

《 计算机组成原理 》

实验报告本

|  |  |
| --- | --- |
| 系 别： | **计算机科学与技术** |
| 班 级： | **计科211** |
| 学 号： | **19001531** |
| 姓 名： | **陈正江** |
| 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。

3、按指定数据完成几种指定的算术运算。

**二、实验装置**

DVCC系列实验计算机系统

**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1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。

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”；

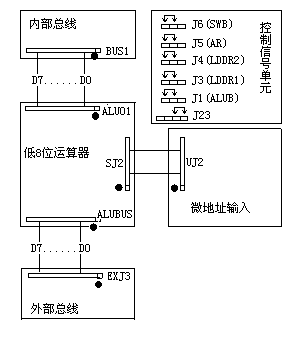
8、JA1,JA2,JA4置“高阻”；

9、JA8置上面“微地址”；

10、EXJ1接BUS3；

11、开关CE、AR置1。

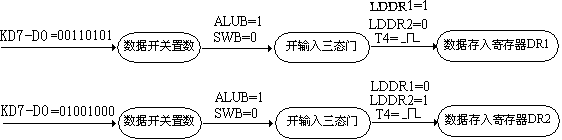
2.2）接线图



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：

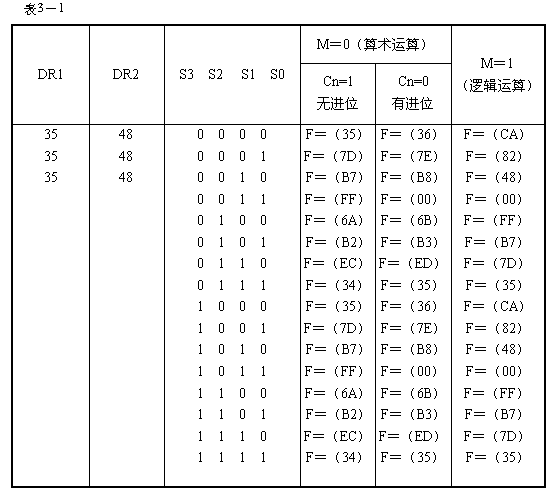


说明：LDDR1、LDDR2、ALUB`、SWB`四个信号电平由对应的开关LDDR1、LDDR2、ALUB、SWB给出，拨在上面为“1”，拨在下面为“0”，电平值由对应的显示灯显示，T4由手动脉冲开关给出。

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11111时，总线指示灯显示DR1中的数，而置成10101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，填入表中，并和理论分析进行比较、验证。



4）实验结果分析

1.实验结果分析：74LS181是算术逻辑运算发生器，S0~S3，M，CN是它的功能控制信号，按照实验步骤接线，关闭输出三态门ALUB’=1，开启输入三态门SWB=0，通过KD0~KD7输入数据，打开对应的锁存器控制信息，同时按下T4手动脉冲，将加数1送到DR1寄存器，加数2送入到DR2寄存器。拨动181的功能控制信号，发现实验结果与理论计算一致。CN=1时，即运算结果，CN=0时，运算器会给原本的运算结果自动加低位进位1。

2.实验总结：在实验时，实验的接线是需要更改的，不然会导致实验结果错误。验证锁存器DR中的数据是，需要使输入三态门关闭，输出三态门打开。还可以通过功能表中的算术运算验证数据。

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”；

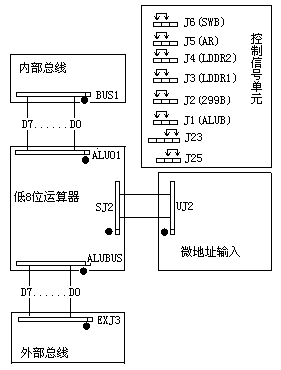
7、JA3置“接通”；

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00000，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10010，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10010，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

(1)实验分析：和上一个实验的接线稍微不同，置数操作相同。最后运算结果是否产生进位，需要通过按下T4脉冲，观察进位标志灯CY来判断。第一个表是在当前进位标志为0即CY=0基础上进行的，拨动181的S0~S3，M=0,CN=0，实现的操作是运算结果自动加1，同时按下手动脉冲，观察当前是否产生进位。若产生进位，需要进行进位标志清零保证不影响下一个计算结果，因为此表需要每一个算术运算的CY=0。第二个表是在CY=1的基础上进行的，每次运算都需要先将CY调为1后，才能拨动181的S0~S3，M=0进行后续计算，实验的操作是将结果加上当前进位标志CY。经实验，结果与理论计算一致并且两个表的实验结果相同。

(2)实验总结：产生进位时，需要按动手动脉冲开关，若此时CY灯亮，则有进位。如果不希望产生的进位对下一步操作造成影响就需要对进位标志清零。

**四、分析思考题**

1.简述8位算术逻辑运算实验的数据通路。

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。

2.简述74LS181的组合功能。

该芯片总共由22个引脚，其中包括8个数据输入端（~A0、~A1、~A2、~A3，~B0、~B1、~B2、~B3，其中八个输入端中A3和B3是高位），这八个都是低电平有效。还包括S0、S1、S2、S3这四个控制端，这四个控制端主要控制两个四位输入数据的运算，例如加、减、与、或。CN端处理进入芯片前进位值，M控制芯片的运算方式，包括算术运算和逻辑运算。F0、F1、F2、F3是四个二进制输出端，以一个四位二进制形式输出运算的结果。CN4记录运算后的进位。

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。

带进位控制8位算术逻辑运算与8位算术逻辑运算接线不同，置数操作相同。前者是CE、299B置“1”，AR置“0”，后者是开关CE 、AR 置1。

而最后的运算结果是否产生进位，需要通过按下T4脉冲，观察进位标志灯CY来判断。CY=0时，拨动芯片的S0~S3，M=0,CN=0，实现的操作是运算结果自动加1，同时按下手动脉冲，观察当前是否产生进位。若产生进位，需要进行进位标志清零保证不影响下一个计算结果。CY=1时，每次运算都需要先将CY调为1后，才能拨动181的S0~S3，M=0进行后续计算，实验的操作是将结果加上当前进位标志CY。

产生进位时，需要按动手动脉冲开关，若此时CY灯亮，则有进位。如果不希望产生的进位对下一步操作造成影响就需要对进位标志清零。

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。

Cy=0 进位灯灭 Cy=1 进位灯亮

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DR1** | **DR2** | **S3 S2 S1 S0** | **M=0,CN=0(带进位算术运算)运算结果** | **运算后进位状态Cy** | **理论计算结果** |
| 8CH | 9FH | 0  0  0  0 | 8D | 0 | 8D |
| 0  0  0  1 | A0 | 0 | A0 |
| 0  1  1  0 | ED | 0 | ED |
| 1  0  0  1 | 2C | 1 | 2C |
| 1  1  0  0 | 19 | 1 | 19 |
| 1  1  0  1 | 2C | 1 | 2C |

### 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系列实验计算机系统

**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3—6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

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；

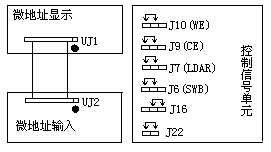
5、JA1，JA2，JA3，JA4置“高阻”；

6、JA5置“接通”；

7、JA8置“微程序”

8、JA6 置“微控”

2.2）接线图



3）实验步骤

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E2PROM2816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E2PROM2816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

(1)实验分析：手动操作时，系统按照程序编写方式，逐一执行代码:提取机器指令、放入指令寄存器、测试、执行所需操作…执行上述操作的同时，也需要手动的操作IN单元、开关单元、WR、RD等，从而完成一系列的指令操作。最后程序正确的执行，并得出最后正确结果。联机操作时，通过写入需要执行的程序、微程序命令，在CMA软件中进行仿真实验，在实验模型中，可以看到执行每一条指令所进行的一系列操作，同时观察出数据的流向、各个门的开闭。运行到最后，得出正确结果。

(2)实验总结：无论是手动操作还是联机操作，都可以正确的执行写入的程序指令，得出正确的结果。通过两次不同实验的对比，我们可以发现，其实微程序起的作用就是将所有指令需要执行的操作编写成一条条的微操作，形成一个微程序库。当执行不同的指令时，同时调出对应的微操作，完成指令操作。

**四、分析思考题**

1.时序信号在这个实验中的作用。

时序信号是用来确定时段执行哪些微操作，它规定这个微操作在什么时候发出去。实验中是由系统时钟引入的时序节拍脉冲的控制，才能逐条读出微代码。

2.微程序是如何写入微程序控制器的？

将控制各部件动作的微命令的集合进行编码，即将微命令的集合仿照机器指令一样，用数字代码的形式表示，用一个微指令序列表示一条机器指令，存储在一种专用的存储器中，称为控制存储器。

实验中的具体操作为：输入的微地址通过锁存器74LS374（U13），在控制信号UA374（1）和控制脉冲T1的控制下，锁存输出到控存的地址总线A0～A5，微地址值通过发光二极管LUA0～LUA5显示，高电平亮，低电平灭。再通过MK1～MK24手工输入24位微程序代码，输入微程序代码值通过3片总线驱动器74LS245（U20～U22），在控制信号K245（19）的控制下，输出到控存的数据总线，微程序代码值通过LMD1～LMD24显示，高电平亮，低电平灭。

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>

**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，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**

8位可控加减法电路的设计方案如下：

（1）输入：两输入数X，Y；加减控制信号Sub

（2）输出：运算结果输出S，进位输出Cout，有符号运算溢出位OF

（3）所需器件：全加器、异或门等

（4）电路引脚：



（5）可控加减法的逻辑设计

·加法部分：利用8个1位全加器串行实现；

·减法部分：根据“减去一个数等于加上这个数的相反数的补码”的原理，通过异或门对减数Y进行按位取反；

·具体过程：

Sub=0时，作加法操作：0和其他数异或得到该数本身，然后通过全加器FA执行加法运算。Sub=1时，作减法操作：1和其他数异或即对该数取反，然后通过全加器FA执行减法运算。注意，需要对加法器FA进行+1操作。

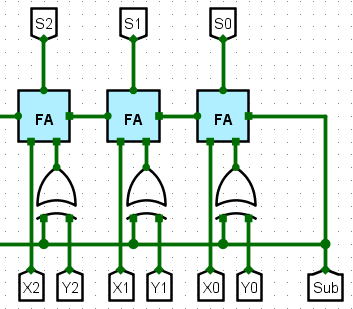
还要进行运算的有符号溢出检测（OF），可以通过最高位进位与符号位进位进行异或，相同为0，表示无溢出；不同为1，表示溢出。

