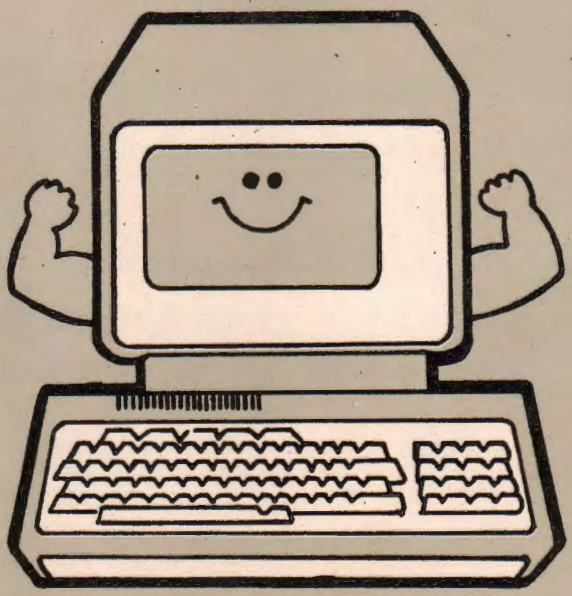
CMC-80

微型电脑使用手册



株洲电子研究所
香港金山公司

前　言

本《使用手册》共分四章。第一章概述CMC—80微型电脑的主要技术特性、功能、操作步骤及使用注意事项；第二章是CMC—80的硬件结构和原理说明，主要介绍了其中的大规模集成电路器件；第三章结合实际操作较为详细地叙述了CMCBUG监控程序(版本2.1)所赋于的键命令功能及其使用方法；第四章为一些实用程序举例，以供入门者学习一些程序设计的方法和技巧。

微型电脑应用软件的编制往往使用手编程序。为了方便广大用户，我们同时提供美国MOSTEK公司编的《Z80袖珍设计手册》，其中编入了Z80CPU的指令系统和Z80系列其它芯片的编程要点。

初次涉猎于微型电脑领域的读者，可以结合我们编印的《Z80微型电脑基础知识》来阅读本手册的内容；打算进一步研究CMC—80硬件结构及监控程序的读者，可以参考《CMC—80微型电脑技术手册》。

本手册系再版，它与初版的差别主要在于介绍的CMCBUG监控程序版本不同，键命令也有些差异，请单独购买本手册的读者注意。

由于我们水平有限，错误之处在所难免，恳请读者不吝指正。

目　　录

第一章　CMC——80微型电脑导论

§1.1　引　言

七十年代以来，微处理器和微计算机的发展十分迅速。它的应用已深入到工业、农业、国防、科研、教育、管理以及家庭生活等各个领域，在自动控制和仪器仪表方面的应用尤为突出。随着大规模集成电路的发展，微型电脑必将对现代社会产生深远的影响。

CMC—80微型电脑是采用功能较强的Z—80系列器件精心设计的双板型计算机，主要的微处理机器件都安装在主板上，辅助板上安装键盘、显示器和直接相关的驱动器件。用户在学习微计算机和利用键盘、显示器开发应用程序时，可将二板用电缆联结起来使用，在编制好程序并写入EPROM后，运行时可撤除辅助板，以简化结构，提高可靠性，特别适合初始设备制造应用。

CMC—80微型电脑可用于生产过程控制、各种仪器仪表或机械设备的数字控制、数据处理等。它既可独立应用在小型自动控制系统中，又可用在分布式控制系统的第一线。该机尤其适合于初学者学习微型电脑的硬件、指令系统、编写程序的方法和技巧。因此，对大专院校学生和各行各业需要应用微型电脑的科技工作者来说，CMC—80微型电脑也是一种经济实用的实验教学设备。

CMC—80微型电脑具有简易的开发功能，具有22条监控命令，还有留给用户的2个命令键，可由用户编制好所需的处理程序然后用命令键调用。监控程序可为用户提供单步，设置（最多5个）断点，检查和修改存储器、寄存器、外围接口的数据，检索数据，追踪、追溯程序的运行，将程序写入EPROM（2716/2758），转储到磁带上和从磁带读入程序到存储器等多种功能。

CMC—80微型电脑具备较强的扩展功能，增配接口板后可与键盘、显示器、打印机等外部设备相衔接，就可构成一种经济实用、功能更为完善的小型微电脑系统。

§1.2　主要技术特性

1.中央处理单元为Z80—CPU

2.时钟频率为1.9968MHZ，晶振的频率为3.9936MHZ。

3.RAM为8K字节2114静态读写存储器。

4.EPROM插座四个，其中第一片EPROM内装CMC—80的监控程序CMCBUG，其余三片留给用户使用。如用户不使用CMCBUG，则全部均可由用户使用，总容量为8K字节。

5.Z80—PIO两片，一共有8位并行可编程I/O接口四个，全供用户使用。

6.Z80—SIO一片，具有串行同步/异步可编程通信接口两个，全供用户使用。

7.Z80—CTC一片，具有可编程计数/定时通道四个。

8.ADC0809一片，为8通道8位A/D转换器，全供用户使用。

9.按键共28个，16个为十六进数字键；12个为命令键，包括：

MEM/DISP（存储器检查/计算转移偏移量）

REG/REG′（寄存器检查/辅助寄存器检查）

NEXT/LOAD（下一组/磁带输入）

LAST/DUMP（上一组/转储磁带）

PROM/USR0（写EPROM/用户子程序0）

MOVE/USR1（存储器传送/用户子程序1）

PORT/TIME（口检查/实时钟）

BRPT/LOAD?（设置断点/转储检查）

STEP/STEP′（单步/主程序单步）

EXEC/SECH（执行/检索）

MON′/MON（监控和上下挡控制）

TRAC/TRAC′（追踪/主程序追踪）

10.六位LED数字显示，通常左四位显示地址，右两位显示数据。

11.配有音频盒式磁带机接口。

12.电源为+5V±5%，1.5A；若对EPROM写入，尚需接入+25V±1V，30mA电源。

§1.3 功能简介

CMC—80使用Zilog公司的Z80系列器件。Z80—CPU的指令系统较强，共有指令158条，包容了INTEL8080A的全部指令，还增加了位操作(置1、置0，测试)、相对转移，变址寻址、16位数据传送和运算，数据块传送与查找等功能。Z80—CPU内部寄存器增加了两个变址寄存器IX和IY，并增设了一套辅助寄存器。这使得Z80的处理功能大为增强，易于进行程序设计。为8080A设计的软件完全可以在Z80上运行。

本机备有8K字节静态读写存储器，足够一般控制和用户实验的需要。配有四个EPROM插座，最多可插入四片2716，容纳8K字节的程序。目前监控程序CMCBUG占了2K字节，提供给用户初步的开发功能。

本机有完整的EPROM编程功能，允许用户把调试好的程序写入2716或2758EPROM中，写入EPROM时还需另加25V电源，电流为30mA。

本机配有音频盒式磁带机接口，用转录线将盒式磁带录音机与CMC—80微型电脑机上插孔相连，可以方便地存取数据和程序。本机上的红色发光二极管能指示磁带上有无信息。利用这只二极管的指示，可以在一盘磁带上分段录制若干文件。存取磁带均由专用命令键控制。

本机辅助板上装有小型键盘和LED数码显示器，28键键盘包括16个十六进制数字键和12个命令键，其功能在第三章介绍。

本机配有三个按钮和开关：按钮S_1用来使整机复位（RESET），开关S_2如果置向

MON RST位置，则复位后，显示器上出现提示符“—”，进入监控状态，准备接受用户命令键；如果S_2置向PROM$_1$ RST位置，则在复位后经状态初始化便进入PROM$_1$中的用户程序。开关S_3用来选择EPROM$_3$中的存储器处于“PGM”（写入）或“READ”（读出）的状态。

此外，主板的左、右、后三方有三个40脚印刷电路板插头，可与国产2CY25Z—40型插座配合，它们把Z80总线、PIO、SIO、ADC、CTC的信息线引出板外，以便用户接入信号和扩充系统。

§1.4 使用CMC—80微型电脑的准备知识

1.4.1 使用MOS器件的注意事项

CMC—80上的集成电路是用MOS工艺制成的，它如果接触到高压电平（包括电源高压和感应静电高压），就会立即受到损伤甚至损坏。

人体上的静电感应往往不被注意，因而是最危险的静电电压，在接触CMC—80内集成电路之前应小心地进行身体放电。未经训练的人不得从电路板上取下集成电路和触摸它们的引脚。

不接地或接地不良的测试设备和电烙铁都是潜在的危险电压源，当需要对本机进行测试和维修时，应将所有的测试设备和电烙铁可靠地接地。

1.4.2 供电要求

CMC—80正常工作只需要单一的+5伏稳压电源，任何一种具有+5V，1.5A和适当过载保护的直流稳压电源都适用，出售CMC—80微型电脑的经销单位也备有该种电源供需要的用户选购。

1.4.3 开机前的连线和设定开关状态。

将CMC—80微型电脑的电源线接到稳压电源的+5伏输出。

如果需使用盒式录音机，则用转录线将本机上的AUX插孔与录音机上的MIC插孔相连；将本机上的EAR插孔与录音机上的MONITOR OUT或EARPHONE插孔相连。

如不需直接进入EPROM1中的用户程序，则将开关S_2设定在MON RST位置。如希望开机或复位后直接进入用户程序，则其入口地此应为0800，将S_2板到PROM1 RST位置即可。

如不需对EPROM进行写入，则将开关S_3设定在READ位置。

1.4.4 通电开机

接通直流稳压电源（打开它的开关）；

此时，最左边的显示器上应显示“—”，表明已进入监控程序并准备接受用户命令。

如不出现“—”，请按下RESET按钮S_1，使计算机进入复位状态，此时应显示“—”。

如果仍不出现“—”，则立即关闭电源，检查连线和开关位置设定是否出错，器件与插座接触是否良好，直流稳压电源是否输出+5伏等。

如果再次检查和重复通电之后，CMC—80仍未出现正常反应，则请送往维修部门。

第二章　CMC——80系统概要

本章将详细地叙述CMC—80微型电脑的硬件和软件，使你了解它的工作原理，包括主要组件的功能，存储空间的分配和监控程序的概述。关于监控命令的使用详见第三章。

§2.1　CPU和主要的LSI器件

CMC—80的原理框图见图2.1

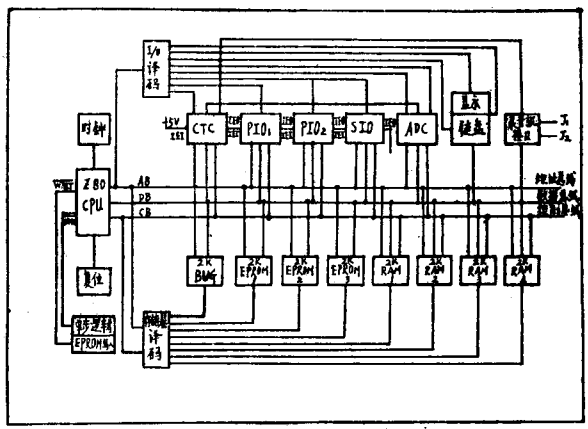


图2.1　CMC—80微型电脑原理框图

CMC—80采取总线式结构，它有三条总线：数据总线（D）、地址总线（A）、控制总线（C）、主要器件均挂在这三条总线上，包括以下三种：

1.中央处理器：Z80—CPU；

2.存储器：RAM、EPROM；

3.接口电路：Z80—PIO、Z80—SIO、Z80—CTC、ADC—0809，键盘接口、显示器接口、录音机接口等。

此外，还有一些辅助电路，如时钟、译码、复位等。详细电原理图见《CMC—80微型电脑技术手册》。

2.1.1　Z80—CPU

CMC—80微型电脑的中央处理器是Z80—CPU，它封装在40脚双列直插塑料外壳中，Z80—CPU采用三总线结构：地址总线16位；双向数据总线8位，控制总线13位。详细了解其性能请参考《Z80—CPU技术手册》。

2.1.1.1　总线结构

1.地址总线：CPU的地址总线用来指定进行信息交换的存储器地址和外围接口地址。在本系统中，CPU是唯一的源地址发生器，即所有地址码均是由CPU发出的。地址总线宽度

为16，故CPU最大直接访存容量为64K字节。

当CPU对外围接口进行数据交换时，仅地址总线的低八位输出有效地址，故访问I/O接口的最大容量为256个（每个8位）。

2.数据总线：CPU与存储器或接口之间通过8位宽的双向数据总线交换信息。

3.控制总线：13条控制总线按其功能可分为四类：4条与CPU状态有关；两条与总线状态有关；两条与中断有关；5条与信息交换有关。下面先叙述与信息交换有关的控制总线信息：

表明地址总线上发送的信息性质的有两条控制线：$\overline{MREQ}$和$\overline{IORQ}$，前者表示地址总线上发送的是存储器地址，准备与存储器进行信息交换；后者表示地址总线低八位上发送的是I/O接口地址，CPU要求与I/O接口电路交换信息。

表明数据总线上交换信息方向的有两条控制线：$\overline{RD}$和$\overline{WR}$，当CPU要把数据送往存储器或I/O接口时，就发出$\overline{WR}$信号，称为“写”，反之，CPU要输入数据时，发出$\overline{RD}$信号，称为“读”。

