

计算机组成与系统结构

课程设计报告

广州大学 计算机科学与网络工程学院
软件工程专业 17 级 xx 班

NAME

(学号:1706xxxx)

2019 年 6 月

一.课设性质，目的，任务

《计算机组成与系统结构课程设计》是计算机学院各专业集中实践性环节之一，是学习完《计算机组成与系统结构》课程后进行的一次全面的综合练习。其目的是综合运用所学计算机原理知识，设计并实现一台模型计算机，以便巩固所学的知识，提高分析问题和解决问题的能力。

二.课设基本理论

- 1、掌握算术、逻辑、移位运算实验，熟悉 ALU 运算控制位的运用。
- 2、掌握存储器组织、读写方式及与总路线组成的数据通路，掌握地址总线、数据总线的工作原理。
- 3、掌握指令结构和指令取指、执行工作过程。
- 4、掌握 CPU 的微程序控制原理。

三.题目

综合运用所学计算机原理知识, 设计并实现具有以下 16 条指令的指令集结构的模型计算机:

编号	助记符	机器指令码	说明
0	SUB Rd,Rs	0000 RdRs	Rd-Rs→Rd
1	ADD Rd,Rs	0001 RdRs	Rd+Rs→Rd
2	AND Rd,Rs	0010 RdRs	Rd&Rs→Rd (Rd 和 Rs 相与)
3	DEC Rd	0011 Rd00	将 Rd 值减 1
4	CLR Rd	0100 Rd00	将 Rd 清零
5	RL Rd	0101 Rd00	Rd 循环左移一位
6	RR Rd	0110 Rd00	Rd 循环右移一位
7	MOV Rd,Rs	0111 RdRs	Rs→Rd
8	LDI Rd,*	1000 Rd00 XXXXXXXX	将指令中的立即数 (第二字节) 送入 Rd
9	OUT IOH,Rs	1001 00Rs	Rs→I/O(数据开关)高字节
10	LDA Rd,M	1010 Rd00 XXXXXXXX XXXXXXXX	[M] →Rd
11	STA M,Rs	1011 00Rs XXXXXXXX XXXXXXXX	Rs→[M]
12	JMP M	1100 0000 XXXXXXXX XXXXXXXX	[M]→PC,即跳转到 M 所指单元
13	JZ M	1101 0000 XXXXXXXX XXXXXXXX	当 Z=1 时, 跳转到 M 所指单元
14	JC M	1110 0000 XXXXXXXX XXXXXXXX	当 CY=1 时, 跳转到 M 所指单元
15	HALT	1111 0000	停机

设计提示：

1、上表中，机器指令码的高 4 位为指令操作码，M 为 16 位存储器地址，Rs 为源寄存器，Rd 为目的寄存器，占 2 位，并规定：

Rs 或 Rd	选定的寄存器
00	R0
01	R1
10	R2
11	R3

2、在微程序中，微地址 001 为取指，微指令为 BF FB F8。

3、各指令指令行阶段微程序入口地址的确定方式：

微地址位号	10	9	8	7	6	5	4	3	2	1	0
内容	1	1	IR7~IR4				0	0	0	0	0

例如，第 5 条指令“RL Rd” 的指令码为 0101 Rd00 则指令码的高 4 位 IR7~IR4 为 0101，由上表知，微程序入口微地址为： 11 **0101** 00000，即 6A0H。

