北京交通大学

研究型学习报告

中文题目：十六位教学计算机微程序设计

学　　院：计算机学院

专　　业：信息安全

姓　　名：张 向 宇

学　　号：**18281059**

指导教师：

2020 年 **5** 月 **30** 日

北京交通大学

研究型学习报告

研究型学习题目：十六位教学计算机微程序设计

一、研究内容：

(1) 详细说明实验系统监控程序的前 7 条机器指令功能及对应的微程序；

(2) 设计指令 ADD DR，[ADR] 指令的微程序，DR 为目标寄存器，ADR 为 16 位的形式地址。该指令完成把 DR 的内容和内存中 ADR 地址所对应的内容相加后，将合存放在 DR 中；

(3) 设计 ADD DR，@ADR 的微程序，该指令的一个操作数在 DR 中，另外一个操作数通过间接寻址得到，首先，由内存中 ADR 地址所对应的内容为操作数的地址，根据此地址，读出其内容作为操作数。两个操作数相加后，将合存放在 DR 中；

(4) 设计 SUB DR，@[ADR]的微程序，该指令的一个操作数在 DR 中，另外一个操作数通过间接寻址得到，首先，由内存中 ADR 地址所对应的内容为操作数的地址，根据此地址，读出其内容作为操作数。两个操作数相减（DR-SR）后，将差存放在 DR 中。

二、基本要求：

（1）以论文格式书写，题目：十六位教学计算机微程序设计

（2）写出微程序： 包括微地址 和微指令编码，对使用到的字段做注释；

（3）提交含有所设计的微程序的 ROM 压缩文件；

（4）编写汇编测试程序，对所设计的微程序正确性加以说明。

（5）对执行测试程序验证所设计指令的截图（反映出指令位置，数据位置，运行结果）

目录

摘要

8251 是通用同步/异步接收器/发送器，包装在 Intel 生产的 28 针 DIP 中。它通常用于串行通信，额定每秒传输速率为 19.2 KB。它通常与更常见的 8250 UART 混淆，后者在 IBM 个人计算机中被广泛用作串行端口。它包括 5 个部分读/写控制逻辑；发射机；接收者数据总线系统；调制解调器控制；

监控程序是指通过植入目标主机的程序动态地监视目标主机，实现将事件传输过去，进行一般操作的程序。监控程序有两种模式，一种是服务器端与客户端在同一台机器运行，另一种是服务器端与客户端在不同机器运行，通过客户端程序监控服务器端主机。

通过分析实验系统监控程序的前 7 条机器指令功能及对应的微程序，深刻理解了实验系统与计算机系统的连接方式，学习了可编程串行通信和接口芯 8251A 的工作原理，通过以上的学习，又加深了对于微程序在实际应用中的理解。

微指令，又称微码，是在 CISC 结构下，运行一些功能复杂的指令时，所分解一系列相对简单的指令。微指令的作用是将机器指令与相关的电路实现分离，这样一来机器指令可以更自由的进行设计与修改，而不用考虑到实际的电路架构。与其他方式比较起来，使用微指令架构可以在降低电路复杂度的同时，建构出复杂的多步骤机器指令。撰写微指令一般称为微程序设计，而特定架构下的处理器实做中微指令有时会称为微程序。

现代的微指令通常由 CPU 工程师在设计阶段编写，并且存储在只读存储器或可编程逻辑数组中。然而有些机器会将微指令存储在静态随机存取存储器（SRAM）或是闪存中。它通常对普通程序员甚至是汇编语言程序员来说是不可见的，也是无法修改的。与机器指令不同的是，机器指令必须在一系列不同的处理器之间维持兼容性，而微指令只设计成在特定的电路架构下运行，成为特定处理器设计的一部分。

通过自行设计微程序，深刻理解了计算机系统的工作方式，理解了计算机的指令系统，CPU 的结构和功能，控制单元的功能，控制单元的设计，真正学会了利用计算机编写固件的能力。在动手实践中，学会了动态的调试程序，学会了编写测试程序，加深了对于微程序在实际应用中的理解。

1 实验系统监控程序

详细说明了实验系统监控程序的前 7 条机器指令功能及对应的微程序。教学机的监控程序是用教学机的汇编语言实现的，运行在教学机的硬件系统之上。它的主要功能是支持把计算机终端或 PC 机仿真终端接入教学机系统，使用这样的设备执行输入/输出操作，运行教学机的有关程序，以更方便直观的形式支持教学机上的各项实验功能，提供教学机汇编语言的可用子程序。在当前的实现中，它被固化在 0000h-0A2F 共 2K 多字的主存 ROM 区。在将来的实现中，新增加的部分将被固化在 0A30h-1FFFh 的主存 ROM 区。

当教学机被正常设置并加电启动之后，首件事情是从内存 0 地址开始启动监控程序，也就是使监控程序进入运行状态，此后便可以从键盘打入监控程序的命令并使其执行。监控程序提供类似 PC 机 DOS 系统下的 Debug 程序的功能，支持 A、U、G、P、T、R、D 和 E 共 8 个监控命令。

1．1 监控指令内存分析

表 1.1 监控程序指令

序号	指令寄存器 IR	附加操作数	功能
1	10001000 00000000	DR = R0，DATA=4E	MVRD DR、DATA
2	10000110 10000001	PORT = 81	I/O PORT
3	10001000 00000000	DR = R0，DATA=37	MVRD DR、DATA
4	10000110 10000001	PORT = 81	I/O PORT
5	10001000 01000000	DR = R4，DATA=2607	MVRD DR、DATA
6	10001000 00000000	DR = R0，DATA= 2000	MVRD DR、DATA
7	10000101 00000000	SR = R0	PUSH SR

表 1.2 监控程序指令内存分布

内存单元地址	存储值（HEX）
0000	88 00
0001	00 4E
0002	86 81
0003	88 00
0004	00 37
0005	86 81
0006	88 40

0007	26 07
0008	88 00
0009	20 00
0010	83 10
0011	01 00

1.2 监控指令功能分析

实验监控程序的作用是实验系统与计算机相连接，能够进行通信交换，其中8251A 是一个可编程的多功能芯片，在使用时必须对其进行初始化编程，用以确定工作方式、命令、波特率、字符格式、同步字等。编程的内容包括两大方面：由 CPU 发给 8251A 的控制字，即方式选择控制字和操作命令控制字；共用一个端口地址，按顺序写入.另一方面是异步方式：在方式选择控制字写入之后，紧接着必须写入操作命令控制字。

