

# 实验报告

课 程	计算机组成与系统结构	姓 名	王磊	学 号	202231060435
指导教师	徐媛媛	专业班级	计科 2202	成 绩	

## 实验四 系统总线与总线接口实验

### 一、实验目的

1. 理解总线的概念及其特性。
2. 掌握控制总线的功能和应用。

### 二、实验设备

PC 机一台，Logism 实验系统一套。

### 三、实验原理及内容

总线是计算机中连接各个功能部件的纽带，是计算机各部件之间进行信息传输的公共通路。总线不只是一组简单的信号传输线，它还是一组协议。分时与共享是总线的两大特征。所谓共享，在总线上可以挂接多个部件，它们都可以使用这一信息通路来和其他部件传送信息。所谓分时，同一总线在同一时刻，只能有一个部件占领总线发送信息，其他部件要发送信息得在该部件发送完释放总线后才能申请使用。总线结构是决定计算机性能、功能、可扩展性和标准化程度的重要因素。

#### 1. 中断接口

本实验在简单模型计算机的基础上，还需要在单总线数据通路中增加与中断相关的硬件模块，图 4-1 为中断接口原理图，主要包括异常程序地址计数器 EPC，中断使能寄存器 IE，中断控制器等模块，需要在主电路中将这模块进行有效连接，并进行最终的联调，测试 CPU 是否能正常响应 2 个按键对应的中断服务程序。

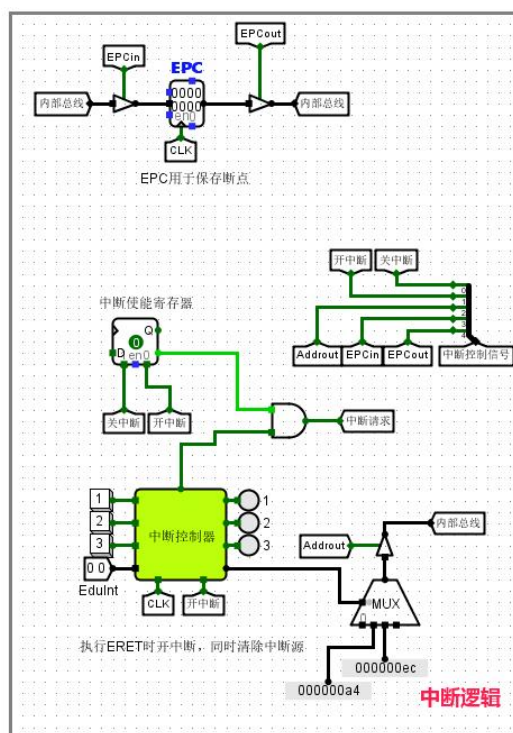


图 4-1 中断接口原理图

2. 中断控制器原理

本中断控制器可处理 3 个中断源的中断请求，3 个中断源分别是按键 Key1、Key2 和 Key3。该中断控制器中的优先级编码器会对 3 个中断源进行判优，输出当前中断请求中号码最大的那个中断源的编码 IntNo.，并提出中断请求 IntR。各个中断源提出中断请求，对应号码的中断请求指示灯会相应点亮，当中断请求提出后，会自动熄灭中断请求指示灯。