4、主要步骤：

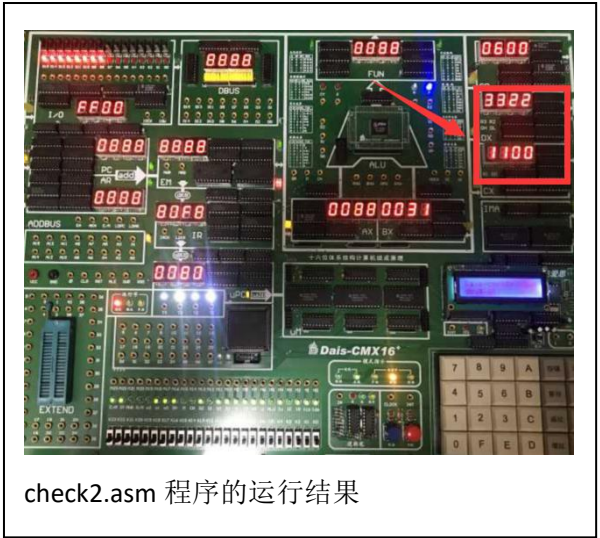
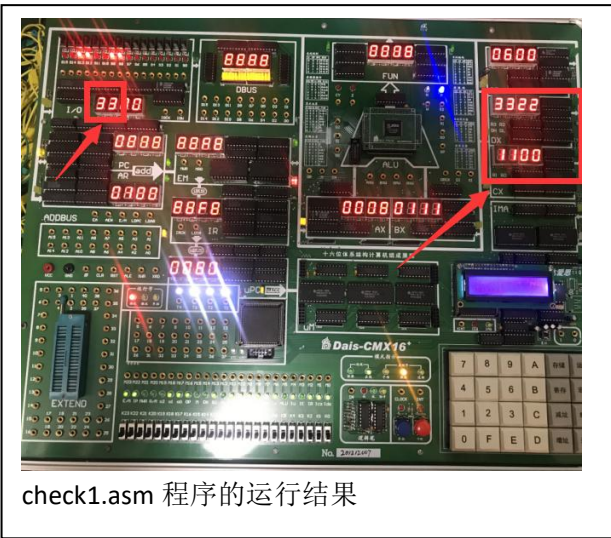
- (1) 按照第 3 点的方法，给出所有 16 条指令的微程序入口微地址；
- (2) 通过分析每条指令的功能明确其的微程序流程，可参考实验指导书图 3-4-1、图 3-2-2、图 3-3-1；
- (3) 写出每条微指令的微命令，即 24 个微控制位信号，可参考实验指导书表 3.4.1、表 3.2.1、表 3.3.1 和“微控制器编程手册”第 2 章。
- (4) 建议采用逐条指令设计实现的方式，一条实现并用汇编语句测试通过后（指令功能、下址顺序均正确）再进行下一条的设计。

5、检查

模型计算机设计完成后，用所给的测试程序 check_1.asm（测 12 条非转移指令）和 check_2.asm（测 3 条转移指令）检查正确性。检查方法：在测试程序中#load s 本人的.IS 微指令程序，实验箱电源关闭重启并连接，装载后选择“运行”或“单步”执行。