控制线$\overline{RFSH}$用来控制动态RAM的刷新，本系统此信号无用。

2.1.1.2 状态信息和控制

控制CPU状态的有两个信号：$\overline{RESET}$使CPU进入复位状态，并从地址0000开始执行复位程序；$\overline{WAIT}$使CPU增补延时的时钟周期，等待所访求信息的到来。

表明CPU所处状态的有两个信号：$\overline{M_1}$表明CPU正在一条指令的取指周期；$\overline{HALT}$表明CPU执行“等待”（HALT）指令，等待着中断的发生。

还有两个控制信号，$\overline{BUSRQ}$和$\overline{BUSAK}$，如果在总线上还挂有主动设备，则利用这两个信号和CPU交换信息以申请使用总线的权利。本系统不使用这两个信号。

2.1.1.3 中断结构

Z80—CPU有两条中断申请输入线：中断请求（$\overline{INT}$）和非屏蔽中断（$\overline{NMI}$）。

中断处理是微处理机系统设计的一个关键问题。Z80系列电路芯片对系统中断的安排较为灵活，为用户提供了较大的方便。

中断请求（$\overline{INT}$）信号被CPU接受后，它将现行指令计数器（PC）的内容保护入栈，置中断屏蔽后，可随用户的要求按如下三种方式之一进行处理。

方式零：接受申请中断的器件从数据总线上发来的一条单字节指令(通常是RST指令)，然后执行这一条指令。这是8080A采取的中断处理方式。

方式一：从地址0038开始执行中断处理程序。

方式二：从数据总线上接受申请中断的器件发来的低八位地址码，与CPU中I寄存器内保存的高八位地址码组合在一起，作为访存地址，取内存中连续存放的两个字节，作为中断处理程序的入口地址。

这三种方式的选择由专门指令来进行。

中断请求是否为CPU接受，还要受CPU内中断屏蔽位（IFF）的影响，如果屏蔽位置0，则CPU不理采$\overline{INT}$上的信号。IFF可由专门指令置1或清0。

非屏蔽中断（$\overline{NMI}$）的请求与此不同，它不受屏蔽位的影响，CPU无条件地接受此中断，保护PC内容后，转到地址0066去执行处理程序。

2.1.2 Z80—PIO

Z80—PIO是与Z80—CPU相匹配的八位并行通用可程输入/输出接口，每片电路上有两个通道，CMC—80微型电脑内直接装有两片PIO，四个并行I/O通道全部留给用户使用。

PIO的每个通道都可由CPU送来的控制字控制选择几种工作方式。A通道可以有四种工作方式：8位输入、8位输出、B位双向和按位任选输入输出；B通道只有三种工作方式：8位输入、8位输出，按位任选输入输出。

PIO的每个通道和外部设备之间除了八条数据线用于交换信息而外，还有两条建立通讯时序关系的应答信号线。每片PIO与CPU之间除了八位数据总线交换信息而外，还有三条地址线，用于选片和选定通道与信息种类（数据或控制信号），另有三条控制线控制PIO的工作方式。

PIO的中断系统是与CPU相配合的，适合于采用功能较灵活的中断方式2，本片申请中断由CPU接受之后，就将片内存放的、由程序预先置定的中断向量（低八位）送上数据总线，以供CPU取得中断处理程序入口。芯片上还有两条信号线用于构成键式优先权中断结构。

每片PIO在I/O地址空间中占用四个地址。CMC—80的两片PIO占用I/O地址10000000～10000111。PIO的详细说明请参看《Z80—PIO技术手册》。

2.1.3 Z80—SIO

Z80—SIO是与Z80—CPU相配的串行同步/异步可程通讯接口，每片上有两个通道。CMC—80微型电脑直接装有一片SIO，两个串行通道全部留给用户使用。

SIO的工作也完全由CPU送来的控制字所决定，每个通道都可以选择同步或异步通讯方式，同步通讯时由程序指定同步方式，内同步字符，校检方式等状态条件；异步通讯时由程序指定字符格式，校验方法，收发速率等状态条件。

SIO的每个通道与外部通讯设备之间除了信息交换线外，还有建立通讯交换关系和控制调制解调设备的8根信号线（由于封装管脚的限制，B通道少一根信号线）。它与CPU之间交换信息的总线使用情况与PIO相似。

SIO采取的中断结构也与PIO一样，特别适用于CPU的中断方式2。每片SIO占用I/O地址空间的四个地址，CMC—80机上SIO占用地址10001000～10001011。

2.1.4 Z80—CTC

Z80—CTC是Z80系列里的4通道计数/定时器，它每个通道的工作完全由CPU送来的控制字确定。在计数工作方式下，它读取外部的计数脉冲并进行计数；计数达到给定值时输出一个信号并可向CPU发出中断请求；在定时工作方式下，它可由内部或外部信号启动计时，计时达到给定值时输出一个信号并可向CPU发出中断请求。

CTC的每个通道与外部设备有两条信号线，一条用于输入计数或外触发脉冲，另一条用于输出计数/定时达到给定值的信号（由于封装管脚的限制，通道3没有输出信号）。它与CPU的信息交换、中断结构和占用I/O地址空间的情况均与PIO相似。CMC—80板上CTC占用地址10001100～10001111。

目前的监控程序使用了CTC的四个通道。TIME命令使用通道0作为实时钟；DUMP命令使用通道1作为调频信号发生器；单步，执行和FPROM写入命令使用了通道2；LOAD和LOAD？命令使用通道3作为波特定时器。

用户如果要使用CTC通道，如果不需使用实时钟，则可利用CTC 0，其中断服务程序入口在3FDA—3FDC(通常可在此处送入一条转移指令，转向实际服务程序的起始地址)，这时不需另行送中断向量。（每次进入监控初态时已送了中断向量）。

如果需用多于1个CTC通道，则用户可自行另送中断向量，原则上其它通道也可使用。但CTC2的C/T2已与辅助板连接，ZC2接往CPU的$\overline{NMI}$输入，CTC3的C/T3已与ADC0809的EOC连接，应避免信号冲突和引起意外的中断请求。

2.1.5 ADC0809

ADC0809是八通道八位模拟/数字转换电路，采用金属栅CMOS工艺制成，每个通道典型的转换时间约为100微秒（采用640KHz钟频），但如果只转换一个通道，则转换时间可下降到约50微秒。

ADC0809可以接收外部送来的八个单端模拟信号，在信号变化速度较慢或外加取样保持电路的情况下，也可利用软件处理来进行四对差分模拟输入的转换。另外，参考电阻网络的正负基准电平是与本机电源分开的，分别由两个管脚（REF(+)REF(-)）由外部送给电路，这样便增加了系统的灵活性，当输入信号变化幅度较小时，可减小V_{REF} = REF(+)—REF(-)的值，提高转换精度。但由于工艺方面的限制，对基准电平的给定也有一定的限制，包括$0.5V < V_{REF} \leqslant V_{CC}$（其中$V_{CC}$系电路的电源电平）和

$$\left|\frac{REF(+)+REF(-)-V_{CC}}{2}\right| < 0.1V$$ 等

0809与微电脑系统总线交换信息的管脚有16个。其中有8个连到数据总线($D_0 \sim D_7$)，用于把转换后的数字量送给CPU；有三个连到地址总线(A、B、C)，用于选择进行转换的模拟输入(八中选一)；时钟(CLOCK)输入转换的节拍，最高可达640KHz；地址闩锁选通（ALE）输入脉冲把管脚A、B、C上给定的输入通道地址锁入寄存器，接通选定的输入信号；启动(START)信号控制0809开始进行A/D转换；转换结束（EOC）信号通知CPU一次A/D转换已完成，可将数据送往总线，该信号通常接到中断线路中去；允许输出(OE)信号控制0809将转换好的数据送上系统数据总线。

由于0809不是Z80系列电路片，它无法按照Z80—CPU的中断方式0或中断方式2的要求送出特殊指令或中断向量，也不能与其它Z80系列外围片直接构成优先中断键。CMC—80中将0809的EOC接到Z80—CTC的CLK/TRG3，以和谐地纳入Z80的中断系统内。

§2.2 存储器，输入输出接口和地址空间分配

2.2.1 存储器和存储空间分配

Z80—CPU可直接访问的存储空间为64K字节。CMC—80板上共使用16K字节的存储器，其中8K字节为只读存储器，（适用2716，如采用2758，则只有4K字节）8K字节为读写存储器。

2.2.1.1 存储分配

CMC—80板上使用地址为0000～3FFF（十六进制）：

其中 0000～07FF 装有CMCBUG监控程序(2K)；

0800～0FFF　为2K字节的EPROM1；

1000～17FF　为2K字节的EPROM2；

1800～1FFF　为2K字节的EPROM3，这三块共计6K字节的EPROM均可由用户存放应用程序，如果用户不需监控程序，则总共可使用8K字节的程序，CMCBUG还提供了向插在EPROM3位置的2716或2758进行写入的能力。

2000～3FFF　为8K字节的RAM

存储分配见表2.1和表2.2

地　址	器　件	A_{15}～A_{11}	A_{10}～A_0	译码器输出
3800～3FFF	2KRAM	00111	可　变	$\overline{Y_7}=\overline{CS7}$
3000～37FF	2KRAM	00110	可　变	$\overline{Y_6}=\overline{CS6}$
2800～2FFF	2KRAM	00101	可　变	$\overline{Y_5}=\overline{CS5}$
2000～27FF	2KRAM	00100	可　变	$\overline{Y_4}=\overline{CS4}$
1800～1FFF	2KEPROM3	00011	可　变	$\overline{Y_3}=\overline{CS3}$
1000～17FF	2KEPROM2	00010	可　变	$\overline{Y_2}=\overline{CS2}$
0800～0FFF	2KEPROM1	00001	可　变	$\overline{Y_1}=\overline{CS1}$
0000～07FF	2KEPROM0	00000	可　变	$\overline{Y_0}=\overline{CS0}=\overline{MONSEL}$

表2.1　CMC—80的存储分配

地址空间	用　途	字节数
3FC0～3FFF	CMCBUG使用的工作单元和断点表	64
3F80～3FBF	CMCBUG栈区(包括用户栈和系统栈)	64
3DC0～3F7F	追踪时使用的保护区（不使用追踪时此区可留给用户使用）	448
3000～3DBF	用户工作区	3520
2000～2FFF	用户选用工作区	4K

表2.2　RAM使用分配

2.2.1.2　存储地址译码

存储器的译码是采用74LS138(U43)译码器和74LS32(U34，U45)或门来完成的，具体电路见技术手册。译码器的每根输出接至2K存储器的$\overline{CE}$和$\overline{CS}$端，共配6K存储器。

2.2.1.3　用户EPROM入口

复位或从键盘命令状态返回监控状态后，程序检查开关S_2的位置。如果S_2置于MONRST位置，则执行CMCBUG监控程序，准备接受用户的键盘输入；如果S_2置于PROM1RST位置，则转移到起始地址为0800H的PROM1中的程序中去。采用这种方式时用户应把自己程

序的入口定在0800H处。

2.2.1.4 EPROM写入的电路

对EPROM的写入，是采用软硬件结合的方式来完成的

EPROM2716或2758的工作方式如表2.3所示

管脚 / 方式	PD/PGM	$\overline{CS}$	Vpp	数据线状态
读	0	0	+5V	输出
禁止	0	1	+5V	高阻抗
待机	1	X	+5V	高阻抗
写入	52ms 脉冲	1	+25V	输入
校核	0	0	+25V	输出
禁止写入	0	1	+25V	高阻抗

表2.3 EPROM工作方式

为了对EPROM进行写入，要将开关S_3置于PGM位置，即把EPROM$_3$的V_{PP}脚接到+25V电源。此时在$\overline{CS}$端送高电平，PD/PGM脚上输入一个脉宽为50～55毫秒的TTL电平正脉冲，即可将地址总线所指定的EPROM$_3$的单元写上数据总线的内容。

实现写入的辅助电路如图2.2所示

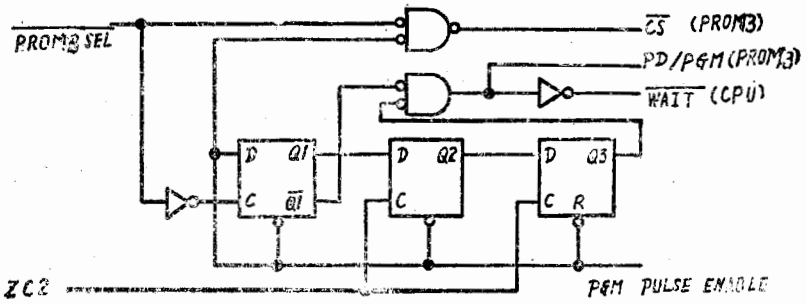
执行写入命令时，程序先将CTC通道2置为定时器工作方式，每26ms送出一个脉冲，再通过锁存器74LS273(U9)使PGM PULSE ENABLE置1，撤除对触发器Q_1～Q_3的清零封锁，将$\overline{CS}$(PROM3)置为1；然后通过写指令选址使$\overline{PROM_3SEL}$为低电平，从而将触发器Q_1置1，PD/PGM送出高电平，$\overline{WAIT}$变为低电平，从而使CPU进入等待状态，数据总线上的内容保持不变，写入EPROM的指定单元。直到CTC的ZC_2发出两个脉冲(52mS)使Q_1的逻辑1信号传送到Q_3后，PD/PGM变低，$\overline{WAIT}$变高，CPU结束等待状态，这一单元也就写入完毕，接着CPU撤消PGM PULSE ENABLE的1状态，从而使Q_1～Q_3清零，准备对下一单元写入数据。