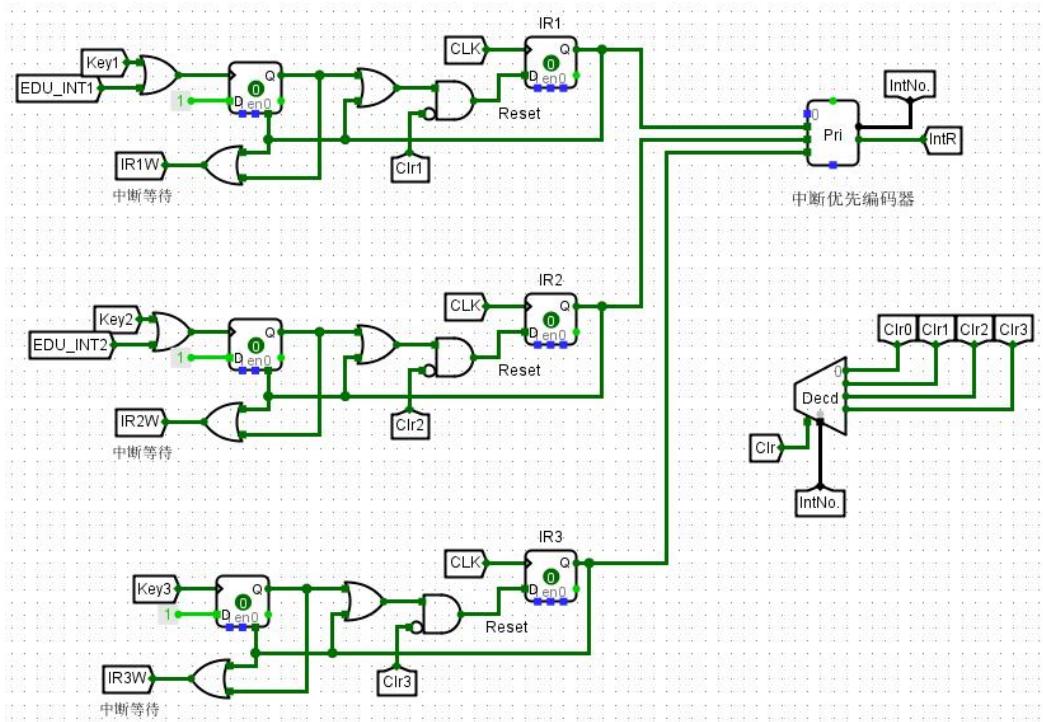


图 4-2 中断控制器原理图

3. 微指令修改

由于增加了中断控制逻辑，需要修改模型机的微指令格式，新增与中断有关的控制字段，包括：STI（开中断）、CLI（关中断）、EPCin（断点保存）、EPCout（断点输出）、Addrout（输出中断服务程序地址）和 P<sub>INTR</sub>（是否有中断请求）字段，模型机微指令修改为图 4-3 所示格式。

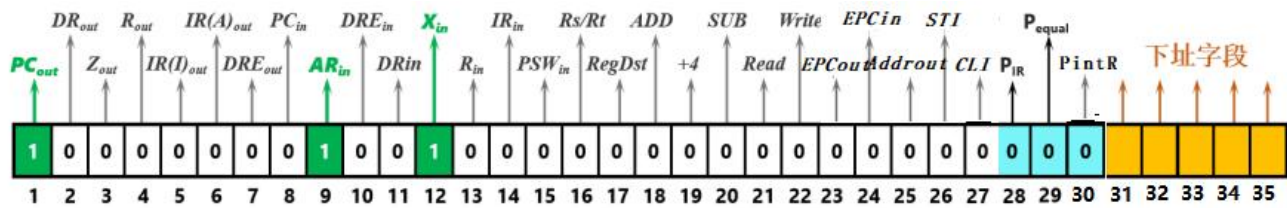


图 4-3 微指令格式（增加中断）

4. 微程序修改

由于增加了中断控制逻辑，需在中断服务程序中增加中断返回的机器指令 erect，在微程序流程图中增加中断响应和 erect 指令分支。修改后的微程序流程图如图 4-4 所示。

