# 实验一、MIPS寄存器文件设计实验

**1、实验目的**

了解MIPS寄存器文件基本概念，熟悉寄存器、多路选择器、译码器、解复用器等Logisim组件的使用。

利用Logisim软件平台构建一个MIPS寄存器文件，为减少实验绘图量，只要求实现4个寄存器，寄存器的编号由分线器引出低2位。其中0号寄存器的值是恒为零。

要求画出具体的MIPS寄存器文件实现电路。

1. **实验环境**

**WIN10、LOGISIM2.7.1**

1. **实验步骤过程**

**首先为寄存器通入CP信号，然后将W#作为2-4解码器的选择端口输入，WE作为解码器使能端输入，因为0寄存器需要恒为0，所以Din只作为1-3寄存器的数据段端口输入。这样，当Din和W#、WE信号准备好的时候，时钟上升沿到来就能够实现将Din的数据输入到W#地址的解码对应的寄存器中。**

**之后把每个寄存器的输出端依次链接到数据选择器的输入端，同时每个寄存器的输出路线上分出另外一条路线，依次连接到另外一个数据选择器的输入端，R1#和R2#作为数据选择器的选择端，这样就能实现同时读出R1#和R2#选择的寄存器数据。**

**之后**在测试子电路中，按Ctrl+K组合键进行功能自动测试

1. 实验结果

电路图：

测试结果：

5、实验分析及反馈

使用译码器的方式是通过WE信号控制译码器选择W#地址的寄存器使能端

使用解复用器是通过为W#地址的寄存器输送WE信号

效果上看是一致的

# 实验二、存储器扩展实验及Cache映射实验

**1、实验目的**

理解存储系统进行位扩展、字扩展的基本原理，能利用相关原理解决实验中的实际汉字字库的存储扩展问题，并了解汉字字库的数据填充。

在Logisim平台中构建GB2312汉字编码的16×16点阵汉字字库。4片4KB×32位芯片扩展为16KB×32位空间代替下图中的空白处的1片16KB×32位芯片。补充完整4片4KB×32位芯片的字库数据，并且在测试电路中加载自己的班级姓名学号并输出

掌握Cache实现的3个关键技术：地址映射、数据查找、替换算法。熟悉译码器、多路选择器、寄存器等器件的使用，利用Logisim软件实现直接相联映射只读Cache的构建，设定Cache模块包含8个块，每个块包含4个字节。

**2、实验环境**

**WIN10、LOGISIM2.7.1**

1. **实验步骤**

**字库拓展：**

**首先利用分线器将14位数据线的高2位和低12位地址分离，高两位作为解码器的选择信号，解码器的输出分别连接四个**4KB×32位芯片的选择端，剩下的低12位均连接到四个4KB×32位芯片的地址端口。四个4KB×32位芯片的输出端连接到一条线上通往D1作为扩展的16KBX32芯片输出端。

然后从参考字库中分离对应我们拓展替代的ROM芯片中的数据，按解码器选择顺序平分到四个4KB×32位芯片上。

最后从网络上查询自身班级姓名学号对应的国标码载入测试电路的ROM，启动测试电路。

Cache直接映射：

首先将字节地址使用分线器进行分位，0-1位作为选择选中Cache块的四个字节数据的选择信号，送入数据选择器的选择信号端口，而被选择的数据则是从被选择的Cache块的输出数据SlotData中分出的四个字；2-4位作为译码器的输入来选择8个Cache块的某一个，译码器分别连接8个Cache块行选中“隧道”；5-13位连接Cache块的标记位Tag“隧道”，同时Tag信号和被选中Cache块的TagData送入比较器进行比较，等于的输出端口的输出和有效位一起送入与门，输出送入Hit隧道，取反送入Miss隧道，而Hit隧道则连接Cache块字节数据选择器的使能端口。

由于当数据未命中时，需要从主存获取数据的过程已经给出，所以Cache直接映射完成

1. 实验结果

字库输出节选：

Cache测验结果：

1. 实验分析及反馈

通过....掌握了存储容量扩展的方法。。。。

明白了直接映射。。。。的缺点（抄课本）

# 实验三、加（减）法器设计实验

**1、实验目的**

掌握1位全加器的逻辑实现，掌握多位可控加减法电路的实现逻辑，熟悉利用Logisim平台中的相关基本功能实现多位可控加减法逻辑电路。用1位全加器扩展实现8位串行进位加法器，将8位串行进位加法器改为8位可控加减法器。

**2、实验环境**

**WIN10、LOGISIM2.7.1**

1. **实验步骤**

1位全加器子电路已经完成可直接调用

将8个1位全加器的进位链串联即可得到8位加法器无符号数的溢出依据最高位的进位判断，有进位，则溢出。补码数加法的溢出依据依据最高数据位的进位及符号位的进位判断，两者异或为1则产生溢出。