当用户使用EPROM2758时，管脚19为AR，用户可根据需要通过跨接线而使它接到+5V或地。

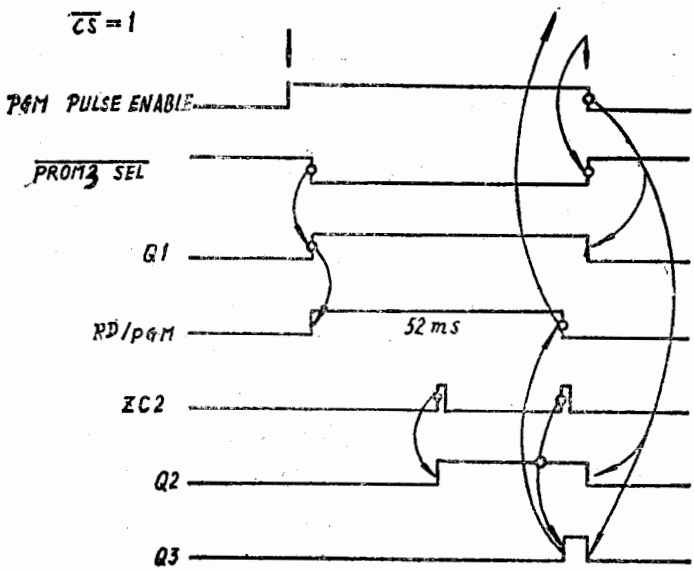
2.2.2 I/O接口及其空间分配

Z80—CPU，可直接访向的I/O空间为256字节。CMC—80板上使用的外围芯片PIO、SIO、CTC每片占四个地址，ADC0809占八个地址，用于显示的锁存器和读键盘输入的缓冲器虽然只有三个单元，但为了简化地址择码电路，实际共占用了八个地住，所以本板上已占用I/O空间的32个单元。

2.2.2.1 择码电路和I/O分配



(a) 电路



(b) 时间图

图2.2 EPROM写入的电路和时间图

器　件	译码器输出	A_7-A_2	A_1A_0	口	口地址
PIO1 U52	$\overline{Y_0}=\overline{\text{PIO1 SEL}}$	100000	0 0 0 1 1 0 1 1	A数据寄存器 B数据寄存器 A控制寄存器 B控制寄存器	80H 81H 82H 83H
PIO2 U42	$\overline{Y_1}=\overline{\text{PIO2 SEL}}$	100001	0 0 0 1 1 0 1 1	A数据寄存器 B数据寄存器 A控制寄存器 B控制寄存器	84H 85H 86H 87H
SIO U51	$\overline{Y_2}=\overline{\text{SIO SEL}}$	100010	0 0 0 1 1 0 1 1	A数据寄存器 B数据寄存器 A命令/状态寄存器 B命令/状态寄存器	88H 89H 8AH 8BH
CTC U33	$\overline{Y_3}=\overline{\text{CTC SEL}}$	100011	0 0 0 1 1 0 1 1	通道0 通道1 通道2 通道3	8CH 8DH 8EH 8FH
74LS273 U 8 74LS244 U10	$\overline{Y_4}=\overline{\text{SEGKB SET}}$	100100	× ×	t段选择(只写) 键盘值 (只读)	90—93H
74LS273 U 9	$\overline{Y_5}=\overline{\text{DIG SEL}}$	100101	× ×	数位选择	94—97H
ADC0809 U50	$\overline{Y_6}=\overline{\text{ADC SEL}}$ $\overline{Y_7}=\overline{\text{ADC SEL}}$	100110 100111	0 0 0 1 1 0 1 1 0 0 0 1 1 0 1 1	通道0 通道1 通道2 通道3 通道4 通道5 通道6 通道7	98H 99H 9AH 9BH 9CH 9DH 9EH 9FH

表2.4　I/O地址分配表

I/O 地址译码电路见技术手册附图，由译码器74LS138(U37)和门电路74LS32(U39)，74LS08(U41)组成，选择指定的输入输出接口，相应的分派见表2.4。

2.2.2.2　各I/O 接口片的使用

CMC—80的监控程序不占用PIO和SIO，用户可根据需要自行编制程序向它们送入控制字和中断向量，进行用户所安排的输入输出工作。PIO、SIO、CTC和ADC的信息交换端已引到CMC—80主板边缘的插头上，用户可根据需要采用印刷电路板插座（例如国产40线插座2CY25Z-40）联接到外部设备上，插脚编号如下：

主板左侧插头J4：

插脚	1	2	3	4	5	6	7	8	9
信号	0809 IN3	0809 IN4	0809 IN5	0809 IN6	GND	PIO2 PA7	PIO2 PA6	PIO2 PA5	PIO2 PA4

插脚	10	11	12	13	14	15	16	17	18
信号	PIO2 PA3	PIO2 PA2	PIO2 PA1	PIO2 PA0	PIO2 $\overline{\text{ASTB}}$	PIO2 $\overline{\text{BSTB}}$	PIO2 $\overline{\text{ARDY}}$	CTC ZC0	CTC C/T3

插脚	19	20	21	22	23	24	25	26	27
信号	CTC ZC1	CTC ZC2	0809 IN2	0809 IN1	0809 IN0	0809 IN7	0809 REF(+)	0809 REF(-)	PIO2 PB7

插脚	28	29	30	31	32	33	34	35	36
信号	PIO2 PB6	PIO2 PB5	PIO2 PB4	PIO2 PB3	PIO2 PB2	PIO2 PB1	PIO2 PB0		PIO2 BRDY

插脚	37	38	39	40
信号	CTC C/T0	CTC C/T1	CTC C/T2	

主板后侧插头J5：

插脚	1	2	3	4	5	6	7	8	9
信号	SIO $\overline{\text{DCDA}}$	SIO $\overline{\text{CTSA}}$	SIO $\overline{\text{RTSA}}$	SIO $\overline{\text{DTRA}}$	SIO TxDA	SIO $\overline{\text{TxCA}}$	SIO $\overline{\text{RxCA}}$	SIO RxDA	SIO $\overline{\text{SYNCA}}$

插脚	10	11	12	13	14	15	16	17	18
信号	SIO $\overline{\text{W/RDYA}}$	PIO1 $\overline{\text{BSTB}}$	PIO1 PA0	PIO1 PA1	PIO1 PA2	PIO1 PA3	GND	PIO1 PA4	PIO1 PA5

插脚	19	20	21	22	23	24	25	26	27
信号	PIO1 PA6	PIO1 PA7	SIO $\overline{\text{DCDB}}$	SIO $\overline{\text{CTSB}}$	SIO $\overline{\text{RTSB}}$	SIO $\overline{\text{DTRB}}$	SIO TxDB	SIO $\overline{\text{RxTxCB}}$	SIO RxDB

插脚	28	29	30	31	32	33	34	35	36
信号	SIO $\overline{\text{SYNCB}}$	SIO W/RDYB	PIO1 BRDY	PIO1 ARDY	PIO1 $\overline{\text{ASTB}}$	PIO1 PB0	PIO1 PB1	PIO1 PB2	PIO1 PB3

插　脚	37	38	39	40
信　号	PIO1 PB4	PIO1 PB5	PIO1 PB6	PIO1 PB7

注：以上是插入SIO/0的情形，如插入 SIO/1则24脚为TxCB，25脚为$\overline{\text{TxCB}}$，26脚为$\overline{\text{RxCB}}$；如插入SIO/2则26脚为$\overline{\text{TxCB}}$，27脚为$\overline{\text{RxCB}}$，28脚为RxDB，其它不变。

CMCBUG使用了CTC的各通道：CTC0用于实时钟；CTC1用于信息转储；CTC3用于信息装入；它们都使用中断方式2。在监控初态时已设置好了中断向量。EPROM写入，单步和执行用户程序时使用了CTC2作为计数或计时器，不向CPU申请中断，而由其ZC2输出信号引起非屏蔽中断。在不更改中断向量的情况下，用户可以使用通道0，其服务程序入口地址为3FDA～3FDC（可在此处放一条转移指令）。当然，用户也完全可以在自己的程序中对CTC的各通道重新作出安排，写入所需的控制字和中断向量。但对通道2 应 特别小心，须仔细研究CMC—80的系统结构后才能使用它，否则就会引起非屏蔽中断。

CTC的信息交换线在主板左侧的插头上引出，其编号为J4

CTC信号	C/T3	ZC0	C/T2	ZC1	C/T1	ZC2	C/T0
插　脚　号	18	17	39	19	38	20	37

须注意C/T2(39脚）在系统内已连接到辅助板上的单稳触发器14538（U12），如果用户还需使用监控程序，不能拆下辅助板，则该引脚不能再由外部输入信号。同样地，C/T3（18脚）在主板上已连接到ADC0809的EOC端，用户如果插上了A/D转换芯片，则C/T3用于A/D转换的中断请求，不能再由外部输入其它信号。

ADC0809 是完全留给用户使用的，它与外部的信息交换线在主板左则的插头上引出，其编号为J4

ADC0809信号	IN0	IN1	IN2	IN3	IN4	IN5	IN6	IN7	REF(+)	REF(-)
插 脚 号	23	22	21	1	2	3	4	24	25	26

注意ADC0809的转换结束信号(EOC)已接到CTC的通道3 C/T3端，如果用户需要由EOC信号引起CPU的中断，除了置中断向量（方式2时）外，还需写入控制字如下

```
LD      A, OD5H
OUT     (CT3), A
LD      A, O1H
OUT     (CT3), A
```

2.2.2.3　键盘和显示

CMC—80有六个七段LED显示器，可以显示十六进制数，也可以显示其它的一些特定字符，在CMCBUG监控程序中可以显示“—”和“'”

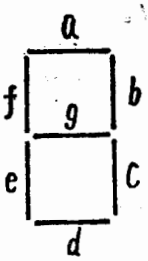
显示采用按位扫描的工作方法，在扫描过程中每位约发光1毫秒，所显 示 的 数据（或

字符）经程序转换为七段格式后，写入U8的锁存器74LS273（口地址为90H），此时要显示的位序则由U9的锁存器74LS273（口地址为94H）指定。

七段LED显示器以及段号与数据位的对应关系如图2.3所示。

U8锁存的段格式经段晶体管驱动送往七段发光显示器阳极，U9锁存的显示字位则经字驱动器75452（U4～U6）送往显示器的阴极，使指定的字段发光。U9还兼用来扫描键盘，例如当口94U 9中数为04H（即L3为低电平，其余扫描线为高电平）时，如果键4、5、6、B、MON中之一被压下，则使某一R线成为低电平。如此时键6被压下，则R_2为低电平，其余线呈高电平，使口90U10的低五位为11011，程序读入这一信息就可判断用户所按下的键。

为了简化译码，U8锁存器和U10键盘值输入口共用一个I/O地址(90H)，但前者只输出，后者只输入，不会互相干扰。



(a) 七段显示器；

数字	D7 D6 D5 D4 D3 D2 D1 D0 g f e d c b a	字形
0	1 0 0 0 0 0 0 0	0
1	1 1 1 1 0 0 1	1
2	0 1 0 0 1 0 0	2
3	0 1 1 0 0 0 0	3
4	0 0 1 1 0 0 1	4
5	0 0 1 0 0 1 0	5
6	0 0 0 0 0 1 0	6
7	1 1 1 1 0 0 0	7
8	0 0 0 0 0 0 0	8
9	0 0 1 1 0 0 0	9
A	0 0 0 1 0 0 0	A
B	0 0 0 0 0 1 1	b
C	1 0 0 0 1 1 0	C
D	0 1 0 0 0 0 1	d
E	0 0 0 0 1 1 0	E
F	0 0 0 1 1 1 0	F
'	1 1 1 1 1 0 1	'
空	1 1 1 1 1 1 1	
-	0 1 1 1 1 1 1	-

(b) 七段显示的代码与字形的关系

图2.3 显示器与代码的对应关系

2.2.2.4 录音机转储电路

存储器里的信息转储到磁带上采取调频记录制式：八个2400HZ的周波表示逻辑1，四个1200HZ的周波表示逻辑0，转储时用程序将CTC通道1设成定时工作方式，ZC1输方波经U7（14013）分频后再经阻容滤波和电阻网络衰减送到录音机MIC端或AUX端录到磁带上。

2.2.2.5 录音机输入电路

从录音机EAR引出的调频信号，经由U11限幅的整形，送往单稳触发器U12，U7与U12组成频率检测器，当输入信号为1200HZ时，鉴频器通过74LS244（U10）送到数据线D7的信号为0，当输入信号为2400HZ时，送往D7的信号为1，程序定时读取U10的值，然后再经处理，就可以把存储在磁带上的程序和数据装入。

