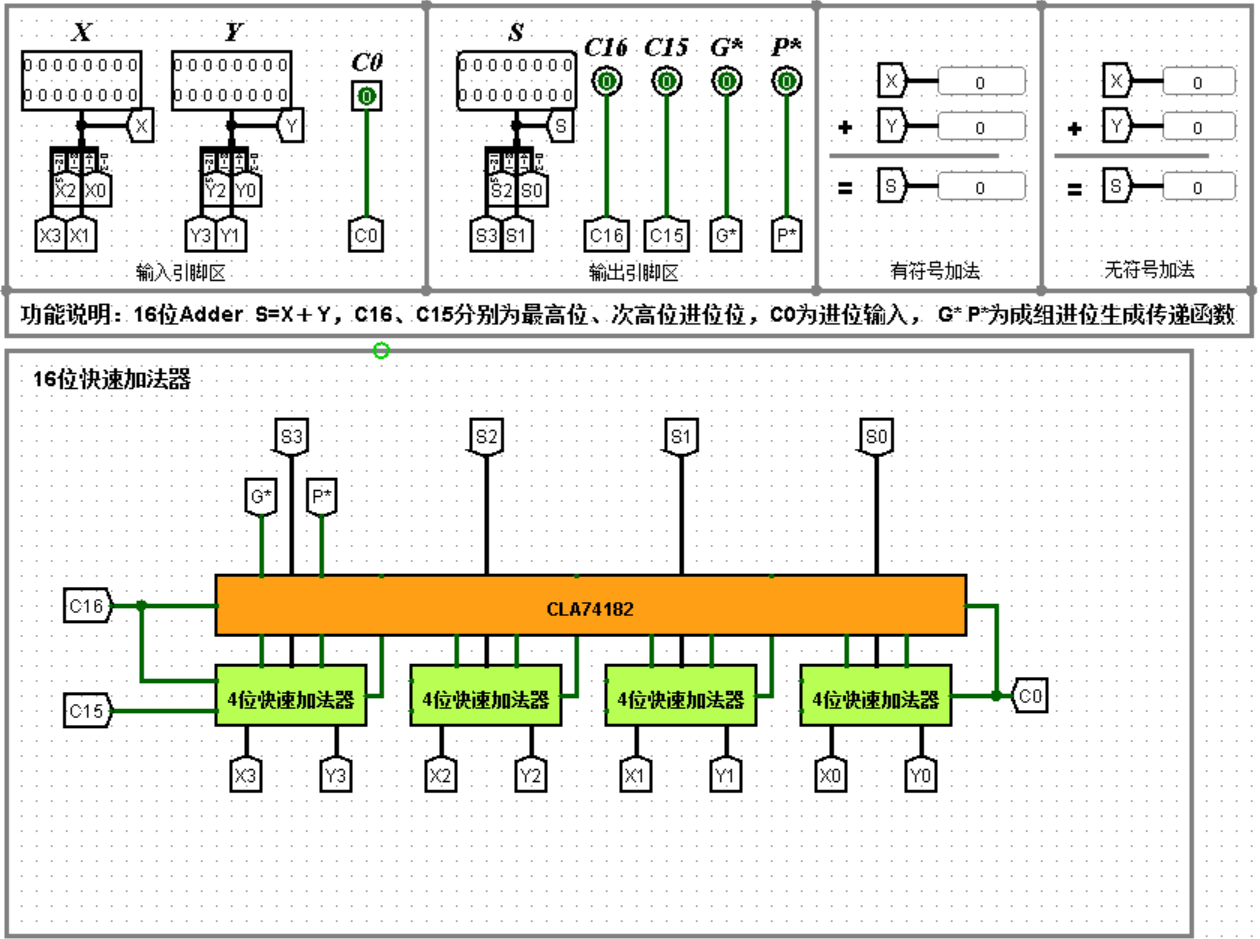
**三、实验内容**

在 Logisim 中打开 alu.circ 文件，在对应的子电路中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，并验证其功能是否正常，快速加法器引脚定义如图所示。其中 X，Y 为16位相加数，Cin 为进位输入，S 为和数输出，Cout 为进位输出，G，P 为16位成组进位生成函数和成组进位传递函数。电路框架和引脚如下所示：





**四、实验方案设计**



本次实验设计的原理是基于上一个四位快速加法器设计的基础上进行了16快速加法器的设计，由于16位数字可以拆分成四个四位数字，对于原始的输入Xi与Yi，所以将X与Y分为四个部分每个部分进行四位快速加法计算，最低位的计算需要考虑将进位信号置入，所以右端接入了C0，输出的信号需要将其与前一位的进位进行加和最终输出门S。与此同时将次高位与最高位的进位情况通过隧道进行输出，最终得到结果

**五、实验步骤**

1. 将输入分为四个四位块：X3X2X1X0、Y3Y2Y1Y0、X7X6X5X4、Y7Y6Y5Y4、X11X10X9X8、Y11Y10Y9Y8、X15X14X13X12、Y15Y14Y13Y12。

2. 将X3X2X1X0和Y3Y2Y1Y0连接到第一个四位快速加法器的X和Y输入。将其和数输出连接到S0。

3. 将X7X6X5X4和Y7Y6Y5Y4连接到第二个四位快速加法器的X和Y输入。将其和数输出连接到S1。

4. 将X11X10X9X8和Y11Y10Y9Y8连接到第三个四位快速加法器的X和Y输入。将其和数输出连接到S2。

5. 将X15X14X13X12和Y15Y14Y13Y12连接到第四个四位快速加法器的X和Y输入。将其和数输出连接到S3。

6.最高4位快速加法器左下端接入次高位进位C15，左上端接入最高位进位C16，最后输出传递生成函数G与P。

7.将本实验用的logisim文件改为记事本下阅读，复制代码到平台上运行即可。

**六、故障与调试**

1.故障一

1. 故障现象

部分导线在运行时出现红色冲突。

1. 原因分析

连接导线时，电子模块背面具有不透明性，四位加法器只有最右端是需要连接到进位器的最上端，而实际操作时有部分不慎连到了下端

1. 解决方案

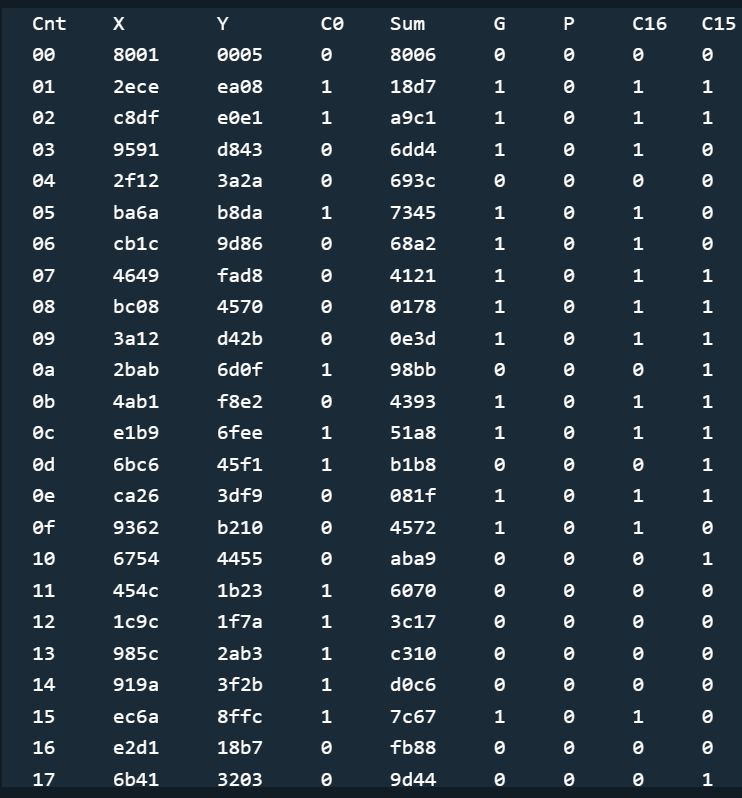
移动大型的电子元件进行检查，反复检查逻辑与测试调整。将测试数据手算找到原因。

1. **实验结果测试与分析**

说明：本次测试结果一共有6x16=96个测试案例，考虑到报告的简洁性，在这里仅仅展示一部分。

提交平台后以第一个例子为例：

在这个例子当中，输入的X为8001（H），Y为0005（H），前位的进位为0，得到的最终结果为8006，而进位C16显示为0也就是在最高位置没有有进位，符合实际情况，因而得到的是正确的结果。平台提供的测试结果如下：