其中方式字确定 8251A 的工作方式;（异步，波特率，字符长度，奇偶校验）;命令字控制 8251A 按方式字所规定的方式工作（允许，禁止收发数据，启动搜索同步字符，8251 复位）；状态字主要是了解 8251A 的工作状态。:

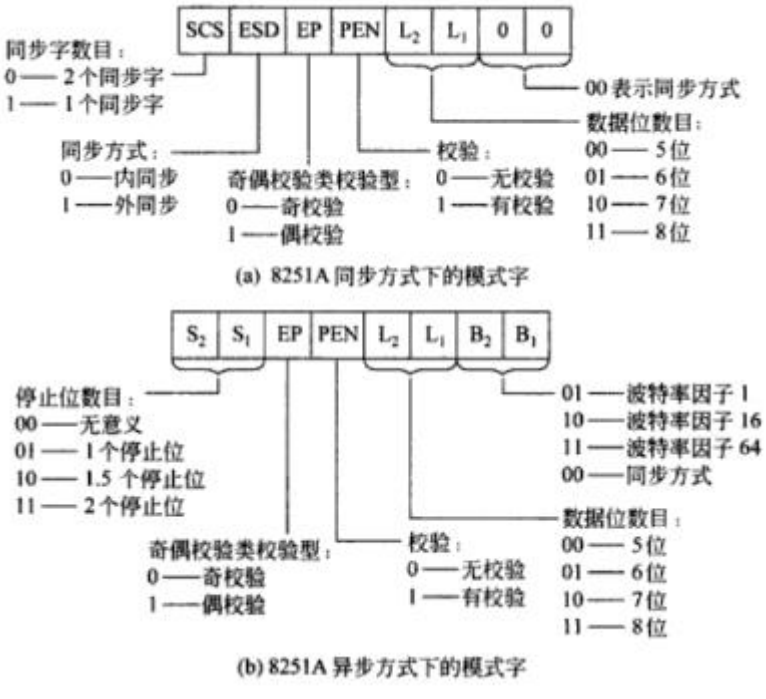

图 1 方式控制字

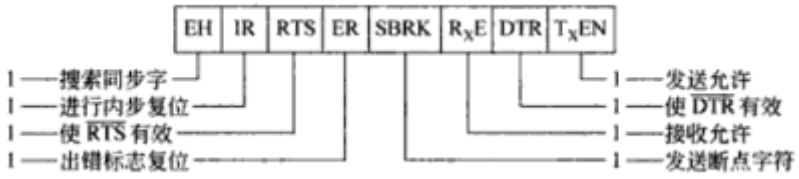

图 2 操作命令控制字

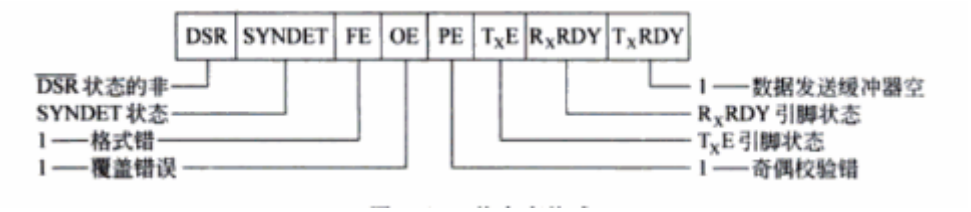

图 3 状态字

对 8251A 的使用,必须首先进行初始化。对 8251A 进行初始化的时候.需要遵守如下的初始化约定:

(1) 芯片复位后,将第一次用奇地址端口写人的值写到模式字寄存器。在模式字中规定为同步方式或者异步方式

(2) 若在模式字中规定同步工作方式,则 CPU 接着往奇地址输出的一个或两个字节就是同步字符,写人同步字寄存器,然后再将命令控制字写人奇端口

(3) 若在模式字中规定异步工作方式,则 CPU 往奇端口输出的一个字就是命令控制字

(4) 以后,只要不是复位命令,用奇地址端口写的是命令控制字,用偶地址端口写的是数据,送到数据输出缓冲器

(5) 初始化流程图见图 4.不管是同步方式还是异步方式,控制字的主要含义是相同的。由初始化流程图可以看出,当 CPU 往 8251A 发送命令控制字以后,8251A 首先会判断是不是给出了复位命令。如是,则重新接收模式字;如果不是,则可以开始传输数据

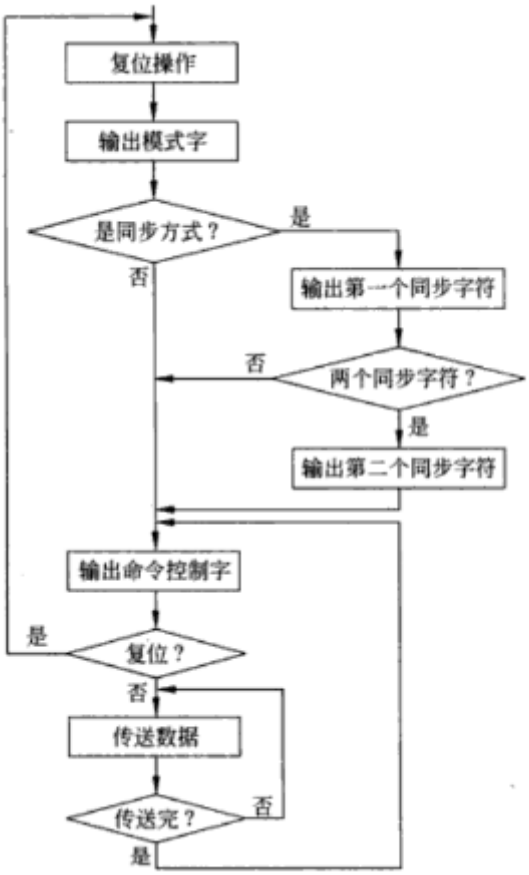

图 4 8251A 初始化操作流程图

1.2.1 MVRD DR、DATA（10001000 00000000）

这条指令的作用是将方式选择控制字 0100 1110（4E）放入 R0 寄存器中，作为缓冲器，其中方式选择控制字的含义是：

S2 S1 = 01：1 个停止位; EP PEN = 00：无奇偶校验位; L2 L1 = 11：字符长度为 8 位; B2 B1 = 10：异步方式 x16,其中这里的作用是定义发送波特率，假设定义时钟信号为 600，则波特率为 600 * 16 = 9600;