§2.3 其它电路部分

2.3.1时钟电路

3.9936MHZ的晶体振荡器与两个反相器74LS04（U48）和阻容一起组成振荡电路，经

由U46分频后得到1.9968MHz的主钟频率送往CPU和其它电路，协调CMC—80机内各部分的动作。

2.3.2 非屏蔽中断

CMC—80的非屏蔽中断在三种情况下使用：运行用户程序时由按下MON键返回监控状态；单步执行程序（STEP）和设置断点后执行用户程序（EXEC）。

CPU的 $\overline{\text{NMI}}$ 端接收CTC通道2的ZC2发来的信号，在单步或设置断点时，该通道被置为定时状态，使得运行一条指令后便引起非屏蔽中断，转到 $\overline{\text{NMI}}$ 中断服务程序进行处理。在执行用户程序时，EXEC命令将八位锁存器74LS273(U9)置定使L3为低电平，并将CTC的通道2置为计数工作方式。如果在程序运行中用户想暂停程序并返回监控，则可按下MON键，使Ro变低，触发单稳14538（U12），其 $\overline{Q}$ 发出的脉冲则使CTC通道2计数，引起非屏蔽中断，退出了用户程序。

由于写入EPROM时也使用了ZC2的信号，此时不应引起非屏蔽中断，所以使用门74LS02(U40)和74LS04(U35)，使得在写入EPROM时ZC2的信号不送往CPU的 $\overline{\text{NMI}}$ 引脚。

2.3.3 复位电路

复位电路由两个非门74LS04(U38)和阻容电路组成，加电时由于电容C的作用，产生复位信号 $\overline{\text{RESET}}$；运行时压下按钮S_1也使全机处于复位状态。

§2.4 系统的扩充能力

主板右侧的电路板插头J6将Z80CPU的主要总线信号引出板外，以备用户在需要时可另配电路板进行系统扩充。但本系统各总线均未加驱动电路，用户外接负载时应考虑其驱动能力的限制。

J6的信号连接如下

插脚	1	2	3	4	5	6	7	8	9
信号	GND	D7	D4	D5	D3	D6	D1	A9	A8

插脚	10	11	12	13	14	15	16	17	18
信号	A10	RD	A13	A15	DO	Ø	$\overline{\text{MREQ}}$	$\overline{\text{BUSAK}}$	SIO IEO

插脚	19	20	21	22	23	24	25	26	27
信号			$\overline{\text{M1}}$	$\overline{\text{IORQ}}$	A7	$\overline{\text{RESET}}$	A6	A5	A4

插脚	28	29	30	31	32	33	34	35	36
信号	A3	A1	A0	A11	A12	A14	D2	A2	$\overline{\text{INT}}$

插　脚	37	38	39	40
信　号	$\overline{BUSRQ}$	$\overline{WR}$		

其中IEO（7脚）为本机板上中断键里优先权最低芯片（SIO）输出的中断开放信号。用户需扩充使用Z80系列的外围芯片（如PIO、SIO、CTC、DMA等）时，可将此信号接往键内次一芯片的IEI信号。

如果用户开发了自己的应用程序而无须使用监控程序，也用不着键盘和显示器时，可将辅助板拆卸下来，但此时需进行如下的改动：

1.将主板中部标有C，P字母的两点用导线连接起来。

2.如果用户不需要使用CTC2申请非屏蔽中断，可将主板左前部标有E，F字母的两点连线断开，而将E点与G点用导线连接起来。

§2.5　监控程序CMCBUG

2.5.1 概述

CMCBUG是CMC—80微型电脑的监控程序，它提供了操作者与硬设备之间的交往手段，帮助用户开发、调试、存储、固化自己编写的机器语言应用程序。

CMCBUG占用2K内存单元，写在一片EPROM2716中，占据存储空间地址为0000～07FFH。它使用64个字节RAM作为工作单元（地址3FC0～3FFFH）。下堆栈指针初始设定在3FCO处，用户栈由此处向下伸展，系统栈紧接着用户栈向下排列，最多占用36个字节RAM。

CMC—80加电或操作者按下复位（RESET)按钮之后，首先进入初始化程序，设置栈指针，清除各标志和显示缓冲单元，对CTC设置中断向量。然后转到显示程序，把显示缓冲单元的内容取出，送往LED显示器，显示采用动态方式工作，每位数字保持约一毫秒。六位数字扫描一遍之后检视键盘看是否有输入，如果没有按键，则返回显示程序，反复循环。如操作者按键，则转入键分析程序，判断输入的信息，如果是数字键，则送入相应的显示缓冲单元，进行需要的处理（例如修改某存储单元的内容）后返回显示程序；如果是命令键，则转到相应的命令处理程序。

CMCBUG有二十四条命令，各自有其处理程序，命令处理完毕之后，多数情况下返回显示程序，也有的返回到监控初始状态，这些程序的作用见第三章。

CMCBUG的框图见图1。

2.5.2　可供用户使用的子程序

CMCBUG中有一些子程序可供用户调用，主要的包括

1. UFOR1（入口0566H）它把累加器A中的高四位作为十六进数字写入（IX）单元，低四位写入（IX+1）单元。该程序使用寄存器A、B、F。

2.UFOR2（入口057FH）

它把显示器前四位数作为两字节数据取到H、L两寄存器中。该程序使用A、H、L、

F、IX寄存器。

3.UFOR5（入口05BAH）

它把H、L中的两字节数据作为四位十六进制数写到显示器前四位缓冲区中去，该程序使用A、B、C、HL、F、IX寄存器。

4.UABIN（入口05EDH）

它把累加器A中的十六进制数的ASCII码转换为二进制数，使用A、F寄存器

5.UBASC（入口0576H）

它把累加器A低四位的十六进制数字转换为相应的ASCII码，占用A、F寄存器

6.D20MS（入口06D7H）

它起延时20ms的作用，占用H、L、F寄存器。

2.5.3 供用户选用的命令、指令、中断处理程序的入口和可资利用的信息。

1.用户专用键USR0和USR1程序入口地址存放在3FC0、3FC1以及3FC2、3FC3中。入口地址低字节在前，高字节在后。但用户也可不必用MEM命令往这些单元里送地址，参见§3.21所介绍的使用方法。

2.重新启动指令中RST0引到复位初始化程序，RST8用于设置断点，其余的重新启动指令均可由用户选用，CMCBUG为每条指令留下了三个单元作为处理程序入口，一般可在相应的单元里放一条转移指令，转到用户编写的处理程序的真正入口去，预留的单元如下

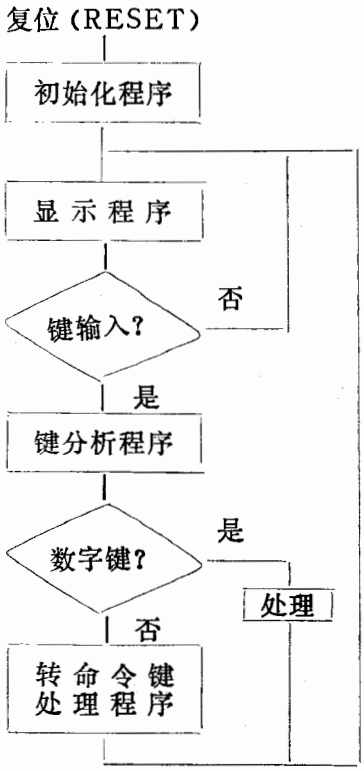


图1. CMCBUG框图

RST16	3FC8～3FCA
RST24	3FCB～3FCD
RST32	3FCE～3FD0
RST40	3FD1～3FD3
RST48	3FD4～3FD6
RST56	3FD7～3FD9

3.使用监控程序时，用户可以使用CTC0作为计数器或定时器，无须另送中断向量（每次返回监控初态时已送了中断向量）。此时CTC0的中断服务程序入口地址为3FDA～3FDC，可放一条转移指令以转向服务程序的真正起点。

4.打开实时钟后，时存放在3FC7单元，分存放在3FC6单元，秒存放在3FC5单元。用户程序也可以读这些单元以取得实时，如向它们写入数据，则修改了实时。

第三章　CMC—80键盘操作说明

CMC80微计算机提供给用户的基本功能包容在2K字节的CMC BUG监控程序中，该监控程序使用辅助板上的28键键盘和六个数码显示器(LED) 作为与使用者交换信息的基本媒介。初学者可借助于它们学习微计算机的基本知识，了解和熟悉Z—80系列芯片的性能和使用方法，初始设备制造、技术改造或科研应用等方面的用户可借助监控程序开发自己的实用程序。如果最终应用不需监控程序，则用户也可取下装有CMCBUG的EPROM而代之以自己编制的固化程序，辅助板便可根据用户的需要撤除或由用户程序指定将键盘及显示器另作它用。

本章主要介绍CMC—80双板机的使用。

§3.1　开关、键盘和显示器

3.1.1 开关

CMC—80有三个开关，按钮S_1是复位(RESET)开关，本机加电时自动复位，需要时操作者也可压下S_1实现复位，复位后机器进入监控程序并置定初始状态，最左一位显示器上出现提示符“-”，准备用户输入命令。

注意，如果在RAM里的用户程序已经运行，则按下复位键可能会产生预料以外的结果而使该程序无法继续运行下去，因为复位产生的初始状态包括：

1.禁止CPU接受中断申请；

2.中断向量寄存器I内容置为07，中断方式置为IMO；

3.将栈指示器SP强行置入初值，（用户的SP值丢失）如果原来设置了断点，则断点被清除，用户寄存器内容全部丢失。

4.CTC、PIO和SIO也被复位为初始状态。（用户建立的输入输出关系被撤消，详见各芯片的用户手册）

开关S_2用于选择复位入口地址。当S_2处于MON位置时，复位后进入监控程序；如果S_2处于PROM1、RST位置，复位时则在初始化后自动转移到入口在0800H的用户程序中去。

开关S_3用于写入EPROM命令，详见§3.17。

3.1.2 键盘

CMC—80键盘由28个按键组成，包括16个数字键和12个命令键。数字键用于输入十六进制数字或CPU寄存器名称；命令键作为上下二挡使用，共有二十四条命令可以使用，详见以下各节。

3.1.3 显示器

辅助板上有六个发光二极管数码显示器，可以显示十六进制数字和提示符，在大多数情况下，左边四位显示地址而右边二位显示数据，具体显示的意义详见以下各节。

辅助板上的发光二极管指示灯LD1用于指示从磁带装入程序时的信号电平，详见§3.14。

§3.2 数字键

十六个数字键用来输入数据，包括存储地址、I/O口地址、CPU寄存器名称、机器码指令，数据等。按下数字键后该数值存入显示缓冲存储区，然后可依次显示在显示器上或根据命令写到存储单元，I/O口或CPU内寄存器里。（详见以下各节）数字键使用的实例见表3.1。

按 键	显 示	说 明
RESET	—	监控初始状态
2	2	逐次输入各个数字键
0	2 0	
3	2 0 3	
4	2 0 3 4	
B	2 0 3 4 b	
6	2 0 3 4 b 6	
7	2 0 3 4 b 6	第七个数字输入不产生反应
8	—	第八个数字输入引起监控处理，由于没有命令输入，返回监控初始状态

表3.1 数字键使用举例

数字键与命令键的联合使用见以下各节。

§3.3 MON/MON′（监控Monitor）和上下档控制键

在键盘操作过程中，MON/MON′命令使计算机退出当前的命令状态，返回到基本的监控状态，准备执行其它监控命令或转入用户程序。例如在检查或修改了存储器内容之后如果想检查和修改CPU内部寄存器的内容，不能直接送REG′/REG命令，而需先按一次或两次MON/MON′键，使监控退出MEM命令状态，然后再送寄存器检查命令才能执行之，见表3.2的例子。

MON/MON′键的另一个功能是中止用户程序的执行。当操作者使用EXEC命令或USR0、USR1键进入用户程序后，停止了对键盘的扫描，这时允许操作者通过按下MON/MON′键强行中止用户程序的运行，保存现场，供用户检查自己程序的运行情况，也可以从被中止的地方继续往下运行。这一功能对排除“死锁”故障特别有用。

MON/MON′键的第三个功能是决定其它各命令键的上下挡状态。CMC—80的键盘上有12个命令键，每个键分上下二挡使用，即用同一个键可发出两个命令。按下MON/MON′

键后，如果显示器上出现下挡提示符“-”，则再接其它命令键都按下挡位命令执行；如果显示器上出现上挡提示符“′”，则再按其它命令键按上挡位命令执行，见表3.3的例子。按下MON/MON′键究竟出现上挡还是下挡提示符，取决于按此键之前键盘处在下挡还是上挡命令状态。每按一次MON/MON′键，挡位变换一次，所以如果提示符不符合待送命令的挡位，只须再按一次MON/MON′键，则提示符就发生了变化。

按　键	显　　示	说　　明
MON/MON′	—	准备接受新命令
3 0 0 0	3 0 0 0	送入四位数作为存储单元地址
DISP/MEM	3 0 0 0 × ×	读出3000单元内容为×××
MON/MON′	′	退出存储器检查命令，
MON/MON′	—	准备接受新命令
1	1	送入PC寄存器代号，
REG′/REG	1 × × × ×	检查PC，读出其内容为××××

表3.2　MON/MON′键使用举例

按　键	显　　示	说　　明
MON/MON′	-	进入监控程序
2	2	准备检查AF寄存器
REG′/REG	2 × × × ×	显示AF内容××××
MON/MON′	′	进入监控准备接受上挡令
2	2	准备检查A′F′寄存器
REG/REG′	2′ × × × ×	显示A′F′内容××