### 第5关 32位快速加法器设计

**一、实验目的**

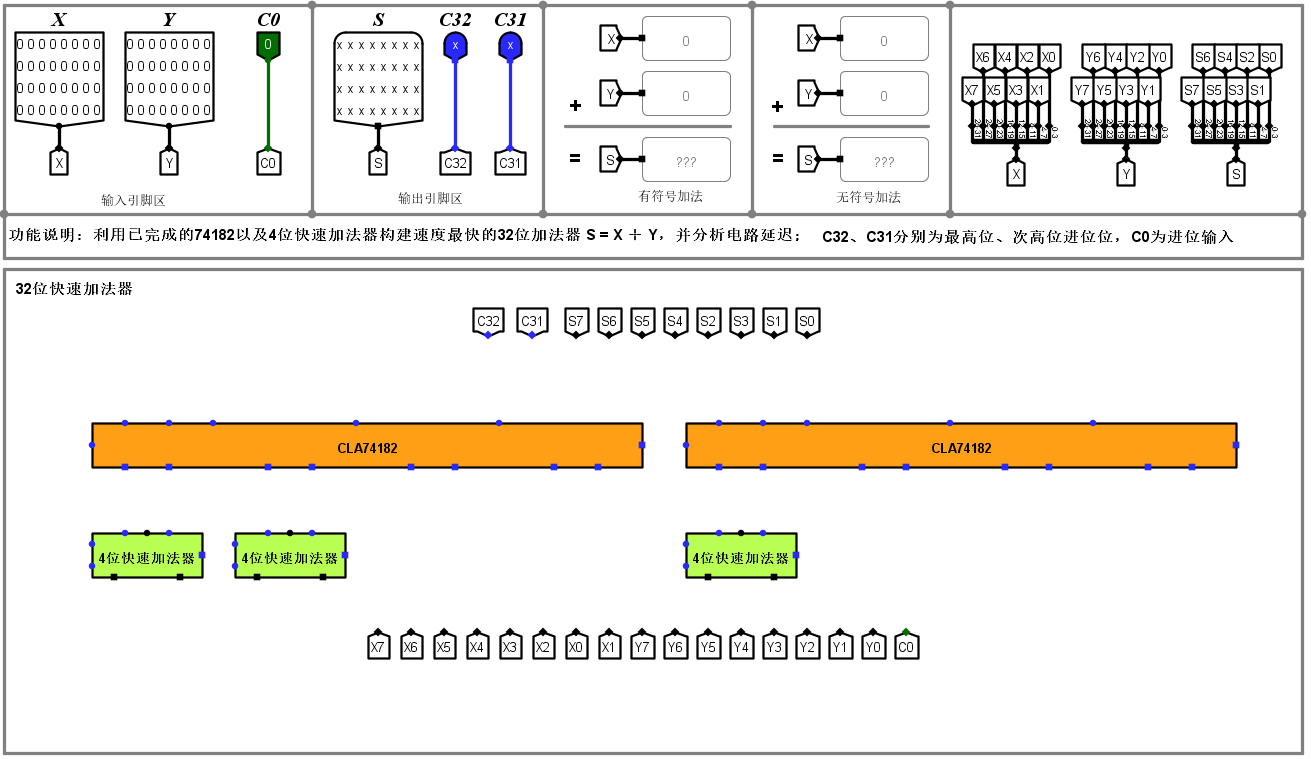
帮助学生理解成组进位产生函数，成组进位传递函数的概念，熟悉 Logisim 平台子电路的概念，能利用前述实验封装好的4位先行进位子电路以及4位快速加法器子电路构建32位快速加法器，并能利用相关知识分析对应电路的时间延迟，理解电路并行的概念。

二、实验环境

1. 实验所使用的工具软件：Logisim
2. 实验平台：[https://www.educoder.net/classrooms/17032/shixun\_homework](https://www.educoder.net/classrooms/13035/shixun_homework)

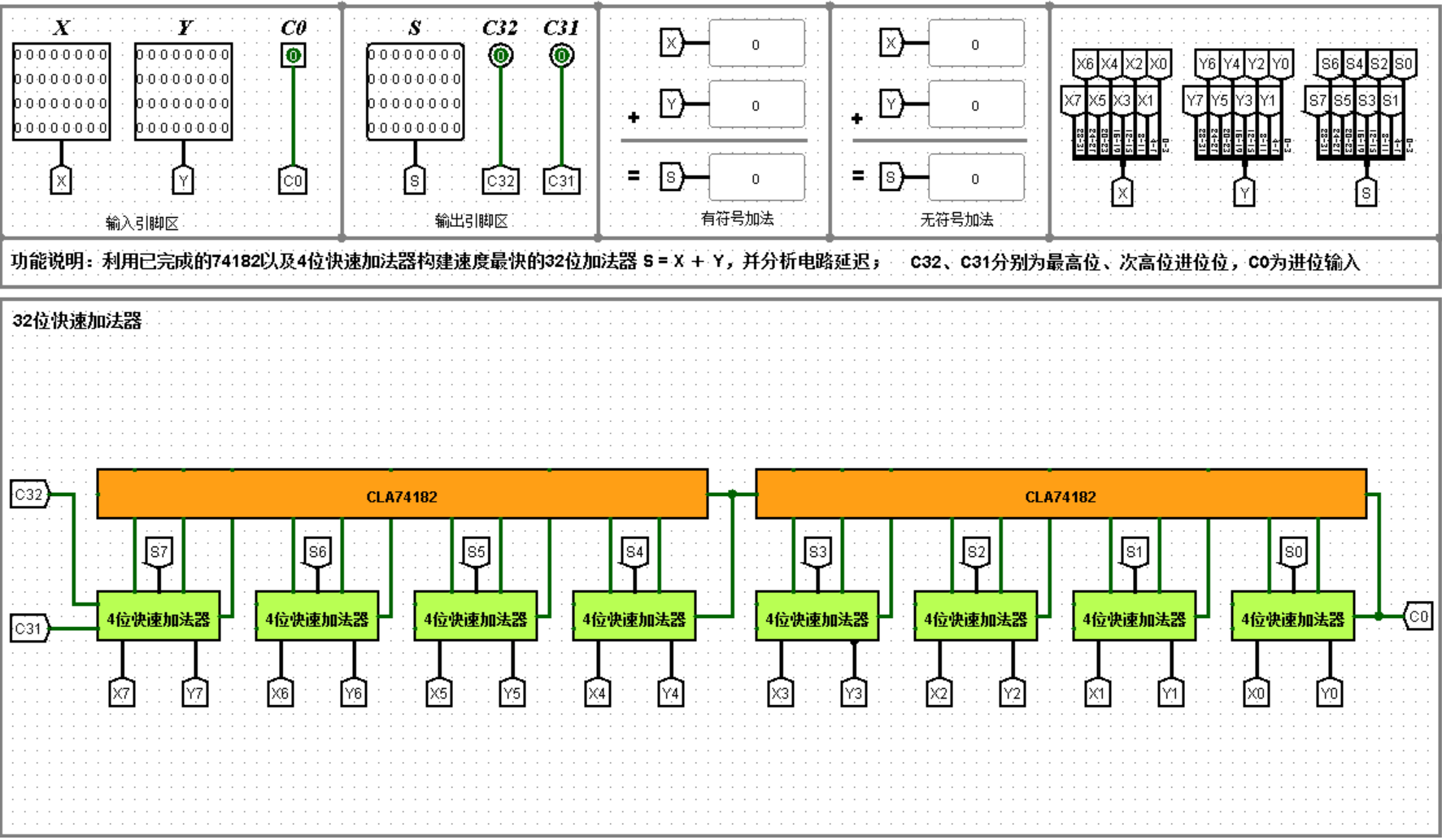
1. **实验内容**

利用16位快速加法器以及先行进位电路构建32位快速加法器，并探讨其时间延迟。 可能方案：（1）2个16位加法器直接串联，C16 信号采用下层的进位输出；（2））2个16位加法器直接串联，C16 进位输入采用上层的进位输出；（3）在16位快速加法器的基础上再增加一级组间先行进位电路，类似64位快速加法器的方法；分别分析3种不同方案可能的总延迟，选择速度最快的方案实现32位快速加法器，并分析其时间延迟，其引脚如图所示。其中 X，Y 为32位相加数，Cin 为进位输入，S 为和数输出，Cout 为进位输出，Overflow 为有符号加法运算溢出信号。电路框架和引脚如下所示：





**四、实验方案设计**



本次实验设计的原理是基于四位快速加法器设计的基础上进行了32快速加法器的设计，由于32位数字可以拆分成八个四位数字，对于原始的输入Xi与Yi，所以将X与Y分为八个部分每个部分进行四位快速加法计算，最低位的计算需要考虑将进位信号置入，所以右端接入了C0，输出的信号需要将其与前一位的进位进行加和最终输出门S。与此同时将次高位与最高位的进位情况通过隧道进行输出，最终得到结果

**五、实验步骤**

1. 将输入分为八个四位块：X3-X0、Y3-Y0、X7-X4、Y7-Y4、X11-X8、Y11-Y8、X15-X12、Y15-Y12、X19-X16、Y19-Y16、X23-X20、Y23-Y20、X27-X24、Y27-Y24、X31-X28、Y31-Y28。

2.将X3-X0和Y3-Y0连接到第一个四位快速加法器的X和Y输入。

3.同理按照上述的分块将剩余输入的四位连接起来连到四位加法器上。

4.最高4位快速加法器左下端接入次高位进位C31，左上端接入最高位进位C32.

5.将本实验用的logisim文件改为记事本下阅读，复制代码到平台上运行即可。

**六、故障与调试**

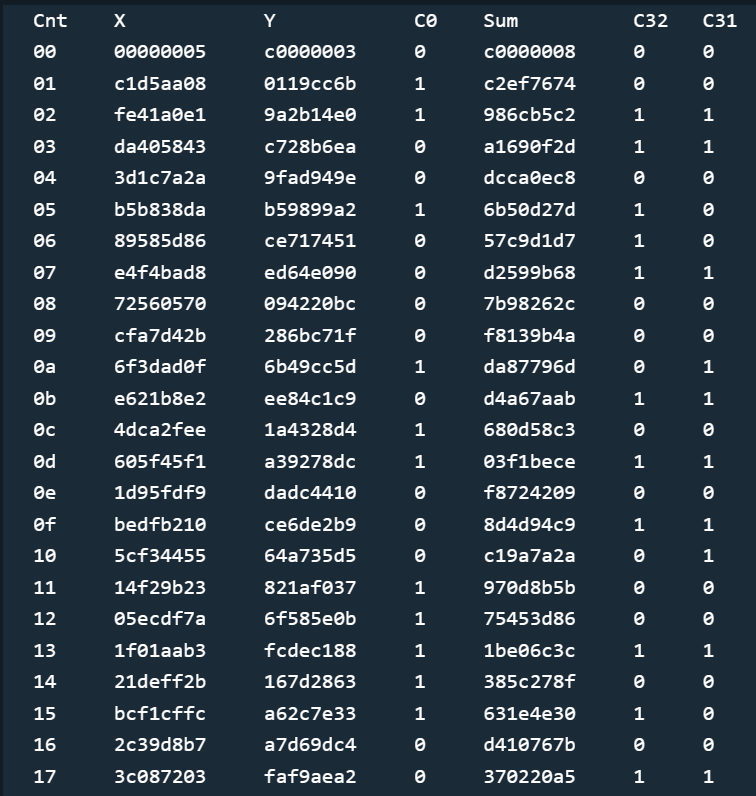
本次实验没有出现故障

1. **实验结果测试与分析**

说明：本次测试结果一共有8x16=144个测试案例，考虑到报告的简洁性，在这里仅仅展示一部分。

提交平台后以第一个例子为例：

在这个例子当中，输入的X为00000005（H），Y为c0000003（H），前位的进位为0，得到的最终结果为c0000008，而进位C32显示为0也就是在最高位置没有有进位，符合实际情况，因而得到的是正确的结果。平台提供的测试结果如下：