1.2.2 I/O PORT（ 10000110 10000001）

这条指令的作用是调用系统的硬件 I/O 功能向端口号 81 发送之前存在 R0 寄存器中的选择控制字

1.2.3 MVRD DR、DATA（10001000 00000000）

这条指令的作用是将操作命令控制字 0011 0111（37）放入 R0 寄存器中，作为缓器，其中操作命令控制字的含义是：

EH = 0：不启动搜索同步字符； IR = 0：不启动内部复位，不使 8251A 回

到方式选择格式;；RTS = 1：使 RTS 非引脚输出低电平； ER = 1：使错误标志 PE、OE、FE 均复位； SBRK = 0：正常工作； RxE = 1：允许接收； DTR = 1：使 DTR 非引脚输出低电平；TxEN = 1：允许发送

1.2.4 I/O PORT（ 10000110 10000001）

这条指令的作用是调用系统的硬件 I/O 功能向端口号 81 发送之前存在 R0 寄存器中的操作命令控制字

1.2.5 MVRD DR、DATA（10001000 01000000）

这条指令的作用应该是，这条指令的作用是将数据 26 07 放入 R4 寄存器中

1.2.6 MVRD DR、DATA （10001000 00000000）

这条指令的作用是将数据 20 00 放入 R0 寄存器中

1.2.7 PUSH SR （10000101 00000000）

这条指令的作用是将 R0 寄存器压入栈中

1.3 监控微指令分析

1.3.1 MVRD DR、DATA（10001000 00000000）

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
ALL	MEM→IR	00	1110	0000	0001	0000	0001	0000	0000	0000	0000	0000	0001	0000	02
ALL	/MAP	00	0010	0000	0100	0000	0001	0000	0000	0000	0000	0000	0000	0000	03
MVRD	PC→AR, PC+1→PC, CC#=0	1C	0011	0000	0100	0011	0010	0000	0101	0101	0000	0001	0011	0000	1D
ALL	MEM→DR, CC#=0	30	0011	0000	001	0111	0011	1000	0000	0000	0000	0000	0000	0000	1C
ALL	STR→Q CC#=INT#	3A	0011	0010	010	0111	0000	0000	0000	0000	0000	0000	0000	0011	30
	PC→AR, PC+1→PC, CC#=0	02	0011	0000	0100	0011	0010	0000	0101	0101	0000	0001	1011	0000	31

图 5 8251A 初始化操作流程图

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
ALL	MEM→IR	00	1110	0000	0001	0000	0001	0000	0000	0000	0000	0000	0001	0000	02

图 6

第一条微指令程序：

CI3~0：#14 命令，表示顺序执行

SCC30：0000：必转移，无需条件即转移

0MRW：0001：内存读

0I2~0：0000：R：A，S：Q

SA，I8~6：00001：因为要实现->IR，Y 输出 F

SB、I5~3：0000 R+ S

B 口：0000 R0

A 口：0000 R0

0SST：无标志位的运算，所以为 000

SSH：00；SCI：00

Cin = 0 不进位

DC2：0001 /GIR：指令寄存器 IR 接收

DC1：0000 /SWTOIB：开关到内部总线

指令功能：内存中取出 AR 寄存器指定的内存单元数据放入指令寄存器中，实现了 MEM -> IR 的功能，指明了下一条指令的地址。

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
ALL	/MAP	00	0010	0000	0100	0000	0001	0000	0000	0000	0000	0000	0000	0000	03

图 7

第二条微指令程序：

CI3~0：0010：MAPROM

SCC30：0000 必转

0MRW：0100 无读写

0I2~0：0000 R：A，S：Q

SA，I8~6：00001 Y：F

SB、I5~3：0000 R+S

B 口：0000 R0

A 口：0000 R0

0SST：0000 C：C、Z：Z、V：V、S：S

SSH：00

SCI：00

Cin = 0 不进位

DC2：0000 NC：NC

DC1：0000 /SWTOIB：开关到内部总线

指令功能是 CI3~0：0010，即使使能信号/MAP 有效，使 D 来源于 MAPROM，用

于实现从刚刚存入 IR 寄存器的机器指令的操作码中找到相应的微程序段首地址

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
MVRD	PC→AR、PC+1→PC、CC#=0	1C	0011	0000	0100	0011	0010	0000	0101	0101	0000	0001	0011	0000	1D

图 8

第三条微指令程序：

CI3~0：0011 条件微转移

SCC30：0000 必转移

0MRW：0100 无读写

0I2~0：0011 R：0，S：B，因为 PC 既要被运算也要被输出，数据来源 0 和 B 口

SA，I8~6：0010 F→B Y：A，因为要实现->AR,PC，所以要输出到 PC 上，所以选 F->B,Y=A，故为 010

SB、I5~3：000 R - S，执行 ADD 功能，R+S,即 0+PC

B 口：0101 R5，即 PC，B 口内容+1 送到 B 口

A 口：0101 R5，即 PC，输出 A 口的内容到 AR，实现 PC→AR，ST（000）：无标志位的运算，所以为 000

0SST：0000 无标志位的运算，所以为 000SSH：00

SCI：01

Cin = 1 PC 要执行自加 1，标志位 Cin=1，即实现 PC+1→PC

DC2：0011 /GARH：AR 高位接收，需要 AR（地址寄存器）高位接收数据。PC（程序计数器）中的地址传递到 AR（地址寄存器）中，由 AR 选中内存中相应的地址。实现 PC→AR 功能

DC1：0000 /SWTOIB：开关到内部总线，送开关内容到内部总线,通过 16 位数据开关置入指令操作码

指令功能：PC→AR、PC+1→PC。实现 PC 的值赋给 AR（地址寄存器），PC 自身实现自增，为之后的读操作中，找到 MAR 内的地址单元在主存中取出对应指令做准备

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
ALL	MEM→DR、CC#=0	30	0011	0000	0001	0111	0011	1000	0000	0000	0000	0000	0000	0000	1C