表3.3　MON/MON′ 决定上下挡状况举例

§3.4　MEM（存储器检查Memory examing）键

MEM键用来检查和修改存储单元的内容，即对RAM进行读写操作和读出ROM的信息。

当监控进入初始状态，显示器最左边出现提示符“-”后，通过键盘输入待检查或写入

的存储单元的地址，（四位十六进制数字）先送高位数字，后送低位数字，每送入一个数字，就立刻显示在显示器上，最高位显示在最左端，然后按下MEM键，则该存储单元的内容显示在右边的两个显示器上（以十六进数字表示）。如果没有送满四个数字就按下MEM键，机器不产生任何反应，原来输入的数字仍然保持，如果继续输入数字，送满四位后再按下MEM键，仍然可以显示该单元的内容。

在完成上述检查存储单元的操作之后，显示器上有六位数字，左边四位为地址，右边两位为存储单元内容，如果再输入两位数字，则这两位数字就被写入该RAM单元，并被显示出来，单元里原先存放的内容被冲消。注意只有送完两位数字后存储内容才会修改。如果只输入一位数字，则既不修改存储单元内容，显示也不发生变化。对于用户修改ROM的内容或往没有存储器的地址写入数据的企图，由于实际上无法实现，显示的内容也就不进行更改，提示操作者这是非法的操作。

连续按下MEM键并不修改显示单元的地址。如果想依次检查存储单元的内容，应配合使用NEXT或LAST命令，见§3.18和§3.19。MEM使用的实例见表3.4

按　键	显　　示		说　　明
MON/MON′	—		监控初始状态
3　1	3　1		送入两位数字
DISP/MEM	3　1		不起任何作用
3　4	3　1　3　4		输入四位数字
DISP/MEM	3　1　3　4	×　×	显示该单元内容
0	3　1　3　4	×　×	只输一位数字不改变原来内容
F	3　1　3　4	0　F	将该单元内容修改为0F
3　C	3　1　3　4	3　C	又修改为3C
MON/MON′	′		
MON/MON′	—		准备继续用下挡命令
0　0　1　0	0　0　1　0		
DISP/MEM	0　0　1　0	C　3	读出该单元内容
4　5	0　0　1　0	C　3	只读存储器内容不能修改

表3.4　MEM 键使用举例

§3.5　DISP（相对转移偏移量计算Displacement）键

DISP命令用于在手编程序时计算相对转移指令的操作数（即转移的相对偏移量），显

示计算结果，提示计算的有效性（即是否超越相对转移指令所能访问的范围），并将计算的结果自动写到该转移指令的操作数字节去。

执行DISP命令时，需使用者先使用REG命令将原指令地址(即相对转移指令所在的单元地址）存放在IX寄存器里，然后将转移目的地址存放在IY寄存器里，再送入DISP命令，则就完成了计算。例如，某JR的指令地址为2300H， 转移目的地址为22FOH，则操作如表3.5所示

按　键	显　　示	说　　明
MON/MON′	—	下档键有效
6 IX	6	
REG′/REG	6　× ×	× × 读出IX值，无用
2 3 0 0	6　2 3	O O 输入源指令地址
NEXT	7　× ×	× × 读出IY值，无用
2 2 F O	7　2 2	F O 输入目的地址
MON/MON′	′	下档键有效
DISP/MEM	′　0 0	E E 计算结果为EE 00表示未超越范围（如为其它值，则超出了跳转允许范围）

表3.5　DISP 键使用举例

§3.6　REG′/REG（寄存器检查Register　Examine）键

这个键（包括上挡和下挡两个命令）有两方面的作用，一是直接检查和修改CPU诸寄存器的内容，包括寄存器SP；PC;IX；IY;寄存器组AF;BC;DE；HL；A′F′；B′C′；D′E′;H′L′;以及寄存器I和IFF(SP的内容只能检查不能修改)。另一个作用是可以把SP；PC；IX；IY；BC；DE；HL；B′C′；D′E′；H′L′的内容当作地址，间接地检查和修改它们作为指针所指示的存储单元的内容。

使用本命令直接检查和修改寄存器内容时，先根据被检查的是主寄存器还是辅助寄存器而按下一次或两次MON/MON′键，然后送入代表寄存器的数字键(0—8)，再按下REG′/REG键，则显示器右四位即显示寄存器的内容，其中使用数字键8时，显示的高二位为I寄存器的内容，低二位若为00，表示禁止中断，若为04，则表示允许中断。

要修改所显示的寄存器内容，则只需输入四位新值即可。其实例见表3.6

使用REG′/REG进行存储器间接访问时，按键顺序与前述方式相同，但存放地址的寄存器由数字键9—F来指定，执行命令后，显示器左边四位显示指定的寄存器的内容，右边二位显示以该内容作为地址的存储单元的内容，此时进入MEM命令状态，可以使用NEXT或LAST命令显示相邻单元的内容，也可以修改存储单元的内容，但不能修改作为指针的寄

按　键	显　示		说　明
MON/MON′	—		准备接受命令
2 AF	2		要检查AF的内容
REG′/REG	2 × ×	× ×	显示AF内容
2 F E 7	2 2 F	E 7	更换AF内容
MON/MON′	′		准备接受命令
5 HL	5′		要检查H′L′的内容
REG′/REG	5′ × ×	× ×	显示H′L′的内容
MON/MON′	—		
0 SP	0		要检查SP内容
REG′/REG	0 3 F	C 0	SP内容为3FC0
3 F C 2	—		SP内容不能修改

表3.6　REG′/REG直接检查寄存器实例

存器的内容。实例见表3.7。

按　键	显　示		说　明
MON/MON′	—		准备输入命令
9 (SP)	9		要检查栈内各单元
REG′/REG	3 F C 0	× ×	栈顶在3FC0,显示其内容为××
C 3	3 F C 0	C 3	修改栈顶内容
LOAD/NEXT	3 F C 1	× ×	检查次单元内容
0 0	3 F C 1	0 0	

表3.7　用REG′/REG间接检查栈内单元实例

§3.7　PORT（口检查Port Examine）键

PORT命令与MEM命令相似，用来检查和修改输入输出接口的内容。

Z—80CPU的I/O空间为256个单元，故I/O接口地址仅需两位十六进数字。输入地址后再送入PORT命令，则在右边两位显示器上出现该口的内容，如该口是可写入的，则再输入两位数字便修改了它的内容。

本机共用32个I/O口地址，如表3.8所示。输入输出接口片上有些地址只能写入，有些地址只能读出，使用时应按其说明书进行操作。

PORT命令使用的实例见表3.9

§3.8 BRPT（设置断点Break Point）键

这个命令用来在用户程序中设置断点（最多五个）。在调试程序时，利用断点可以在用

口地址	输入输出口	说明
80H	PIO1 A口数据寄存器	口内有数据输入寄存器和数据输出寄存器，读写过程与操作方式有关
81H	PIO1 B口数据寄存器	同上
82H	PIO1 A口控制寄存器	只能写入，不能读出
83H	PIO1 B口控制寄存器	同上
84H	PIO2 诸寄存器，同上述	
～	PIO1 情况	同上述PIO1 情况
87H		
88H	SIO A口数据寄存器	按操作方式可以读写
89H	SIO B口数据寄存器	同上
8AH	SIO A口命令/状态寄存器	按指定方式可对多个寄存器进行读或写操作
8BH	SIO B口命令/状态寄存器	同上
8CH	CTC 通道0	可读写，但读出的是减1计数器内容
8DH	CTC 通道1	同上
8EH	CTC 通道2	同上
8FH	CTC 通道3	同上
90H	七段选择锁存器和键盘选择	七段选择锁存器只能写入
～		
93H	三态缓冲器	键盘选择缓冲器只能读出
94H		只能写入，不能读出
～	数位选择锁存器	
97H		
98H	ADC通道0	可以读写
99H	ADC通道1	同上
9AH	ADC通道2	同上
9BH	ADC通道3	同上
9CH	ADC通道4	同上
9DH	ADC通道5	同上
9EH	ADC通道6	同上
9FH	ADC通道7	同上

3.8 CMC—80使用的输入输出接口

按　键	显　示		说　明
MON/MON′	—		准备接受命令
8　0	8　0		输入口地址
TIME/PORT	8　0	×　×	读出80口内容为××H
MON/MON′	′		
MON/MON′	—		返回监控初态
9　0	9　0		输入口地址
TIME/PORT	9　0	7　F	键盘输入内容7FH

表3.9　PORT命令使用实例

户指定的点上停止运行，以便检查在这一段程序设计中是否发生差错，并可根据需要进行修改。

用户需在程序中某地址设置断点时，先用数字键打入其地址，再按下BRPT键，便设定了一个断点。如果已经输入了五个断点，再设置的断点不被接受，显示器上内容不变化，但不影响原来设定的五个断点。

断点设定之后，用户可用EXEC命令开始执行程序，当程序执行中进行到所设定的断点时，中止它的运行，显示器左四位显示此时PC的内容（即断点地址），右二位显 示累加器A的内容。这时用户可以用其它命令检查和修改程序、寄存器、存储器和I/O 口的内容。如果未修改PC的内容，则按下EXEC键还可继续向下执行程序。

按　键	显　示		说　明
MON/MON′	—		准备接受命令
3　0　1　5	3　0　1　5		断点地址
LOAD?/BRPT	—		接受一个断点
3　0　2　4	3　0　2　4		第二个断点地址
LOAD?/BRPT	—		接受第二个断点
3　0　0　0	3　0　0　0		程序的入口地址
SECH/EXEC	3　0　1　5	×　×	从3000开始运行程序，中止于第一个断点
SECH/EXEC	3　0　2　4	×　×	显示A的内容××H，再从3015继续运行到第二个断点3024，显示此时A的内容

表3.10　设置断点的操作举例

要清除已设置的断点，可采用下列三种方法：

1.按下复位（RESTE）按钮。

2.按下STEP′/STEP键。

3.未输入满四位数便按下BRPT键。

设置断点操作的实例见表3.10，先用MEM命令将3000H～3024H诸单元全部送00，然后按表3.10执行即可。

§3.9 STEP′/STEP（单步执行Single step）键

这两个命令都是用来帮助用户分析和调试程序的，STEP命令执行 用户PC所指向的那一条指令，然后返回监控状态，显示PC的新值（即程序中下一条指令的地 址）和累加器A的内容，此时可以再送入STEP命令来继续向下执行，也可以利用其它监控命令 对程序的执行情况进行检查和修改。STEP′命令与STEP不同之处在于，它遇到子程 序 调用时，则把整个子程序当作一条宏指令执行完毕，然后再返回监控状态，这有利于加速对复杂程序的调试工作。

使用STEP′/STEP命令时，操作者应先用REG′/REG命令将程序的起始地 址送 入PC，然后逐次压下STEP′/STEP键就可以一步步地执行程序。如果修改了PC，也就强使程序转移到了一个新地址。

§3.10 EXEC（执行Execute）键

这个键用来执行存储器内进的程序，它有两种使用方式：

1.先输入待执行的程序起始地址的四位十六进制数字，然后按下EXEC键 。即从这个地址开始执行程序。

2.直接按下EXEC键，则从用户PC所指示的地址开始执行程序。程序在运行中遇到断点而中止，运行中由操作者按下MON键而中止，单步(STEP′/STEP) 执行 了一 定步数后，都可以按下EXEC键而继续往下执行程序。

EXEC键使用的实例见表3.10

§3.11 TRAC′/TRAC（追踪Trace）键

这两个命令用来使用户监视自己程序的运行流程，使用户程序以大约每秒一条指令的速度执行，在显示器上不断显示当前用户PC和A的内容，使操作者可追踪程序的运行。TRAC命令严格一条条地显示每条指令的执行过程，而TRAC′命令只显示主程序的指令流程，把子程序当作一条宏指令一次执行完而不再显示其内部的流程。

要停止追踪，可按下MON/MON′键。

追踪时监控程序自动保存前十六条指令执行的现场状 态，停 止追踪后， 可以用 LAST命令追溯停止追踪前十六条指令的现场状态，用监控命令检查当时CPU寄 存器和堆栈的状态，执行追溯命令（LAST）在形式上好象倒过来的STEP命令，但 实质 上指令无法逆转执行，追溯命令只恢复当时CPU各寄存器和栈的状态，已执行了 写入存储器 或输出数据到接口的指令是无法消除其效果的。

追踪时使用448个内存单元（3DCOH～3F7FH）作为保护区，栈区被限制在64个单元，如果用户栈需要30个单元以上，则可能发生冲突，此时可在用户程序开始时先将用户SP设定到内存其它区域（例如设定SP为3DCOH）。如果操作者不使用追踪追溯命令，则上述限制不存在，448个单元也完全可用用户自行安排使用。

TRAC命令还有另一个用法，在MEM（存储器检查和修改）命令状态下不按MON/MON′键就送入TRAC命令，则以大约每秒一个单元的速度依次显示相邻各内存单元的内容，这相当于一般监控里的LIST命令，可用于检查操作者送入的程序或数据的内容。退出LIST状态也需使用MON/MON′命令。

§3.12 SECH（检索Search）键

这个键用来在一段程序或数据中查找指定的单字节数据或双字节数据。

查找数据时，先用REG命令将待查找区域的起始地址送入IX，将该区域的结束地址送入IY。查找是依地址的递增次序进行的，因此一般IY内的地址值应大于IX内的地址值。如果IX内的地址值较大，则SECH命令将从(IX)开始一直查到FFFF单元，然后再从0000单元查到(IY)，这一情况在后述的DUMP、PROG、MOVE等指令的执行中也是相似的。