**五、实验步骤**

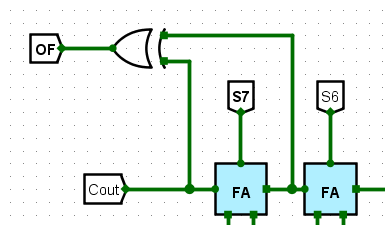
（1）在Logisim中打开alu.circ文件，选择8位可控加减法器子电路；

（2）准备8组封装好的全加器FA和异或门，对X、Y的各位数字进行分类排序，从左到右依次为7~0位；

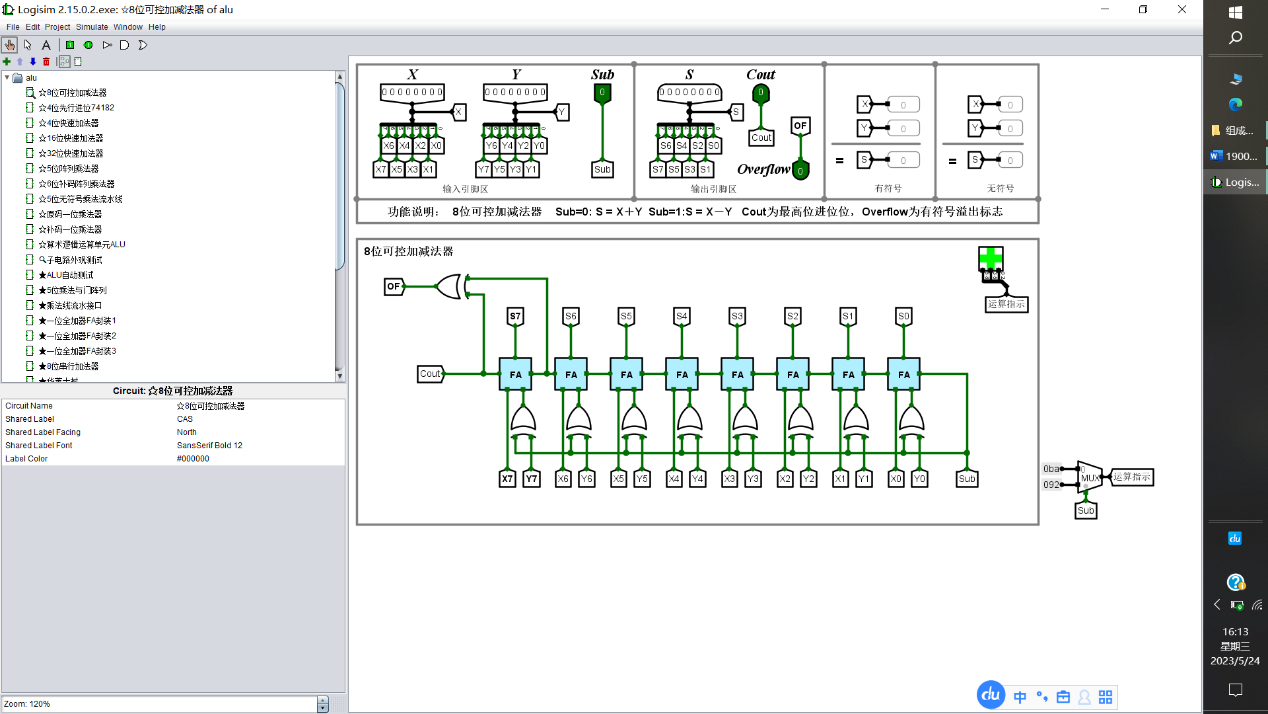
（3）将所有低位运算结果引脚连接全加器的输出端，X的位连接到FA的第一个输入端，Y的位与Sub加减控制信号进行异或以后接入FA的第二个输入端，同时也将Sub信号与全加器低位进位输入端相联，高位进位输出端与高一位的全加器低位进位输入端相联；



（4）将OF与最高位符号位进位和次高位进位异或得到的结果相连；



（5）最终得到的8位可控加减法器的电路图如下：



（6）进行电路测试：

利用给出的测试电路，自定义X、Y、Sub值，进行8位可控加减法器的自动测试。将alu.circ文件所有文字信息复制粘贴到Educoder平台代码区域，点击评测按钮进行测试。

**六、故障与调试**

故障一

1. 故障现象

按照草图进行连接之后，出现了红线，并且无法正常运行，存在错误。

1. 原因分析

出现红线，经过搜索是由于存在布线错误，产生了冲突，需要再对连线进行检查。

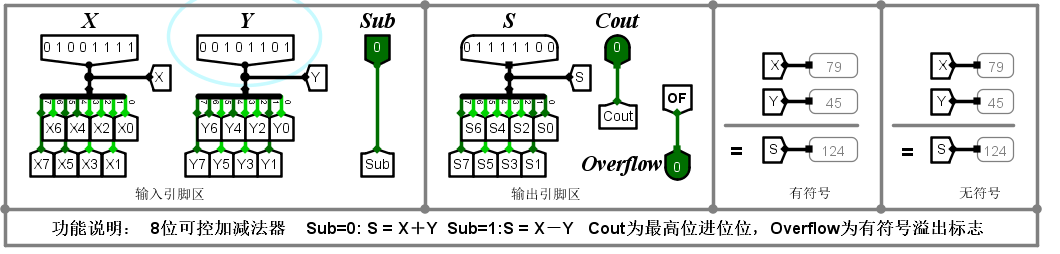
1. 解决方案

通过检查，发现问题出现在右边第二个逻辑门的连线上，有多余线的两点会出现加粗的连接点。

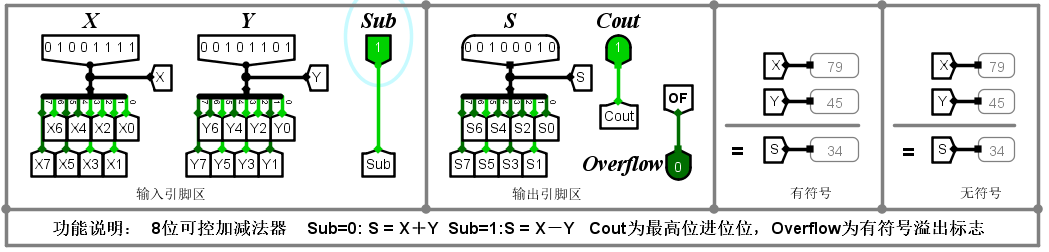
**七、实验结果测试与分析**

自定义X = 01001111，Y = 00101101；

Sub=0时，加法：



Sub=1时，减法：



平台测试结果：



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](https://so.csdn.net/so/search?q=%E7%BA%A7%E8%81%94&spm=1001.2101.3001.7020" \t "_blank)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。