### 第6关 MIPS运算器设计

**一、实验目的**

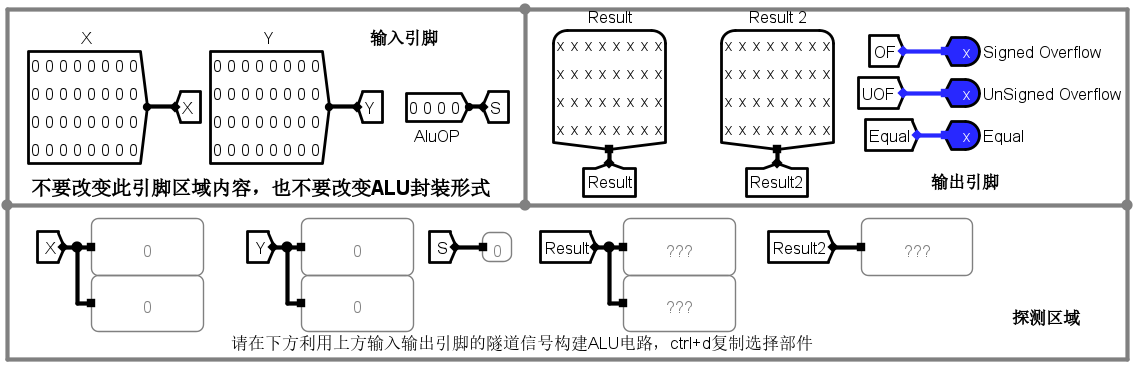
学生理解算术逻辑运算单元（ALU）的基本构成，掌握 Logisim 中各种运算组件的使用方法，熟悉多路选择器的使用，能利用前述实验完成的32位加法器、 Logisim 中的运算组件构造指定规格的 ALU 单元。

二、实验环境

1. 实验所使用的工具软件：Logisim
2. 实验平台：[https://www.educoder.net/classrooms/17032/shixun\_homework](https://www.educoder.net/classrooms/13035/shixun_homework)

**三、实验内容**

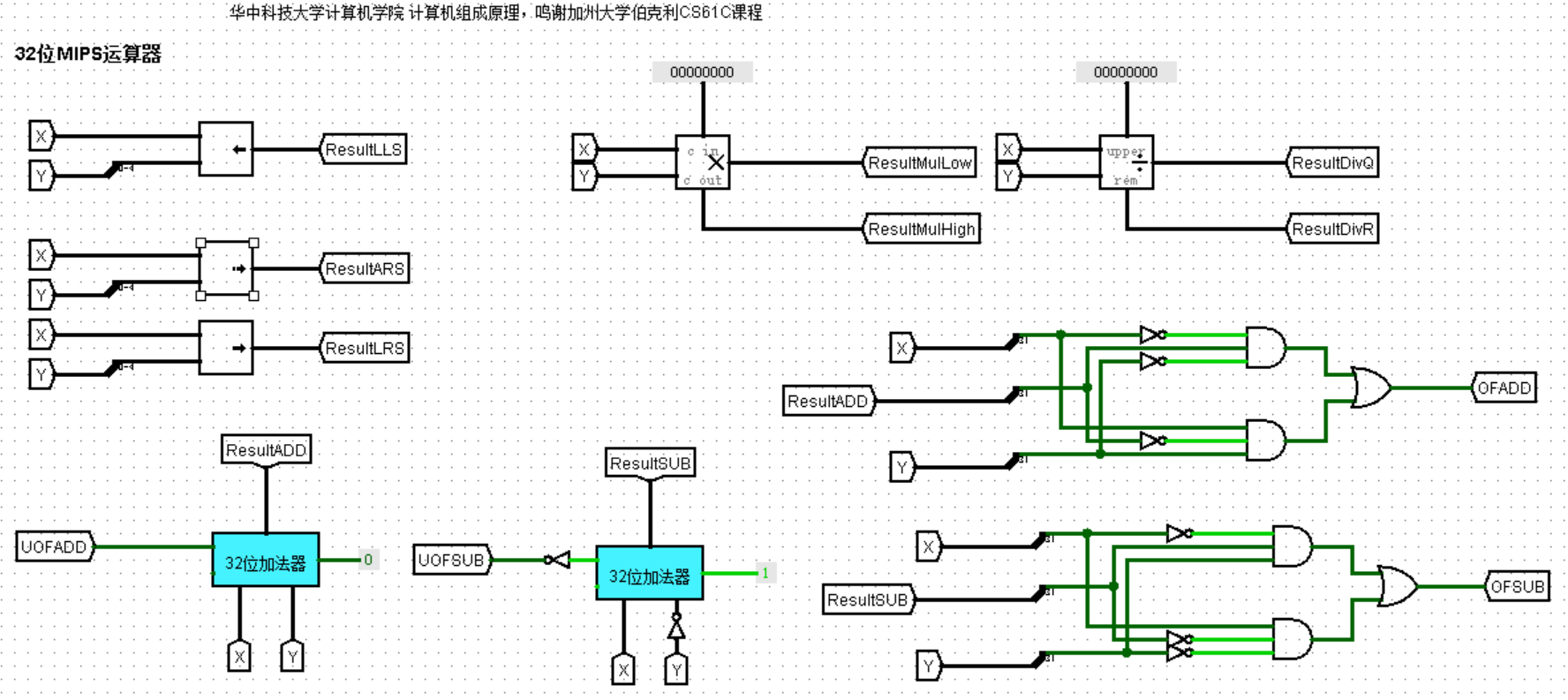
利用前面实验封装好的32位加法器以及 Logisim 平台中现有运算部件，构建一个32位算术逻辑运算单元（禁用 Logisim 系统自带的加法器，减法器），可支持算术加、减、乘、除，逻辑与、或、非、异或运算、逻辑左移、逻辑右移、算术右移运算，支持常用程序状态标志（有符号溢出 OF 、无符号溢出 UOF ，结果相等 Equal ），ALU 功能以及输入输出引脚见后表，在主电路中详细测试自己封装的 ALU ，并分析该运算器的优缺点。电路框架和引脚如下所示：

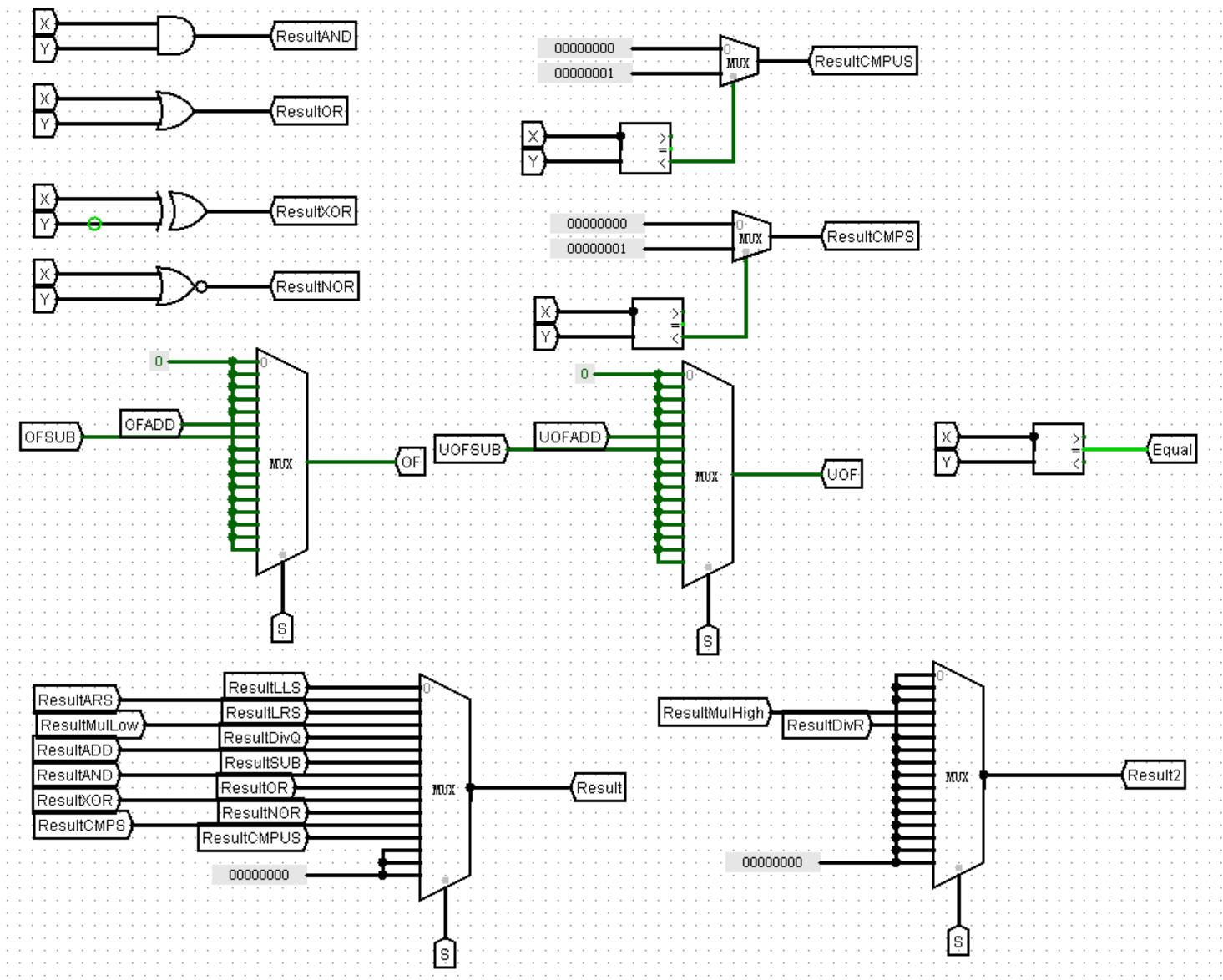






**四、实验方案设计**





按照题目要求的表格中各个OP信号代表的意义来进行设计：前三位为算数或逻辑的移动，然后是乘除法，然后是加减法，注意要使用上一个实验所设计出来的32位加法器来进行设计，然后是逻辑门的运算，只需要调用Logisim内部的相应的逻辑门即可，接下来是比较运算，也使用本来有的逻辑门实现，最后设计等于判断运算，OF，UOF，将上述所有功能的输出隧道按照顺序连接到Multiplexer中即可完成设计。