图 9

第四条微指令程序：

CI3~0：0011 条件微转移

SCC30：0000 必转移

0MRW：0001 内存读

0I2~0：0111 R：D，S：0

SA，I8~6：0011 F→B Y：F，Y 输出 F 且把输出结果送到 B 口
SB、I5~3：1000 R + S
B 口：0000 R0
A 口：0000 R0 无效不用
0SST：0000 四个标志位的值保持不变
SSH：00
SCI：00
Cin = 0 执行 ADD 指令，Cin=0
DC2：0000 NC：NC
DC1：0000 /SWTOIB：开关到内部总线
指令功能：完成 MEM→R0 本条微指令是将存储单元中读到的立即数送至 DR（R1），给 DR 赋值，通过运算器实现 0+D→DR，其中 D 是从存储单元中读到的立即数通过 245 缓冲器经过数据总线传送至 CPU 的内部总线，作为 ALU 的外部输入 D 的数据来源，参与后面的 ADD 运算。

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
ALL	STR→Q CC#=INT#	3A	0011	0010	0100	0111	0000	0000	0000	0000	0000	0000	0000	0011	30

图 10

`

第五条微指令程序：
CI3~0：0011 条件微转移
SCC30：0010 /INT=0 时，必转
0MRW：0100 无读写
0I2~0：0111 R：D，S：0
SA，I8~6：0000 Q：F→Q
SB、I5~3：0000 R + S
B 口：0000 R0
A 口：0000 R0
0SST：0000 C：C、Z：Z、V：V、S：S
SSH：00
SCI：00
Cin = 0
DC2：0000 NC：NC
DC1：0011 /FTOIB：状态到内部总线
指令功能：完成 STR → Q,

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
	PC→AR、PC+1→PC、CC#=0	02	0011	0000	0100	0011	0010	0000	0101	0101	0000	0001	1011	0000	31

图 11

此部分与前理相同，不再赘述

1.3.2 I/O PORT（ 10000110 10000001）

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
ALL	MEM→IR	00	1110	0000	0001	0000	0001	0000	0000	0000	0000	0000	0001	0000	02
ALL	/MAP	00	0010	0000	0100	0000	0001	0000	0000	0000	0000	0000	0000	0000	03
IN/OUT	PORT→AR	14	0011	0110	0100	0111	0001	0000	0000	0000	0000	0000	0011	0010	12
	R0→IO	30	0011	0000	0010	0011	0001	0000	0000	0000	0000	0000	0000	0001	13
ALL	STR→Q CC#=INT#	3A	0011	0010	010	0111	0000	0000	0000	0000	0000	0000	0000	0011	30
	PC→AR、PC+1→PC、CC#=0	02	0011	0000	0100	0011	0010	0000	0101	0101	0000	0001	1011	0000	31

图 12

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
ALL	MEM→IR	00	1110	0000	0001	0000	0001	0000	0000	0000	0000	0000	0001	0000	02

图 13

第一条微指令程序此部分与前理相同，不再赘述

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
ALL	/MAP	00	0010	0000	0100	0000	0001	0000	0000	0000	0000	0000	0000	0000	03

图 14

第二条微指令程序此部分与前理相同，不再赘述

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
IN/OUT	PORT→AR	14	0011	0110	0100	0111	0001	0000	0000	0000	0000	0000	0011	0010	12

图 15

第三条微指令程序：

CI3~0：0011 条件微转移

SCC30：0110 IR10=0 时，转移

0MRW：0100 无读写

0I2~0：0111 R：D，S：0

SA，I8~6：0001 Y：Y→F 因为要实现->AR，所以选 Y→F

SB、I5~3：0000 R + S

B 口：0000 R0
A 口：0000 R0
0SST：0000 四个标志位不变
SSH：00
SCI：01
Cin = 1 进位+1
DC2：0011 /GARH：AR 高位接收
DC1：0010 /ETOIB：16 位拓展符号到内部总线
指令功能：/GARH 信号使得 AR 高位接收总线上的数据，将数据寄存器中保存的内容发送至 AR 寄存器

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
	R0→IO	30	0011	0000	0010	0011	0001	0000	0000	0000	0000	0000	0000	0001	13

图 16

第四条微指令程序：
CI3~0：0011 条件微转移
SCC30：0000 必转移
0MRW：0010 无读写
0I2~0：0011 R：0，S：B
SA，I8~6：0001 Y：Y→F
SB、I5~3：0000 R + S
B 口：0000 R0
A 口：0000 R0
0SST：0000 四个标志位不变
SSH：00
SCI：00
Cin = 0 执行 add，cin=0
DC2：0000 NC = NC
DC1：0001 /ETOIB：ALU 输出到内部总线
指令功能：首先使得 ALU 的输出到内部总线也就是送到 I/O 总线上，将保存的方式选择控制字发送到 IO 上

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
ALL	STR→Q CC#=INT#	3A	0011	0010	0100	0111	0000	0000	0000	0000	0000	0000	0000	0011	30

图 17

第五条微指令程序此部分与前理相同，不再赘述

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
	PC→AR、PC+1→PC、CC#=0	02	0011	0000	0100	0011	0010	0000	0101	0101	0000	0001	1011	0000	31

图 18

第六条微指令程序此部分与前理相同，不再赘述

1.3.3 PUSH SR （ 10000101 00000000）

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
ALL	MEM→IR	00	1110	0000	0001	0000	0001	0000	0000	0000	0000	0000	0001	0000	02
ALL	/MAP	00	0010	0000	0100	0000	0001	0000	0000	0000	0000	0000	0000	0000	03
PSH/F	SP-1→SP,AR	1A	0011	0111	0100	0011	0011	0001	0100	0000	0000	0000	0000	0011	15
ALL	SP→MEM、CC#=0	30	0011	0000	0000	0100	1001	0000	0000	0000	0000	0000	0000	0001	1A
ALL	STR→Q CC#=INT#	3A	0011	0010	010	0111	0000	0000	0000	0000	0000	0000	0000	0011	30
	PC→AR、PC+1→PC、CC#=0	02	0011	0000	0100	0011	0010	0000	0101	0101	0000	0001	1011	0000	31

图 19

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
ALL	MEM→IR	00	1110	0000	0001	0000	0001	0000	0000	0000	0000	0000	0001	0000	02

图 20

第一条微指令程序此部分与前理相同，不再赘述

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
ALL	/MAP	00	0010	0000	0100	0000	0001	0000	0000	0000	0000	0000	0000	0000	03