**四、实验方案设计**

CLA182四位先行进位电路的设计方案如下：

（1）输入：进位输入Cin，进位生成函数Gi，传递函数Pi

（2）输出：进位输出Ci，成组进位生成函数G，成组进位传递函数P

（3）所需器件：基本逻辑门电路等

（4）电路引脚：



（5）先行进位加法器的设计逻辑

4位先行进位加法器表示进位输出，并不依赖于低位的进位，而是与最低位的进位输入有关；根据先行进位加法器的原理，可以设计出以下逻辑表达式：

C4 = P4 P3 P2 P1 Cin + P4 P3 P2 G1 + P4 P3 G2 + P4 G3 + G4

C3 = P3 P2 P1 Cin + P3 P2 G1 + P3 G2 + G3

C2 = P2 P1 Cin + P2 G1 + G2

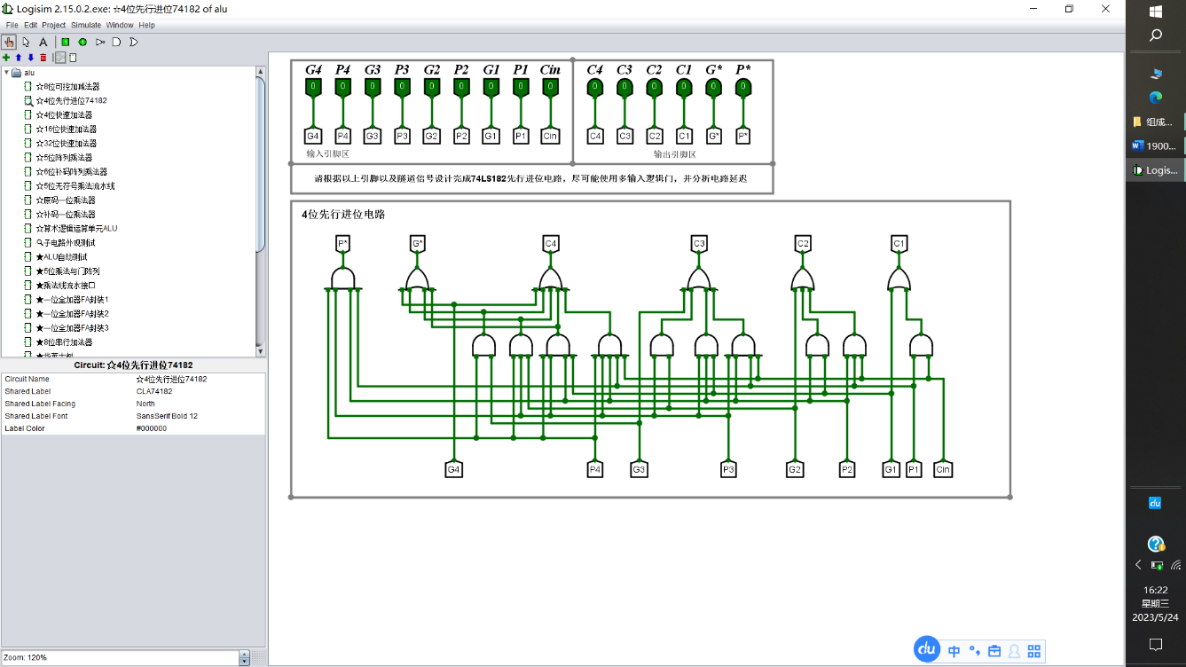
C1 = P1 Cin + G1

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

P = P4 P3 P2 P1

**五、实验步骤**

（1）在Logisim中打开alu.circ文件，选择4位先行进位74182子电路；

（2）根据C1、C2、C3、C4进位项的表达式（只与P\*、G\*、Cin相关），将与门、或门及其他各项引脚连接；

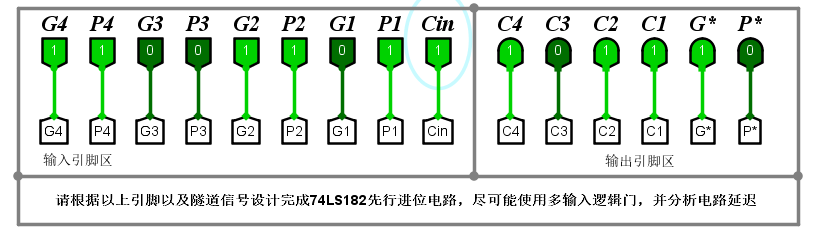
（3）进行电路测试

利用给出的测试电路，自定义输入值，进行4位先行进位的自动测试。将alu.circ文件所有文字信息复制粘贴到Educoder平台代码区域，点击评测按钮进行测试。

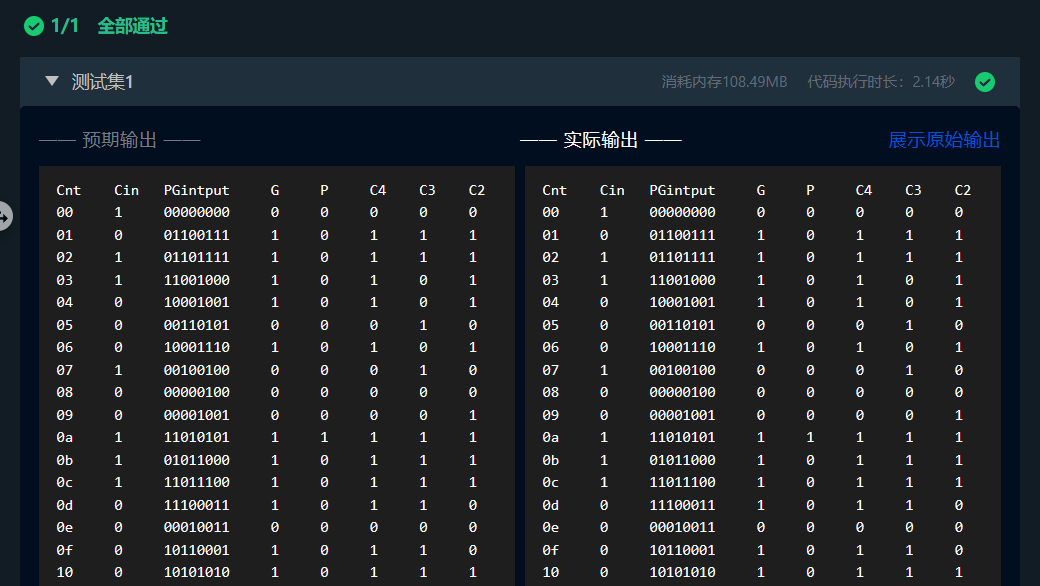
**六、故障与调试**

无故障与报错。

**七、实验结果测试与分析**

自定义Pi、Gi的值进行测试，观察进位信号Ci的输出；

平台测试结果：



### 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。

**四、实验方案设计**

在前一个子实验已经设计好的4位先行进位电路的基础之上，构造4位快速加法器。4位快速加法器的设计方案如下：

（1）输入：四位相加数X、Y，进位输入C0

（2）输出：和数输出S，进位输出C，4位成组进位生成函数G，4位成组进位传递函数P

（3）所需器件：CLA74182先行进位器，基本逻辑门电路等

（4）电路引脚：



（5）快速加法器的设计逻辑

根据快速加法器的原理，可以得到输出信号的逻辑表达式，经过相应的化简整理可以得到如下的逻辑表达式：

S4 = X4⊕Y4⊕C3 = P4⊕C3

S3 = X3⊕Y3⊕C2 = P3⊕C2

S2 = X2⊕Y2⊕C1 = P2⊕C1

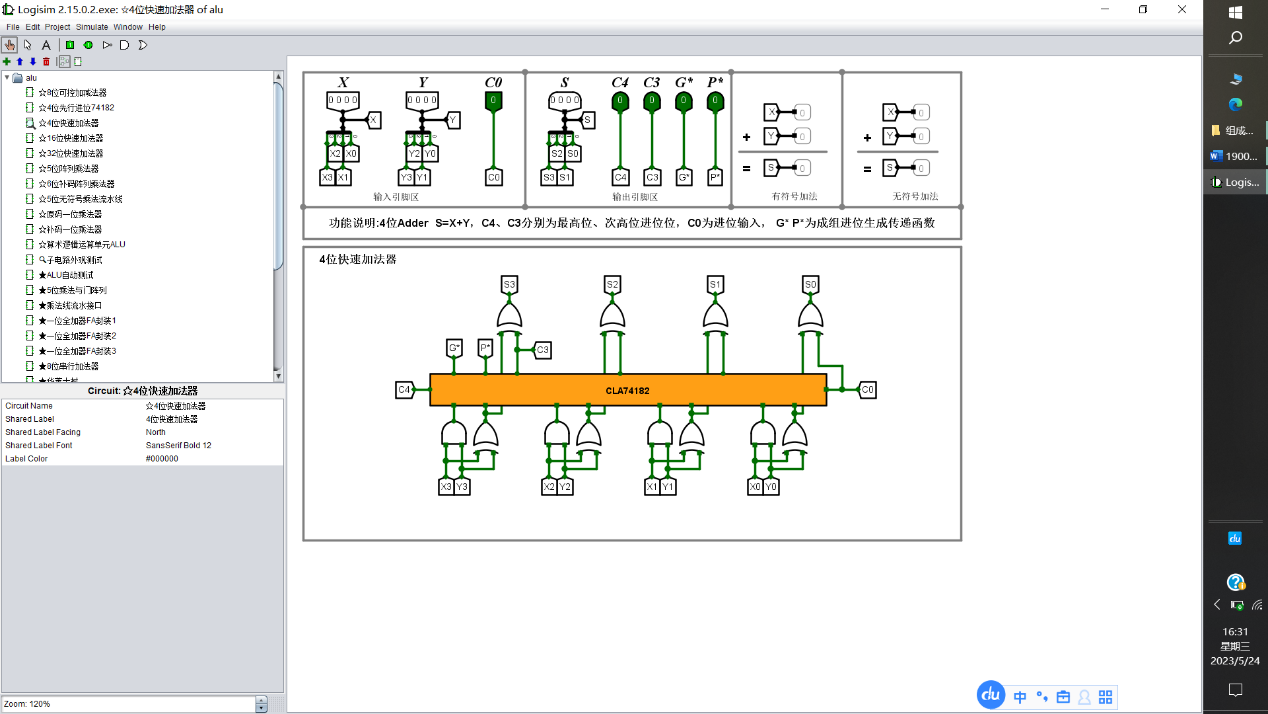
S1 = X1⊕Y1⊕C0 = P1⊕C0

Gi = Xi Yi Pi = Xi⊕Yi

**五、实验步骤**

（1）在Logisim中打开alu.circ文件，选择4位快速加法器子电路；

（2）根据成组生成函数和成组传递函数的表达式，利用与门和异或门设计Gi、Pi电路；

（3）根据各个和数输出S的逻辑表达式，基于上个实验已经封装好的CLA74182先行进位器，将各个引脚及逻辑门连接完整；

（4）进行电路测试

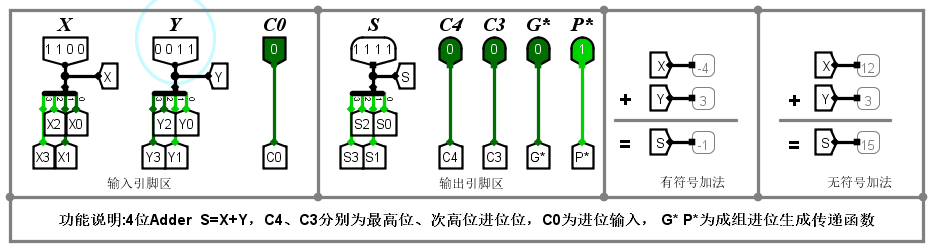
利用给出的测试电路，自定义输入，进行4位快速加法器的自动测试。将alu.circ文件所有文字信息复制粘贴到Educoder平台代码区域，点击评测按钮进行测试。

