桂林航天工业学院

课程设计存档材料

课程名称：计算机组成与结构课程设计

开课学期：2019 — 2020 学年第一学期

专　　业：软件工程

班　　级：2017 软件工程 4 班

姓　　名：钟祯

学　　号：2017070030429

指导老师：张亚红　张新伦

报告日期：2019 年 12 月 31 日

桂林航天工业学院课程设计任务书

设计题目： 基本模型机的设计与实现

学生姓名	钟祯		
课程名称	计算机组成与结构课程设计	专业班级	2017 软件工程 4 班
地　　点	巡天楼 317A	起止时间	2019.12.23 - 2019.12.31
设计内容	1.设计机器指令，并设计其对应的微程序，将微程序下载到微程序控制器.将微程序控制器同执行部件（整个数据通路）联机，完成模型计算机的设计。 2.用微程序控制器控制模型计算机的数据通路。通过 TEC-5G 执行由 16 条机器指令组成的简单程序，掌握机器指令与微指令的关系，牢固建立计算机的整机概念。		
设计参数	1.设计机器指令 8 条。 2.设计机器指令的微程序 1 个。 3.运用设计的机器指令编写简单程序验证微程序是否能对机器指令进行解释执行。		
设计进度	第一阶段：先了解 TEC-5G 实验台机器指令的指令格式，根据指令格式设计机器指令。 第二阶段：根据设计的机器指令设计对应的微程序，画出微程序流程图。 第三阶段：用设计的机器指令编写简单程序验证微程序是否能对机器指令进行解释执行。 第四阶段：将微程序下载到微程序控制器，接线进行验证。 第五阶段：整理上述资料，提交课程设计报告。		
设计成果	1.机器指令 2.机器指令对应的微程序		
参考资料	1.《计算机组成原理》 白中英，科学出版社，2013. 2.《计算机组成与结构》实验指导书		
说明	1．本表应在每次实施前由指导教师填写一式 2 份，审批后所在教学单位和指导教师各留 1 份。 2．多名学生共用一题的，在设计内容、参数、要求等方面应有所区别。 3．若填写内容较多可另纸附后。		

指导教师：张亚红
教研室主任：张新伦
2019 年 12 月 31 日

桂林航天工業學院
GUILIN UNIVERSITY OF AEROSPACE TECHNOLOGY

本科课程设计报告

课程名称：计算机组成与结构课程设计

开课学期：2019 — 2020 学年第一学期

学　　院：计算机科学与工程学院

题　　目：基本模型机的设计与实现

专业班级：2017 软件工程 4 班

学　　号：2017070030429

学生姓名：钟祯

指导教师：张亚红 张新伦

报告日期：2019 年 12 月 31 日

成绩：＿＿＿＿＿＿＿＿（五级）

课程设计评分标准

平时（30%）	全勤、学习态度端正、实验认真、积极回答问题	优秀（90-100）	
	偶有缺勤、实验认真、回答问题较积极	良好（80-89）	
	旷课 2 次以内、偶有迟到、实验认真、回答问题较好	中（70-79）	
	旷课 2 次以上、学习态度一般、基本能回答出问题	及格（60-69）	
	经常旷课，实验过程不认真、问题回答不积极	不及格(<60)	
考核（70%）	作品功能完全实现、作品演示操作熟练、问题回答准确；设计论文结构合理、内容符合要求、论文格式符合规范要求、参考文献新	优秀（90-100）	
	作品功能基本完全实现、作品演示操作熟练、问题回答准确；设计论文结构合理、内容符合要求、论文格式符合规范要求、参考文献新	良好（80-89）	
	作品功能实现 70%、作品演示操作基较熟练、问题回答基本准确；设计论文结构基本合理、内容基本符合要求、论文格式基本符合规范要求、参考文献较新	中（70-79）	
	作品功能实现 50%、作品演示操作较熟练、部分问题回答准确；设计论文结构较合理、 论文格式基本符合规范要求、有参考文献	及 格（60-69）	
	作品功能不能实现、作品演示操作不熟练、问题回答效果差；设计论文结构不合理、 论文格式不符合规范要求、无参考文献	不及格(<60)	
总评			

目 录

一、 设计目的

1. 将微程序控制器同执行部件（整个数据通路）联机，组成一台模型计算机。
2. 用微程序控制器控制模型计算机的数据通路。
3. 通过 TEC-5G 执行由 16 条机器指令组成的简单程序，掌握机器指令与微指令的关系，牢固建立计算机的整机概念。

二、 设计所需设备

1. TEC-5G 计算机组成实验系统 1 台
2. 逻辑测试笔 1 支
3. 双踪示波器 1 台
4. 万用表 1 只

三、 设计原理

部件实践过程中，各部件单元的控制信号是人为模拟产生的。而本次课程设计能在微程序控制下自动产生各部件单元控制信号,实现特定指令的功能。这里，计算机数据通路的控制将交由微程序控制器来完成。CPU 从内存中取出一条机器指令到执行指令结束的一个指令周期，是由微程序来完成的，即一条机器指令对应一段微程序。