所需查找的数据用数字键送入显示缓冲区，如果查找单字节数据，则送入两位数，如果

按　键	显　示		说　明
MON/MON′	—		准备送入命令
6 IX	6		
REG′/REG	6 × ×	× ×	准备向IX送地址
0 0 0 0	6 0 0	0 0	检索区起始地址为0000
NEXT	7 × ×	× ×	准备向IY送地址
0 7 F F	7 0 7	F F	检索区末地址为07FF
MON/MON′	′		准备送入上档命令
C 3	C 3		检索数据C3
SECH/EXEC	0 0 0 C	C 3	检索区内第一个C3在000C单元
LOAD/NEXT	0 0 1 0	C 3	检索区内第二个C3在0010单元
LOAD/NEXT	0 0 1 8	C 3	检索区内第三个C3
MON/MON′	′		返回监控初态

表3.11　SECH命令使用实例（其中数据是假设的）

查找双字节数据，则送入四位数。然后执行SECH命令。在待查找区域内如果检索到了该数据，则显示器左四位显示该数据所在的地址，右二位显示该数据（当检索的是双字节数据时，所显示的是第二字节所在的地址及第二字节数据）。如果在待查找区域内没有指定的数据，则显示该区域的末地址及该地址的内容。

在检索区内已查找到了数据后，如果想再往下继续查找第二个或更后面的该指定数据时，可逐次按下NEXT键，此时监控自动进入下挡命令状态，每按一次，就可以从前面已查找的地址后继续执行SECH命令，直到找到下一个检索数据或达到检索区终点为止。

SECH键的使用实例见表3.11

§3.13 MOVE（存储区传送）键

该键用来将某一区域的程序或数据传送到RAM中的另一区域中去，两个区域可以有重叠的部分，此时重叠部分的原来数据被冲掉。

使用MOVE命令时，待传送字块的起始地址和结束地址用REG命令送入IX和IY寄存器内，目标字块的起始地址用数字键送入显示缓冲区内（若未送满四位数，则目标字块的起始地址取待传送字块起始地址加1，即将原字块向后移动一个单元），然后送入MOVE命令，传送结束后显示器显示一个地址和该地址的数据，当数据块是向后传送时（目标块地址大于待传送块地址），显示的地址是待传送块的起始地址减1；当数据块是向前传送时，显示的地址是待传送块的结束地址加1。

使用MOVE命令的实例见表3.12

按　键	显　示		说　明
MON/MON′	—		
6 IX	6		准备送入起始地址
REG′/REG	6 × ×	× ×	
3 0 0 0	6 3 0	0 0	起始地址3000
NEXT	7 × ×	× ×	准备送入末地址
3 0 1 0	7 3 0	1 0	末地址3010
MON/MON′	′		
MON/MON′	—		
3 8 0 0	3 8 0 0		要送到从3800开始的区域去
USR1/MOVE	—		传送完毕

表3.12 MOVE 命令使用实例

§3.14 DUMP（转储）键

这个命令用来将存储器里的程序或数据转储到音频盒式录音机的磁带中，转储采用美国

Kansas标准。传送信息速率为300波特，以2400H音频信号表示“1”，以1200H音频信号表示“0”。在一个文件的转储过程中，先传送40秒全“1”的导引信号，然后转储文件标号（也可以不传送标号），接着依次传送待转储的信息，最后有5秒全“1”的结束信号。

转储的操作顺序如下：

1.用转录线将本机的插口J_2(AUX)与录音机的“AUXIARY”或“MIC”相连。

2.将磁带装入录音机。

3.利用REG命令将内存中待转储信息区的起始地址送入IX，结束地址送入IY。

4.用数字键将这一段信息（文件）的标号送入显示缓冲区，标号限定为两位十六进制数字，如果多于两位，则只有前两位有效，如果不送数字或只送一位数字，则认为没有标号。

5.送DUMP命令并将录音机置成录音方式，此时显示器不显示信息。录音机内一般有自动音量控制，无须进行调节。

6.转储完成后，显示器出现提示符“-”，返回监控初态。整个转储过程至少要45秒。

DUMP命令的使用实例见表3.13。

按　键	显　　示		说　　明
MON/MON′	—		
6 IX	6		
REG′/REG	6 × ×	× ×	
3 0 0 0	6 3 0	0 0	转储起始地址为3000
NEXT	7 × ×	× ×	
3 7 F F	7 3 7	F F	转储结束地址为37FF，共转储2K字节信息
MON/MON′	′		
3 A	3 A		文件标号3A，此时须连好录音机，放好磁带盒，按下录音键
DUMP/LAST			执行上挡命令 DUMP开始转储
⋮	⋮	⋮	⋮ 经过若干秒钟 ⋮
	—		转储完成

表3.13　DUMP命令使用实例

§3.15　LOAD（装入）键

使用DUMP命令将RAM中的信息转储到磁带上之后，在需要时可使用LOAD命令将它

重新装回到RAM的原来区域，输入的操作过程如下：

1.用转录线将录音机的“MONITOR OUT”或“EAR PHONE”与本机插口J_1(EAR)相连。

2.将磁带走到适当的位置。

3.将录音机的高音控制调到最大，低音控制调到最小，音量控制调到最小。

4.用数字键送入待装入文件的标号，标号为两位十六进数字。此时将单向寻找具有该标号的第一个文件并装入RAM。也可以不送标号，此时将装入所遇到的第一个文件。

5.送LOAD命令(注意它是上档命令，在送标号之前应按MON′键)，并压下录音机的放音键，此时显示内容消失。

6.逐渐增大录音机音量，直至CMC—80板上LED（发光二极管）出现亮光。然后再增大音量20%。在整个装入过程中，LED将保持发亮。

7.如果输入过程中未发现错误，则完成后显示提示符“-”，返回监控初始状态，可关闭录音机。

8.如果输入过程中显示器上显示数字，则左四位所表示的地址以前的十六个单元中可能有传输错误，此时应重新输入，并核对音调与音量的设置是否恰当。如果磁带走过了相应区域仍未出现提示符，则可能是送入的标号不符。

9.同一条磁带上记录若干个文件时，最好各指定特定的标号，以便装入时进行检索和查找。当然也可以利用录音机上的走带计数器或文件间隙中LED不亮的特点来识别不同的文件。

§3.16 LOAD?（转储检查）键

LOAD？命令用于检查转储到磁带的文件是否正确，它的操作过程与LOAD命令完全相同，其差别在于它不把磁带上的数据写入RAM，而仅是与RAM中的相应内容进行比较，如果没有错误，则磁带走完后出现提示符“-”；如果发现有错误，则显示一个地址，表示该地址的数据出错或该地址前一段记录的校验出错。此时应该查音量与音调的开关设置，如果排除其它因素后仍然出错，则应重新进行转储或更换磁带再转储。如果磁带走过了相应区段仍未出现提示符，则可能是标号送错或录放系统有故障，应检查、修复后重新转储。

这个命令有助于确保用户的信息不致因操作不当或录放设备故障而遭到损失。

§3.17 PROG（EPROM写入Programmer）键

本机可以将内存中的一段程序或数据写入插在PROM3插座中的２７１６(或２７５８)型EPROM。写入时还需要将25伏稳压电源（本机所配稳压电源盒内已装有）连到机器上。

写入EPROM的操作过程如下：

1.将需要写入的EPROM用紫外灯穿过器件窗口进行照射，以擦除其中原有的内容，即使各单元的内容均恢复为FF。未使用过的新器件可以免除这一步骤。

所用紫外线的波长为2537埃，照射强度为1200uw/cm²，照射能量为15w—sec/cm²。通常可使用医用紫外灯（15—30W），照射距离为2.5—5.0cm，照射时间为20—40分钟，即可擦除EPROM的内容。

2.在关闭电源的情况下，将EPROM插入PROM3插座。

3.合上5V和25V的电源。

4.准备好要往EPROM写入的程序或数据。

5.用REG命令把准备往EPROM里写入的信息区域的首地址送到IX，末地址送到IY。把准备写到EPROM中的首地址送到显示缓冲区（如果采用2716，则它在PROM3插座中的地址为1800H—1FFFH；如果采用2758，则地址范围为1800H—1BFFH，所写入的整个信息区都不应超过这个范围），如果不送写入EPROM的首地址，则认为首地址为1800H。

6.将开关S_3置于PGM位置，送入PROG命令，则显示器熄灭，开始写入EPROM，每个字节的写入约需52ms。

7.写入完成后自动进行核对，如果写入完全成功，则显示器上出现提示符“-”，表示核对无误，进入监控初态。如果检查出有错误，则显示出错的地址（左四位数字）及该地址的内容（右二位数字）。此时如果送入NEXT命令，则继续往下检查EPROM的内容，如没有错误则出现“-”；如有错误继续显示，直至用NEXT检查完毕为止。写入EPROM出现错误，往往是使用了没有插净的EPROM，或者EPROM已经损坏。

写入过程完成后，应将S_3置于READ位置。

写入的实例见表3.14

按　　键	显　　示		说　　明
MON/MON′	—		
6 IX	6		
REG′/REG	6 × ×	× ×	送入待写入区首地址为3000
3 0 0 0	6 3 0	0 0	
NEXT	7 × ×	× ×	送入待写入区末地址为37FF
3 7 F F	7 3 7	F F	
MON/MON′	′		
MON/MON′	—		此时将开关S3置于PGM位置
USR0/PROG			开始写入。未送写入起始地址，则从1800地址开始
⋮	⋮	⋮	⋮
			约需两分钟
			⋮
	—		写入并检查完毕

表3.14　PROG命令使用实例

§3.18 NEXT（继续）键

NEXT键有多种用法，与其它键配合使用有五种用法：

1.在进行存储器访问(MEM)时，使用NEXT键可以转到对下一个存储单元进行检查和修改，见表3.15所示的实例。注意在REG′/REG命令进行间接存储器访问时就进入了MEM状态，因此也可以使用NEXT命令，见§3.6和表3.7的实例。

按　　键	显　　示		说　　明
MON/MON′	—		
3 3 A 7	3 3 A 7		送检查地址3 3 A 7
DISP/MEM	3 3 A 7	B 6	该地址内容为B 6
LOAD/NEXT	3 3 A 8	3 5	33A8单元内容为3 5
9 F	3 3 A 8	9 F	修改其内容
LOAD/NEXT	3 3 A 9	7 1	继续往下检查
LOAD/NEXT	3 3 A A	4 C	
DUMP/LAST	3 3 A 9	7 1	往回检查
DUMP/LAST	3 3 A 8	9 F	

表3.15　NEXT与LAST命令使用实例（其中数据是假设的）

2.在进行输入输出接口访问（PORT）命令时，使用NEXT键可检查和修改下一个接口地址单元的内容，其操作方法与存储器访问（MEM）很相似，不再赘述。

3,在进行数据检索(SECH)命令时，使用NEXT命令可以继续向下查找第二、第三……个数据，见§3.12及表3.12的实例。

4.在写入EPROM(PROG)命令时，如果写完检查时发现出错，则可使用NEXT命令继续向下检查所有的写入出错地址，见§3.17。

5.在主寄存器直接检查(REG)命令状态时，如果使用NEXT键，则检查按数字键顺序下一个寄存器的内容，在表3.12、3.13、3.14等都列出了检查IX寄存器之后用NEXT键检查IY寄存器的使用方法。这种使用方式不适于辅助寄存器，因为REG′命令为上档状态，无法直接送入NEXT命令。

NEXT命令也可以作为直接的一级命令使用，在显示器上打入四位RAM地址后送入NEXT命令，则响应的情况与送入MEM命令是一样的，可以检查和修改该单元的内容，但如果修改了该单元的内容，则地址立刻自动加一，可供检查和修改次一单元的内容，并可一直进行下去。这相当于一般监控里的AUTO命令，可用于操作者继续向内存送程序或数据，而且可以在AUTO状态下使用NEXT和LAST命令检查邻近单元的内容而不退出AUTO状态。如果送入MEM命令或MON/MON′命令则退出AUTO状态。NEXT命令的这一用法见表3.16

按　　键	显　　示		说　　明
MON/MON′	—		
3 0 0 0	3 0 0 0		从3000H起送程序
LOAD/NEXT	3 0 0 0	× ×	
3 E	3 0 0 1	× ×	3EH送入3000H单元
0 0	3 0 0 2	× ×	00H送入3001H单元
D 3	3 0 0 3	× ×	D3H送入3002H单元
9 6	3 0 0 4	× ×	96H送入3003H单元
DUMP/LAST	3 0 0 3	9 6	刚才数送错，准备修改
9 0	3 0 0 4	× ×	90H送入3003H单元
⋮	⋮	⋮	⋮
MON/MON′	′		退出AUTO状态

表3.16　NEXT命令用来进入AUTO状态

§3.19　LAST（追溯和前位）键

LAST键与其它键配合使用的方法很类似于NEXT键，不过它仅与MEM、PORT和REG命令配合以使存储地址，I/O接口地址或主寄存器标号减1以访问前一个单元，表3.15示例了在MEM命令状态下LAST键的使用方法。

LAST键单独使用时为追溯命令，它必须在追踪（TRAC′/TRAC）之后使用才有意义，如§3.11中所述，追踪时保存的十六条指令状态可用LAST命令调出来，形式上好象执行了逆转的STEP命令，因此可用REG′/REG等命令检查前一段程序执行过程中的响应与状态。