**六、故障与调试**

无故障与报错。

**七、实验结果测试与分析**

自定义X、Y、C0的值进行测试，观察S、C4、C3、G\*、P\*的输出；



平台测试的结果：

****

### 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，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。

**四、实验方案设计**

在前述子实验已封装好的4位先行进位电路、4位快速加法器电路的基础之上，构造16位快速加法器。16位快速加法器的设计方案如下：

（1）输入：16位相加数X、Y，进位输入C0

（2）输出：和数输出S，进位输出C，16位成组进位生成函数G，16位成组进位传递函数P

（3）所需器件：CLA74182先行进位器，4位快速加法器，基本逻辑门电路等

（4）电路引脚：



（5）16位快速加法器的设计逻辑

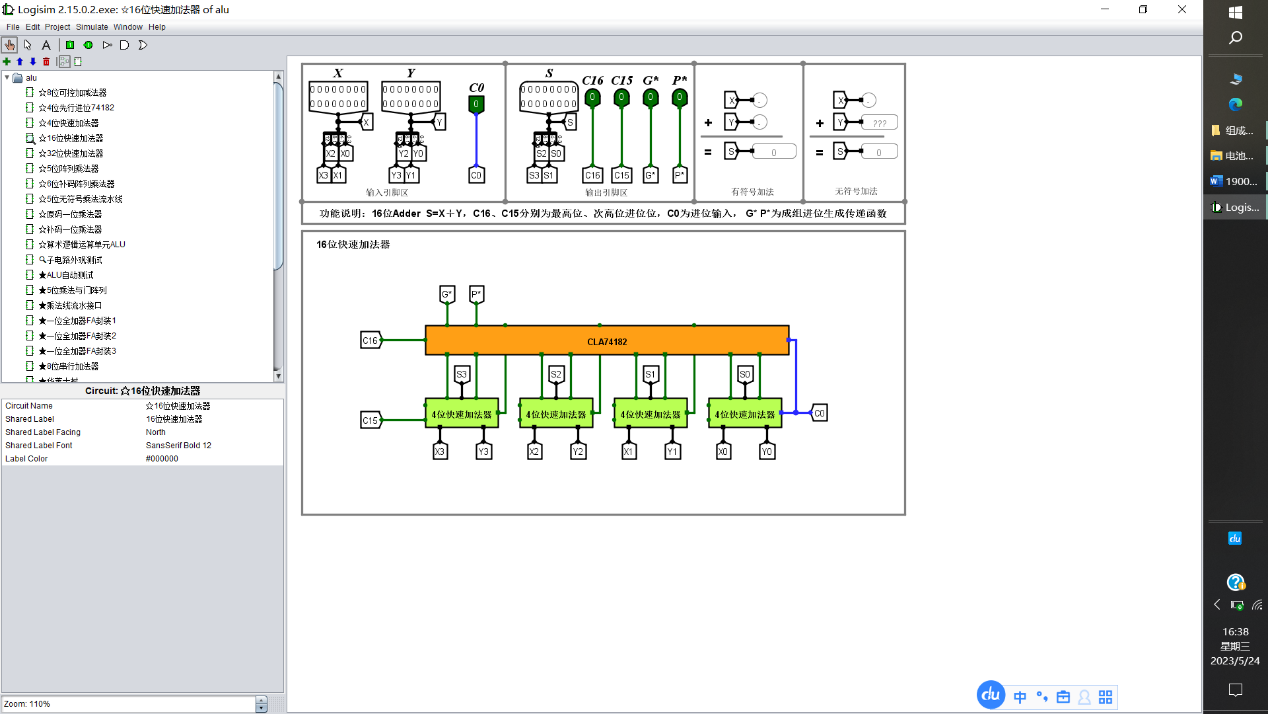
16位快速加法器需要将加数的16个数据位分为4组，每组包含4位数据；利用四个4位快速加法器和一个CLA74128先行进位电路，得到每一组的进位输入，即C0、C4、C8和C12；最后，利用4位快速加法器依次计算得到运算结果S3、S2、S1和S0。

**五、实验步骤**

（1）在Logisim中打开alu.circ文件，选择16位快速加法器子电路；

（2）先将两个加数分成四组，分别与四个4位快速加法器相连；

（3）再将上述四组器件依次与CLA74128先行进位电路相连；注意C0需要同时连接CLA74128和第一个4位快速加法器的输入端；



（4）进行电路测试

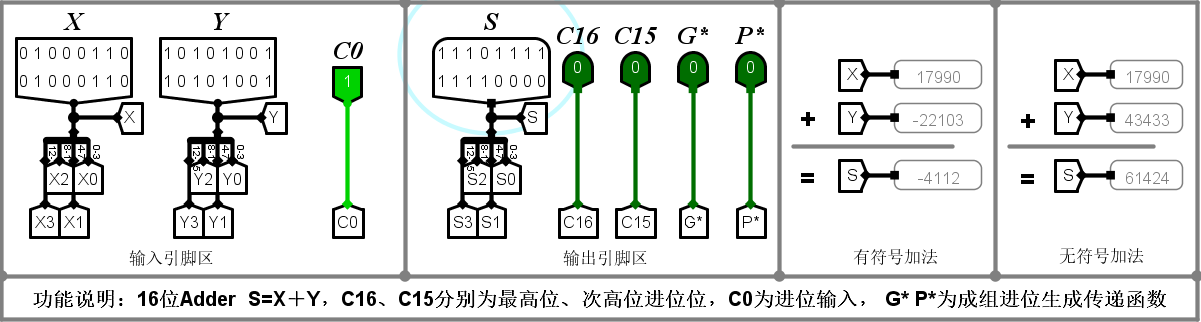
利用给出的测试电路，自定义输入，进行16位快速加法器的自动测试。将alu.circ文件所有文字信息复制粘贴到Educoder平台代码区域，点击评测按钮进行测试。

**六、故障与调试**

无故障与报错。

**七、实验结果测试与分析**

自定义X、Y、C0的值进行测试，观察S、C16、C15、G\*、P\*的输出，右侧显示有符号和无符号加法结果；

****

平台测试的结果：



### 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，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**

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，

**四、实验方案设计**

在前述子实验已封装好的4位先行进位电路、16位快速加法器电路的基础之上，构造32位快速加法器。32位快速加法器的设计方案如下：

（1）输入：32位相加数X、Y，进位输入C0

（2）输出：和数输出S，进位输出C，有符号加法运算溢出信号Overflow

（3）所需器件：CLA74182先行进位器，4位快速加法器，基本逻辑门电路等

（4）电路引脚：



（5）32位快速加法器的设计逻辑

32位快速加法器设计的可能方案有以下三种：

·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；

·方案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；

·方案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；

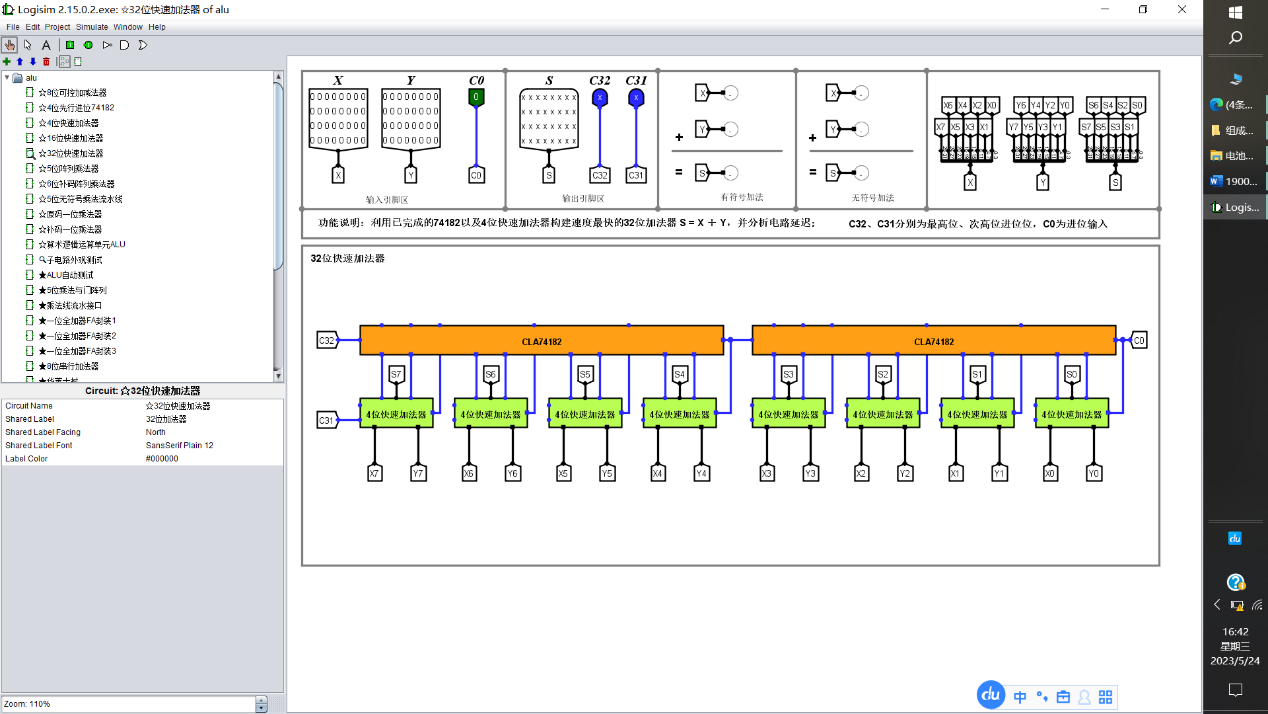
本实验选择方案3：在16位快速加法器的基础上，再增加一级组间先行进位电路，构建32位快速加法器；以第一个CLA74128先行进位电路的最高位进位输出作为第二个CLA74128的进位输入；最后再利用八个4位快速加法器依次计算得到运算结果S7~S0。

**五、实验步骤**

（1）在Logisim中打开alu.circ文件，选择32位快速加法器子电路；

（2）先利用上个子实验的分组分块方法，将32位数据分为8组，再将每4个为一大组，构建两个16位快速加法器；

（3）再将第一个CLA74128先行进位电路最高位进位输出，作为第二个CLA74128的进位输入；将C0连接到第一个CLA74128和4位快速加法器的进位输入；



（4）进行电路测试

利用给出的测试电路，自定义输入，进行32位快速加法器的自动测试。将alu.circ文件所有文字信息复制粘贴到Educoder平台代码区域，点击评测按钮进行测试。

**六、故障与调试**

故障一

1. 故障现象

接口处位宽具有较大问题。

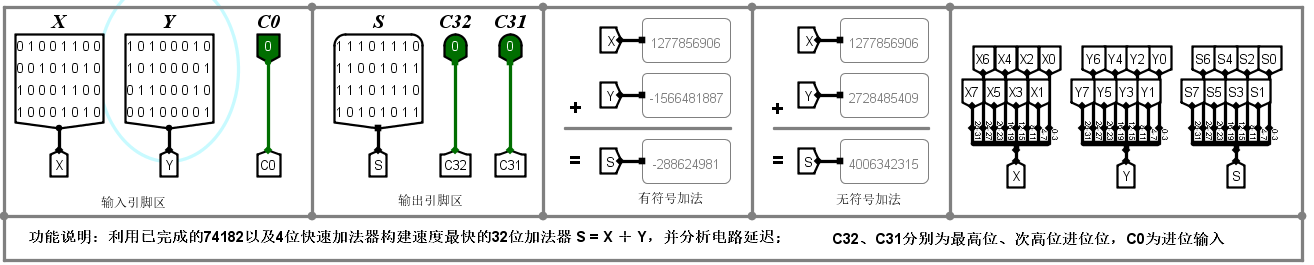
1. 原因分析

在进行32位加法器的设计时，首先需要对于4位先行进位进行连接设计，但在连接时，对于接口大小数值调配不成功。

1. 解决方案

一个一个根据其数据接口处大小进行调节，调整到电路无错。

**七、实验结果测试与分析**

自定义X、Y、C0的值进行测试：

平台测试：



### 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，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。

**四、实验方案设计**

在前述子实验已封装好的32位快速加法器电路的基础之上，结合Logisim平台中现有运算部件，构造MIPS运算器。MIPS运算器的设计方案如下：

（1）输入：32位相加数X、Y，进位输入C0

（2）输出：和数输出S，进位输出C，有符号加法运算溢出信号Overflow

（3）实现功能：实现算术加、减、乘、除运算，逻辑与、或、非、异或运算，逻辑左移、逻辑右移、算术右移运算；实现常用程序状态标志的功能（有符号溢出OF、无符号溢出UOF，结果相等Equal）