四、 设计内容

4.1 机器指令的格式与功能

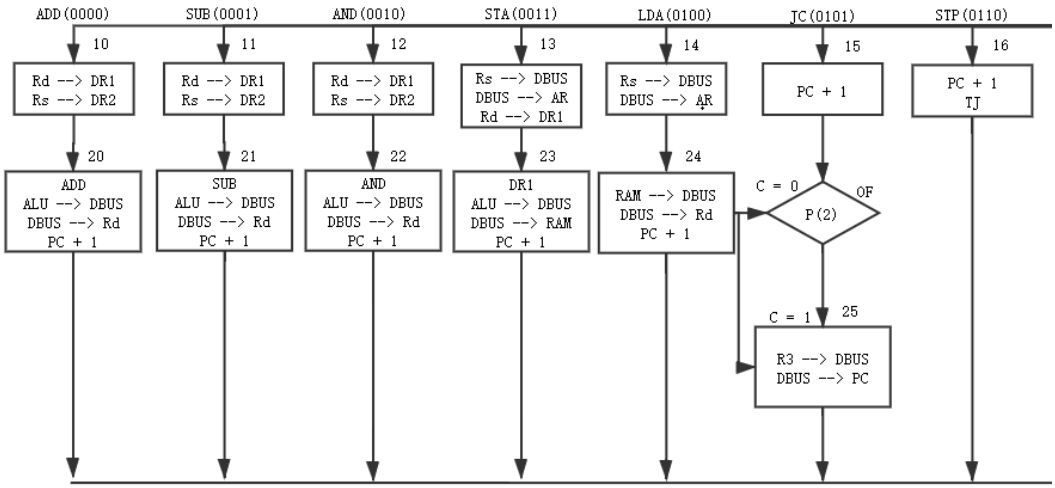
名称	助记符	功能	指令格式		
			IR7 IR6 IR5 IR4	IR3 IR2	IR1 IR0
加法	ADD Rd,Rs	Rd + Rs → Rd	0 0 0 0	Rs1 Rs0	Rd1 Rd0
减法	SUB Rd,Rs	Rd - Rs → Rd	0 0 0 1	Rs1 Rs0	Rd1 Rd0
逻辑与	AND Rd,Rs	Rd & Rs → Rd	0 0 1 0	Rs1 Rs0	Rd1 Rd0
存数	STA Rd,[Rs]	Rd → [Rs]	0 0 1 1	Rs1 Rs0	Rd1 Rd0
取数	LDA Rd,[Rs]	[Rs] → Rd	0 1 0 0	Rs1 Rs0	Rd1 Rd0
条件转移	JC R3	若 C = 1 则 R3 → PC	0 1 0 1	1 1	× ×
停机	STP	暂停执行	0 1 1 0	× ×	× ×
输出	OUT Rs	Rs → DBUS	0 1 1 1	Rs1 Rs0	× ×
自加	SelfAdd Rd	Rd + Rd → Rd	1 0 0 0	× ×	Rd1 Rd0
逻辑非	Non Rd	!Rd → Rd	1 0 0 1	× ×	Rd1 Rd0
逻辑或	Or Rd,Rs	Rd || Rs → Rd	1 0 1 0	Rs1 Rs0	Rd1 Rd0
逻辑或非	OrNon Rd,Rs	!(Rd || Rs)→Rd	1 0 1 1	Rs1 Rs0	Rd1 Rd0
逻辑与非	AndNon Rd,Rs	!(Rd && Rs)→Rd	1 1 0 0	Rs1 Rs0	Rd1 Rd0
逻辑异	Diff Rd,Rs	Rd ⊕ Rs → Rd	1 1 0 1	Rs1 Rs0	Rd1 Rd0
非 A 与 B	NonAAndB Rd,Rs	!Rd && Rs → Rd	1 1 1 0	Rs1 Rs0	Rd1 Rd0
A 或非 B	AOrNonB Rd,Rs	Rd || !Rs → Rd	1 1 1 1	Rs1 Rs0	Rd1 Rd0