图 21

第二条微指令程序此部分与前理相同，不再赘述

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
PSH/F	SP-1→SP,AR	1A	0011	0111	0100	0011	0011	0001	0100	0000	0000	0000	0011	0000	15

图 22

第三条微指令程序功能：

CI3~0：0011 条件微转移

SCC30：0000 必转移

OMRW：0100 无读写

OI2~0：0011 R：0，S：B

SA，I8~6：0011 Y：F F → B

SB、I5~3：0001 S - R

B 口：0100 R4

A 口：0000 R0

OSST：0000 四个标志位不变

SSH：00

SCI：00

Cin = 0,根据 AM2901 定义，当执行 SUB 运算时，Cin=0，表示-1，所以实现了 SP-1→SP

DC2：0011 /GARH：AR 高位接收

DC1：0000 /SWTOIB：开关到内部总线

指令功能：SP 就是 R4 寄存器，发送/GARH 信号使得 AR 高位接收总线上的数据，实现 SP-1→SP,AR

指令	操作功能	下地址	CI3~0	SCC30	OMRW	OI2~0	SA、I8~I6	SB、I5~I3	B口	A口	OSST	SSH、SCI	DC2	DC1	微地址
ALL	SP→MEM、CC#=0	30	0011	0000	0000	0100	1001	0000	0000	0000	0000	0000	0000	0001	1A

图 23

第四条微指令程序功能：

CI3~0：0011 条件微转移

SCC30：0000 必转移

OMRW：0000 无读写

OI2~0：0100 R：0，S：A

SA，I8~6：1001 Y：Y→F

SB、I5~3：0000 R + S

B 口：0000 R0

A 口：0000 R0

OSST：0000 四个标志位不变

SSH：00

SCI：00

Cin = 0

DC2：0000 NC

DC1：0001 /SWTOIB：ALU 输出到内部总线

指令功能：将当前 SP 指针的值放入之前存入 AR 所指定的内存单元中

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
ALL	STR→Q CC#=INT#	3A	0011	0010	0100	0111	0000	0000	0000	0000	0000	0000	0000	0011	30

图 24

第五条微指令程序此部分与前理相同，不再赘述

指令	操作功能	下地址	CI3~0	SCC30	0MRW	0I2~0	SA、I8~I6	SB、I5~I3	B口	A口	0SST	SSH、SCI	DC2	DC1	微地址
	PC→AR、PC+1→PC、CC#=0	02	0011	0000	0100	0011	0010	0000	0101	0101	0000	0001	1011	0000	31

图 25

第六条微指令程序此部分与前理相同，不再赘述

1.4 综述

8251 是通用同步/异步接收器/发送器，包装在 Intel 生产的 28 针 DIP 中。它通常用于串行通信，额定每秒传输速率为 19.2 KB。它通常与更常见的 8250 UART 混淆，后者在 IBM 个人计算机中被广泛用作串行端口。它包括 5 个部分读/写控制逻辑；发射机；接收者数据总线系统；调制解调器控制；

监控程序是指通过植入目标主机的程序动态地监视目标主机，实现将事件传输过去，进行一般操作的程序。监控程序有两种模式，一种是服务器端与客户端在同一台机器运行，另一种是服务器端与客户端在不同机器运行，通过客户端程序监控服务器端主机。

通过分析实验系统监控程序的前 7 条机器指令功能及对应的微程序，深刻理解了实验系统与计算机系统的连接方式，学习了可编程串行通信和接口芯 8251A 的工作原理，通过以上的学习，又加深了对于微程序在实际应用中的理解。

2 自主指令设计

2.1 设计指令 ADD DR,[ADR]

2.1.1 指令格式分析

汇编指令格式即：ADD DR, [ADR]；

机器指令格式由于软件中并未设计 ADD DR,[ADR]的指令或拓展指令，故占用数据来源类似的 LDRA DR,[ADR]微址来实现：

E4	DR	0000
ADR		

图 26

操作码机器码即：1110 0100（E4），LDRA 的操作码；

指令功能把 DR 的内容和内存中 ADR 地址所对应的内容相加后，将合存放在 DR 中

2.1.2 设计指令

下地址	Ci3~0	SCC	0MRW	0I2~0	SA I8~6	SB I5~3	B口	A口	0 SST	SSH SCI	DC2	DC1	OP	微地址
0000 0000	1110	0000	0100	0011	0010	0000	0101	0101	0000	0001	0011	0000	E4	5B
0000 0000	1110	0000	0001	0111	0011	1000	0000	0000	0000	0000	0000	0000	E4	5C
0000 0000	1110	0000	0100	0011	0001	1000	0000	0000	0000	0000	0011	0000	E4	5D
0000 0000	0010	0000	0001	0111	0011	1000	0001	0000	0000	0000	0000	0000	E4	5E
0011 0000	1110	0000	0100	0001	0011	1000	0000	0001	0000	0000	0000	0000	E4	5F

图 27

5B：PC->AR，PC+1->PC

5C：MEM -> RO

5D：RO -> AR

5E：MEM -> R1

5F：RO + R1 -> RO

这样设计的原因是发现倘若直接从 MEM 送入 AR，AR 得出 DR 模拟器会发送重启

2.1.3 具体实现

下地址	Ci3~0	SCC	0MRW	0I2~0	SA I8~6	SB I5~3	B口	A口	0 SST	SSH SCI	DC2	DC1
0000 0000	1110	0000	0100	0011	0010	0000	0101	0101	0000	0001	0011	0000

图 28

首先根据实验指导书 P50 找到 LDRA 指令的入口地址为 5B

操作功能：PC→AR、PC+1→PC。实现 PC 的值赋给 AR（地址寄存器），PC 自身实现自增，为之后的读操作中，找到 MAR 内的地址单元在主存中取出对应指令做准备

下址（00H）：顺序执行时无效

CI3-0（1110）：#14 命令，表示顺序执行

MRW（100）：无读写操作

SCC3-0（0000）：必转，无需条件即转移

I2-0（011）：因为 PC 既要被运算也要被输出，数据来源 0 和 B 口；

I8-6（010）：因为要实现->AR,PC，所以要输出到 PC 上，所以选 F->B,Y=A，

故为 010

I5-3（000）：执行 ADD 功能，R+S,即 0+PC;