（4）所需器件：32位快速加法器，Logisim平台中其他现有运算部件

（5）电路引脚：



（6）MIPS运算器设计方法

·运算实现

将运算器整体分为多个功能模块来逐一设计，以此简化硬件线路的实现。每次运算时，先通过ALU\_OP信号选择对应的运算方式进行运算，然后通过多路选择器，将所选运算方式对应的结果给Result，将乘法运算时高位结果或除法运算时余数给Result2，其余情况下Result2的输出为0。

·溢出判断

判断加法溢出时，UOF加连C32引脚，OF加连C31引脚；判断减法溢出时，可以利用比较器，通过对X和Y的比较得到UOF减，通过加法器引脚得到OF减。

（7）运算器功能码的功能对照表如下：



**五、实验步骤**

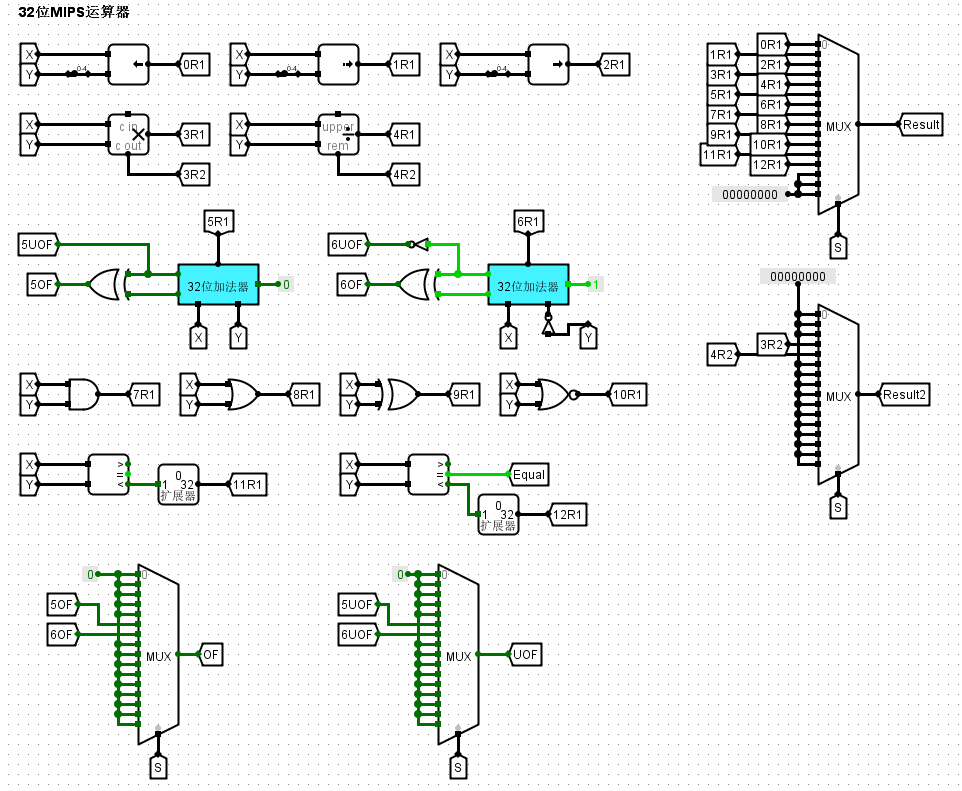
（1）在Logisim中打开alu.circ文件，选择算术逻辑运算单元子电路；

（2）利用前述子实验中已经设计封装好的32位加法器，以及Logisim平台中的现有运算部件，实现基本运算的逻辑电路；

（3）再利用多路选择器，设计加减溢出判断标记的电路图；

（4）最后再利用多路选择器，将上述各种运算结果合并，以ALU\_OP（S）为选择控制信号，实现最终的结果输出；

（5）将上述部件组合，最终得到的MIPS运算器的电路图如下：



（6）进行电路测试

**六、故障与调试**

1. 故障一
2. 故障现象

测试结果不正确。

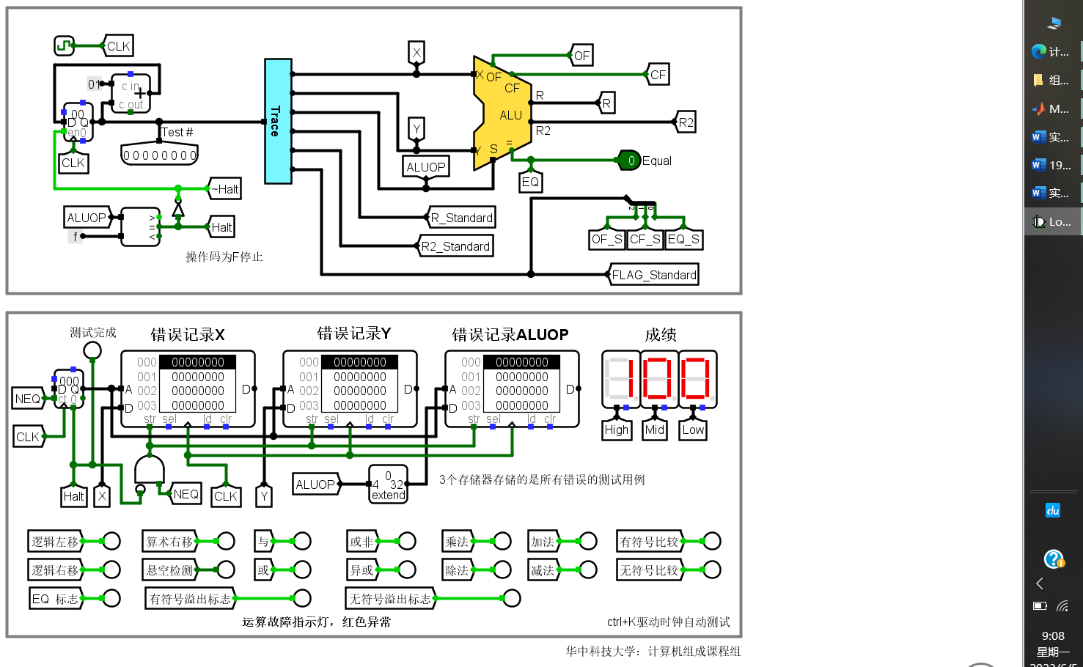
1. 原因分析

忘记对减数取补码。首先是溢出的判断错误，第一轮芯片功能测试之后得到了错误的判断方法，主要原因在于没有理解两个溢出的含义，加法溢出判断的两个引脚弄反，修改提交之后还是出现错误，分析后发现无符号减法运算的判断不能直接通过芯片的引脚(减数符号位为1，无符号溢出位永远为0)，应当通过对运算数真值的比较得出。

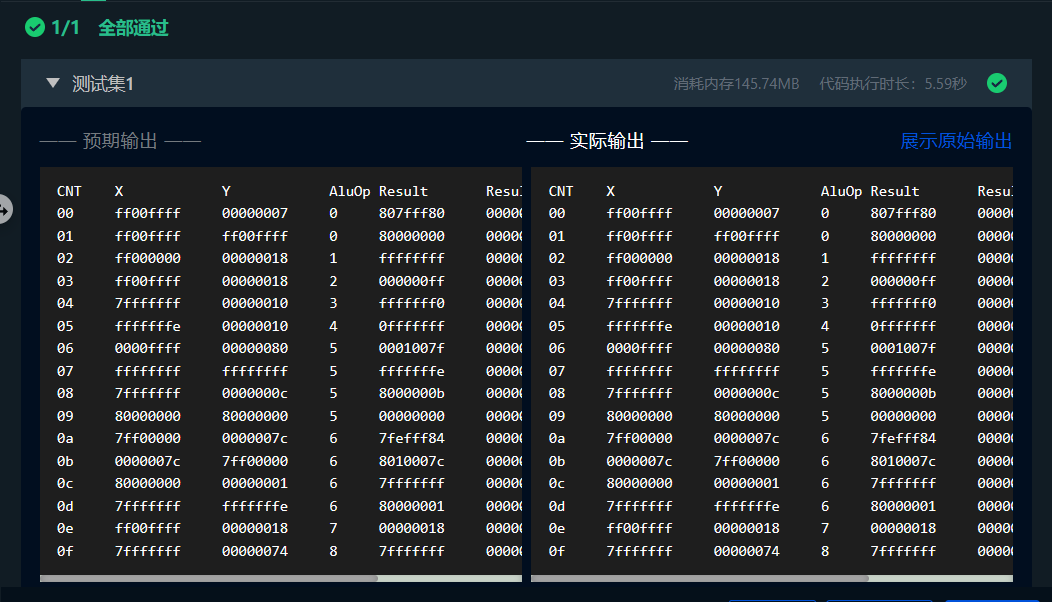
1. 解决方案

连接将乘法运算高位或除法余数位结果输出给Result2的电路中左移右移取的是y低五位，而不是全部。在电路连接的过程中将数据位宽调整匹配。

**七、实验结果测试与分析**

自动测试：

平台测试：



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

**四、实验方案设计**

MIPS寄存器文件又称为MIPS寄存器组或MIPS寄存器堆。利用Logisim构建一个简化的MIPS寄存器文件，设计方案如下：

1. 寄存器个数：包含4个32位寄存器
2. 所需器件：多路选择器、译码器、解复用器、寄存器等
3. 电路引脚：
4. 实现功能：对所有寄存器的写入、读取

**五、实验步骤**

（1）在Logisim中打开已下载的storage.circ文件，选择MIPS Regfile子电路；

（2）利用分线器，将5位读寄存器地址编号的低两位，作为所选的寄存器编号0~3，并赋给R1#、R2#、W#；

（3）利用解复用器控制写使能信号WE以及写入寄存器编号W#；

（4）结合时钟信号CLK，以及解复用器输出端的写入信息，将写入数据Din存入4个寄存器中；

（5）再利用两个多路选择器，四个寄存器数据作为备选数据，由R1#、R2#提供所选寄存器的编号，然后读出对应数据，将读出的两个寄存器的值分别给RD1、RD2进行输出；