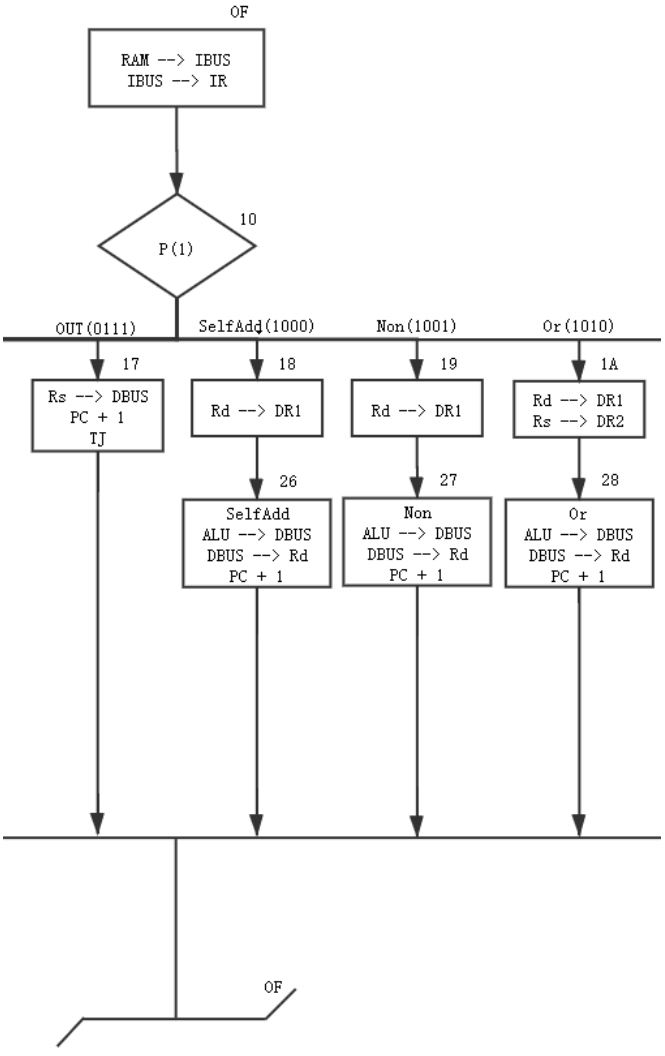
4.2 机器指令的代码形式

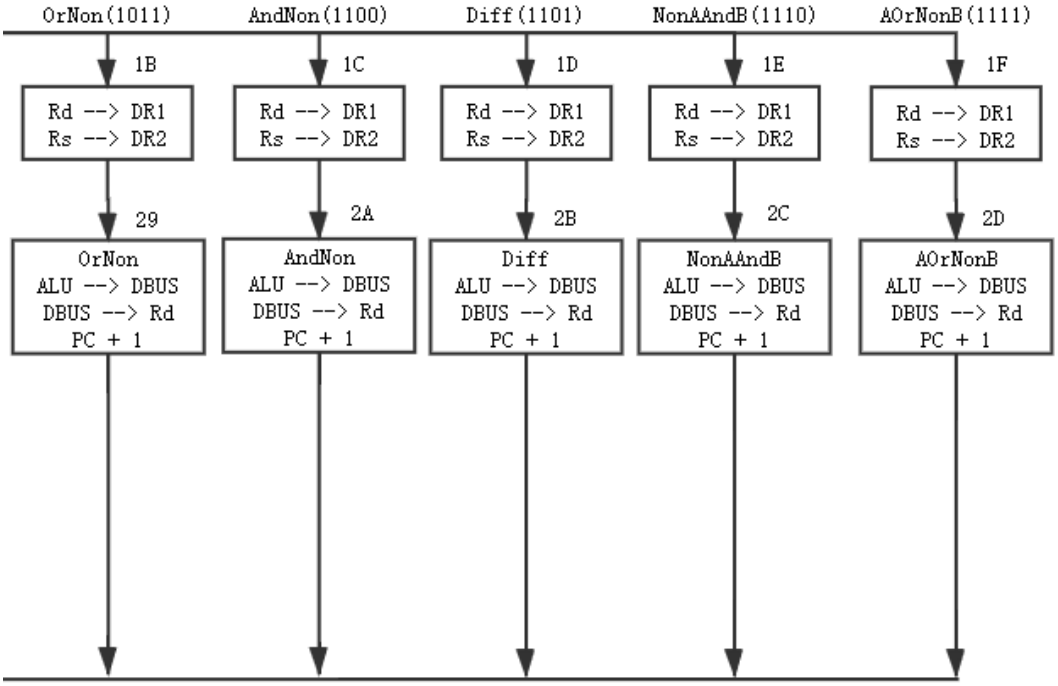
内存地址	机器指令	机器代码（十六进制）
00H	ADD R1,R0	01H
01H	JC R3	5CH
02H	STA R1,[R2]	39H
03H	LDA R2,[R2]	4AH
04H	AND R2,R0	22H
05H	SUB R2,R3	1EH
06H	SelfAdd R0	83H
07H	Non R1	95H
08H	Or R3,R1	A7H
09H	OrNon R0,R1	B4H
0AH	AndNon R0,R2	C8H
0BH	Diff R3,R2	DBH
0CH	NonAAndB R1,R3	EDH