将减数Y的各位分别连接一个异或门，另一个输入端口则由SUB信号输入，同时SUB信号传入最低位的加法器进位输入端口，这样就能够完成减去一个数[Y]补的实现。

1. 实验结果

加法测试：

减法测试：

5、实验分析及反馈

串行加法的高位正确输出需要低位进位输入的支持，得到正确结果的速度比较慢。

可控加减法器的Sub信号可由所需操作的CU控制信号选择

# 实验四、ALU设计实验

**1、实验目的**

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32位的加法器和Logisim平台中的运算组件构成指定规格的ALU运算单元。在给定的子电路中，设计182、4位加法器，扩展组成32位加法器，在ALU电路中连接剩余的线路，得到线路准确的ALU单元。

**2、实验环境**

**WIN10、LOGISIM2.7.1**

1. 实验步骤

设计4位先行进位74182：

根据分析得到74182逻辑表达式为：

G=d p=t

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。

P\*和G\*，将作为使用双重分组跳跃进位链设计32位快速加法器的高层182输入端使用，代表的是分配的4位数据输入组成的小组产生的本地进位信号和传送条件信号。

根据逻辑表达式连接线路

设计4位快速加法器：

使用设计好的74182先行进位器产生第1位到第3位的加法的进位，作为各个对应的1位加法器进位输入端信号输入，初始进位c0送入第0位的加法器进位输入，数据输入对应两加数各位的输入，输出对应各个位的加法输出，这样就设计了4位快速加法器

使用双重分组跳跃进位设计32位快速加法器：

首先将两个32位加数都分别分成8份每份四位的数据，分别送入设计好的4位加法器的输入端而8份数据分为低4组即第一大组和高4组即第二大组，将4位快速加法器编号从低到高编为0-7，每个4位加法器的传送条件信号G\*和本地进位信号P\*作为74182的输入端传入，初始进位信号C0作为第0号4位快速加法器和第一大组的74182Cin端输入，1-3号4位快速加法器的进位由第一大组74182产生，第二大组的4号快速加法器进位输入为第一大组的C4进位信号，同样其他输入由第二大组的各个4位快速加法器产生的传送条件信号G\*和本地进位信号P\*，第二大组产生的74182产生的进位信号又接入相应的5-7号4位加法器，最终加法结果是由各个4位快速加法器的输出按顺序组合在一起。

同时我们设置第7号4位快速加法器的最高进位输出和次高进位输出为C32和C31留做之后设计判断溢出使用

设计ALU：

按照要求，除了加减法之外使用LOGISIM自带组件作为ALU的功能组件，加减法使用设计好的32位快速加法器。符号数溢出判断就用我们之前设计32位快速加法器留下的C31和C32输出异或处理的结果，无符号加法溢出就用最高位进位C32判断，无符号减法就用减法结果和被减数送入比较器，分别作为A、B端，而比较器A>B的输出端则作为无符号减法溢出信号

最后将各个功能部件的结果，通过实验规定的运算器功能吗实验数据选择器进行编辑，数据选择器输出则为ALU输出Result，同理Result2用于乘法指令结果的高位或除法指令的余数位的ALU结果第二部分、OF有符号加减运算的溢出标志、UOF无符号加减运算溢出标志也使用数据选择器进行对应设置。最终ALU设计完成

1. 实验结果

4位先行进位74182：

4位快速加法器：

32位快速加法器：

ALU：

1. 实验分析及反馈

通过。。。初步实现ALU的基本。。。

同时理解。。有符号和无符号溢出判断的不同。有符号可使用最高进位输出和次高进位输出异或结果判断，无符号加法溢出可直接由最高进位输出判断，减法溢出可以通过差是否小于被减数的判断进行

# 实验五、CPU设计实验

**1、实验目的**

掌握控制器设计的基本原理，能根据硬布线控制器设计原理或微程序控制器设计原理设计实现MIPS处理器。画出单周期或多周期的MIPS处理器数据通路，选择单周期硬布线设计、多周期硬布线设计、多周期微程序设计的其中一种设计任务，完成控制单元的设计，要求能支持数条MIPS核心指令，进而完成MIPS处理器的设计。