**五、实验步骤**

1. 根据题目要求的表格，确定所需的各个操作码（OP）信号以及它们的意义。根据这些意义，确定需要实现的功能模块，例如算数或逻辑移动、乘除法、加减法、逻辑门运算、比较运算和等于判断运算。

2. 根据上一个实验中设计的32位加法器，将其作为基本模块引入MIPS运算器设计中。确保正确连接输入和输出信号。

3. 对于算数或逻辑移动操作，根据具体需求设计相应的电路模块。这可能涉及位移寄存器、逻辑门和选择器等。根据指令的要求，连接输入和输出信号。

4. 对于乘除法操作，根据题目要求的表格，设计相应的电路模块。

5. 对于加减法操作，已经在上一个实验中设计了32位加法器，可以直接将其作为基本模块引入MIPS运算器设计中。根据题目要求的表格，确定输入和输出信号的连接方式。

6. 对于逻辑门运算，可以使用Logisim内部的逻辑门模块来实现。根据指令的要求，将相应的输入和输出信号连接到逻辑门模块。

7. 对于比较运算，使用逻辑门模块实现。根据指令的要求，将相应的输入和输出信号连接到逻辑门模块。

8. 对于等于判断运算，根据题目要求的表格，确定等于判断的条件和相应的逻辑操作。使用逻辑门模块实现，并连接输入和输出信号。

9. 将上述所有功能模块的输出信号按照顺序连接到多路复用器（Multiplexer）中。根据指令的操作码（OP）信号，选择正确的功能模块输出作为最终的输出。

10. 完成连接后，根据题目要求的表格，验证设计的MIPS运算器是否能正确执行各种指令，并生成预期的输出结果。

11.将本实验用的logisim文件改为记事本下阅读，复制代码到平台上运行即可。

**六、故障与调试**

1. 故障一
2. 故障现象

当OP码为0001时，计算结果出了问题

1. 原因分析

调用逻辑门的时候，题目要求的是算术逻辑右移，但误用了逻辑右移的门电路，导致计算的是逻辑右移与题目要求不符合。

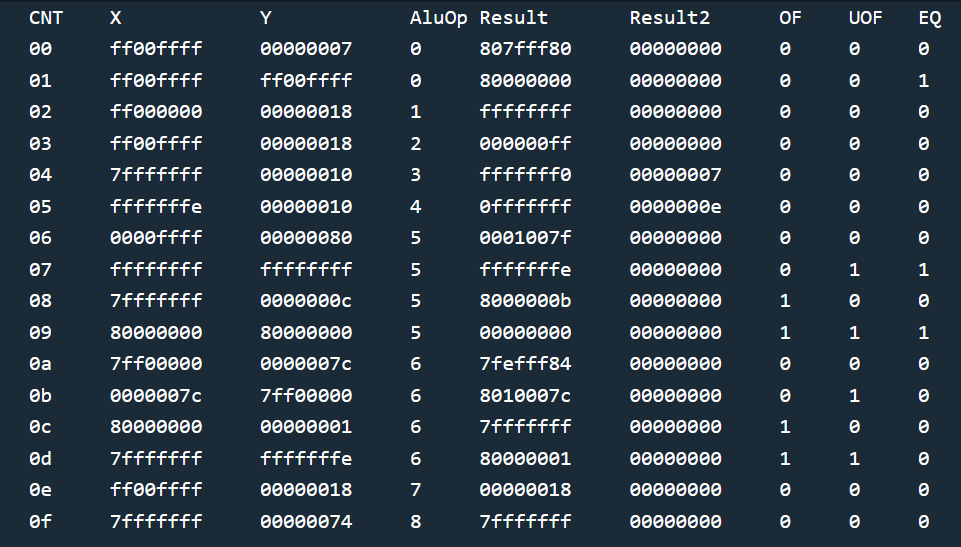
1. 解决方案

更换门电路为逻辑右移门电路。

1. **实验结果测试与分析**

提交平台后以第一个例子为例：

在这个例子当中，输入的X为ff00ffff（H），Y为00000007（H），OP码为0，也就是说进行的操作是将X进行逻辑左移左移的量为Y个单位，因为第一个例子中Y=7，所以左移7个单位，X=11111111000000001111111111111111（2），左移后结果为X=10000000011111111111111110000000（2），转化为16进制就是807fff80，符合实际情况，因而得到的是正确的结果。其余其他结果也均为正确。平台提供的测试结果如下：



### 实验四 存储器设计

**实验四成绩： 教师签名： 日期：**

### 第1关 MIPS寄存器文件设计

**一、实验目的**

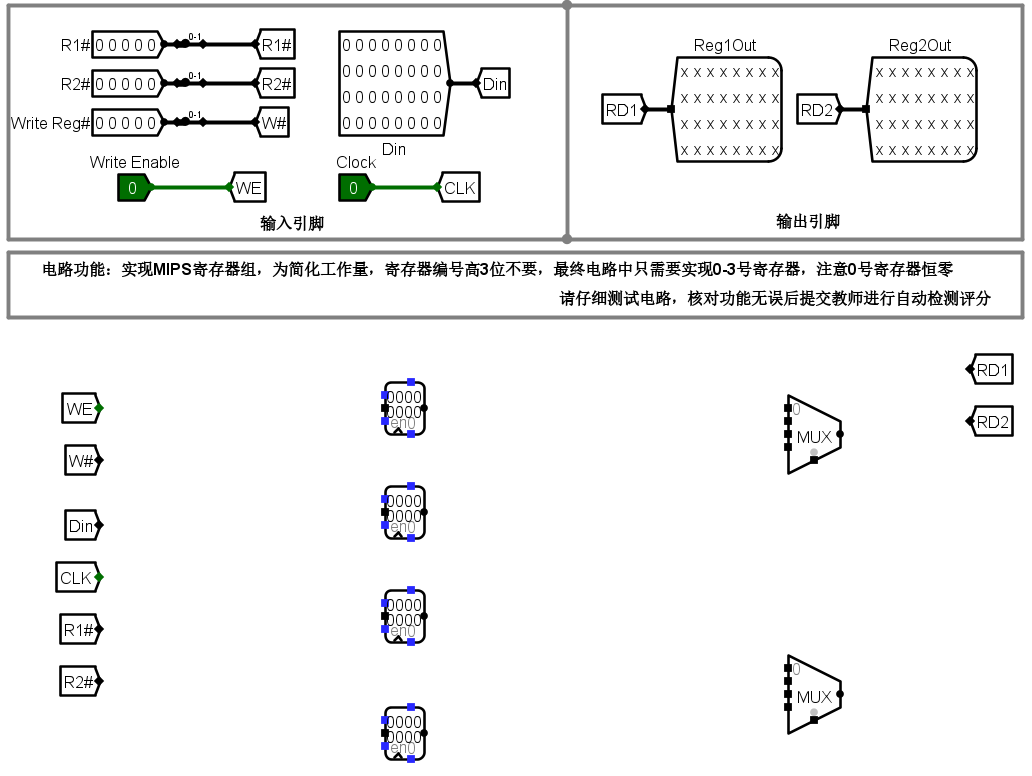
学生了解 MIPS 寄存器文件基本概念，进一步熟悉多路选择器、译码器、解复用器等 Logisim 组件的使用，并利用相关组件构建 MIPS 寄存器文件。

**二、实验环境**

1. 实验所使用的工具软件：Logisim
2. 实验平台：[https://www.educoder.net/classrooms/17032/shixun\_homework](https://www.educoder.net/classrooms/13035/shixun_homework)