0DH	AOrNonB R2,R1	F6H
0EH	OUT R2	7FH
0FH	STP	60H

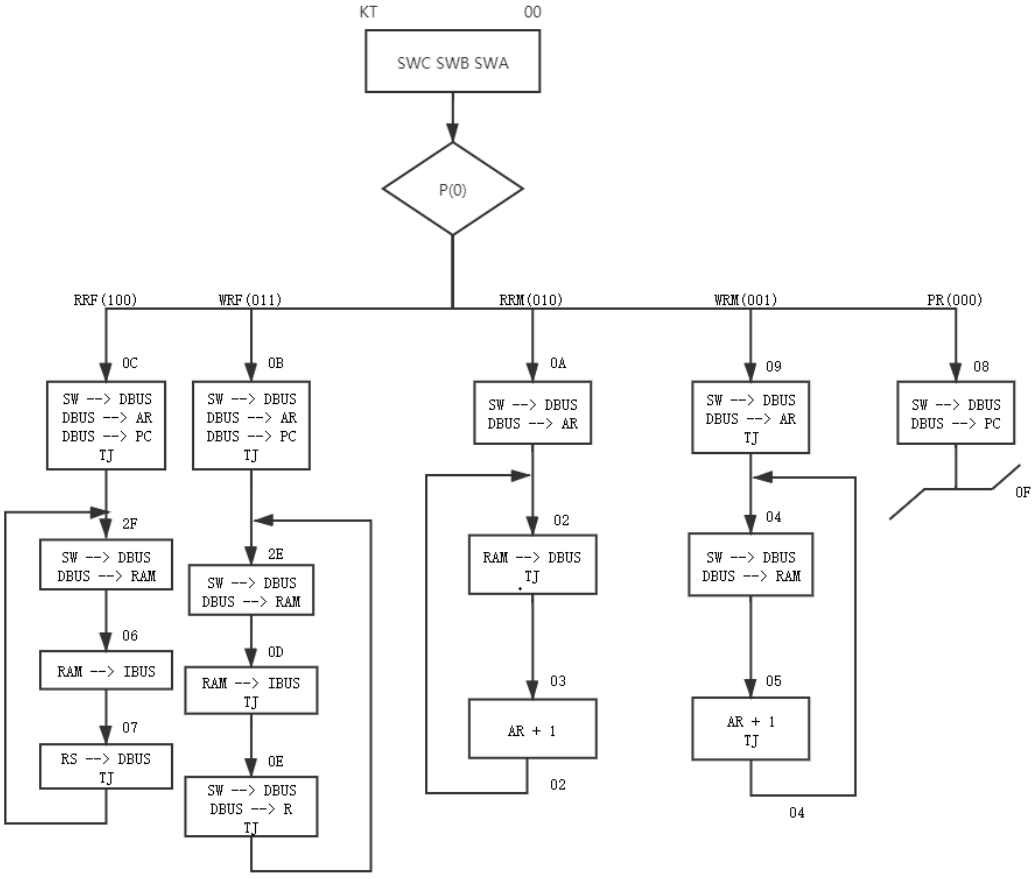
4.3 微指令流程图






OrNon(1011)
AndNon(1100)
Diff(1101)
NonAAndB(1110)
AOrNonB(1111)
1B
1C
1D
1E
1F
Rd --> DR1
Rs --> DR2
29
2A
2B
2C
2D
OrNon
AndNon
Diff
NonAAndB
AOrNonB
ALU --> DBUS
DBUS --> Rd
PC + 1

4.4 微程序流程图


KT
00
SWC SWB SWA
P(0)
RRF(100)
WRF(011)
RRM(010)
WRM(001)
PR(000)
0C
0B
0A
09
08
SW --> DBUS
DBUS --> AR
DBUS --> PC
TJ
DBUS --> RAM
RAM --> IBUS
RS --> DBUS
DBUS --> R
RAM --> DBUS
AR + 1
AR + 1
0F
2F
2E
0D
0E
02
03
04
05
07

4.5 微指令代码表

微地址	CM3	CM2	CM1	CM0
00	00	00	00	48
01	00	00	00	00
02	03	40	04	03
03	00	00	40	02
04	01	08	00	05
05	00	00	44	04
06	00	80	08	07
07	00	10	04	2F
08	00	08	20	0F
09	00	08	84	04
0A	00	08	80	02
0B	00	08	A4	2E
0C	00	08	A4	2F
0D	00	80	0C	0E
0E	00	0C	04	2E
0F	00	80	08	90
10	00	03	00	20
11	00	03	00	21
12	00	03	00	22
13	00	11	80	23
14	00	10	80	24
15	00	00	11	0F
16	00	00	14	0F
17	00	10	14	0F
18	00	24	10	26
19	00	24	10	27
1A	00	24	10	28
1B	00	20	10	29
1C	00	44	10	2A
1D	00	08	00	2B
1E	00	08	00	2C
1F	00	10	20	2D
20	90	24	10	0F
21	64	24	10	0F
22	B8	24	10	0F
23	01	24	10	0F
24	03	24	10	0F
25	01	24	10	0F

26	C0	24	10	0F
27	00	24	10	0F
28	E8	24	10	0F
29	18	24	10	0F
2A	48	24	10	0F
2B	68	24	10	0F
2C	28	24	10	0F
2D	D8	24	10	0F
2E	01	24	10	0D
2F	01	24	10	06

五、 设计步骤

1、连线

控制器	LDIR	PC+1	LDPC#	AR+1	LDAR#	LDDR1	LDDR2	LDRi
数据通路	LDIR	PC+1	LDPC#	AR+1	LDAR#	LDDR1	LDDR2	LDRi

表 5.1

控制器	SW_BUS#	Rs_BUS#	ALU_BUS#	RAM_BUS#	CER#	CEL#
数据通路	SW_BUS#	Rs_BUS#	ALU_BUS#	RAM_BUS#	CER#	CEL#

表 5.2

控制器	LR/W#	Cn#	M	S0	S1	S2	S3
数据通路	LR/W#	Cn#	M	S0	S1	S2	S3

表 5.3

控制器	进位 C	IR7	IR6	IR5	IR4
数据通路	进位 C	IR7	IR6	IR5	IR4

表 5.4

数据通路	IR3	IR2	IR1	IR0	IR1	IR0
数据通路	RS1	RS0	RD1	RD0	WR1	WR0

表 5.5

控制器	TJ
时序电路	TJ

表 5.6

把上面表中的同列的信号用线连接即可，一共接线 33 条。

接线后，将编程开关拨到正常位置。合上电源，按 CLR#按钮，使 TEC-5G 实验系统处于初始状态。

2、存程序代码，设置通用寄存器 R0、R1、R2 和 R3 的第一组值及存储器相关单元的数据。

本组的寄存器数据是 R0=35H、R1=43H、R2=10H、R3=07H。存储器 10H 单元的内容为 55H。这组数据在执行 ADD R1，R0 指令时不产生进位 C，从而在执行 JC R3 指令时不产生跳转，而是顺序执行。

（1）、设置通用寄存器 R0、R1、R2、R3 的值

① 令 DP=0，DB=0，使系统处于连续运行状态。令 SWC=0，SWB=1，SWA=1，使系统处于写寄存器状态 WRF。按 CLR#按钮，使实验系统处于初始状态。