B 口（0101）：R5,即 PC，B 口内容+1 送到 B 口

A 口（0101）：R5，即 PC，输出 A 口的内容到 AR，实现 PC→AR

SST（000）：无标志位的运算，所以为 000

SSHSCI（001）：PC 要执行自加 1，标志位 Cin=1，即实现 PC+1→PC

DC1（000）：送开关内容到内部总线,通过 16 位数据开关置入指令操作码

DC2（011）：需要 AR（地址寄存器）高位接收数据。PC（程序计数器）中的地址传递到 AR（地址寄存器）中，由 AR 选中内存中相应的地址。实现 PC→AR 功能

下地址	Ci3~0	SCC	0MRW	0I2~0	SA I8~6	SB I5~3	B口	A口	0 SST	SSH SCI	DC2	DC1
0000 0000	1110	0000	0001	0111	0011	1000	0000	0000	0000	0000	0000	0000

图 29

操作功能：MEM→R0。MEM 中存储的是形式地址（2100H），此条微指令是为了实现读取形式地址中对应的内存单元的内容放入指定寄存器中

下址（00H）：下一条微指令的入口地址

CI3-0（1110）：#14 命令，表示顺序执行

MRW（001）：向内存发出读信号

SCC3-0（0000）：必转，无需条件即转移

I2-0（111）：数据来源 0 和 D 口；

I8-6（011）：因为要实现 F->B，Y 输出 F

I5-3（000）：执行 ADD 功能，R+S,即 0+MEM;

B 口（0000）：R0，

A 口（0000）：R0，无意义

SST（000）：无标志位的运算，所以为 000

SSHSCI（000）：执行 add，cin=0

DC1（000）：送开关内容到内部总线,通过 16 位数据开关置入指令操作码

DC2（000）：不执行操作

下地址	Ci3~0	SCC	0MRW	0I2~0	SA I8~6	SB I5~3	B口	A口	0 SST	SSH SCI	DC2	DC1
0000 0000	1110	0000	0100	0011	0001	1000	0000	0000	0000	0000	0011	0000

图 30

操作功能：R0→AR。R0 中存储的是形式地址（2100H），此条微指令是为了实现指定寄存器中读取内存单元的内容放入 AR

下址（00H）：下一条微指令的入口地址

CI3-0（1110）：#14 命令，表示顺序执行

MRW（1010）：无读写操作

SCC3-0（0000）：必转，无需条件即转移

I2-0（011）：数据来源 0 和 B 口；

I8-6（001）：Y 输出 F。

I5-3（000）：执行 ADD 功能，R+S,即 0+MEM;

B 口（0000）：R0，
A 口（0000）：R0，无意义
SST（000）：无标志位的运算，所以为 000
SSHSCI（000）：执行 add，cin=0
DC1（000）：送开关内容到内部总线,通过 16 位数据开关置入指令操作码
DC2（011）：需要 AR（地址寄存器）高位接收数据。R0 中的地址传递到 AR（地址寄存器）中，由 AR 选中内存中相应的地址。实现 R0→AR 功能

下地址	Ci3~0	SCC	0MRW	0I2~0	SA I8~6	SB I5~3	B口	A口	0 SST	SSH SCI	DC2	DC1
0000 0000	0010	0000	0001	0111	0011	1000	0001	0000	0000	0000	0000	0000

图 31

操作功能：MEM→R1。MEM 中存储的是形式地址（2100H）指定的数据，此条微指令是为了实现指定寄存器中暂存接下来计算要使用的数据
下址（00H）：下一条微指令的入口地址
CI3-0（1110）：#14 命令，表示顺序执行
MRW（001）：向内存发出读信号
SCC3-0（0000）：必转，无需条件即转移
I2-0（111）：数据来源 0 和 D 口；
I8-6（011）：因为要实现 F->B，Y 输出 F
I5-3（000）：执行 ADD 功能，R+S,即 0+MEM;
B 口（0001）：R1，
A 口（0000）：R0，无意义
SST（000）：无标志位的运算，所以为 000
SSHSCI（000）：执行 add，cin=0
DC1（000）：送开关内容到内部总线,通过 16 位数据开关置入指令操作码
DC2（000）：不执行操作

下地址	Ci3~0	SCC	0MRW	0I2~0	SA I8~6	SB I5~3	B口	A口	0 SST	SSH SCI	DC2	DC1
0011 0000	1110	0000	0100	0001	0011	1000	0000	0001	0000	0000	0000	0000

图 32

指令功能：完成 R0+R1→R0。本条微指令是将上一步得到的 R1 的值与目的寄存器相加，并将结果赋给 R0，通过运算器实现 R0+R1→R0 。本条微指令执行结束后，指令的执行周期已经结束，即 LDRA 指令的功能已经完成，所以按照指令周期的流程就需要检查是否有中断请求，根据必转所以转到下址字段中的微指令地址 30H。
下址（30）：下址操作是公共操作
CI3-0（1110）：#14 命令，表示顺序执行。
SCC（0000）：必转，无需条件即转移。
MRW（100）：无读写操作
I2-0（001）：数据来源 A 和 B
I8-6（011）：Y 输出 F 且把输出结果送到 B 口；
I5-3（000）：执行 ADD 功能

B 口（0000）：R0
A 口（0001）：R1
SST（001）：接收 ALU 的标志位输出的值
SSHSCI（000）：执行 ADD 指令，Cin=0
DC1（000）：送开关内容到内部总线
DC2（000）：不操作

2.1.4 编写测试程序

```
>e 2000
2000    8800:e400  0000:2100
>e 2100
2100    0036:0036
>a 2002
2002: ret
2003:
```