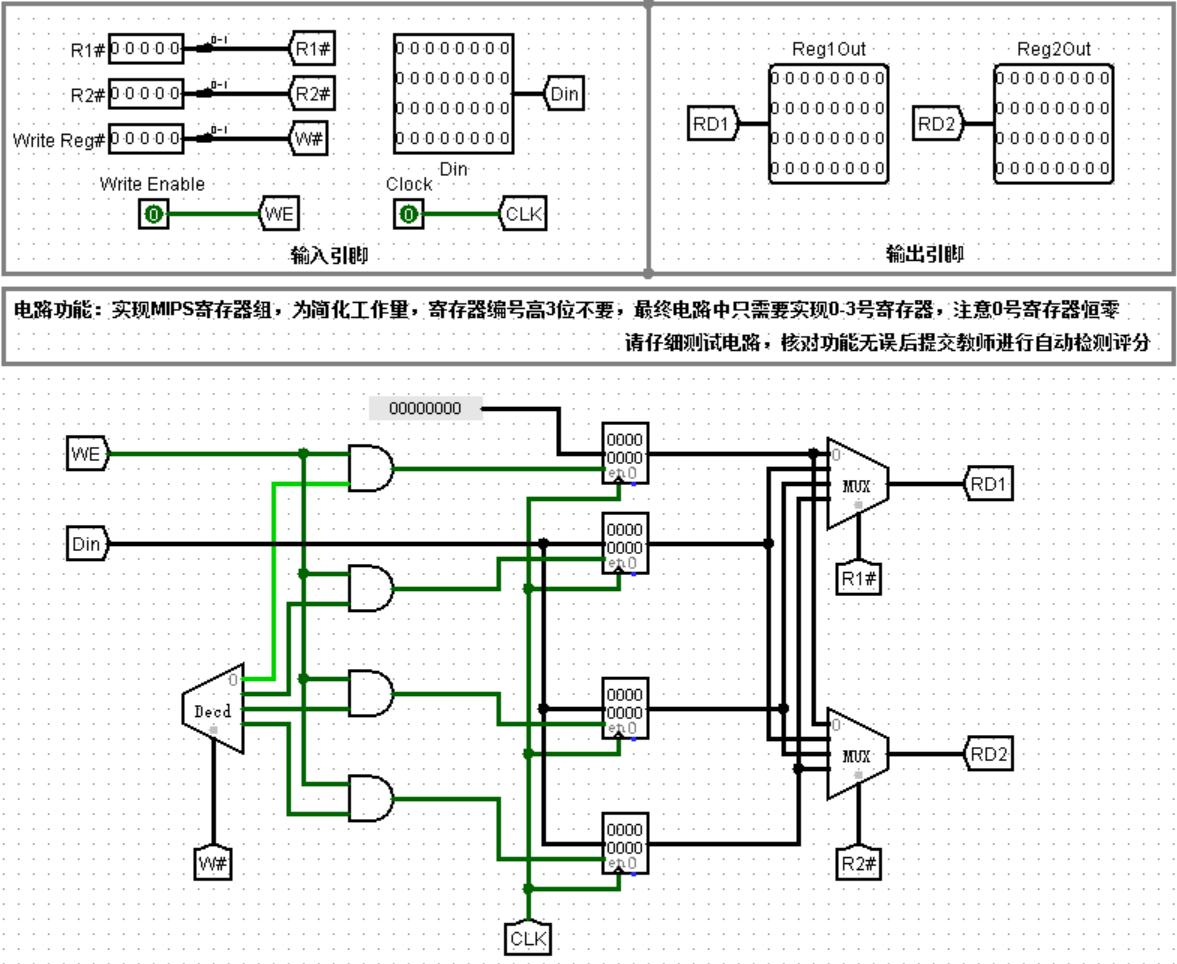
**三、实验内容**

利用 Logisim 平台构建一个简化的 MIPS 寄存器文件，内部包含4个32位寄存器，其具体引脚与功能描述如下表。





**四、实验方案设计**



本实验主要使用多路复用器（Multiplexer），将存储器的信号输入到其中最后进行输出，前位使用Decoder将写入的寄存器编号解码，W#写使能信号，为 1 时在 CLK 上跳沿将 Din 数据写入W#寄存器，所以使用与门与寄存器编号连接，最后将输出信号与写入数据信号Din一并写入寄存器中。同时还需要接入时钟确保数据准确写入。

五、实验步骤

1. 根据题目要求的需求，确定寄存器的位数和数量。根据位数确定寄存器的宽度，并根据数量确定所需的多路复用器（Multiplexer）和解码器（Decoder）的输入和输出端口数量。

2. 设计解码器（Decoder）电路模块，将寄存器编号进行解码，生成对应的选通信号。确保解码器的输出端口数量与寄存器的数量一致。

3. 设计多路复用器（Multiplexer）电路模块，用于选择要输出的寄存器内容。将解码器的输出信号作为多路复用器的控制信号，将各个寄存器的内容作为多路复用器的输入信号。

4. 连接写使能信号（W#）到与门（AND gate），并将与门的输出与寄存器编号连接。这样，在时钟（CLK）上升沿时，当写使能信号为1时，与门输出为1，允许将输入数据（Din）写入对应编号的寄存器。

5. 确保时钟信号（CLK）正确连接到所有寄存器，以保证数据在时钟上升沿时准确写入。

6. 将多路复用器的输入信号连接到寄存器的输出端口，作为寄存器的输出数据。

7. 根据题目要求的功能，对寄存器进行读取和写入操作的连接。将写入数据信号（Din）与多路复用器的输入信号连接，根据解码器的输出信号选择要写入的寄存器。

8. 确保所有信号连接正确，根据题目要求的时序和操作流程，验证设计的MIPS寄存器是否能够正确地读取和写入数据。

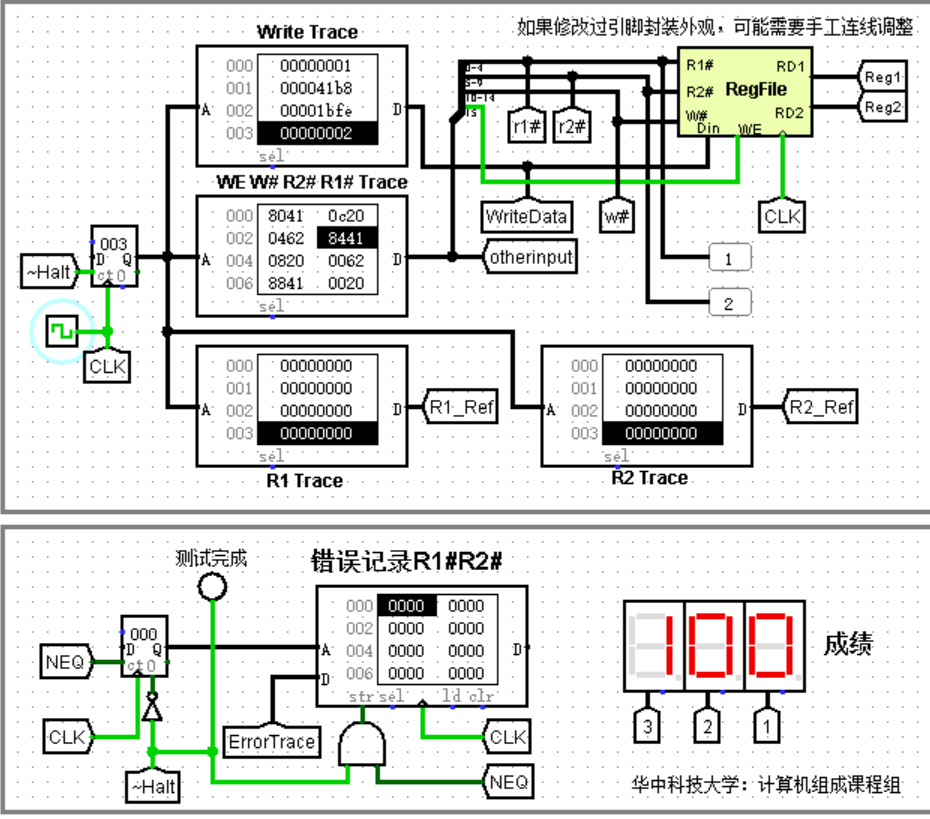
9.将本实验用的logisim文件改为记事本下阅读，复制代码到平台上运行即可。

**六、故障与调试**

本次实验没有出现故障

1. **实验结果测试与分析**

由头歌平台上提供的程序中已经有了测试程序电路，在测试电路中点击运行按钮得到寄存结果，可以看到各个寄存器的数据都正确地存入存储器当中，运行结果如下所示：



### 第2关 直接相联cache设计

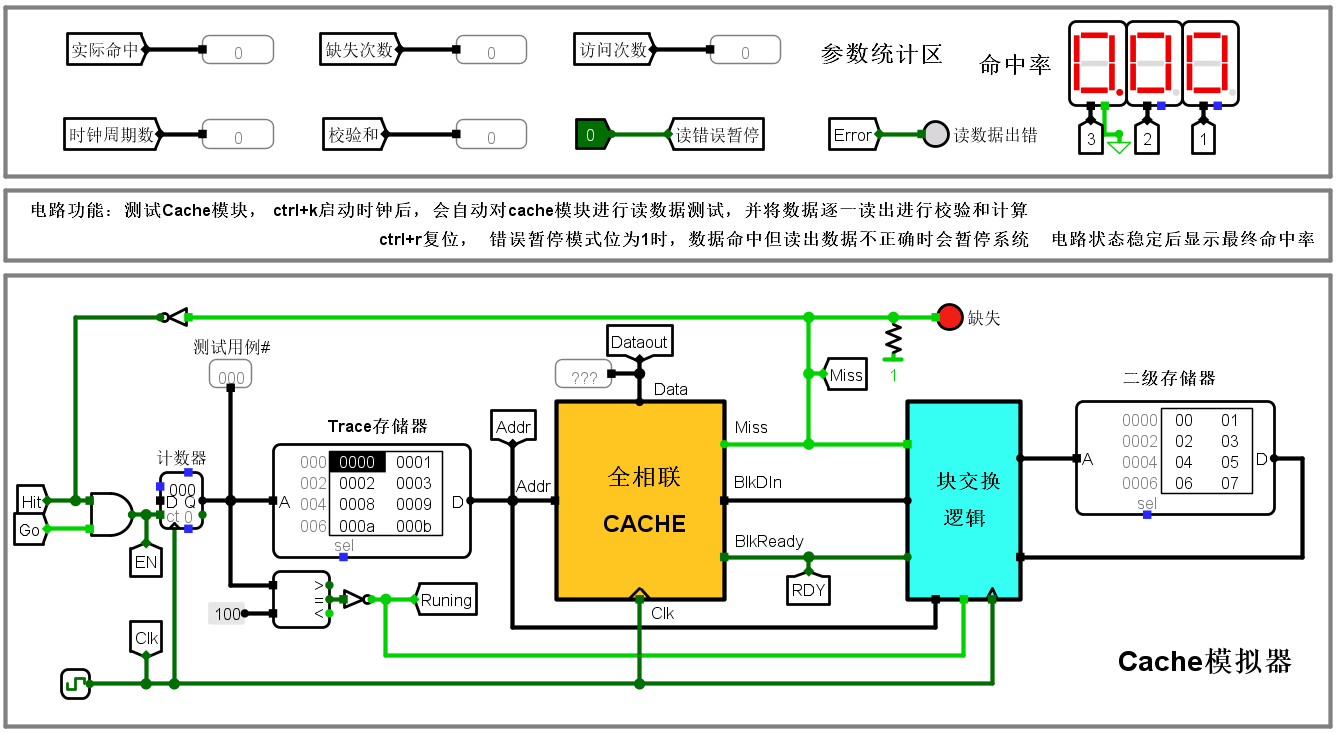
**一、实验目的**

学生掌握 cache 实现的三个关键技术：数据查找，地址映射，替换算法，熟悉译码器，多路选择器，寄存器的使用，能根据不同的映射策略在 Logisim 平台中用数字逻辑电路实现 cache 机制。