check_1.asm 运行的正确结果为：寄存器 R0R1R2R3 分别显示 00112233，IOH 显示 33。

check_2.asm 运行的正确结果为：寄存器 R0R1R2R3 分别显示 00112233，如果显示 EE 则执行有错误。



四 . 数据格式定义

编号	助记符	机器指令码	入口地址	对应微程序
1	SUB Rd, Rs	0000 RdRs	600	600:F8 F9 7F
				601:FA C1 FF
				602:F0 6E 6D
2	ADD Rd, Rs	0001 RdRs	620	620:F8 F9 7F
				621:FA C1 FF
				622:F0 66 6D
3	AND Rd, Rs	0010 RdRs	640	640:F8 F9 7F
				641:FA C1 FF
				642:F0 EE 6D
4	DEC Rd	0011 Rd00	660	660:F8 F9 5F
				661:F0 D6 4D
5	CLR Rd	0100 Rd00	680	680:F0 DE 6D
6	RL Rd	0101 Rd00	6A0	6A0:F0 D6 4D
				6A1:F0 76 4D
7	RR Rd	0110 Rd00	6C0	6C0:F8 F9 5F
				6C1:F0 7E 4D
8	MOV Rd, Rs	0111 RdRs	6E0	6E0:F0 F9 ED
9	LDI Rd, *	1000 Rd00 XXXXXXXX	700	700:FA FB FF
				701:B0 C6 6D
10	OUT IOH, Rs	1001 00Rs	720	720:F5 F9 ED
11	LDA Rd, M	1010 Rd00 XXXXXXXX XXXXXXXX	740	740:F8 FB FF
				741:B9 FB FF
				742:BC FE 3F
				743:70 FB ED
12	STA M, Rs	1011 00Rs XXXXXXXX XXXXXXXX	760	760:FA FB 7F
				761:BB FB FF
				762:BC C6 3F
				763:5F F9 AD
13	JMP M	1100 0000 XXXXXXXX XXXXXXXX	780	780:F8 FB FF
				781:B9 FB FF
				782:3F FE 2D
14	JZ M	1101 0000 XXXXXXXX XXXXXXXX	7A0	7A0:FA C3 FF
				7A1:BB FB FC
				7A4:FF FF ED
				7A5:3F C6 2D
15	JC M	1110 000 XXXXXXXX XXXXXXXX	7C0	7C0:FA FB FF
				7C1:BB FB FD

				7C4:FF FF ED
				7C5:3F C6 2D
16	HALT	1111 0000	7E0	7E0:F8 F8 3F
				7E1:3F D6 0D

五．模型机指令系统

助记符	操作数	指令码	长度	
;-----				
;1、减法运算：600h				
SUB	R0,R0	00	1	;R0-R0->R0
SUB	R0,R1	01	1	;R0-R1->R0
SUB	R0,R2	02	1	;R0-R2->R0
SUB	R0,R3	03	1	;R0-R3->R0
SUB	R1,R0	04	1	;R1-R0->R1
SUB	R1,R1	05	1	;R1-R1->R1
SUB	R1,R2	06	1	;R1-R2->R1
SUB	R1,R3	07	1	;R1-R3->R1
SUB	R2,R0	08	1	;R2-R0->R2
SUB	R2,R1	09	1	;R2-R1->R2
SUB	R2,R2	0A	1	;R2-R2->R2
SUB	R2,R3	0B	1	;R2-R3->R2
SUB	R3,R0	0C	1	;R3-R0->R3
SUB	R3,R1	0D	1	;R3-R1->R3
SUB	R3,R2	0E	1	;R3-R2->R3
SUB	R3,R3	0F	1	;R3-R3->R3
;2、加法运算：620h				
ADD	R0,R0	10	1	;R0+R0->R0
ADD	R0,R1	11	1	;R0+R1->R0
ADD	R0,R2	12	1	;R0+R2->R0
ADD	R0,R3	13	1	;R0+R3->R0
ADD	R1,R0	14	1	;R1+R0->R1
ADD	R1,R1	15	1	;R1+R1->R1
ADD	R1,R2	16	1	;R1+R2->R1
ADD	R1,R3	17	1	;R1+R3->R1
ADD	R2,R0	18	1	;R2+R0->R2
ADD	R2,R1	19	1	;R2+R1->R2
ADD	R2,R2	1A	1	;R2+R2->R2
ADD	R2,R3	1B	1	;R2+R3->R2
ADD	R3,R0	1C	1	;R3+R0->R3
ADD	R3,R1	1D	1	;R3+R1->R3

ADD	R3,R2	1E	1	;R3+R2->R3
ADD	R3,R3	1F	1	;R3+R3->R3

;3、逻辑与运算： 640h

AND	R0,R0	20	1	;R0&R0->R0
AND	R0,R1	21	1	;R0&R1->R0
AND	R0,R2	22	1	;R0&R2->R0
AND	R0,R3	23	1	;R0&R3->R0
AND	R1,R0	24	1	;R1&R0->R1
AND	R1,R1	25	1	;R1&R1->R1
AND	R1,R2	26	1	;R1&R2->R1
AND	R1,R3	27	1	;R1&R3->R1
AND	R2,R0	28	1	;R2&R0->R2
AND	R2,R1	29	1	;R2&R1->R2
AND	R2,R2	2A	1	;R2&R2->R2
AND	R2,R3	2B	1	;R2&R3->R2
AND	R3,R0	2C	1	;R3&R0->R3
AND	R3,R1	2D	1	;R3&R1->R3
AND	R3,R2	2E	1	;R3&R2->R3
AND	R3,R3	2F	1	;R3&R3->R3