图 33 汇编测试程序

```
>g 2000

R0=0000  R1=0000  R2=0000  R3=0000  SP=2780  PC=2000  R6=0000  R7=0000  R8=0000
R9=0000 R10=0000 R11=0000 R12=0000 R13=0000 R14=2612 R15=0000   F=00100000
>e 2000
2000    8800:e400  0000:2100
>e 2100
2100    0036:0036
>a 20002
2000:
>
>a 2002
2002: ret
2003:
>d 2000
2000    E400  2100  8F00  0000  0000  0000  0000  0000  ..!.............
2008    0000  0000  0000  0000  0000  0000  0000  0000  ................
2010    0000  0000  0000  0000  0000  0000  0000  0000  ................
2018    0000  0000  0000  0000  0000  0000  0000  0000  ................
2020    0000  0000  0000  0000  0000  0000  0000  0000  ................
2028    0000  0000  0000  0000  0000  0000  0000  0000  ................
2030    0000  0000  0000  0000  0000  0000  0000  0000  ................
2038    0000  0000  0000  0000  0000  0000  0000  0000  ................
2040    0000  0000  0000  0000  0000  0000  0000  0000  ................
2048    0000  0000  0000  0000  0000  0000  0000  0000  ................
2050    0000  0000  0000  0000  0000  0000  0000  0000  ................
2058    0000  0000  0000  0000  0000  0000  0000  0000  ................
2060    0000  0000  0000  0000  0000  0000  0000  0000  ................
2068    0000  0000  0000  0000  0000  0000  0000  0000  ................
2070    0000  0000  0000  0000  0000  0000  0000  0000  ................
>g 2000

R0=0036  R1=0000  R2=0000  R3=0000  SP=2780  PC=2000  R6=0000  R7=0000  R8=0000
R9=0000 R10=0000 R11=0000 R12=0000 R13=0000 R14=2612 R15=0000   F=00100000
>
```

图 34 程序运行结果

可见 R0 寄存器已被修改为 R0 + [2100]的值为 36，结果正确，故微程序设计正确

ADD DR, [2100]，表示将形式地址内存单元 2100 的内容通过寻址给寄存器 R1，里面的值为 0003H。首先程序的入口地址为 2000，通过 2000H 首先执行 MVRD 指令给目的寄存器 R0 赋初值 0036H，由于 MVRD 是双字长指令，执行完之后转到地址 2002,2002 找到了指令的操作码 E4,根据 E4 找到微程序的入口地址 2100H，在微程序过程中， 内存中 ADR 地址所对应的内容存放在 R1,R1 与 R0 相加之后将结果存放在 R0， 所以微程序执行结束之后，R0 是 R0 与 R1 的和，最后结束程序。

2.2 设计指令 ADD DR, @ADR

2.2.1 指令格式分析

汇编指令格式即：ADD DR, @[ADR]；

机器指令格式由于软件中并未设计 ADD DR,@[ADR]的指令或拓展指令，故占用数据来源类似的 LDRA DR,[ADR]微址来实现：

E4	DR	0000
ADR		

图 35

操作码机器码即：1110 0100（E4），LDRA 的操作码；

指令功能在内存中，ADR 地址所对应的内容为第二个操作数的形式地址，根据此地址，可以通过间接寻址读取第二个操作数。两个操作数相加后，将合存放在 DR 中

2.2.2 设计指令

下地址	Ci3~0	SCC	0MRW	0I2~0	SA I8~6	SB I5~3	B口	A口	0 SST	SSH SCI	DC2	DC1	OP	微地址
0000 0000	1110	0000	0100	0011	0010	0000	0101	0101	0000	0001	0011	0000	E4	5B
0000 0000	1110	0000	0001	0111	0011	1000	0000	0000	0000	0000	0000	0000	E4	5C
0000 0000	1110	0000	0100	0011	0001	1000	0000	0000	0000	0000	0011	0000	E4	5D
0000 0000	0010	0000	0001	0111	0011	1000	0000	0000	0000	0000	0000	0000	E4	5E
0000 0000	1110	0000	0100	0011	0001	1000	0000	0000	0000	0000	0011	0000	E4	5F
0000 0000	1110	0000	0001	0111	0011	1000	0001	0000	0000	0000	0000	0000	E4	60
0011 0000	0011	0000	0100	0001	0011	1000	0000	0001	0000	0000	0000	0000	E4	61

图 36

5B：PC->AR，PC+1->PC

5C：MEM -> RO
5D：RO -> AR
5E：MEM -> RO
5F: RO -> AR
60: MEM -> R1
61: RO + R1 -> RO

2.2.3 具体实现

与之前部分相同，实现的功能也相似。需要注意的是，5C 中 MEM 存储的是形式地址（2100H），指令实现的是读取形式地址中的有效地址（2200H）单元，需要在 5E 实现读取有效地址中的值之后在进行 60 操作，将读到的值赋给 R1

所有微指令在之前部分已经解释过，不再赘述。

2.2.4 编写测试程序

```
>e 2100
2100    2200:2200
>e 2200
2200    0036:0046
>e 2000
2000    8800:e400  0000:2100
>a 2002
2002: ret
2003:
```

图 37 汇编测试程序

```
>g 2000

R0=0000  R1=0000  R2=0000  R3=0000  SP=2780  PC=2000  R6=0000  R7=0000  R8=0000
R9=0000 R10=0000 R11=0000 R12=0000 R13=0000 R14=2612 R15=0000   F=01000000
>e 2100
2100    2200:2200
>e 2200
2200    0036:0046
>e 2000
2000    8800:e400  0000:2100
>a 2002
2002: ret
2003:
>g 2000

R0=0046  R1=0000  R2=0000  R3=0000  SP=2780  PC=2000  R6=0000  R7=0000  R8=0000
R9=0000 R10=0000 R11=0000 R12=0000 R13=0000 R14=2612 R15=0000   F=01000000
>
```

图 38 程序运行结果

可见 R0 寄存器已被修改为 R0 +[[2100]]的值为 46，结果正确，故微程序设计正确

ADD DR, @[2100]，表示通过间接寻址将形式地址内存单元 2100 的内容给寄存器 R0，其中形式地址为 2100H，存放的有效地址为 2200H，2200H 存放的是操作数 0046H，首先程序的入口地址为 2000，通过 2000H 首先执行 MVRD 指令给目的寄存器 R0 赋初值 0001H，由于 MVRD 是双字长指令，执行完之后转到地址 2002，2002 找到了微指令的操作码 E4，根据 E4 找到微程序的入口地址 2100H，在微程序过程中， 内存中 ADR 地址所对应的内容存放在 R1，R1 与 R0 相加之后将结果存放在 R0， 所以微程序执行结束之后，R0 是 R0 与 R1 的和，最后结束程序。

2.3 设计指令 SUB DR,@[ADR]