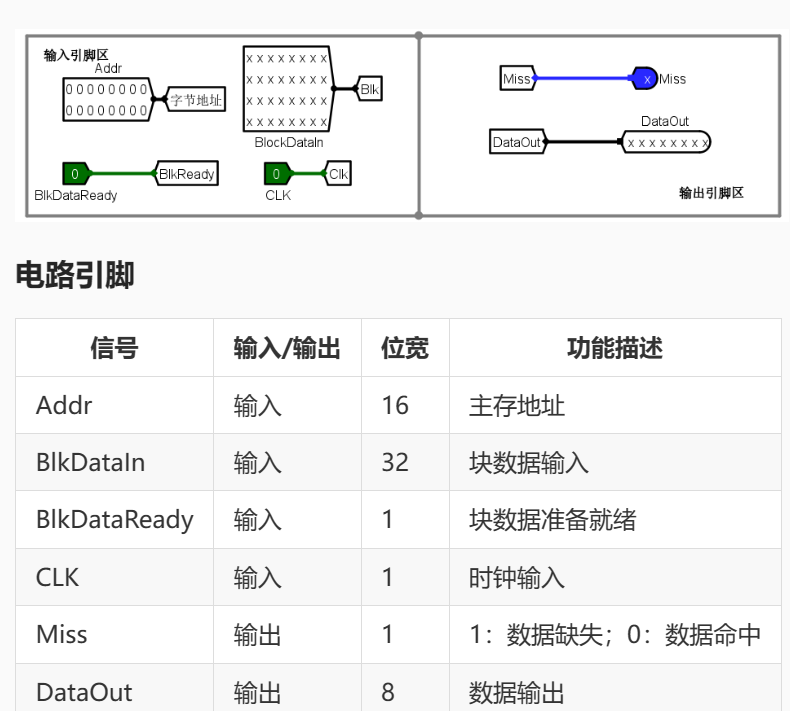
**二、实验环境**

1. 实验所使用的工具软件：Logisim
2. 实验平台：[https://www.educoder.net/classrooms/17032/shixun\_homework](https://www.educoder.net/classrooms/13035/shixun_homework)

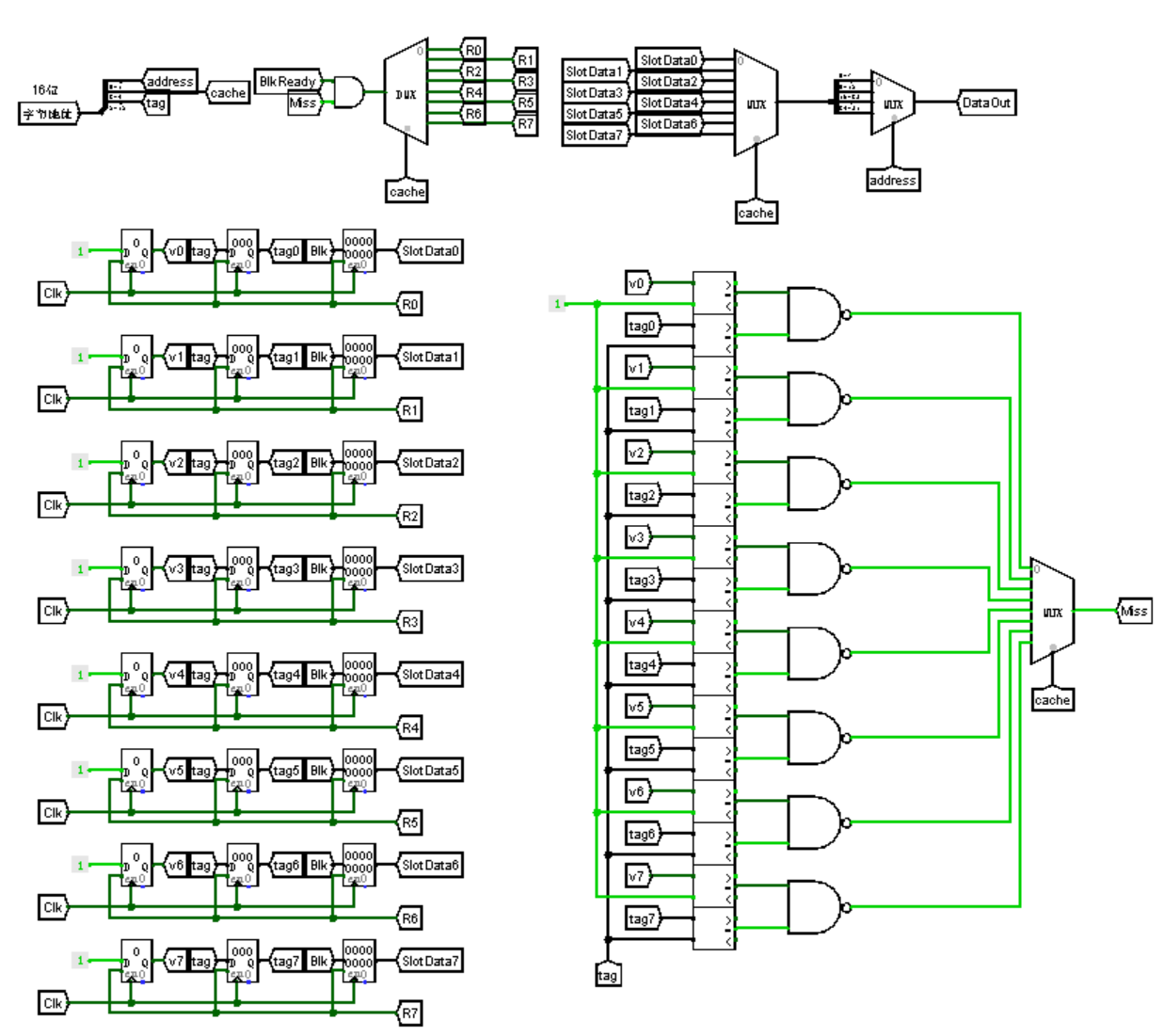
**三、实验内容**



上图给出了一个在 Logisim 中设计完成的 cache 系统自动测试电路，为简化实验设计，这里所有 cache 模块均为只读 cache（类似指令 cache），无写入机制。电路左侧计数器与存储器部分会在时钟驱动下逐一生成地址访问序列给 cache 模块。计数器模块的使能端受命中信号驱动，缺失时使能端无效，计数器不计数，等待系统将待请求数据所在块从二级存储器中调度到 cache 后才能继续计数。cache 与二级存储器之间通过块交换逻辑实现数据块交换，由于二级存储器相比 cache 慢很多，所以一次块交换需要多个时钟周期才能完成，cache 模块判断数据块准备好的逻辑是 blkready 信号有效，该信号有效且时钟到来时，cache 将块数据从 BlkDin 端口一次性载入到对应 cache 行缓冲区中，此时 cache 数据命中，直接输出请求数据，解锁计数器使能端，继续访问下一个地址。 自动测试电路会逐一取出 trace 存储器中的主存地址去访问存储系统，并逐一将数据从 cache 模块取出送校验和计算电路计算校验和，计数器值为256时会停止电路运行，此时所有存储访问的 cache 命中率将会在右上角 LED 数码管显示。本次实验的主要任务就是设计该电路的核心模块 cache 子电路。 结合引脚功能说明，实现直接相联 cache 模块，该 cache 模块共包括8个 cache 行，每个数据块包含包括4个字节共32位数据。



**四、实验方案设计**



对于每个cache行其设计电路是一致的：主要使用寄存器与时钟进行设计寄存器用于存储数据，而时钟用于控制和同步各个电路模块的操作，确保正确的数据传输和处理。而对于是否命中的逻辑设计使用在寄存器左右端各放置标志的方式再将标志与经过寄存器的tagi进行比较，最后将8个cache行的数据进行汇总输入多路复用器（Multiplexer），得到Miss隧道信号，再将Blkready与Miss结合判断是否为有效上升信号，最后将该信息存入寄存器当中，得到SlotDatai，最后汇总输入多路复用器（Multiplexer）得到最后结果。