;4、Rd 值减 1:660

DEC	R0	30	1	;R0-1
DEC	R1	34	1	;R1-1
DEC	R2	38	1	;R2-1
DEC	R3	3C	1	;R3-1

;5、Rd 清空： 680

CLR	R0	40	1	;R0 清空
CLR	R1	44	1	;R1 清空
CLR	R2	48	1	;R2 清空
CLR	R3	4C	1	;R3 清空

;6、Rd 循环左移一位： 6A0

RL	R0	50	1	;R0 循环左移一位
RL	R1	54	1	;R1 循环左移一位
RL	R2	58	1	;R2 循环左移一位
RL	R3	5C	1	;R3 循环左移一位

;7、Rd 循环右移一位： 6C0

RR	R0	60	1	;R0 循环右移一位
RR	R1	64	1	;R1 循环右移一位
RR	R2	68	1	;R2 循环右移一位
RR	R3	6C	1	;R3 循环右移一位

;8、将 Rs 移到 Rd: 6E0

MOV	R0,R0	70	1	;R0->R0
MOV	R0,R1	71	1	;R1->R0
MOV	R0,R2	72	1	;R2->R0
MOV	R0,R3	73	1	;R3->R0
MOV	R1,R0	74	1	;R0->R1
MOV	R1,R1	75	1	;R1->R1
MOV	R1,R2	76	1	;R2->R1
MOV	R1,R3	77	1	;R3->R1
MOV	R2,R0	78	1	;R0->R2
MOV	R2,R1	79	1	;R1->R2
MOV	R2,R2	7A	1	;R2->R2
MOV	R2,R3	7B	1	;R3->R2
MOV	R3,R0	7C	1	;R0->R3
MOV	R3,R1	7D	1	;R1->R3
MOV	R3,R2	7E	1	;R2->R3
MOV	R3,R3	7F	1	;R3->R3

;9、将指令中的立即数送入 Rd: 700

LDI	R0,*	80	2	;将指令中的立即数（第二字节）送入 R0
LDI	R1,*	84	2	;将指令中的立即数（第二字节）送入 R1
LDI	R2,*	88	2	;将指令中的立即数（第二字节）送入 R2
LDI	R3,*	8C	2	;将指令中的立即数（第二字节）送入 R3

;10、将 Rs 送往 IOH: 720

OUT	IOH,R0	90	1	;将寄存器 R0 的数据写入到 IOH
OUT	IOH,R1	91	1	;将寄存器 R1 的数据写入到 IOH
OUT	IOH,R2	92	1	;将寄存器 R2 的数据写入到 IOH
OUT	IOH,R3	93	1	;将寄存器 R3 的数据写入到 IOH