§3.20　TIME（实时钟）键

该命令为用户提供了一个二十四小时实时钟的功能，可以随时检查实际时间，也可供用户在程控中加以利用。

该键有两种用法。开机后第一次打开时钟时，应先用REG命令把当时的时(24小时制)和分以四位数的形式送入IX寄存器，然后用数字键把秒数以二位数的形式送入显示缓冲器，再送入TIME命令，则实时钟被打开，显示器上出现时、分、秒的数字显示，并且不断更新，和电子钟的显示情况相似。

如果用户要进行其它工作，则可按下MON/MON′键，时钟不再显示，但内部仍然继续计时，此时可以进行调试修改程序的工作，也可以运行实时要求不十分严格的用户程序。但应注意，实时钟每秒钟引起中断申请60次，它会打乱有严格时间要求的程序，因此在执行DUMP、LOAD、PROG等命令时应关闭时钟。此外，实时钟利用了CTC通道1，如果用户要使用这个通道，则实时钟就被关闭了。

TIME键的第二种用法用在随时查询实时间的情况，在时钟已被打开的情况下，无须向IX和显示缓冲区送数，在上档命令初始状态时按下TIME/PORT键，则内部计时时在显示器上显示出来，要停止显示仍可用MON/MON′命令。

TIME命令使用实例见表3.17

按　　键	显　　示		说　　明
MON/MON′	—		
6 IX	6		送入初始时间为9点27分30秒
REG′/REG	6 × ×	× ×	
0 9 2 7	6 0 9	2 7	
MON/MON′	′		
3 0	3 0		
TIME/PORT	0 9 2 7	3 0	打开实时钟，进入计时显示状态
	0 9 2 7	3 1	
⋮	0 9 2 7	3 2	
	⋮	⋮	⋮
MON/MON′	—		停止显示，仍在计时，可进行其它工作
⋮	⋮	⋮	
MON/MON′	′		查询时间
TIME/PORT	0 9 3 5	2 8	已过去了八分钟
MON/MON′	—		准备做其它工作

表3.17　TIME命令使用实例

如果用户需运行有严格时间要求的程序，为了避免定时中断引起干扰或出错，此时应将开打的实时钟关闭。用户在向磁带转储信息，从磁带读入信息和写EPROM时应先关闭实时钟。

关闭实时钟有两个方法：

1.按复位按钮。

2.先按MON/MON′键进入上档状态，然后输入两位数字，使其值大于60H，再按TIME/PORT键，则时钟被关闭。

§3.21 USR0和USR1（用户程序）键

这是留给用户调用自己编写的程序的专用键。用户写好的程序如果需要经常调用，虽然可以用打入起始地址再送EXEC（执行）命令的办法来运行它，但未免太烦，使用USR0和USR1则可用单键调用这一程序。

第一次调用用户程序时，需将程序入口地址数字键送入显示缓冲区，然后送入USR0或USR1命令，则开始运行用户程序。以后调用该程序便无须再送入口地址。当然也可以用MEM命令把用户程序的入口地址写到RAM单元3FC0和3FC1(低字节在前，高字节在后)，然后使用USR0命令就可调用。同样，把入口地址写在3FC2和3FC3，使用USR1就可调用，使用实例见表3.18。

按　　键	显　　示	说　　明
MON/MON′	′	
3 8 0 0	3 8 0 0	首次使用USR0应送用户程序入口地址3800
USR0/PROG	根据程序安排显示	开始执行用户程序
RESET	—	复位停止用户程序
MON/MON′	′	
USR0/PROG	根据程序安排显示	第二次调用用户程序无须再送入口地址

表3.18　USR0命令使用实例

第四章　程序举例

本章目的是想通过几个程序的导入、执行和调试，使用户进一步熟悉键盘的操作功能和本机的使用方法，以及程序设计的一些基本知识。

§4.1　熟悉键盘操作和CMCBUG命令的使用

设有以下程序使用CMCBUG命令，通过键盘操作，输入单板机*。

```
                        ORG     3000
3000    3EAA    LD  A,  0AAH;  将立即数AA送入A累加器
3002    06BB    LD  B,  0BBH;  将立即数BB送入B寄存器
3004    78      LD  A,  B     ;  寄存器B内容送入A累加器
3005    0ECC    LD  C,  0CCH;  将立即数CC送入C寄存器
3007    79      LD  A,  C     ;  寄存器C内容送入A累加器
3008    3EAA    LD  A,  0AAH;  立即数AA送入A累加强
300A    76      HALT          ;  暂停
```

使用TPBUG命令进行的操作如表4.1

按　　键	显　　示 ADDRESS　DATA	说　　明
RESET　(S1)	—	按下开关S1-RESET，立刻显示标志“—”
3　0　0　0	3 0 0 0	输入程序的首地址3000
DISP/MEM	3 0 0 0　× ×	使用MEM　EXAM键读出存储单元3000的内容 × ×
3　E	3 0 0 0　3 E	对3000存储单元写入操作码3E
LOAD/NEXT	3 0 0 1　× ×	用NEXT键顺序对存储单元进行读出操作
A　A	3 0 0 1　A A	对存储单元3001写入操作数AA
LOAD/NEXT	3 0 0 2　× ×	读出3002存储单元

*在本例以及下面各例中，所有用户按下的按键，都在它下面划一横线，而CMCBUG的响应在显示器上显示的就表示在方框中。且数字无特殊标记，均为十六进制数字。

续表4.1

按　　键	显　　示 ADDRESS　　DATA	说　　明
0　6	3 0 0 2　　0 6	对存储单元3002写入操作码06
LOAD/NEXT	3 0 0 3　　× ×	读出3003存储单元
B　B	3 0 0 3　　b b	对存储单元3003写入操作数B B
LOAD/NEXT	3 0 0 4　　× ×	读出存储单元3004
7　8	3 0 0 4　　7 8	对存储单元3004写入单字节指令78
LOAD/NEXT	3 0 0 5　　× ×	读出存储单元3005
0　E	3 0 0 5　　0 E	对存储单元3005写入操作码0 E
LOAD/NEXT	3 0 0 6　　× ×	读出存储单元3006
C　C	3 0 0 6　　C C	对存储单元3006写入操作数C C
LOAD/NEXT	3 0 0 7　　× ×	读出存储单元3007
7　9	3 0 0 7　　7 9	对存储单元3007写入单字节指令79
LOAD/NEXT	3 0 0 8　　× ×	读出存储单元3008
3　E	3 0 0 8　　3 E	对存储单元3008写入操作码3 E
LOAD/NEXT	3 0 0 9　　× ×	读出存储单元3009
A　A	3 0 0 9　　A A	对存储单元3009写入操作数A A
LOAD/NEXT	3 0 0 A　　× ×	读出存储单元300A
7　6	3 0 0 A　　7 6	对存储单元300A写入单字节暂停指令76

续表4.1

按　　键	显示 ADDRESS	DATA	说　　明
MON/MON′	′		中止存储单元操作
MON/MON′	—		以便进行其它操作进入下挡状态
2 AF	2		读出AF累加器
REG′/REG	2 × ×	× ×	在程序未执行前，AF的内容是随机数
LOAD/NEXT	3 × ×	× ×	程序未执行，寄存器BC的内容是随机数
MON/MON′ MON/MON′	—		中止上述操作。下面用STEP键，采用单步方式执行ORG 3000程序
1 PC	1		
REG′/REG	1 × ×	× ×	读出程序计数器PC
3 0 0 0	1 3 0	0 0	单步执行程序方式，首先要修改程序计数器PC内容，使它指向程序首地址
STEP′/STEP	3 0 0 2	A A	执行完3000指令，立即数AA送入累加器A，PC指向下一条指令的地址。“DATA”显示器显示A的内容“AA”
STEP′/STEP	3 0 0 4	A A	3002指令执行完毕，立即数BB送入B寄存器，PC指向下一条指令地址，这时累加器A内容不变，“DATA”仍然显示“AA”
MON/MON′ MON/MON′	—		按两次MON键，进入下挡状态
3 BC	3		
REG′/REG	3 b b	× ×	读出BC寄存器的内容，验证指令的执行。
STEP′/STEP	3 0 0 5	b b	B寄存器内容送入A累加器，PC指向下一条指令的地址

续表4.1

按　　键	显　　示 ADDRESS　DATA	说　　明
STEP′/STEP	3 0 0 7　b b	立即数CC送入C寄存器，PC指向下一条指令的地址。A累加器的内容不变
STEP′/STEP	3 0 0 8　C C	C寄存器的内容送入A累加器，PC指向下一条指令的地址
STEP′/STEP	3 0 0 A　A A	
MON/MON′ MON/MON′	—	中止STEP操作方式
2 REG′/REG	2　A A　× ×	先按下2，后按REG键，读出AF累加器内容
LOAD/NEXT	3　b b　C C	读出BC寄存器内容
MON/MON′ MON/MON′	—	
1 REG′/REG	1　3 0　0 A	读出PC的内容，它现在指向300A 存储单元，是存贮暂停指令的地址
MON/MON′ MON/MON′	—	中止读出方式，下面开始设置多个断点操作，如在3002、3005、3008、300A设置断点
3 0 0 2	3 0 0 2	先送入四位数码表示断点单元地址
LOAD?/BRPT	—	按下BREAK POINT键，断点被接受
3 0 0 5	3 0 0 5	先按四位地址数码键
LOAD?/BRPT	—	在3005设置断点
3 0 0 8	3 0 0 8	在3008单元设置断点

续表4.1

按　　键	显示 ADDRESS	DATA	说　　明
LOAD?/BRPT	—		
3 0 0 A	3 0 0 A		在300A单元设置断点
LOAD?/BRPT	—		中止设置断点的操作，下面用EXEC键执行程序
3 0 0 0	3 0 0 0		送入程序首地址3000，按EXEC键，执行程序
SECH/EXEC	3 0 0 2	A A	程序从3000单元开始执行，显示第一个断点地址和执行第一条指令后，A的内容AA
SECH/EXEC	3 0 0 5	b b	显示第二个断点地址和执行此断点前面的指令后，A的内容“bb”
SECH/EXEC	3 0 0 8	C C	显示第三个断点地址以及完成此断点前面的指令后，A的内容“CC”
SECH/EXEC	3 0 0 A	A A	显示第四个断点，以及完成此断点前面的指令后，A的内容“AA”
RESET	—		按动RESET或STEP或BREAK POINT键，均可取消断点
3 0 0 0	3 0 0 0		输入程序首地址
SECH/EXEC			从程序首地址开始执行程序，一直执行到暂停指令。显示器不显示
MON/MCN′ MON/MON′	—		按下MON键，回到CMCBUG控制，显示标志“—”
1 REG′/REG	1 3 0	0 b	检查PC，它指向存贮HALT指令的下一个存储单元的地址
MON/MON′	′		重新回到CMCBUG控制，可相应检查各寄存器内容，与上述执行程序方式相比较

§4.2 相对转移指令中偏移量的计算

通过键盘操作，输入下面程序：

```
                        ORG  3000
3000    3E00            LD A,00
3002    0605            LD B,05H
3004    3C          LOOP:INC A
3005    10?             DJNZ  LOOP-$
3007    76              HALT
```

表4.2

按　　键	显示 ADDRESS　DATA	说　　明
RESET	—	等待键盘命令
3 0 0 0	3 0 0 0　× ×	输入程序，用NEXT命令进入AUTO状态
3 E	3 0 0 1　× ×	逐字输入程序
0 0	3 0 0 2　× ×	
0 6	3 0 0 3　× ×	
0 5	3 0 0 4　× ×	
3 C	3 0 0 5　× ×	
1 0	3 0 0 6　× ×	
NEXT	3 0 0 7　× ×	3006单元不装入信息，这是准备填入计算的结果——相对转移偏移量
7 6	3 0 0 8　× ×	
MON/MON′ MON/MON′	—	中止输入程序的操作，准备进行计算偏移量的操作
7 REG′/REG	7　× ×　× ×	目的地地址→ I Y

续表4.2

按　　键	显　　示 ADDRESS　DATA	说　　明
3 0 0 4	7 3 0　0 4	
DUMP/LAST	6 × ×　× ×	源地址→IX
3 0 0 5	6 3 0　0 5	
MON/MON′	′	
DISP/MEM	′ 0 0　F D	显示偏移量OOFDH，即—0003H的2的补码，因Z80指令系统规定相对偏移量为两字节（即从+127D——128D），所以实际偏移量为FDH，前面的00表示未超过允许范围。
MON/MON′	—	
3 0 0 6 DISP/MEM	3 0 0 6　F d	检查3006单元已自动填入偏移量FD
MON/MON′ MON/MON′	—	下面用SINGLE STEP方式执行主程序
1 REG′/REG	1 × ×　× ×	修改程序计数器
3 0 0 0	1 3 0　0 0	将PC指向程序的首地址
STEP′/STEP	3 0 0 2　0 0	执行完第一条指令，显示A的内容和下一条指令的地址
STEP′/STEP	3 0 0 4　0 0	
STEP′/STEP	3 0 0 5　0 1	第一次执行“INC A”指令，A第一次增量