**2、实验环境**

**WIN10、LOGISIM2.7.1**

1. 实验步骤
2. 了解MIPS指令，并实验中使用的指令进行相关查询和学习：

J型指令：实验设计的8条指令无J指令

I型指令：LW、SW、BEQ、BNE、ADDI

R指令：SLT、ADD

特殊指令：SYSCALL

LW 从存储器中读取一个字的数据到寄存器中

SW 把一个字的数据从寄存器存储到存储器中

beq: 数据跳转指令，标志寄存器中Z标志位等于零时, 跳转到BEQ后标签处

bne: 数据跳转指令，标志寄存器中Z标志位不等于零时, 跳转到BNE后标签处

SLT 如果Rs的值小于Rt，那么设置Rd的值为1，否则设置Rd的值为0

ADDI 把一个Rs寄存器的内容加上一个立即数存入Rt寄存器

ADD 把一个Rs寄存器和Rt寄存器内容相加，结果存入Rd寄存器

Syscall 停机指令

1. 查询MIPS32指令手册和MIPS GREEN SHEET，仔细分析每条指令的执行过程，设计数据通路
2. 进行指令设计

首先，分析实验所给的器件想要实现的功能：微指令程序中第一个微指令是取指令信号：从而一开始启动，就开始取命令到IR寄存器中，IR寄存器的数据经过数据通路分线，产生CU的输入信号OP和FUC第二部分，通过输入的OP和FUC字段输入指令译码信号，本来是需要使用译码器的，但是由于我们只有8条MIPS指令，且是使用LOGISIM进行实现，所以只需要在MIPS指令集中查找对应的MIPS指令代表的OP和FUNC字段，通过比较器比较相等即可产生译码信号，根据不同的译码信号通过地址转移逻辑进入相应的指令微程序入口

然后再次分析8条指令，对每条指令需要的每个时钟周期进行分析，得出每个指令在每个时钟周期需要产生的控制信号，填入微指令自动生成EXCEL表格，生成16进制的各个指令对应的微程序

之后设计好各个指令的微程序后将16进制微程序码存入控存。这样我们就能够根据设计好的不同指令的微程序入口设计地址转移逻辑:为不同指令填写相应入口地址生成相应的S0到S3产生的逻辑使用LOGISIM的分析电路功能完成转移逻辑设计

最后载入冒泡排序程序测试

1. 实验结果

数据通路:

微指令设计表：

微地址入口转移逻辑设计：

CU设计电路：

转移逻辑电路：

运行结果：

1. 实验分析及反馈

通过本次实验，对微指令CPU的运行机制有了更加深刻的了解，体会到了寄存器在多周期CPU中扮演的保存前态的重要功能，对最基本的MIPS指令的运行有了初步的了解，体会到了时序性在CPU工作中的重要性，以及数据通路的设计是如此重要。

课程设计

1. 实验目的

深入理解和巩固计算机组成原理课程讲授的计算机部件工作原理及设计的相关知识，加深对计算机各功能模块协同工作的认识，清晰地建立计算机的整机概念，获取系统设计与调试的实践经验，提高学生对工程观、系统观的认知，培养学生独立学习、工作和创新思维的能力。

实现简单计算机系统的模型机，统一设计8条MIPS核心指令lw/sw/add/slt/addi/beq/bne/syscal，自行设计R类型、I类型、J类型指令各一条，编写程序，生成8个整数，分别实现排序、找最大数及最小数、求平均值程序验证指令及CPU。

1. 实验环境

WIN10、Logisim2.7.1

1. 实验步骤、过程

（1）了解MIPS指令，并实验中使用的指令进行相关查询和学习：

J型指令：J

I型指令：LW、SW、BEQ、BNE、ADDI、LUI

R指令：SLT、ADD、SRL

特殊指令：SYSCALL

统一设计指令：

LW 从存储器中读取一个字的数据到寄存器中

SW 把一个字的数据从寄存器存储到存储器中

beq: 数据跳转指令，标志寄存器中Z标志位等于零时, 跳转到BEQ后标签处

bne: 数据跳转指令，标志寄存器中Z标志位不等于零时, 跳转到BNE后标签处

SLT 如果Rs的值小于Rt，那么设置Rd的值为1，否则设置Rd的值为0

ADDI 把一个Rs寄存器的内容加上一个立即数存入Rt寄存器

ADD 把一个Rs寄存器和Rt寄存器内容相加，结果存入Rd寄存器

Syscall 停机指令

新增指令：

LUI 16位立即数后面补0后存储到rt（由于Logisim平台中拓展器的原因，这里实际上是16位立即数的符号拓展，所以实际效果是伪指令LI的存储立即数到rt中）

SRL rt寄存器中的值算数右移shamt位后存储到rd存储器中

J 无条件跳转到标签处

（2）查询MIPS32指令手册和MIPS GREEN SHEET，仔细分析每条指令的执行过程，设计数据通路

（3）由于本实验使用微程序产生信号代替了组合逻辑电路产生信号，所以这里我们类似之前微程序CU设计实验方式，设计微程序。

（4）根据设计的微程序，填写硬布线 控制器状态转换逻辑自动生成表格，生成硬布线控制器的状态机FSM内部电路逻辑表达式，使用电路分析功能生成电路。此时发现Logisim平台不支持12个以上的电路输入，SRL可以归为R指令不用增加输入，最后选择增加LUI指令、舍弃J指令。

（5）由于新增指令原因，所以我们也要对CU内部的译码指令逻辑、ALU控制器逻

辑、和信号产生序列进行相应调整

（6）由于信号产生序列的调整，外部封装接口改变，需要对数据通路再次修改。

（7）根据要求使用Mars汇编设计相应的汇编程序，载入测试。

4、实验结果

5、分析及反馈 000000000