**六、故障与调试**

1. 故障一
2. 故障现象

开始有几组数据的输出结果不合预期。

1. 原因分析

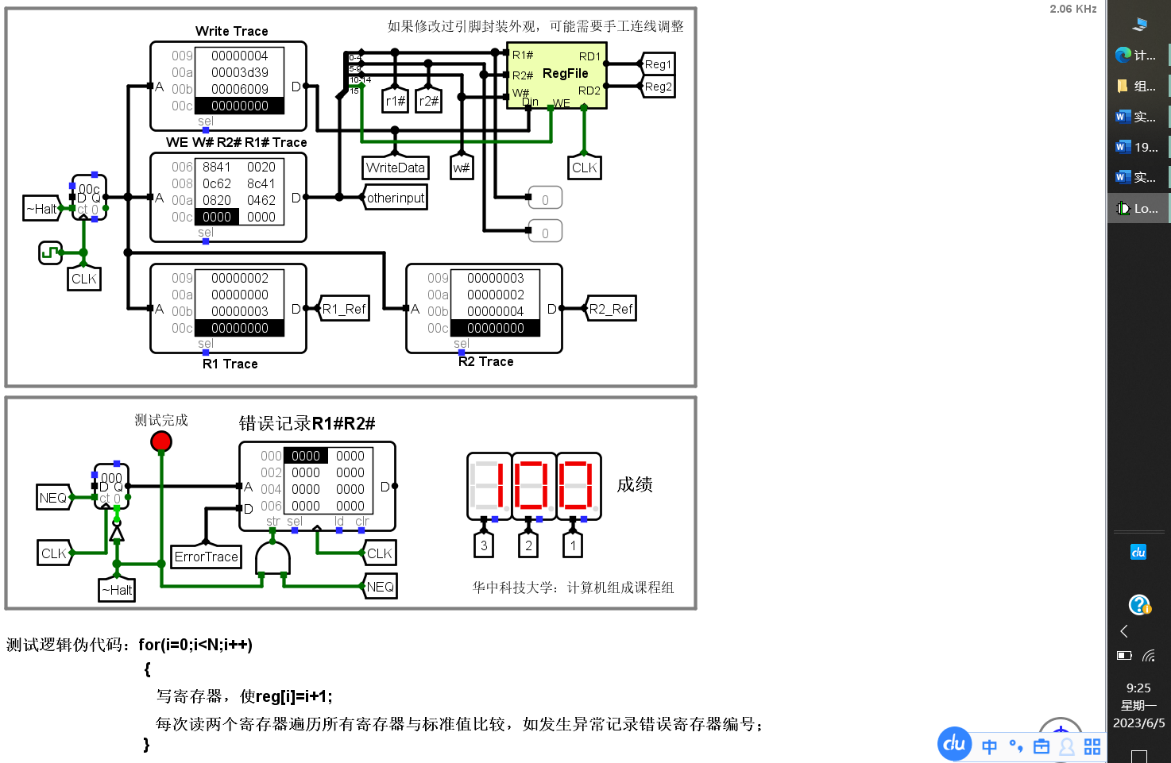
最终发现是[多路选择器](https://so.csdn.net/so/search?q=%E5%A4%9A%E8%B7%AF%E9%80%89%E6%8B%A9%E5%99%A8&spm=1001.2101.3001.7020" \t "_blank)的两根输入线并在了一起，并的位置与多路选择器的边框重合，需要对多路选择器进行拖拽才能发现问题。

1. 解决方案

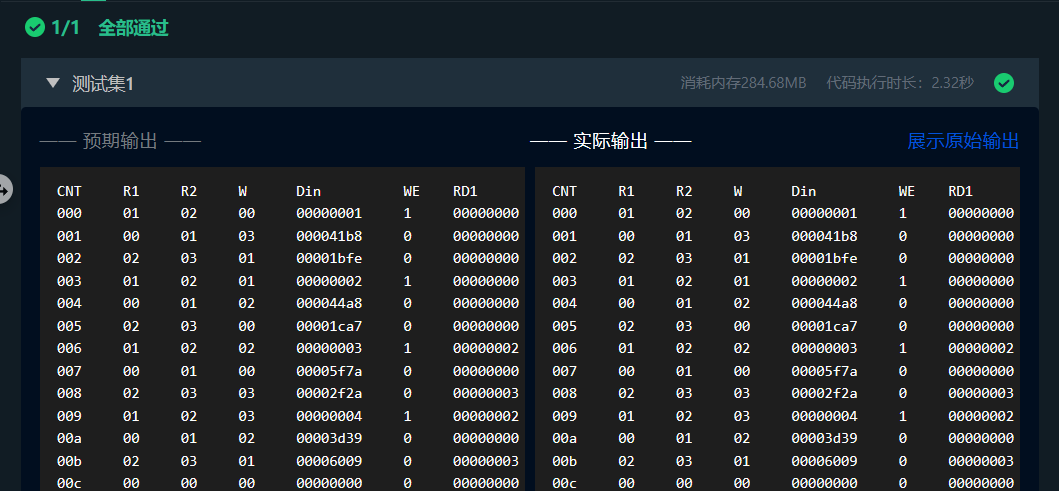
在排查故障的过程中更换了另一种方法，通过片选端控制寄存器的写操作（禁用信号与使能端的忽略脉冲信号相反，因此在解复用器后添加了非门）。

**七、实验结果测试与分析**

自动测试：



平台测试：

****

第2关 直接相联cache设计

**一、实验目的**

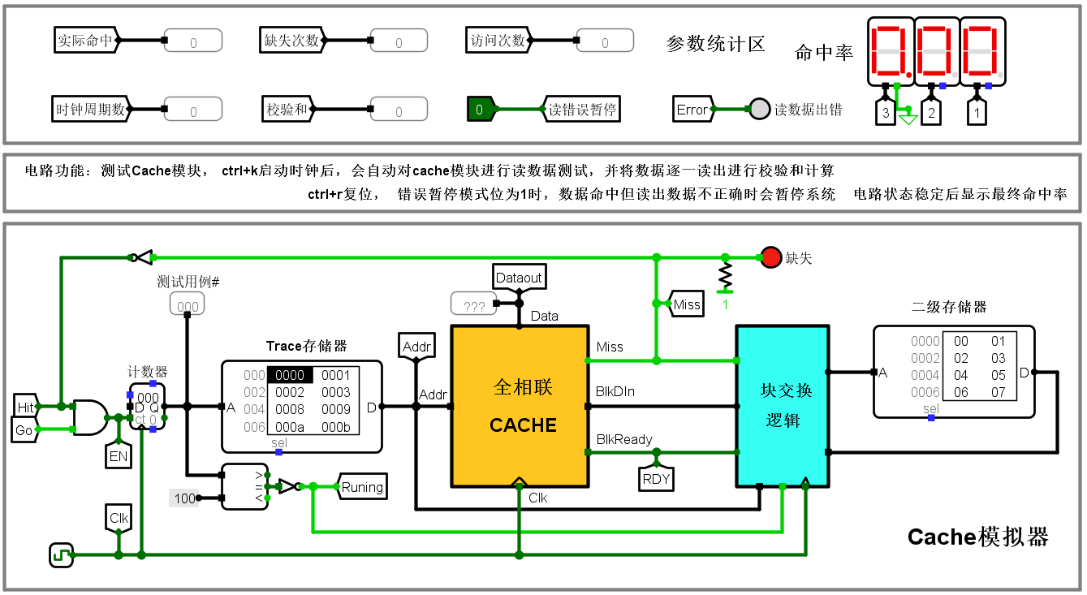
学生掌握 cache 实现的三个关键技术：数据查找，地址映射，替换算法，熟悉译码器，多路选择器，寄存器的使用，能根据不同的映射策略在 Logisim 平台中用数字逻辑电路实现 cache 机制。

**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**

下图给出了一个在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**

直接相联的cache的设计方案如下：

（1）cache槽：所设计的cache具有8个cache槽

（2）设计规格：主存地址16位，一个cache模块共包括8个cache行，每个数据块包含4个字节共32位数据。

（3）所用器件：译码器、多路选择器、编码器、寄存器、比较器等

（4）电路引脚：



（5）直接相联映射的逻辑设计：

在明确主存地址划分（标记字段tag为11位，行索引字段index为3位，块内偏移字段offset为2位）以后，cache需要实现逻辑：

**·数据查找逻辑**

对于直接相联映射，由于一个数据块只能对应到cache中的特定行，查找算法较为简单，只需查看tag和有效位，即可判断数据是否命中。

从硬件实现的角度来设计，需要一个行索引译码器和一个比较器：只需要将index字段输送到一个行索引译码器，就可以得到8个行索引译码输出，其中输出高电平的那一行就是当前主存地址对应的cache行；利用8个行译码输出信号，控制8个三态门，将所有行的有效位以及标记字段输出到一个比较器；根据译码器的定义，只有当前主存地址对应的那一行的数据才会输出，因此我们将当前行的有效位和1进行比较，同时将tag位和主存地址中的区地址进行比较，若两个都相等表示命中，否则表示缺失。

**·数据访问逻辑**

用上述的8路译码输出信号控制cache块的数据输出。将所有cache行的数据块都通过三态门输出到数据总线（槽数据）上，三态门的控制端就是这8个译码输出信号。

命中时，命中行的数据会输出到数据总线上。接下来具体选择数据块中的哪一个字，由主存地址中的offset字段决定，硬件上可以使用多路选择器实现。注意，命中信号要直接连接到多路选择器的使能端，缺失时多路选择器的输出为高阻态。

**·数据载入逻辑**

将主存地址访问序列先进行地址划分，将区地址字段直接送入比较器，行索引字段送入行地址译码器，选中输出为高电平的那一行，控制该行的有效位和标记位通过三态门输出到比较器，与区地址和有效位1进行比较，判断比较结果是否相等。

若结果不相等，则数据发生缺失，需要对新的数据进行载入：需要从二级存储器（内存）中载入该主存地址所在的数据块。由于内存的DRAM比cache的SRAM要慢很多，所以这个数据块的载入过程比较漫长，在这段时间内，CPU会死锁等待相应数据块的载入，此时CPU不能进行任何其他操作。