;11、[M] → Rd: 740

LDA	R0,*	A0	3	;[M]->R0
LDA	R1,*	A4	3	;[M]->R1
LDA	R2,*	A8	3	;[M]->R2
LDA	R3,*	AC	3	;[M]->R3

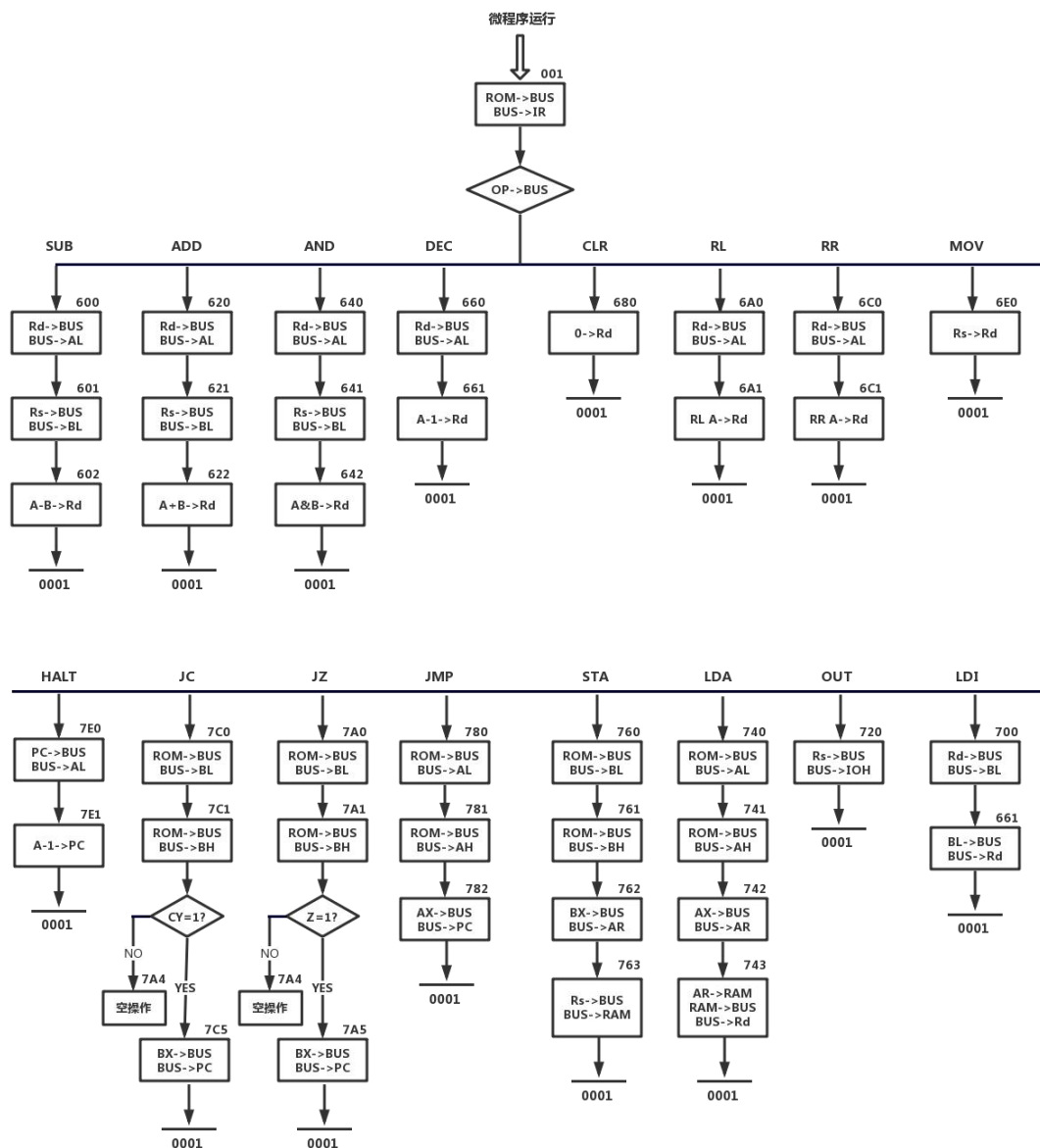
;12、Rs→[M]: 760

STA	*,R0	B0	3	;R0->[M]
STA	*,R1	B1	3	;R1->[M]
STA	*,R2	B2	3	;R2->[M]
STA	*,R3	B3	3	;R3->[M]