② 在 SW7-SW0 上设置一个存储器地址，该存储器地址是设置通用寄存器使用，最好是不常用的一个地址如 0FH，以免设置通用寄存器操作时重要的存储器单元的内容被破坏。按一次 QD 按钮，将 0FH 写入左端口地址寄存器 AR。

③ 在 SW7-SW0 上设置 00H，作为通用寄存器 R0 的寄存器号，按一次 QD 按钮，将 00H 写入指令寄存器 IR。

④ 在 SW7-SW0 上设置 35H，按一次 QD 按钮，将 35H 写入 IR 指定的寄存器 R0。

⑤ 在 SW7-SW0 上设置 01H，作为通用寄存器 R1 的寄存器号，按一次 QD 按钮，将 01H 写入指令寄存器 IR。

⑥ 在 SW7-SW0 上设置 43H，按一次 QD 按钮，将 43H 写入 IR 指定的寄存器 R1。

⑦ 在 SW7-SW0 上设置 02H，作为通用寄存器 R2 的寄存器号，按一次 QD 按钮，将 02H 写入指令寄存器 IR。

⑧ 在 SW7-SW0 上设置 10H，按一次 QD 按钮，将 10H 写入 IR 指定的寄存器 R2。

⑨ 在 SW7-SW0 上设置 03H，作为通用寄存器 R3 的寄存器号，按一次 QD 按钮，将 03H 写入指令寄存器 IR。

⑩ 在 SW7-SW0 上设置 07H，按一次 QD 按钮，将 07H 写入 IR 指定的寄存器

R3。

设置寄存器内容完毕。按 CLR#按钮，使系统恢复到初始状态。

注意：设置完 R0、R1、R2、R3 的值后，最好用读寄存器控制台操作检查一下写入的内容是否正确。

（2）、向存储器写入机器代码

本操作中，从 00 地址开始存 16 个机器代码：01H，5CH，39H，4AH，22H，1EH，83H，95H，A7H，B4H，C8H，DBH，EDH，F6H，7FH，60H。在 10H 单元存入 55H，作为 10H 单元的初值，以检查 LDA 和 STA 两条指令的作用。

① 令 DP=0，DB=0，使实验系统处于连续运行状态。令 SWC=0、SWB=0，SWA=1，使实验系统处于写双端口存储器工作方式 WRM。按 CLR#按钮，使实验系统处于初始状态。

② 置 SW7-SW0 为 00H，按 QD 按钮，将 00H 写入左端口地址寄存器 AR。

③ 置 SW7-SW0 为 01H，按 QD 按钮，将 01H 写入存储器 00H 单元。AR 自动加 1，变为 01H。

④ 置 SW7-SW0 为 5CH，按 QD 按钮，将 5CH 写入存储器 01H 单元。AR 自动加 1，变为 02H。

⑤ 置 SW7-SW0 为 39H，按 QD 按钮，将 39H 写入存储器 02H 单元。AR 自动加 1，变为 03H。

⑥ 置 SW7-SW0 为 4AH，按 QD 按钮，将 4AH 写入存储器 03H 单元。AR 自动加 1，变为 04H。

⑦ 置 SW7-SW0 为 22H，按 QD 按钮，将 22H 写入存储器 04H 单元。AR 自动加 1，变为 05H。

⑧ 置 SW7-SW0 为 1EH，按 QD 按钮，将 1EH 写入存储器 05H 单元。AR 自动加 1，变为 06H。

⑨ 置 SW7-SW0 为 83H，按 QD 按钮，将 1EH 写入存储器 06H 单元。AR 自动加 1，变为 07H。

⑩ 置 SW7-SW0 为 95H，按 QD 按钮，将 1EH 写入存储器 07H 单元。AR 自动加 1，变为 08H。

⑪ 置 SW7-SW0 为 A7H，按 QD 按钮，将 1EH 写入存储器 08H 单元。AR 自动加 1，变为 09H。

⑫ 置 SW7-SW0 为 B4H，按 QD 按钮，将 1EH 写入存储器 09H 单元。AR 自动加 1，变为 0AH。

⑬ 置 SW7-SW0 为 C8H，按 QD 按钮，将 1EH 写入存储器 0AH 单元。AR 自动加 1，变为 0BH。

⑭ 置 SW7-SW0 为 DBH，按 QD 按钮，将 1EH 写入存储器 0BH 单元。AR 自动加 1，变为 0CH。

⑮ 置 SW7-SW0 为 EDH，按 QD 按钮，将 1EH 写入存储器 0CH 单元。AR 自动加 1，变为 0DH。

⑯ 置 SW7-SW0 为 F6H，按 QD 按钮，将 1EH 写入存储器 0DH 单元。AR 自动加 1，变为 0EH。

⑰ 置 SW7-SW0 为 7FH，按 QD 按钮，将 78H 写入存储器 0EH 单元。AR 自动加 1，变为 0FH。

⑱ 置 SW7-SW0 为 60H，按 QD 按钮，将 60H 写入存储器 0FH 单元。AR 自动加 1，变为 10H。

按 CLR#按钮，使实验系统恢复到初始状态。

① 置 SW7-SW0 为 10H，按 QD 按钮，将 10H 写入左端口地址寄存器 AR。

② 置 SW7-SW0 为 55H，按 QD 按钮，将 55H 写入存储器 10H 单元。AR 自动加 1，变为 11H。

往存储器写入程序和数据结束，按 CLR#按钮，使实验系统恢复到初始状态。

注意：设置完程序和数据后，最好用存储器控制台操作检查一下写入的内容是否正确。

（3）、用单拍（DP）方式执行一遍程序

置 SWC=0、SWB=0、SWA=0，DP=1，DB=0，使实验系统处于单拍运行状态。置 SW7-SW0=00H，使程序从地址 00H 开始执行。按 CLR#按钮，使实验系统处于初始状态，然后 一次一次按 QD 按钮，使程序一拍一拍地执行。