修改微程序流程图，截图置于此处

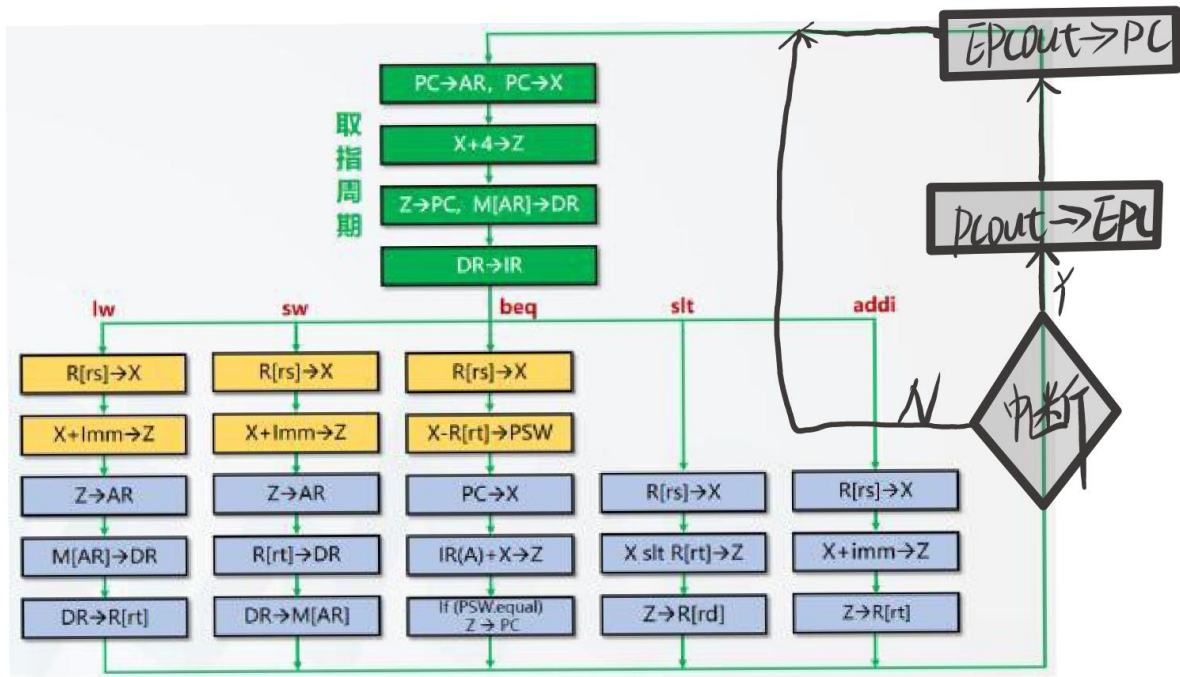


图 4-4 微程序流程图（增加中断）

在原模型机微程序基础上增加中断响应和中断返回 erect 指令微程序，修改后的微程序如图 4-5 所示：  
截取微程序（支持中断）excel 表格图置于此处

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AI				AJ					
1	微指令功能	PCout	DRout	Zout	Rout	IRout	PCin	ARin	DRin	Xin	Rin	IRin	PCin	ARin	DRin	Xin	Rin	IRin	PCin	ARin	DRin	Xin	Rin	IRin	PCin	ARin	DRin	Xin	Rin	IRin	PCin	ARin	DRin	Xin	Rin	IRin	PCin	ARin	DRin	Xin	Rin	IRin
2	取指令	0	1								1			1																					10000000100100000000000000000000					20240000		
3	取指令	1																					1												00000000000000000000000000000000					800		
4	取指令	2				1					1		1											1											00100001010000000000000000000000					8500200		
5	取指令	3		1														1														1			01000000000000000000000000000000					10010004		
6	lw	4				1											1																	00010000000100000000000000000000					4040000			
7	lw	5					1																1											00001000000000000000000000000000					2001000			
8	lw	6																																00100000100000000000000000000000					8200000			
9	lw	7				1										1																		00000000010000000000000000000000					100200			
10	lw	8					1																										1		10000000000000000000000000000000					10020001		
11	sw	9																																	00010000000100000000000000000000					4040000		
12	sw	10																																	00001000000000000000000000000000					2001000		
13	sw	11					1																												00100000010000000000000000000000					8200000		
14	sw	12																																	00010000000100000000000000000000					4084000		
15	sw	13																																	10000000000000000000000000000000					800101		
16	beq	14																																	00010000000100000000000000000000					4040000		
17	beq	15																																	10000000000000000000000000000000					400C003		
18	beq	16																																	10000000000000000000000000000000					20040000		
19	beq	17																																	00000100000000000000000000000000					1001000		
20	beq	18																																	10100001000000000000000000000000					8400001		
21	slt	19																																	00010000000100000000000000000000					4040000		
22	slt	20																																	00010000000000000000000000000000					4004400		
23	slt	21																																	10100000000000000000000000000000					8022001		
24	addi	22																																	00010000000100000000000000000000					4040000		
25	addi	23																																	00001000000000000000000000000000					2001000		
26	addi	24																																	10100000000000000000000000000000					8020001		
27	eret	25																																	10000001000000000000000000000000					400091		
28	中断	26																																	10000000000000000000000000000000					20000048		
29	中断	27																																	10000001000000000000000000000000					400021		