**五、实验步骤**

1.为每个缓存行设计模块，包括有效位、标记和数据存储单元。使用寄存器或存储器组件来实现每个缓存行。

2. 将主存地址中的标记与缓存行中的标记进行比较。使用比较器电路实现标记比较逻辑。

3. 根据blkready信号判断数据块是否准备就绪。当blkready信号有效且时钟到来时，将数据从BlkDin端口一次性载入到对应缓存行的数据存储单元中。

4. 在每次访存时，根据命中与否更新命中次数和访问次数的计数器。使用计数器和逻辑门电路实现命中率计算逻辑。

5.根据直接相联缓存的连接方式，将缓存行模块、标记比较逻辑、缓存块交换逻辑和命中率计算逻辑进行连接。

6.将本实验用的logisim文件改为记事本下阅读，复制代码到平台上运行即可。

**六、故障与调试**

1. 故障一
2. 故障现象

Tag寄存器在运行时出现结果不正确情况

1. 原因分析

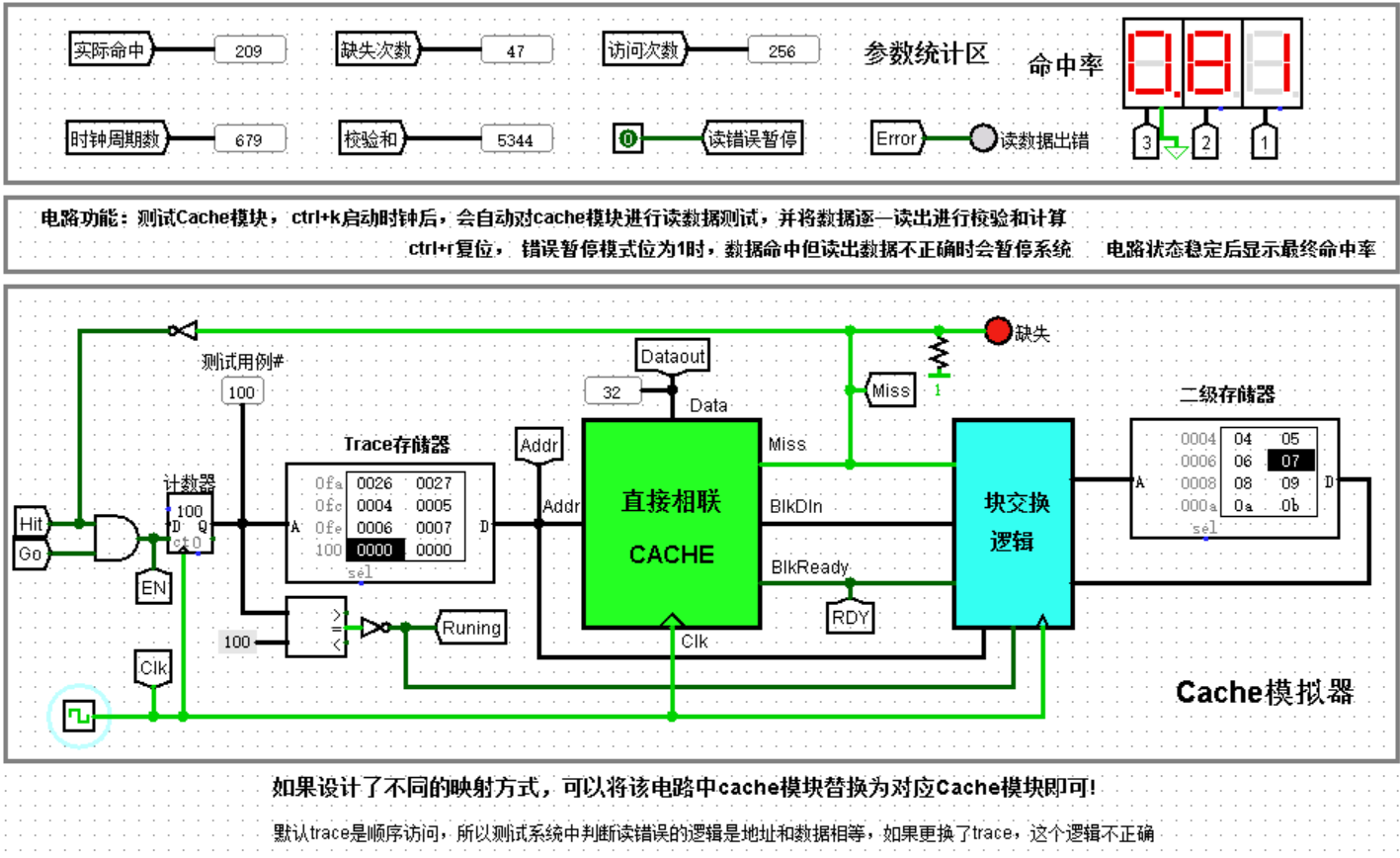
本来应当连接寄存器的元件错误使用了计数器元件，导致无法寄存。

1. 解决方案

更换为正确的电子元件：Register，即寄存器即可。

1. **实验结果测试与分析**

根据实验软件提供的测试程序，使用Ctrl+K来启动时钟，等待数据写入全部完毕，电路稳定之后，可以看到时钟周期数为679，实际命中为209，缺少次数为47，命中率为81%，符合头歌平台上的设计要求。



### 实验五 MIPS CPU设计

**实验五成绩： 教师签名： 日期：**

### 第1关 单周期MIPS CPU设计

**一、实验目的**

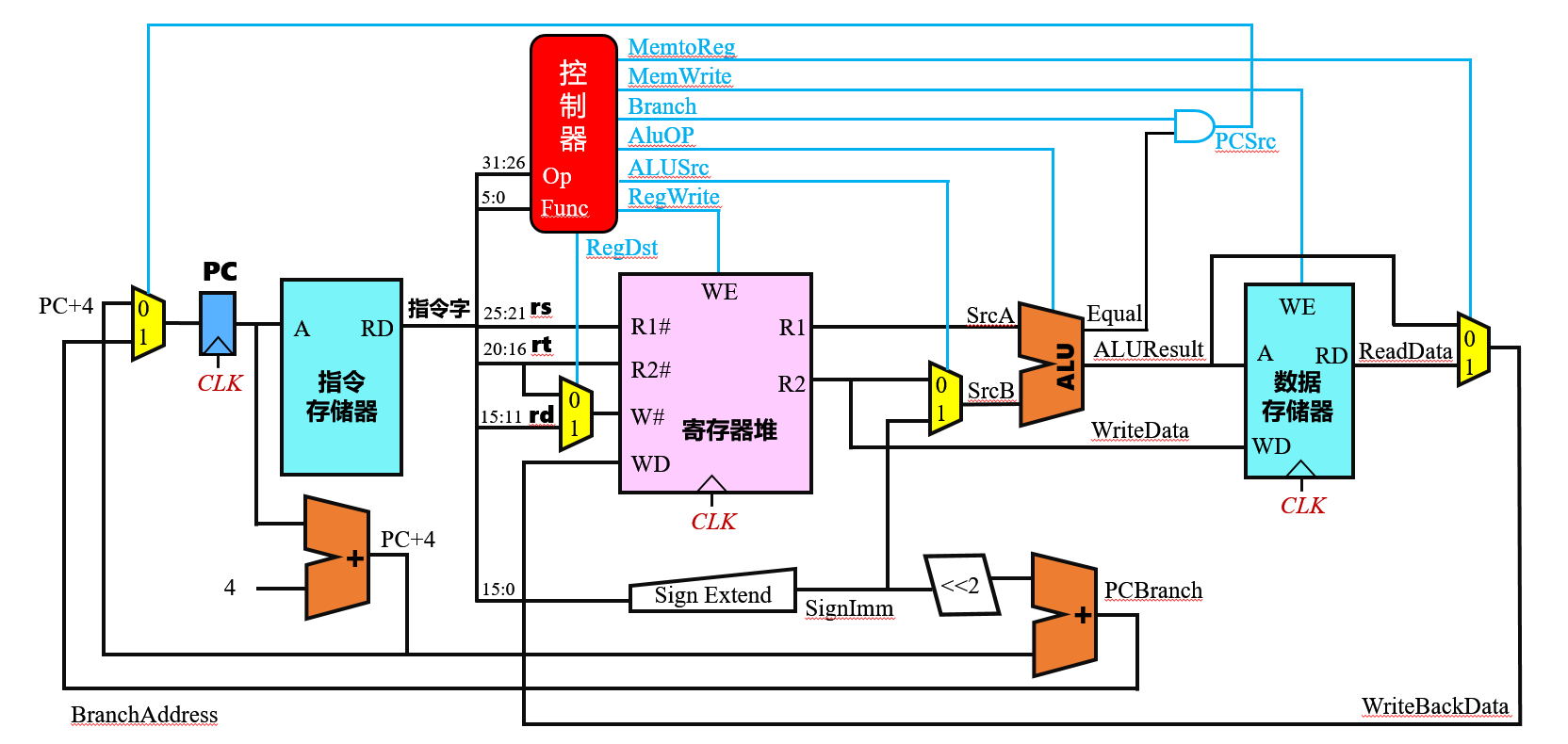
学生掌握控制器设计的基本原理，能利用硬布线控制器的设计原理，在 Logisim 平台中设计实现 MIPS 单周期 CPU。

**二、实验环境**

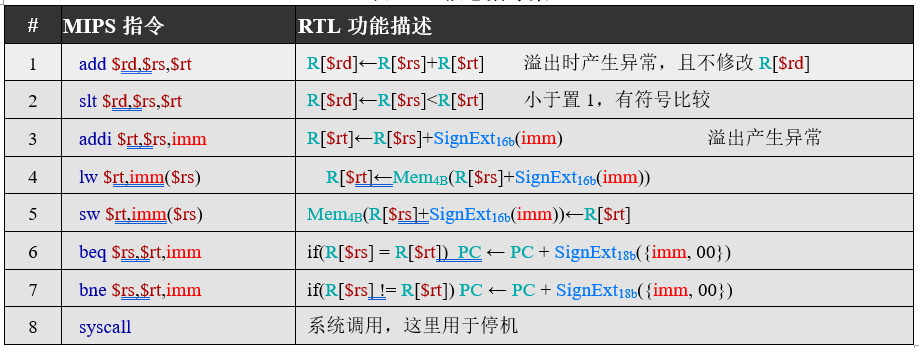
1. 实验所使用的工具软件：Logisim
2. 实验平台：[https://www.educoder.net/classrooms/17032/shixun\_homework](https://www.educoder.net/classrooms/13035/shixun_homework)

**三、实验内容**

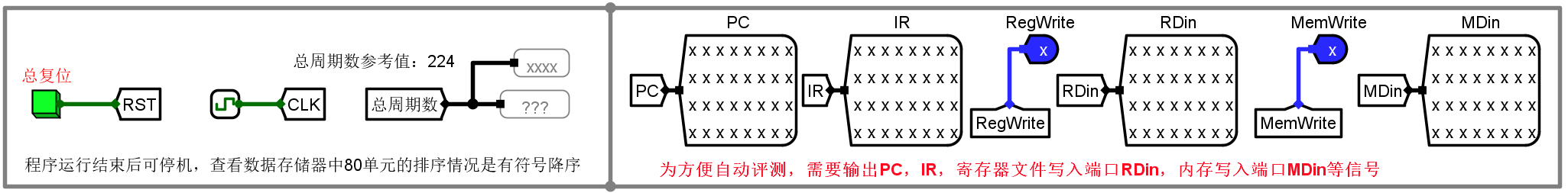
利用运算器实验，存储系统实验中构建的运算器、寄存器文件、存储系统等部件以及 Logisim 中其它功能部件，构建一个32位 MIPS CPU 单周期处理器。数据通路如下图所示：