在单拍执行过程中，首先要随时监测 AR、PC、 μA5 ~ μA0 和 IR 的值，以判定程序执行到何处，正在执行哪条指令和微指令。对照微程序流程图，可以判断出微指令的地址和正在进行的微操作。程序执行的结果如下：

初值：R0=35H，R1=43H，R2=10H，R3=07H。存储器单元 10H 单元的内容为 55H。

① ADD R1，R0

执行结果 R0=35H，R1=78H，R2=10H，R3=07H。存储器单元 10H 单元的内容为 55H，无进位 C。

② JC R3

执行结果 R0=35H，R1=78H，R2=10H，R3=07H。存储器单元 10H 单元的内容为 55H。PC 为 02H。进位 C 不变。

③ STA R1，[R2]

执行结果 R0=35H，R1=78H，R2=10H，R3=07H。存储器单元 10H 单元的内容为 78H。

④ LDA R2，[R2]

执行结果 R0=35H，R1=78H，R2=78H，R3=07H。存储器单元 10H 单元的内容为 78H。

⑤ AND R2，R0

执行结果 R0=35H，R1=78H，R2=30H，R3=07H。存储器单元 10H 单元的内容为 78H。

⑥ SUB R2，R3

执行结果 R0=35H，R1=78H，R2=29H，R3=07H。存储器单元 10H 单元的内容为 78H，进位 C 为 1。

⑦ SelfAdd R0

执行结果 R0=36H，R1=78H，R2=29H，R3=07H。存储器单元 10H 单元的内容为 78H，无进位 C。

⑧ Non R1

执行结果 R0=36H，R1=87H，R2=29H，R3=07H。存储器单元 10H 单元的内容为 78H，无进位 C。

⑨ Or R3，R1

执行结果 R0=36H，R1=87H，R2=29H，R3=37H。存储器单元 10H 单元的内容为 78H，无进位 C。

⑩ OrNon R0，R1

执行结果 R0=48H，R1=87H，R2=29H，R3=37H。存储器单元 10H 单元的内容为 78H，无进位 C。

⑪ AndNon R0，R2

执行结果 R0=F7H，R1=87H，R2=29H，R3=37H。存储器单元 10H 单元的内容为 78H，无进位 C。

⑫ Diff R3，R2

执行结果 R0=F7H，R1=87H，R2=29H，R3=1CH。存储器单元 10H 单元的内容为 78H，无进位 C。

⑬ NonAAndB R1，R3

执行结果 R0=F7H，R1=08H，R2=29H，R3=1CH。存储器单元 10H 单元的内容为 78H，无进位 C。

⑭ AOrNonB R2，R1

执行结果 R0=F7H，R1=08H，R2=FFH，R3=1CH。存储器单元 10H 单元的内容为 78H，无进位 C。

⑮ OUT R2

执行结果 R0=F7H，R1=08H，R2=FFH，R3=1CH。存储器单元 10H 单元的内容为 78H，可在数据总线 DBUS 指示灯上观察到 FFH。

⑯ STP

执行结果 R0=F7H，R1=08H，R2=FFH，R3=1CH。存储器单元 10H 单元的内容为 78H。

最后的执行结果可以通过控制台的读寄存器操作和读存储器操作观察到，在观察最后的结果之前，首先应按 CLR#按钮，使实验系统处于初始状态。

实验执行结果：（R0=F7H，R1=08H，R2=FFH，R3=1CH 存储器单元 10H 单元的内容为 78H）。

（4）、用连续方式执行一遍程序

① 由于上面的单拍执行程序，已破坏了寄存器 R1、R2 和存储器 10H 单元的内容（程序没有破坏），因此需要重新设置寄存器 R1、R2 和存储器 10H 单元的值。初值：R0=35H，R1=43H，R2=10H，R3=07H。存储器单元 10H 单元的内容为 55H。

② 置 SWC=0、SWB=0、SWA=0，DP=0，DB=0，使实验系统处于连续运行状态。置 SW7-SW0=00H，使程序从地址 00H 开始执行。按 CLR#按钮，使实验系统处于初始状态，然后按一次 QD 按钮，则程序自动连续运行到地址为

07H 的 STP 指令。最后的执行结果可以通过控制台的读寄存器操作和读存储器操作观察到，在观察最后的结果之前，首先应按 CLR#按钮，使实验系统处于初始状态。

3、存程序代码，设置通用寄存器 R0、R1、R2 和 R3 的第二组值及存储器相关单元的数据，再用连续方式执行一遍程序。

本组的寄存器数据是 R0=86H，R1=88H，R2=10H，R3=0FH。存储器 10H 单元的内容为 55H。这组数据执行 ADD R1，R0 指令时产生了进位 C，从而在执行 JC R3 指令时产生跳转，而不是顺序执行。