;13、跳转到 M 所指单元: 780

JMP	*	C0	3	;[M]→PC,即跳转到 M 所指单元
;14、跳转到 M 所指单元: 7A0				
JZ	*	D0	3	;当 Z=1 时, 跳转到 M 所指单元
;15、跳转到 M 所指单元: 7C0				
JC	*	E0	3	;当 CY=1 时, 跳转到 M 所指单元
;16、停机: 7E0				
HALT	""	F0	1	;停机

六 . 微程序流程图



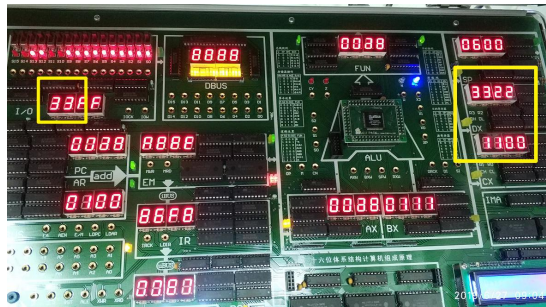
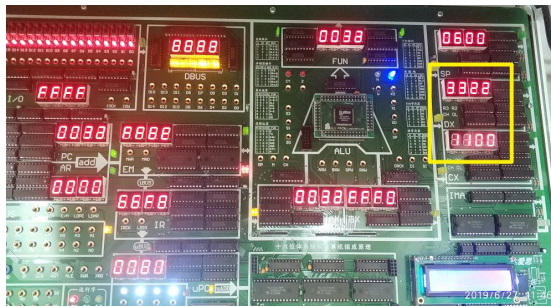
七．模型机微指令表