图 4-5 模型机微程序（增加中断）

## 5. 实验结果分析

在控制存储器中载入设计好的微程序，在主存 MEM 中载入 sort-5-int.hex 文件，运行程序，观察结果。分别按下 key1 和 key2，再观察结果。



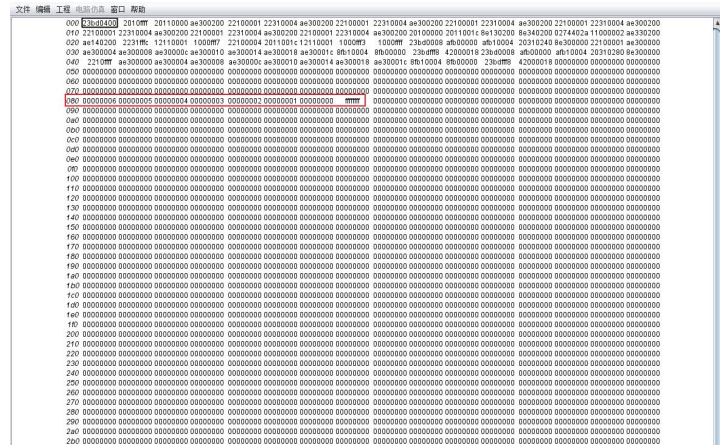


图 4-6 实验结果 1

图 4-6 为运行 sort-5-int.hex 的实验结果，结果表明程序对数据进行了降序排序

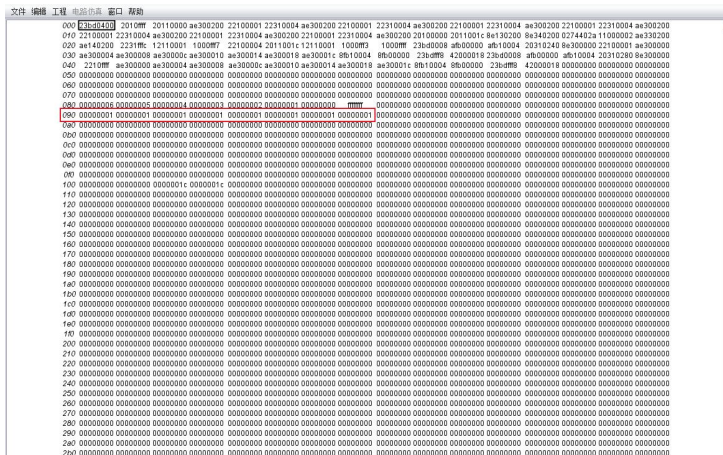


图 4-7 实验结果 2

图 4-7 为按下 key1 后的实验结果，结果表明按下 key1 后，从地址 0x90 起始的连续 8 个字节的数据将各自增加 1。

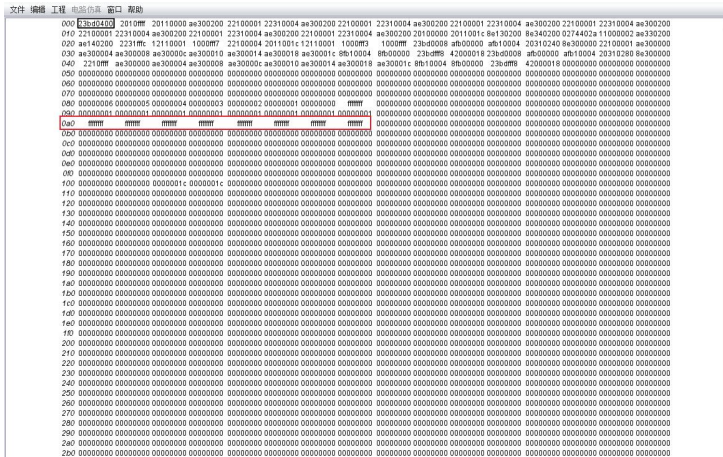


图 4-8 实验结果 3

图 4-8 为按下 key2 后的实验结果，结果表明按下 key2 后，从地址 0xa0 起始的连续 8 个字节的数据将各自增加-1。

#### 四、思考题

请为 key3 实现中断控制功能。