数据载入过程一般需要若干个时钟周期，对应的数据块会载入到之前选中的行中；同时，会对选中的行的标记位进行更新，并将有效位设置为1。

这些字段更新完毕以后，比较器结果会变为相等，即数据最终会变为命中。然后将该行数据块中的数据输出到数据总线，再由offset字段进一步选择具体的字，进行最后的输出。

若起初比较时有效位为1，但标记位与区地址字段并不相同，则接下来应该进行替换操作：将主存地址访问序列对应内存中的数据块载入选中的这一行替换原来的数据，同时更新对应的标记字段。

**五、实验步骤**

（1）在Logisim中打开已下载的storage.circ文件，选择Cache（直接相联）子电路；

（2）先利用分线器对主存地址进行地址划分；

（3）再利用解复用器构造行索引译码器，对index字段进行译码，选定某一行；

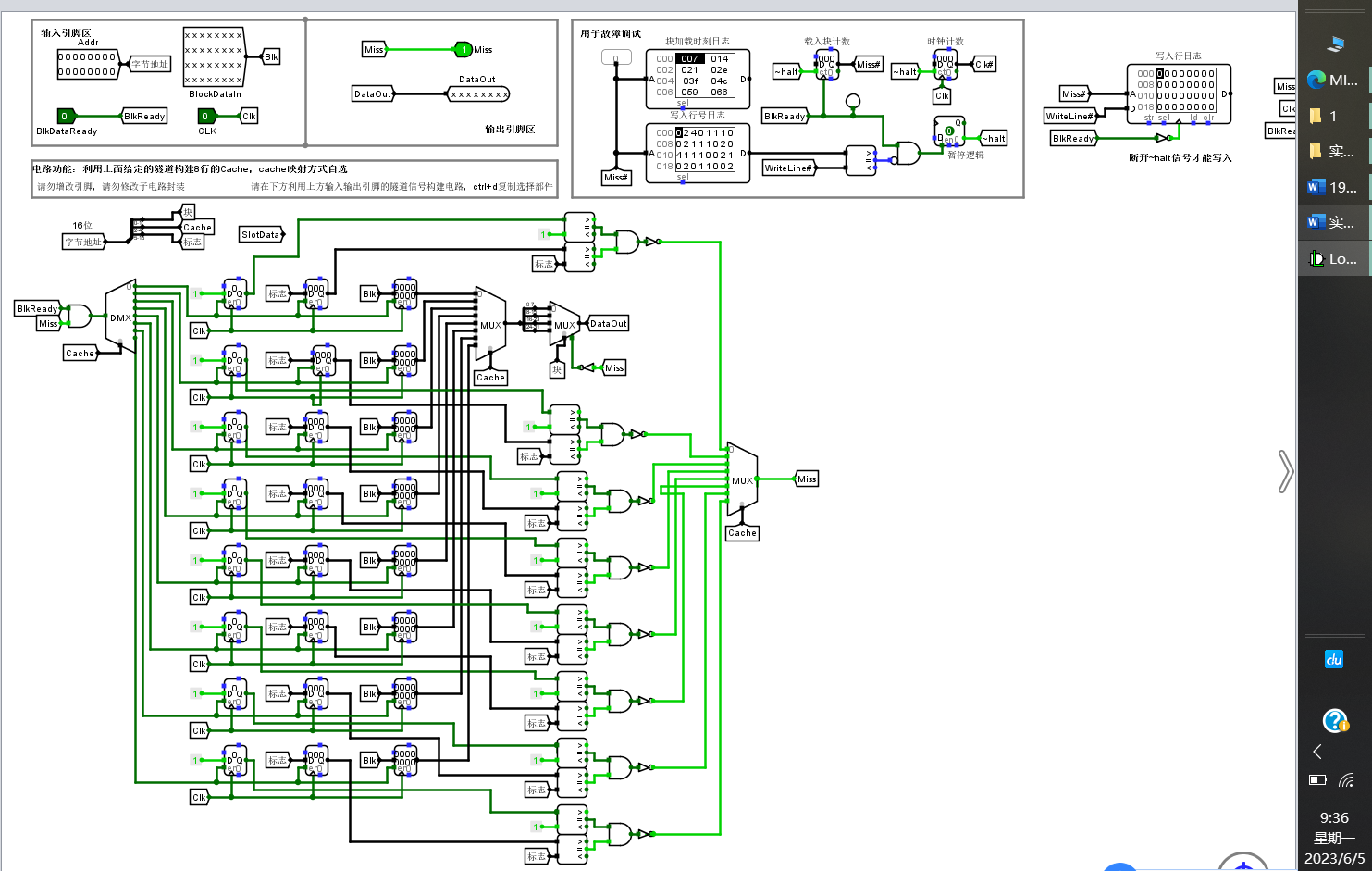
（4）每行中对应地址三个寄存器存储，第一个寄存器存储有效位（数据位宽为1），第二个寄存器存储标记位（数据位宽为11），第三个寄存器存储块内数据（4个字节，数据位宽为32）；8行共使用24个寄存器；