微址	M23	M22	M21	M20	M19	M18	M17	M16	代码	M15	M14	M13	M12	M11	M10	M9	M8	代码	M7	M6	M5	M4	M3	M2	M1	M0	代码	后续微址	说明
	E/M	IP	MWR	R/M	o2	o1	00	0P		M	CN	S2	S2	S0	X2	X1	X0		XP	W	ALU	lu	IE	IR	lcz	lds			
0001	1	0	1	1	1	1	1	1	BF	1	1	1	1	1	0	1	1	FB	1	1	1	1	1	0	0	0	F8	+1	取指变值
0600	1	1	1	1	1	0	0	0	F8	1	1	1	1	1	0	0	1	F9	0	1	1	1	1	1	1	1	7F	+1	AX=REG
0601	1	1	1	1	1	0	1	0	FA	1	1	0	0	0	0	0	1	C1	1	1	1	1	1	1	1	1	FF	+1	BX=REG
0602	1	1	1	1	0	0	0	0	F0	0	1	1	0	1	1	1	0	6E	0	1	1	0	1	1	0	1	6D	0001	REG=A-B
0620	1	1	1	1	1	0	0	0	F8	1	1	1	1	1	0	0	1	F9	0	1	1	1	1	1	1	1	7F	+1	AX=REG
0621	1	1	1	1	1	0	1	0	FA	1	1	0	0	0	0	0	1	C1	1	1	1	1	1	1	1	1	FF	+1	BX=REG
0622	1	1	1	1	0	0	0	0	F0	0	1	1	0	0	1	1	0	66	0	1	1	0	1	1	0	1	6D	0001	REG=A+B
0640	1	1	1	1	1	0	0	0	F8	1	1	1	1	1	0	0	1	F9	0	1	1	1	1	1	1	1	7F	+1	AX=REG
0641	1	1	1	1	1	0	1	0	FA	1	1	0	0	0	0	0	1	C1	1	1	1	1	1	1	1	1	FF	+1	BX=REG
0642	1	1	1	1	0	0	0	0	F0	1	1	1	0	1	1	1	0	EE	0	1	1	0	1	1	0	1	6D	0001	REG=A&B
0660	1	1	1	1	1	0	0	0	F8	1	1	1	1	1	0	0	1	F9	0	1	0	1	1	1	1	1	5F	+1	TmpA=REG
0661	1	1	1	1	0	0	0	0	F0	1	1	0	1	0	1	1	0	D6	0	1	0	0	1	1	0	1	4D	+1	REG=A-1
0680	1	1	1	1	0	0	0	0	F0	1	1	0	1	1	1	1	0	DE	0	1	1	0	1	1	0	1	6D	0001	A 清零
06A0	1	1	1	1	1	0	0	0	F8	1	1	1	1	1	0	0	1	F9	0	1	0	1	1	1	1	1	5F	+1	TmpA=REG
06A1	1	1	1	1	0	0	0	0	F0	0	1	1	1	0	1	1	0	76	0	1	0	0	1	1	0	1	4D	0001	左移
06C0	1	1	1	1	1	0	0	0	F8	1	1	1	1	1	0	0	1	F9	0	1	0	1	1	1	1	1	5F	+1	TmpA=REG
06C1	1	1	1	1	0	0	0	0	F0	0	1	1	1	1	1	1	0	7E	0	1	0	0	1	1	0	1	4D	0001	右移
06E0	1	1	1	1	0	0	0	0	F0	1	1	1	1	1	0	0	1	F9	1	1	1	0	1	1	0	1	ED	0001	Rs->Rd
0700	1	1	1	1	1	0	1	0	FA	1	1	1	1	1	0	1	1	FB	1	1	1	1	1	1	1	1	FF	+1	BX=ROM
0701	1	0	1	1	0	0	0	0	B0	1	1	0	0	0	1	1	0	C6	0	1	1	0	1	1	0	1	6D	0001	REG=B PC++
0720	1	1	1	1	0	1	0	1	F5	1	1	1	1	1	0	0	1	F9	1	1	1	0	1	1	0	1	ED	0001	Rs→I/O
0740	1	1	1	1	1	0	0	0	F8	1	1	1	1	1	0	1	1	FB	1	1	1	1	1	1	1	1	FF	+1	AL=ROM
0741	1	0	1	1	1	0	0	1	B9	1	1	1	1	1	0	1	1	FB	1	1	1	1	1	1	1	1	FF	+1	AH=ROM PC++
0742	1	0	1	1	1	1	0	0	BC	1	1	1	1	1	1	1	0	FE	0	0	1	1	1	1	1	1	3F	+1	AR=A PC++
0743	0	1	1	1	0	0	0	0	70	1	1	1	1	1	0	1	1	FB	1	1	1	0	1	1	0	1	ED	0001	[M] →Rd
0760	1	1	1	1	1	0	1	0	FA	1	1	1	1	1	0	1	1	FB	0	1	1	1	1	1	1	1	7F	+1	BL=ROM
0761	1	0	1	1	1	0	1	1	BB	1	1	1	1	1	0	1	1	FB	1	1	1	1	1	1	1	1	FF	+1	BH=ROM PC++
0762	1	0	1	1	1	1	0	0	BC	1	1	0	0	0	1	1	0	C6	0	0	1	1	1	1	1	1	3F	+1	AR=B PC++
0763	0	1	0	1	1	1	1	1	5F	1	1	1	1	1	0	0	1	F9	1	0	1	0	1	1	0	1	AD	0001	Rs→[M]
0780	1	1	1	1	1	0	0	0	F8	1	1	1	1	1	0	1	1	FB	1	1	1	1	1	1	1	1	FF	+1	AL=ROM