1. 请问中断控制逻辑图中哪些地方需要修改，如何修改？

修改后截图并配文字说明

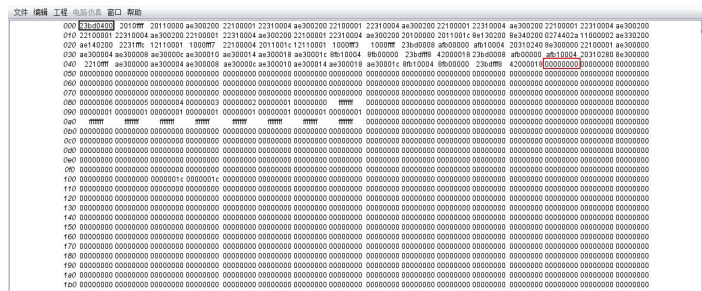


图 4-9 地址 0x4d

通过计算该位置的地址信息，将 0x4d 转换成二进制形式得到 1001101。由于在读取过程中仅读 2~11 位，所以完整的地址信息为 100110100。将其转换为十六进制，结果为 134。在下图中提供常量 0x134，并且连接到选择 key3 的引脚上。

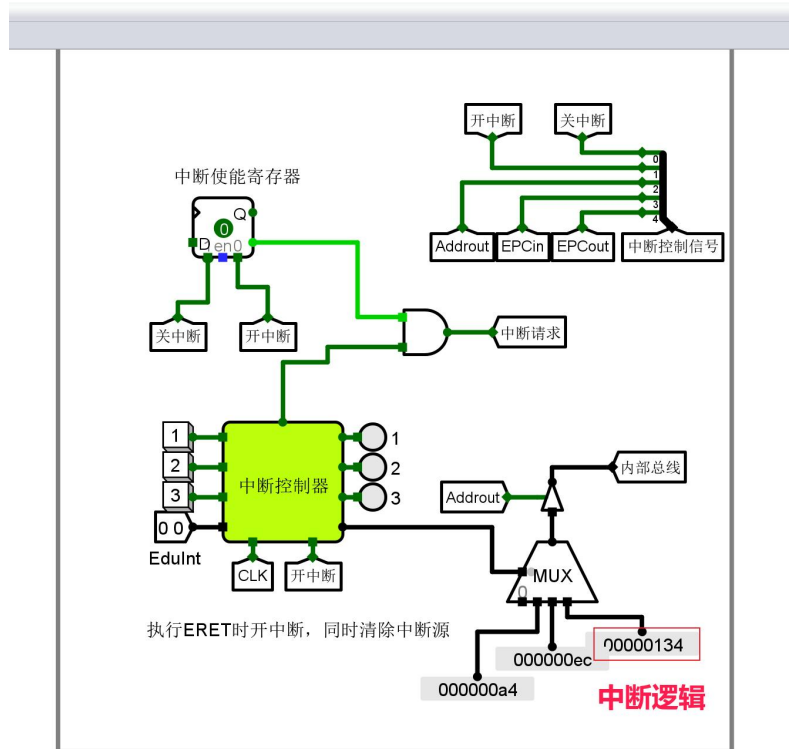


图 4-10 0x134 连接到选择 key3 的引脚上

2. 为 key3 编写中断服务程序，该中断服务程序的功能是：将本人学号的前 4 位、中间 4 位和后 4 位分别写入三个不同的寄存器中。例如，某生学号为 202131060101，选择 \$20、\$21 和 \$22 寄存器分别存放学号的前 4 位、中间 3 位和最后 4 位，按下 key3，响应中断，能观察到 \$20、\$21 和 \$22 寄存器的值变为 0x00002021、0x00003106 以及 0x00000101。