（5）利用比较器和基本逻辑门构造综合比较器，判断有效位是否为1，且标记位与区地址字段是否相等；

（6）再结合多路选择器对判断结果进行汇总，用于判断结果是命中、缺失还是替换；

（7）最后利用多路选择器，结合offset字段，实现对数据块中具体字的选择和输出；

（8）综上设计得到MIPS寄存器文件的电路图如下所示：



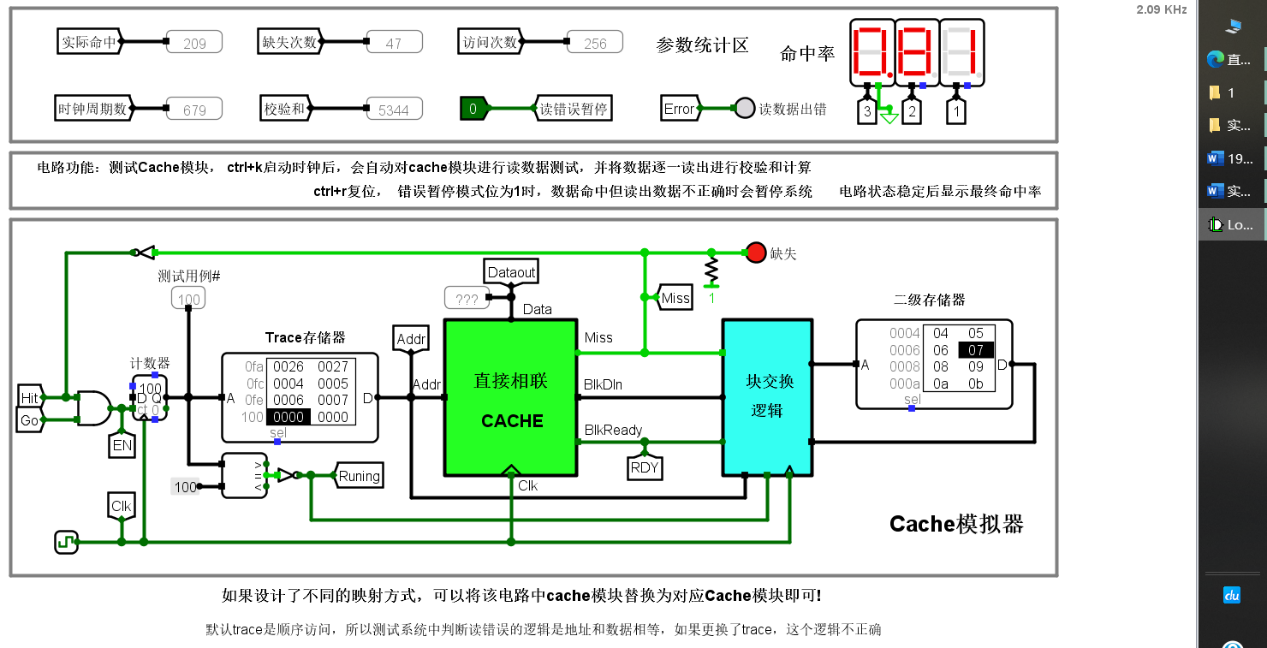
（9）进行电路测试：

**六、故障与调试**

无故障与报错。

**七、实验结果测试与分析**

自动测试：



平台测试：

****

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。

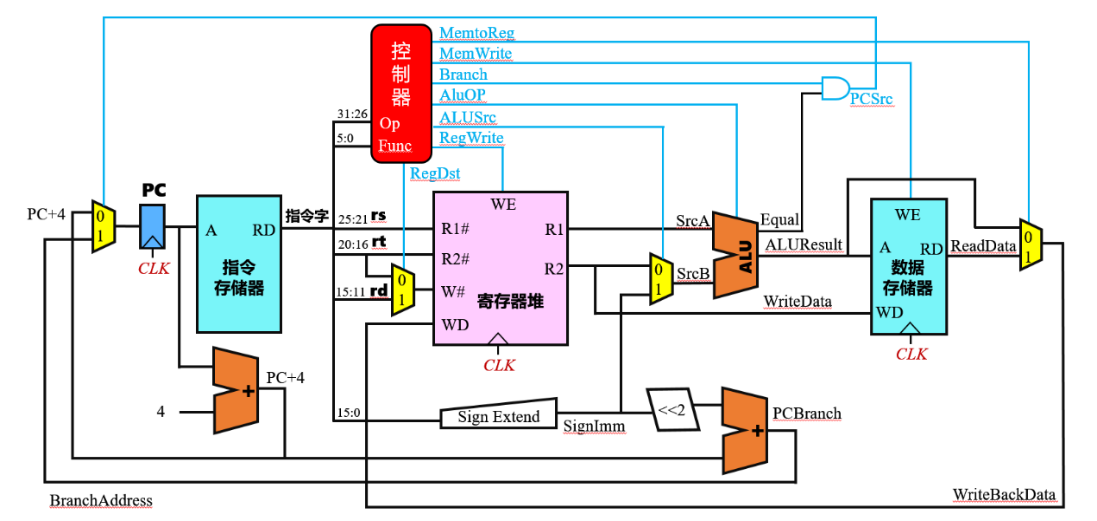
**二、实验环境**

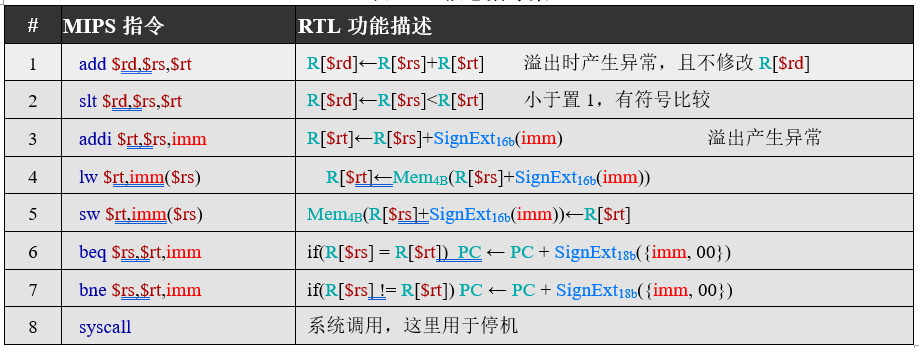
1.实验所使用的工具软件：Logisim

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。





**四、实验方案设计**

单周期MIPS CPU的设计方案如下：

（1）主要任务：绘制MIPS CPU数据通路，实现单周期硬布线控制器，并完成测试联调；

（2）所用器件：程序计数器PC、寄存器文件Regfile、指令存储器IMEM、数据存储器DMEM、运算器ALU、单周期硬布线控制器Controller；

（3）电路引脚：

|  |  |  |  |
| --- | --- | --- | --- |
| **信号** | **输入/输出** | **位宽** | **功能描述** |
| CLK | 输入 | 1 | 时钟信号 |
| PC | 输出 | 32 | 程序计数器的值 |
| IR | 输出 | 32 | 当前指令字 |
| RegWrite | 输出 | 1 | 寄存器文件写使能控制信号 |
| RDin | 输出 | 32 | 寄存器文件写入端口的数据 |
| MemWrite | 输出 | 1 | 存储器写使能控制信号 |
| MDin | 输出 | 32 | 存储器写入端口的数据 |

（4）单周期硬布线控制器的设计：

1）输入信号：指令字OP、Func字段（12位）

2）输出信号：多路选择器选择信号、内存访问控制信号、寄存器写使能信号、运算器控制信号、指令译码信号，具体表示如下表所示：

3）实现逻辑：

·指令译码

对于非R型指令，可以根据给出的6位指令操作码OP产生对应的指令译码信号，LW、SW、BEQ、BNE、ADDI等。硬件实现中为了简化设计，选择利用比较器进行相应比较，得出译码信号；

将OP与常量0进行比较，若相等，则为R型指令。对于R型指令要根据Func字段来确定具体是ADD、SLT还是SysCall指令。注意：R\_TYPE表示R型运算类指令（不包括SysCall这样的特殊指令）。

·ALU控制器

利用OP、Func字段生成对应的运算器AluOP信号，即4位的运算器功能选择信号。

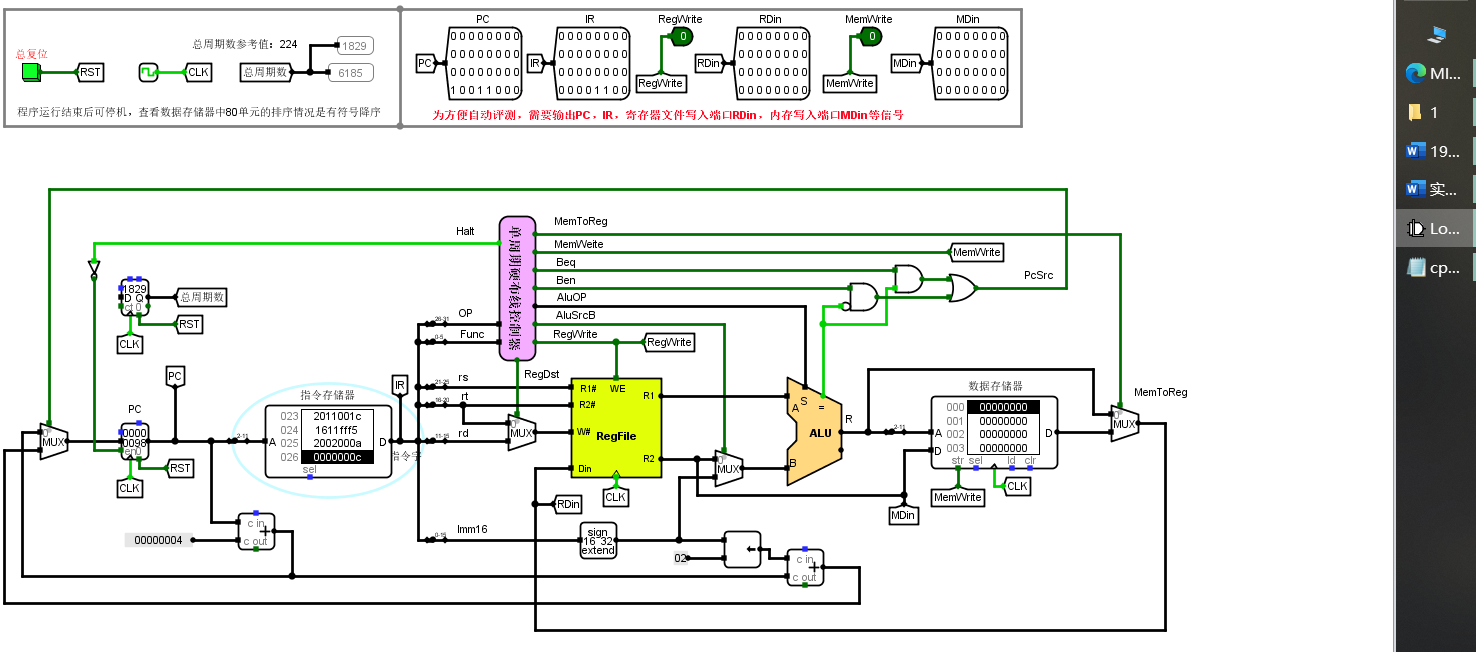
·其余控制信号

通过增加简单的组合逻辑，根据给出的指令译码信号，实现剩下的8个控制信号逻辑。

（5）单周期MIPS CPU通路设计：参照“实验设计要求”中数据通路图进行设计。

**五、实验步骤**

（1）在Logisim中打开已下载的cpu.circ文件，同时加载cs3410.jar文件（包含设计所需器件）；

（2）先选择单周期硬布线控制器子电路，在已有框架的基础之上，完成单周期硬布线控制器的设计：

（3）停机处理的设计：

设计停机逻辑处理时，不能对时钟进行直接的与操作，而是应该控制相应的寄存器的写使能信号来达到停机的目的，即应当控制电平信号而不是控制时钟信号；

（4）选择单周期MIPS（硬布线）电路，开始构建MIPS主机通路，依照数据通路图完成通路的连接：

其中，左上方的计数器用来记录CPU运行的总周期数（参考值为224），受时钟周期和复位信号的控制；

（5）进行电路测试：

·自动测试

在指令存储器中载入排序程序sort.hex，利用Logisim的时钟自动仿真（快捷键Ctrl+k）运行程序；

当程序运行完成以后，整个CPU处于停机状态（因为最后一个指令设置为SysCall）。程序停机以后，查看数字存储器中的排序情况，在080位置应该得到一个有符号降序排列。

·平台测评

自动测试结果满足实验要求后，再利用记事本打开cpu.circ文件，将所有文字信息复制粘贴到Educoder平台代码区域，点击评测按钮进行测试。

**六、故障与调试**

1. 故障一
2. 故障现象

输出报错：“Cannot find Jar Library：cs3410.jar”；

1. 原因分析

根据平台常见故障说明，原因可能是由于本地cs3410.jar和测试平台中的cs3410.jar路径不一样。

1. 解决方案

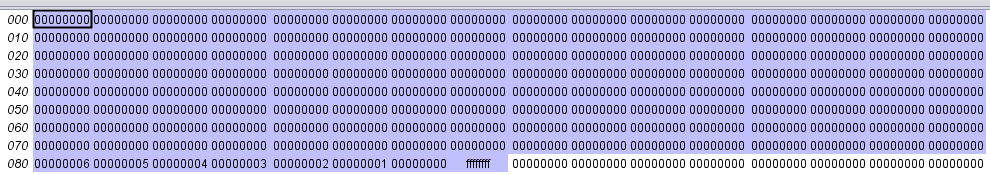
直接修改平台代码：

<lib desc="jar#6.CPU设计实验/cs3410.jar#edu.cornell.cs3410.Components" name="7">，将上面的绝对路径删除，修改为：

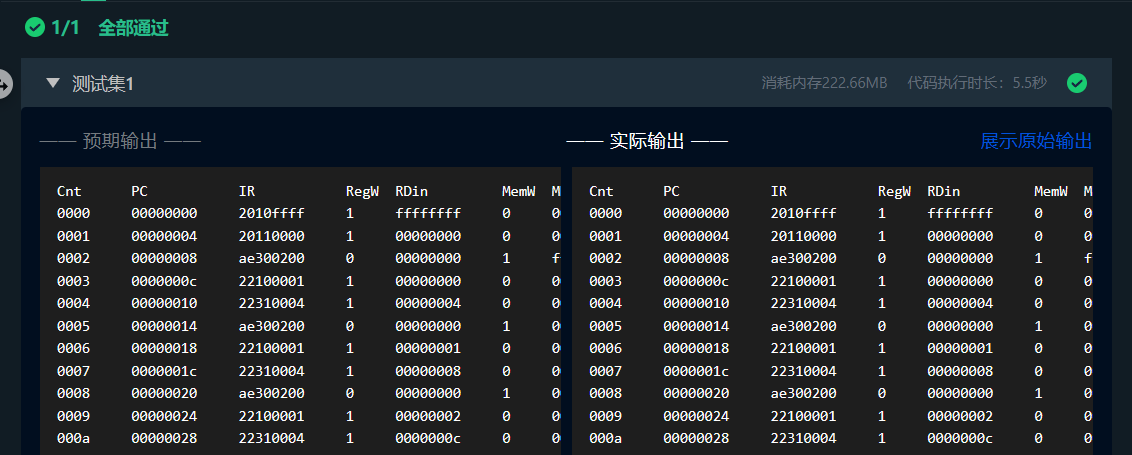
<lib desc="jar#cs3410.jar#edu.cornell.cs3410.Components" name="7">

**七、实验结果测试与分析**

自动测试：

****

平台测试：

****