（1）、设置通用寄存器 R0、R1、R2、R3 的值

① 令 DP=0，DB=0，使系统处于连续运行状态。令 SWC=0，SWB=1，SWA=1，使系统处于写寄存器状态 WRF。按 CLR#按钮，使实验系统处于初始状态。

② 在 SW7-SW0 上设置一个存储器地址，该存储器地址是设置通用寄存器使用，最好是不常用的一个地址如 0FFH，以免设置通用寄存器操作时重要的存储器单元的内容被破坏。

③ 在 SW7-SW0 上设置 00H，作为通用寄存器 R0 的寄存器号，按一次 QD 按钮，将 00H 写入指令寄存器 IR。

④ 在 SW7-SW0 上设置 86H，按一次 QD 按钮，将 86H 写入 IR 指定的寄存器 R0。

⑤ 在 SW7-SW0 上设置 01H，作为通用寄存器 R1 的寄存器号，按一次 QD 按钮，将 01H 写入指令寄存器 IR。

⑥ 在 SW7-SW0 上设置 88H，按一次 QD 按钮，将 88H 写入 IR 指定的寄存器 R1。

⑦ 在 SW7-SW0 上设置 02H，作为通用寄存器 R2 的寄存器号，按一次 QD 按钮，将 02H 写入指令寄存器 IR。

⑧ 在 SW7-SW0 上设置 10H，按一次 QD 按钮，将 10H 写入 IR 指定的寄存器 R2。

⑨ 在 SW7-SW0 上设置 03H，作为通用寄存器 R3 的寄存器号，按一次 QD 按钮，将 03H 写入指令寄存器 IR。

⑩ 在 SW7-SW0 上设置 07H，按一次 QD 按钮，将 07H 写入 IR 指定的寄存器

R3。

设置寄存器内容完毕。按 CLR#按钮，使系统恢复到初始状态。

注意：设置完 R0、R1、R2、R3 的值后，最好用读寄存器控制台操作检查一下写入的内容是否正确。

（2）、向存储器写入机器代码

本操作中，从 00 地址开始存 8 个机器代码：01H，5CH，39H，4AH，22H，1EH，78H，60H。在 10H 单元存入 55H，作为 10H 单元的初值，以检查 LDA 和 STA 两条指令的作用。

① 令 DP=0，DB=0，使系统牌连续运行状态。令 SWC=0，SWB=0，SWA=1，使系统处于写寄存器状态 WRF。按 CLR#按钮，使实验系统处于初始状态。

② 置 SW7-SW0 为 00H，按 QD 按钮，将 00H 写入左端口地址寄存器 AR。

③ 置 SW7-SW0 为 01H，按 QD 按钮，将 01H 写入存储器 00H 单元。AR 自动加 1，变为 01H。

④ 置 SW7-SW0 为 5CH，按 QD 按钮，将 5CH 写入存储器 01H 单元。AR 自动加 1，变为 02H。

⑤ 置 SW7-SW0 为 39H，按 QD 按钮，将 39H 写入存储器 02H 单元。AR 自动加 1，变为 03H。

⑥ 置 SW7-SW0 为 4AH，按 QD 按钮，将 4AH 写入存储器 03H 单元。AR 自动加 1，变为 04H。

⑦ 置 SW7-SW0 为 22H，按 QD 按钮，将 22H 写入存储器 04H 单元。AR 自动加 1，变为 05H。

⑧ 置 SW7-SW0 为 1EH，按 QD 按钮，将 1EH 写入存储器 05H 单元。AR 自动加 1，变为 06H。

⑨ 置 SW7-SW0 为 83H，按 QD 按钮，将 83H 写入存储器 06H 单元。AR 自动加 1，变为 07H。

⑩ 置 SW7-SW0 为 95H，按 QD 按钮，将 95H 写入存储器 07H 单元。AR 自动加 1，变为 08H。

⑪ 置 SW7-SW0 为 A7H，按 QD 按钮，将 A7H 写入存储器 08H 单元。AR 自动加 1，变为 09H。

⑫ 置 SW7-SW0 为 B4H，按 QD 按钮，将 B4H 写入存储器 09H 单元。AR 自动加

1，变为 0AH。

⑬ 置 SW7-SW0 为 C8H，按 QD 按钮，将 C8H 写入存储器 0AH 单元。AR 自动加 1，变为 0BH。

⑭ 置 SW7-SW0 为 DBH，按 QD 按钮，将 DBH 写入存储器 0BH 单元。AR 自动加 1，变为 0CH。

⑮ 置 SW7-SW0 为 EDH，按 QD 按钮，将 EDH 写入存储器 0CH 单元。AR 自动加 1，变为 0DH。

⑯ 置 SW7-SW0 为 F6H，按 QD 按钮，将 F6H 写入存储器 0DH 单元。AR 自动加 1，变为 0EH。

⑰ 置 SW7-SW0 为 7FH，按 QD 按钮，将 7FH 写入存储器 0EH 单元。AR 自动加 1，变为 0FH。

⑱ 置 SW7-SW0 为 60H，按 QD 按钮，将 60H 写入存储器 0FH 单元。AR 自动加 1，变为 10H。

按 CLR#按钮，使实验系统恢复到初始状态。

① 置 SW7-SW0 为 10H，按 QD 按钮，将 10H 写入左端口地址寄存器 AR。

② 置 SW7-SW0 为 55H，按 QD 按钮，将 55H 写入存储器 10H 单元。AR 自动加 1，变为 11H。

往存储器写入程序和数据结束，按 CLR#按钮，使实验系统恢复到初始状态。

注意：设置完程序和数据后，最好用存储器控制台操作检查一下写入的内容是否正确。

（3）、用单拍（DP）方式执行一遍程序

置 SWC=0、SWB=0、SWA=0，DP=1，DB=0，使实验系统处于单拍运行状态。置 SW7-SW0=00H，使程序从地址 00H 开始执行。按 CLR#按钮，使实验系统处于初始状态，然后一次一次按 QD 按钮，使程序一拍一拍地执行。