#### (1) 编写程序

设计思路：使用 addi 指令进行赋值，之后利用 erect 指令返回。

实际操作：

1. \$20 赋初值为 2022: 001000 00000 10100 00000111111001110 即 0x201407e6
2. \$21 赋初值为 3106: 001000 00000 10101 0000110000100010 即 0x20150c22
3. \$22 赋初值为 0435: 001000 00000 10110 0000000110110011 即 0x201601b3
4. erect 参考已有程序 0x42000018

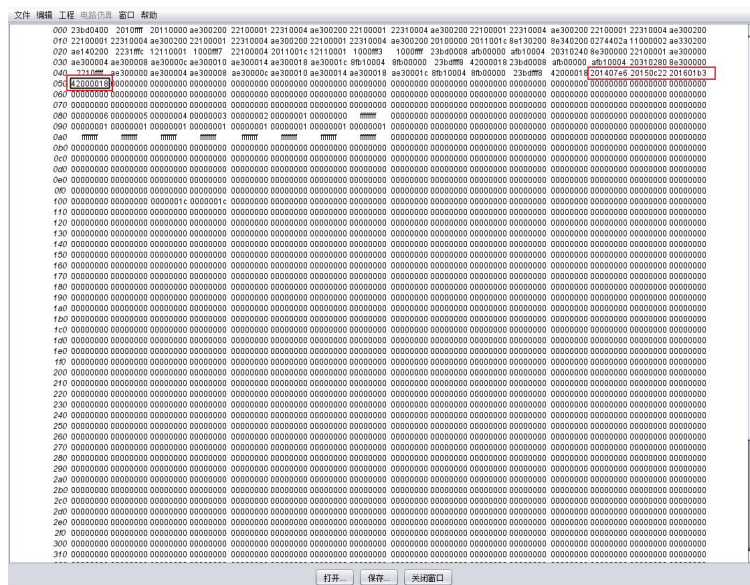


图 4-11 编写程序

## (2) 实验结果及分析

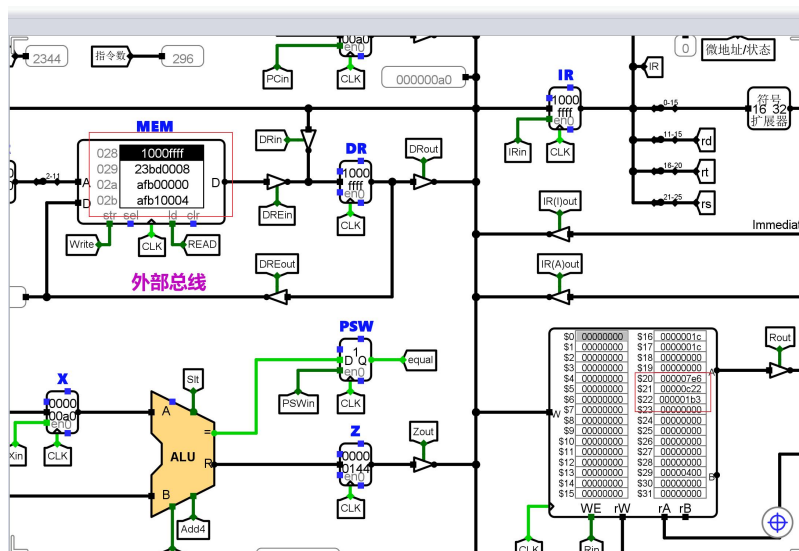


图 4-12 实验结果

由图可知，\$20 被赋值为 0x7e6 即 2022，\$21 被赋值为 0xc22 即 3106，\$22 被赋值为 0x1b3 即 0435，在主存中可以看到指令正确中断返回，实验结果正确，