0781	1	0	1	1	1	0	0	1	B9	1	1	1	1	1	0	1	1	FB	1	1	1	1	1	1	1	1	FF	+1	AH=ROM PC++
0782	0	0	1	1	1	1	1	1	3F	1	1	1	1	1	1	1	0	FE	0	0	1	0	1	1	0	1	2D	0001	[M]→PC
07A0	1	1	1	1	1	0	1	0	FA	1	1	1	1	1	0	1	1	FB	0	1	1	1	1	1	1	1	7F	+1	BL=ROM
07A1	1	0	1	1	1	0	1	1	BB	1	1	1	1	1	0	1	1	FB	1	1	1	1	1	1	0	0	FC	07A4	BH=ROM PC++
07A4	1	1	1	1	1	1	1	1	FF	1	1	1	1	1	1	1	1	FF	1	1	1	0	1	1	0	1	ED	0001	空操作
07A5	0	0	1	1	1	1	1	1	3F	1	1	0	0	0	1	1	0	C6	0	0	1	0	1	1	0	1	2D	0001	跳到 M
07C0	1	1	1	1	1	0	1	0	FA	1	1	1	1	1	0	1	1	FB	0	1	1	1	1	1	1	1	7F	+1	BL=ROM
07C1	1	0	1	1	1	0	1	1	BB	1	1	1	1	1	0	1	1	FB	1	1	1	1	1	1	0	1	FD	07C4	BH=ROM PC++
07C4	1	1	1	1	1	1	1	1	FF	1	1	1	1	1	1	1	1	FF	1	1	1	0	1	1	0	1	ED	0001	空操作
07C5	0	0	1	1	1	1	1	1	3F	1	1	0	0	0	1	1	0	C6	0	0	1	0	1	1	0	1	2D	0001	跳到 M
07E0	1	1	1	1	1	0	0	0	F8	1	1	1	1	1	0	0	0	F8	0	0	1	1	1	1	1	1	3F	+1	PC→AX
07E1	0	0	1	1	1	1	1	1	3F	1	1	0	1	0	1	1	0	D6	0	0	0	0	1	1	0	1	0D	0001	AX-1→PC

八．检测结果

check_1 程序	check_2 程序
<pre> #LOAD "XZ.IS" org 0 start: LDI r0,12h sta 100h,r0 lda r1,100h dec r1 rl r0 add r0,r1 rr r0 add r0,r1 ldi r2,76h and r2,r0 mov r3,r2 add r3,r1 clr r0 out ioh,r3 halt end </pre>	<pre> #LOAD "XZ.IS" org 0 start: LDI r0,12h sub r0,r0 jz tag1 ldi r3,0eeh jmp tag2 tag1:ldi r3,33h tag2:ldi r0,88h add r0,r0 jc tag3 ldi r2,0eeh jmp tag4 tag3:ldi r2,22h tag4:ldi r1,11h ldi r0,0 add r1,r0 jz tag5 jmp tag6 tag5:ldi r1,0eeh tag6:jc tag7 </pre>

	<pre>ldi r0,0h jmp tag8 tag7:ldi r0,0eeh tag8:halt end</pre>
	

九．存在的问题及体会

通过本次课程设计，我对计算机系统结构和指令系统的创建有了进一步的认识，同时对程序运行过程有了更深入的理解，相信对以后的编程会有一定帮助。

实验的关键在于要理解 24 个微控制位 M0~M23 对应的功能。新建指令首先要计算入口地址，同时将指令系统里的助记符写好，计算指令码。对于一些需要用到两个通用寄存器的指令，会有对应 16 (2^4) 种助记符。接着分析指令的作用，根据总线规则从原编码取数据送往目标编码地址。

在完成课程设计过程中会遇到很多问题，例如：

1. 在微单步微指令时，指令经常不按照预期转跳。其中的一个原因是没有处理好条件变址、结尾变址等情况。
2. 在存取数据时会出现不能正确地把数据送往指定寄存器的问题，原因在于没有选对总线规则，其中 Rs 寄存器是奇数位，Rd 寄存器是偶数位。
3. 在研究 JZ、JC 指令的时候，需要条件变址，要搞清楚何时跳转第一条指令，何时跳转第二条指令，其中情况有两种，一种是零标志 Z 灯亮时，另一种是进位标志 CY 灯亮时，分别跳转不同的指令。这两条指令及 RR 指令都要连接实验箱才能进行调试。

.....