2.3.1 指令格式分析

汇编指令格式即：SUB DR, @[ADR]；

机器指令格式由于软件中并未设计 SUB DR,@[ADR]的指令或拓展指令，故占用数据来源类似的 LDRA DR,[ADR]微址来实现：

图 39

操作码机器码即：1110 0100（E4），LDRA 的操作码；

指令功能在内存中，ADR 地址所对应的内容为第二个操作数的形式地址，根据此地址，可以通过间接寻址读取第二个操作数。两个操作数相减后，将合存放在 DR 中

2.3.2 指令设计

下地址	Ci3~0	SCC	0MRW	0I2~0	SA I8~6	SB I5~3	B口	A口	0 SST	SSH SCI	DC2	DC1	OP	微地址
0000 0000	1110	0000	0100	0011	0010	0000	0101	0101	0000	0001	0011	0000	E4	5B
0000 0000	1110	0000	0001	0111	0011	1000	0000	0000	0000	0000	0000	0000	E4	5C
0000 0000	1110	0000	0100	0011	0001	1000	0000	0000	0000	0000	0011	0000	E4	5D
0000 0000	0010	0000	0001	0111	0011	1000	0000	0000	0000	0000	0000	0000	E4	5E
0000 0000	1110	0000	0100	0011	0001	1000	0000	0000	0000	0000	0011	0000	E4	5F
0000 0000	1110	0000	0001	0111	0011	1000	0001	0000	0000	0000	0000	0000	E4	60
0011 0000	0011	0000	0100	0001	0011	1010	0000	0001	0000	0000	0000	0000	E4	61

图 40

5B：PC->AR，PC+1->PC
5C：MEM -> R0
5D：R0 -> AR
5E：MEM -> R0
5F：R0 -> AR
60：MEM -> R1
61：R0 - R1 -> R0

2.3.3 具体实现

与之前部分相同，实现的功能也相似。需要注意的是，5C 中 MEM 存储的是形式地址（2100H），指令实现的是读取形式地址中的有效地址（2200H）单元，需要在 5E 实现读取有效地址中的值之后在进行 60 操作，将读到的值赋给 R1

前六条微指令在之前部分已经解释过，不再赘述。目前分析第七条指令

下地址	Ci3~0	SCC	0MRW	0I2~0	SA I8~6	SB I5~3	B口	A口	0 SST	SSH SCI	DC2	DC1
0011 0000	0011	0000	0100	0001	0011	1010	0000	0001	0000	0000	0000	0000

图 41

指令功能：完成 R0-R1→R0。本条微指令是将上一步得到的 R1 的值与目的寄存器相减，并将结果赋给 R0，通过运算器实现 R0-R1→R0 。本条微指令执行结束后，指令的执行周期已经结束，即 LDRA 指令的功能已经完成，所以按照指令周期的流程就需要检查是否有中断请求，根据必转所以转到下址字段中的微指令地址 30H。

下址（30）：下址操作是公共操作
CI3-0（1110）：#14 命令，表示顺序执行。
SCC（0000）：必转，无需条件即转移。
MRW（100）：无读写操作
I2-0（001）：数据来源 A 和 B
I8-6（011）：Y 输出 F 且把输出结果送到 B 口；
I5-3（010）：执行 SUB 功能
B 口（0000）：R0
A 口（0001）：R1
SST（001）：接收 ALU 的标志位输出的值
SSHSCI（000）：执行 ADD 指令，Cin=0
DC1（000）：送开关内容到内部总线
DC2（000）：不操作

2.3.4 编写测试程序

```
>e 2100
2100    2200:2200
>e 2200
2200    0036:0046
>e 2000
2000    8800:e400  0000:2100
>a 2002
2002: ret
2003:
```

图 42 汇编测试程序

```
TEC-2000 CRT MONITOR
Version 2.0   2001.10
Computer Architectur Lab.,Tsinghua University
Copyright Jason He
>e 2100
2100    0000:2200
>e 2200
2200    0000:36
>e 2000
2000    0000:e400  0000:2100
>a 2002
2002: ret
2003:
>g 2000

R0=FFC9  R1=0000  R2=0000  R3=0000  SP=2780  PC=2000  R6=0000  R7=0000  R8=0000
R9=0000 R10=0000 R11=0000 R12=0000 R13=0000 R14=2612 R15=0000   F=11111111
```

图 43 程序运行结果

R0 原值为 0，可见 R0 寄存器已被修改为 R0 - [[2100]]的值为 FFC9，结果正确，故微程序设计正确

SUB DR, @[2100]，表示通过间接寻址将形式地址内存单元 2100 的内容给寄存器 R0，其中形式地址为 2100H，存放的有效地址为 2200H，2200H 存放的是操作数 0046H，首先程序的入口地址为 2000，通过 2000H 首先执行 MVRD 指令给目的寄存器 R0 赋初值 0001H,由于 MVRD 是双字长指令,执行完之后转到地址 2002，2002 找到了微指令的操作码 E4，根据 E4 找到微程序的入口地址 2100H，在微程序过程中， 内存中 ADR 地址所对应的内容存放在 R1,R1 与 R0 相减之后将结果存放在 R0， 所以微程序执行结束之后，R0 是 R0 与 R1 的和，最后结束程序。

2.4 综述

微指令（英语：microcode），又称微码，是在 CISC 结构下，运行一些功能复杂的指令时，所分解一系列相对简单的指令。微指令的作用是将机器指令与相关的电路实现分离，这样一来机器指令可以更自由的进行设计与修改，而不用考虑到实际的电路架构。与其他方式比较起来，使用微指令架构可以在降低电路复

杂度的同时，建构出复杂的多步骤机器指令。撰写微指令一般称为微程序设计（microprogramming），而特定架构下的处理器实做中微指令有时会称为微程序（microprogram）。

现代的微指令通常由 CPU 工程师在设计阶段编写，并且存储在只读存储器（ROM, read-only-memory)或可编程逻辑数组(PLA, programmable logic array）中。然而有些机器会将微指令存储在静态随机存取存储器(SRAM)或是闪存(flash memory）中。它通常对普通程序员甚至是汇编语言程序员来说是不可见的，也是无法修改的。与机器指令不同的是，机器指令必须在一系列不同的处理器之间维持兼容性，而微指令只设计成在特定的电路架构下运行，成为特定处理器设计的一部分。

通过自行设计微程序，深刻理解了计算机系统的工作方式，理解了计算机的指令系统，CPU 的结构和功能，控制单元的功能，控制单元的设计，真正学会了利用计算机编写固件的能力。在动手实践中，学会了动态的调试程序，学会了编写测试程序，加深了对于微程序在实际应用中的理解。