在单拍执行过程中，首先要随时监测 AR、PC、 μA5 ~ μA0 和 IR 的值，以判定程序执行到何处，正在执行哪条指令和微指令。对照微程序流程图，可以判断出微指令的地址和正在进行的微操作。程序执行的结果如下：

初值：R0=86H，R1=88H，R2=10H，R3=0FH。存储器单元 10H 单元的内容为 55H。

① ADD R1，R0

执行结果 R0=86H，R1=0EH，R2=10H，R3=0FH。存储器单元 10H 单元的内容为 55H，有进位 C。

② JC R3

执行结果 R0=86H，R1=0EH，R2=10H，R3=0FH。存储器单元 10H 单元的内容为 55H。PC 为 0FH，进位 C 不变。

③ STP

执行结果 R0=86H，R1=0EH，R2=10H，R3=0FH。存储器单元 10H 单元的内容为 55H，进位 C 不变。

最后的执行结果可以通过控制台的读寄存器操作和读存储器操作观察到，在观察最后的结果之前，首先应按 CLR#按钮，使实验系统处于初始状态。

实验执行结果：（R0=86H，R1=0EH，R2=10H，R3=0FH。存储器单元 10H 单元的内容为 55H）。

（4）、用连续方式执行一遍程序

① 由于上面的单拍执行程序，已破坏了寄存器 R1 的内容（程序没有破坏），因此需要重新设置寄存器 R1 值。初值：R0=86H，R1=88H，R2=10H，R3=0FH。存储器单元 10H 单元的内容为 55H。

② 置 SWC=0、SWB=0、SWA=0，DP=0，DB=0，使实验系统处于连续运行状态。置 SW7-SW0=00H，使程序从地址 00H 开始执行。按 CLR#按钮，使实验系统处于初始状态，然后按一次 QD 按钮，则程序自动连续运行到地址为 0FH 的 STP 指令。

最后的执行结果可以通过控制台的读寄存器操作和读存储器操作观察到，在观察最后的结果之前，首先应按 CLR#按钮，使实验系统处于初始状态。

六、 设计总结

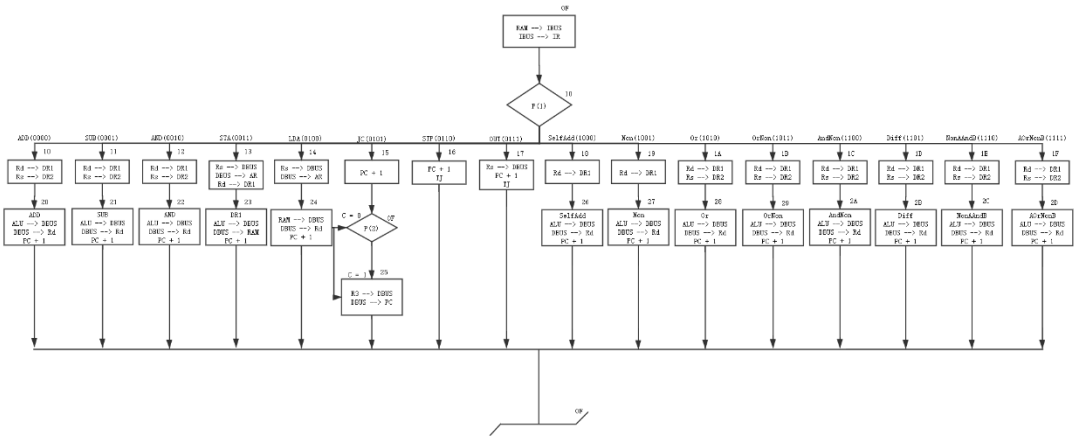
不知不觉的，一个星期就过去了，通过这一个星期的计算机组成原理课程设计实践，我对计算机的基本组成，计算机各个部件的基本功能，各个部件间的连接方法，微程序控制器的设计，微指令和微程序的编制都有了一定的了解，同时也增强了我的实际动手能力。从遇到难题到解决难题，是对自己能力的肯定，自信心大增。课程设计是结束了，不过这次实践教会了我许多，包括知识、能力。

通过这次课程设计，我不仅应用到了许多的理论知识，同时更多的是理论和实践相结合，本次课程设计真的不是很简单，不管是查资料还是其他部分。在这次课程设计中，我对存储器有了进一步的了解，理解它的功能特性和使用方法。做这次的课设时我把原理弄清楚了，做起课设来自然顺手不少，但还是需要极大的细心和耐心。以前觉得这是个不可能完成的事实，但是我却做到了，我觉得自己还是要认真的学习更多的东西，看更多的书籍。同时我也发现自身的不足之处，对于太多细节的不注重和学的东西太少，直接导致了我在设计中遇到很多困难，基础知识的不牢固，也是我自身的缺陷之一，往后我会继续努力，更进一步。

学生签名：钟祯

2019 年 12 月 31 日

附录 A 完整微指令流程原图



附录 B 完整微程序流程原图