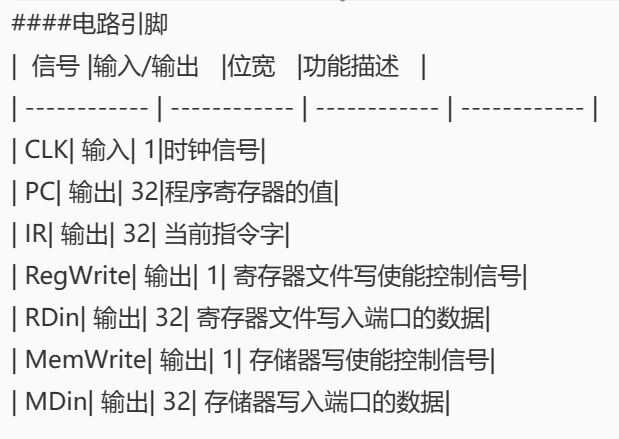


要求支持8条 MIPS 核心指令，最终设计实现的 MIPS 处理器能运行实验包中的冒泡排序测试程序 sort.asm，该程序自动在数据存储器0~15号字单元中写入16个数据，然后利用冒泡排序将数据升序排序，要求统计指令条数与 MARS 中的指令统计数目进行对比。

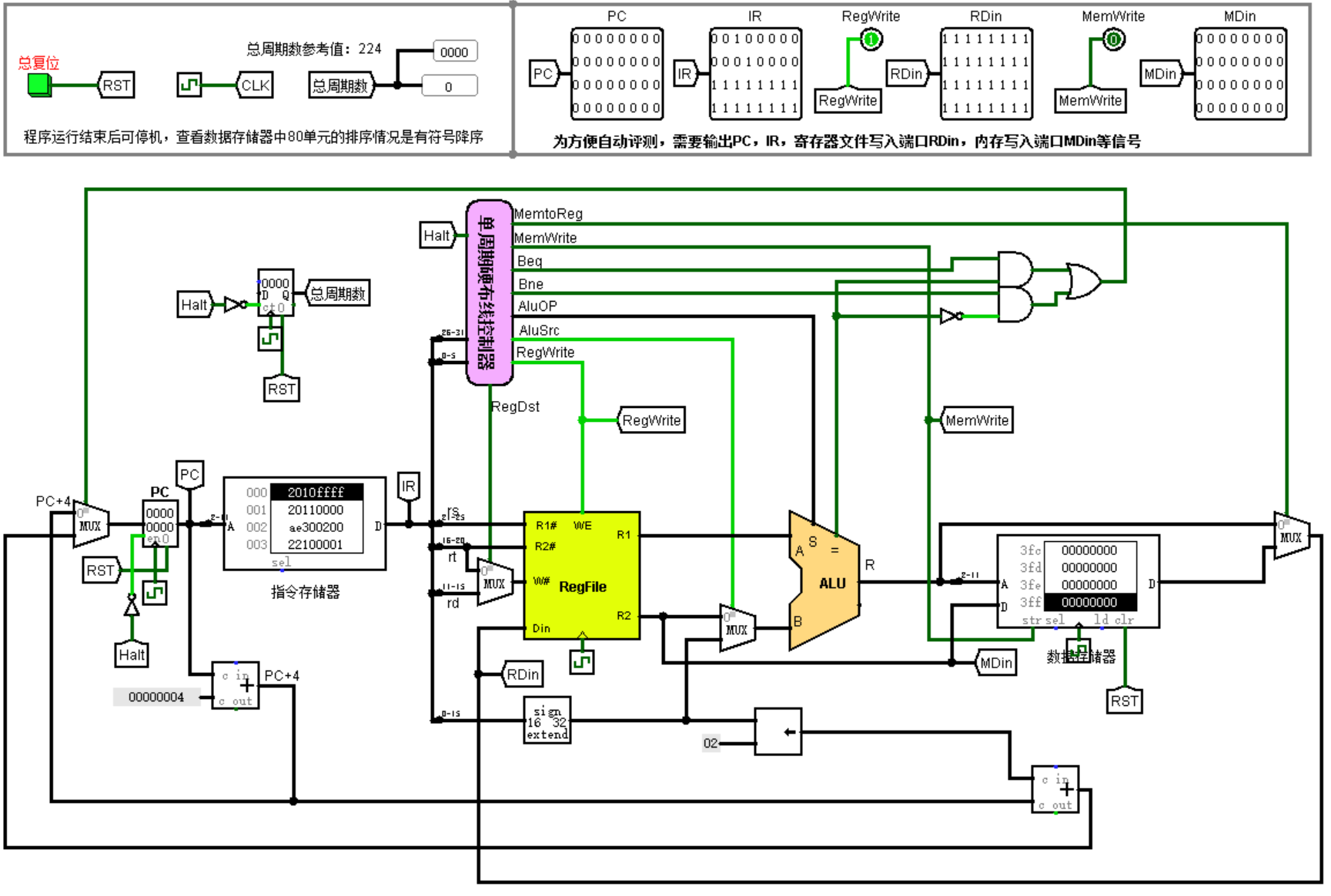


电路框架引脚如下所示：





**四、实验方案设计**



按照题目要求的数据通路图，在Logisim中找到相应的电子元件进行对应搭建即可完成：按照指令的处理顺序进行电路设计：先是PC端的输入，然后进入指令存储器，然后进入IR译码环节，通过引入单周期硬布线控制器根据功能描述将输出引脚对应接入不同部分，其中最主要为ALU为运算器。

**五、实验步骤**

1. 设计指令解码单元：根据 MIPS 指令格式，设计指令解码单元，将指令分解为操作码、寄存器编号等字段，并确定每个指令对应的操作。

2. 设计寄存器文件：根据 MIPS 的寄存器文件结构，设计寄存器文件模块，包括存储器组件和相关控制逻辑，用于存储和访问寄存器数据。

3. 设计运算器：利用之前设计的运算器实验构建算术逻辑单元(ALU)，支持 MIPS 指令集中的算术和逻辑运算操作，并考虑相关控制信号

4. 设计控制单元：根据 MIPS 的指令执行流程和控制信号，设计控制单元，包括时钟控制、指令跳转、分支跳转等功能。

5. 设计存储系统：将之前构建的存储系统与 CPU 进行连接，包括指令存储器和数据存储器，用于存储程序指令和数据。

6. 设计指令执行单元：结合指令解码单元、寄存器文件、运算器、控制单元和存储系统，设计指令执行单元，实现指令的执行和数据的传输。

7. 连接各个模块：将指令解码单元、寄存器文件、运算器、控制单元和存储系统按照 MIPS CPU 的连接方式进行连接，确保信号的正确传递和控制的协调。

**六、故障与调试**

1. 故障一
2. 故障现象

无法打开头歌平台上下载的原题程序

1. 原因分析

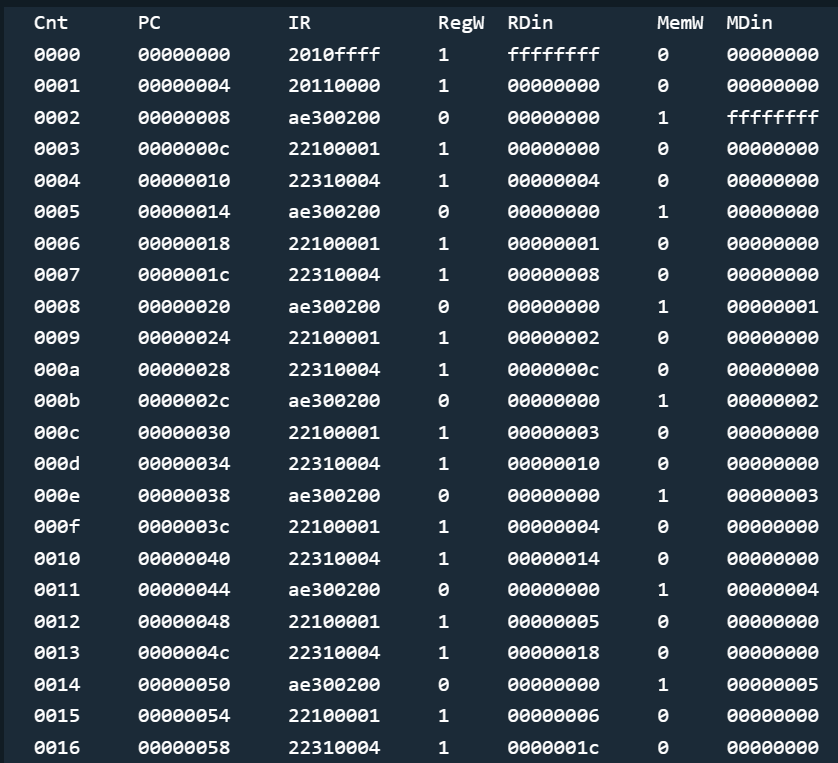
缺少文件cs3410.jar

1. 解决方案

在网络上搜索下载cs3410.jar，并将其放到与下载的circ文件的同一个目录下重新打开文件即可。

1. **实验结果测试与分析**

说明：本次测试结果一共有8x16=144个测试案例，考虑到报告的简洁性，在这里仅仅展示一部分：



经过平台运行，电路设计正确，以第一个分析为例：IR译码得到结果是2010ffff，得到输出分别为ffffffff与00000000，在电路中可见均为正确结果，表明所设计的CPU具有正确的译码、存储、执行等功能。具体结果如下所示：