续表4.2

按　　键	显示 ADDRESS　DATA	说　　明
STEP′/STEP	3 0 0 4　0 1	循环回去，这时若检查B寄存器，B的内容是多少？
STEP′/STEP	3 0 0 5　0 2	累加器A第二次增量
STEP′/STEP	3 0 0 4　0 2	循环回去
STEP′/STEP	3 0 0 5　0 3	A第三次增量
STEP′/STEP	3 0 0 4　0 3	循环回去
STEP′/STEP	3 0 0 5　0 4	A第四次增量
STEP′/STEP	3 0 0 4　0 4	循环回去
STEP′/STEP	3 0 0 5　0 5	A第五次增量，这时若检查B的内容，它是01
STEP′/STEP	3 0 0 7　0 5	执行条件转移指令“DJNZ”，当条件B－1＝0成立时，即依次执行下一条指令，而不作转移
MON/MON′	′	等待键盘命令

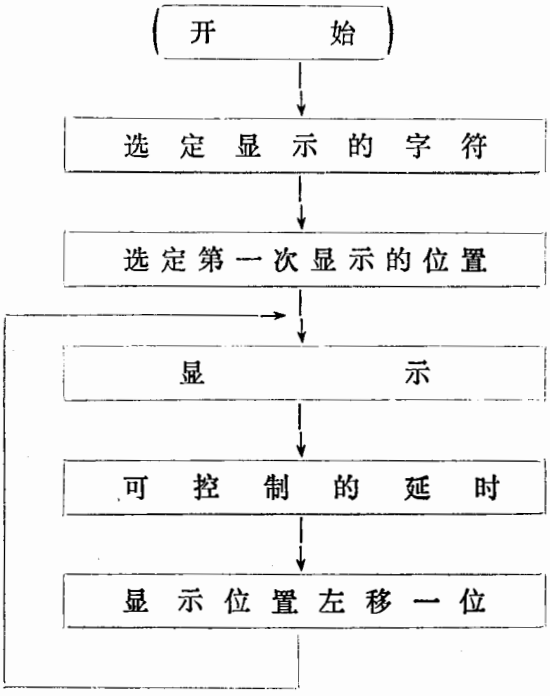
§4.3 软件延时

下面这个程序是使字符“8”，从显示器的右端向左端不停地循环移动。在每个位置上停留的时间，可以由改变装入B寄存器的时间常数来决定，约由20ms到5.2sec。

```
；程序
                        ORG  3000
3000    3E00            LD  A,00H
3002    D390            OUT  (90H), A      ;选定字符“8”
3004    3E01            LD  A,01H          ;选定左边第一个显示器亮
3006    D394    LOOP:   OUT  (94H),A
3008    0601            LD  B,01H          ;B←延时时间常数
300A    CDDB06          CALL DMS           ;调用CMCBUG中延时子程序
300D    07              RLCA               ;左移一位
300E    18F6            JR  LOOP-$         ;循环回去
```

首先输入程序，按下RESET，输入程序首地址3000，按下EXEC，字符“8”则在显示器上从右到左循环移动。按下MON/MON′，程序则停止执行。控制延时子程序D20MS的调用次数（也就是控制装入B寄存器的时间常数）就可以控制字符在显示器上的停留时间。

程 序 流 程 图



§4.4 Z80—CTC的应用

```
                ORG   3000
3000  3E31      LD    A,31H
3002  ED47      LD    I,A      ;设定中断矢量高字节I←31H
3004  310033    LD    SP,3300H;建立栈指示器
3007  3E00      LD    A,00
3009  D38C      OUT   (8CH),A;外都设备提供中断矢量低字节
300B  3EA5      LD    A,0A5H
300D  D38C      OUT   (8CH),A;设定CTC的工作方式，输入通道控制字
300F  3EFF      LD    A,0FFH
3011  D38C      OUT   (8CH),A;输入时间常数，延时约33ms
3013  3E01      LD    A,01H ;设定第一次显示01.
3015  ED5E      IM2         ;设定中断方式2
3017  FB        LOOP: EI
```

```
3018  76        HALT
3019  C31730    JP LOOP
   ;  中断服务程序入口地址表
3100  0032
   ;  中断服务程序
3200  FB                EI          ; 开中断
3201  07                RLC A       ; A累加器左移一位
3202  ED4D              RETI        ;中断返回
```

本例程序是使Z80-CTC的0通道工作在定时器方式，经过一个整定延迟时间后(本例整定延迟时间约33ms)，发出一个中断请求。改变输入到CTC的时间常数，可以改变请求中断的时间。

输入程序，按以下步骤进行调试：

1、采用STEP工作方式，首先修改程序计数器PC为3000H。步进执行程序到显示地址为3018H时，稍停一会。

2、继续按动STEP键，注意观察显示器显示A累加器的内容。每执行完一次3201H指令，A的内容便左移一位，显示器按01，02，04，08，10，20，40、80的次序循环变化。

3、当3201H地址显示时，停止按STEP。按下MON键，检查栈指示器SP的内容，这时应为32FEH。这是因为从栈顶地址3300H压入了两字节。检查32FEH和32FFH单元的内容，应该是18H和30H，这是中断返回的地址3018H。

4、按下MON键，然后再按下STEP键，直至显示3018H。这时，再检查栈指示器SP内容，应该是3300H。这表明已经执行了中断返回RETI指令，中断返回地址3018H已从栈中弹出。

5、用BREAK POINT键，在3200H设置一个断点，用EXEC键连续执行程序。先输入首地址3000H，按下EXEC键，立刻显示断点地址3200H和这时A累加器的内容。连续不断按动EXEC，A累加器内容不断变化，而且每按一次，改变显示一次，按01，02，04，08，10，20，40，80的顺序改变。

可利用RESET键使CTC复位。

下面这个程序，是将本例稍加修改后，再与4.3例结合起来的一个简单表演程序。

输入程序，按下RESET键，输入程序首地址3000H，按下EXEC键，字符“8”快速从显示器右端移动到左端，经过十次循环之后，立即变为较慢的一次移动，接着又是十次较快的循环移动，又接着一次慢速移动……，这样不停地在显示器上循环移动。根据需要，可以很方便地改变显示的字符和循环的速度。按下RESET键，移动立刻停止，回到CMCBGU控制。

```
   ;  程序
                  ORG  3000
3000  3E31        LD   A,31 H
3002  ED47        LD   I,A          ; 设定中断矢量高字节
3004  3E00        LD   A,00
```

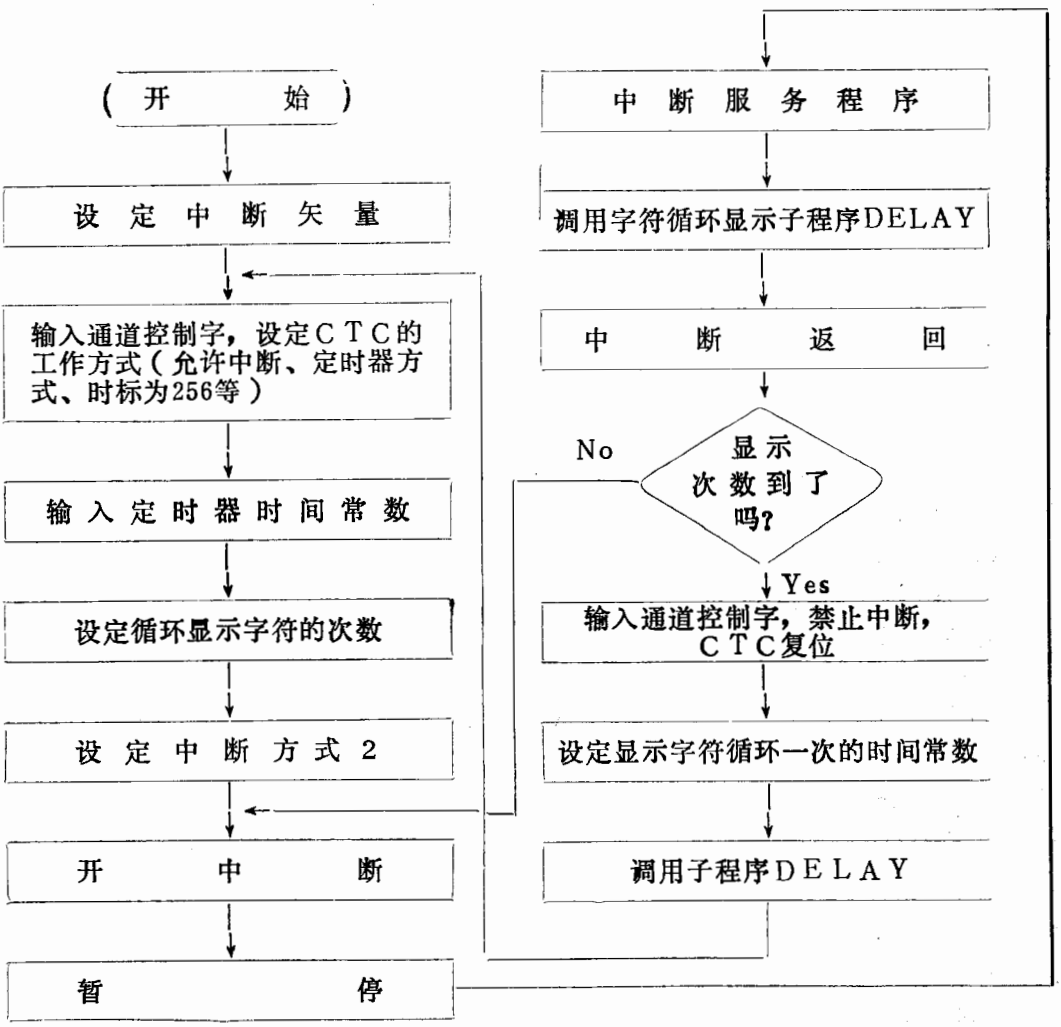
```
3006  D38C        OUT  (8CH),  A     ; 外都设备提供中断矢量低字节
3008  3EA5        ST:LD  A,0A5H
300A  D38C            OUT (8CH), A  ;输入通道控制字,设定CTC工作方式
300C  3EFF            LD  A,0FFH
300E  D38C            OUT (8CH), A  ;输入时间常数
3010  0EF6            LD  C,0F6H    ;设定快速循环次数(10D次)
3012  ED5E            IM  2         ;设定中断方式2
3014  FB        LOOP:EI             ;开中断
3015  76              HALT
3016  0C              INC  C
3017  20FB            JR NZ,LOOP-$;C≠O循环返回
3019  3E03            LD  A,03H
301B  D38C            OUT (8CH),A   ;禁止通道中断
301D  1E0A            LD  E,0AH     ;设定字符循环一次的时间常数
301F  CD5030          CALL  DELAY   ;调用子程序
3022  C30830          JP  ST        ;循环返回，重新工作
   ;中断服务程序入口地址表
3100  0032
   ;中断服务程序
3200  1E01            LD  E，01H    ;设定字符循环速度的时间常数
3202  CD5030          CALL  DELAY
3205  ED4D            RETI
   ; 子程序
3050  3E00     DELAY:LD  A,00H      ;选定字符“8”
3052  D390            OUT  (90H),A
3054  3E01            LD  A,01H
3056  D394     LOOP1:OUT (94H),A
3058  43              LD  B,E
3059  CDDB06          CALL  DMS
305C  CB6F            BIT  5,A      ;测试A的第5位
305E  2003            JR  NZ, LOOP3-$
3060  07              RLC  A
3061  18F3            JR  LOOP1-$
3063  C9        LOOP3:RET
```

注：（1）这个〔DELAY〕子程序，是将4.3例稍加修改而成。这种设计方法，不仅易于调试和检查，而且也节省存储单元。

（2）若想显示别的字符，可参考下表，修改DELAY子程序的第一条指令LD A,00H中的立即数（即修改3051存储单元的内容）。

显示的字符	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
立即数	40	79	24	30	19	12	02	78	00	18	08	03	46	21	06	0E

程 序 流 程 图



§4.5 Z80—PIO的应用

首先用一个10KΩ电阻，将Z80—PIO1的$\overline{\text{ASTB}}$脚连接到电源+5V端，然后输入下列程序，并用BREAK POINT键在3200H单元设置断点。

输入程序首地址3000H，按下EXEC键，CPU执行程序到暂停指令HALT，显示器无显示。

取一导线，使$\overrightarrow{\text{ASTB}}$脚碰一下地(GND)，相当于输入一负脉冲。此脉冲的上升沿使

PIO发出一中断请求($\overline{INT}$)，CPU将执行中断服务程序，把由BREAK POINT键设置的断点3200H显示出来。

```
        ；程序
                              ORG   3000
3000    3E31                  LD    A，31H
3002    ED47                  LD    I，A          ；设定中断矢量高字节
3004    3E00                  LD    A，00
3006    D382                  OUT   (82H)，  A    ；外部设备提供中断矢
                                                    量低字节
3008    3E4F                  LD    A，4FH
300A    D382                  OUT   (82H)，A ；对PIO的A口设置方式1
300C    3E87                  LD    A,87H
300E    D382                  OUT   (82H)，A ；设置允许中断控制字
3010    ED5E                  IM2             ；中断方式2
3012    FB          LOOP:     EI
3013    76                    HALT
3014    C31230                JP    LOOP—$
        ；中断服务程序入口地址表
3100    0032
        ；中断服务程序
3200    FB                    EI
3201    ED4D                  RETI                ；中断返回
```

可以将4.4例的中断服务程序及DELAY子程序作为本例的中断服务程序。这时，输入程序之后，按下RESET键，输入程序首地址3000H，按下EXEC键，显示器无显示，将$\overline{ASTB}$脚碰一下地，字符“8”立刻从显示器右端移到左端，最后停留在左端不动。再碰一次，再重复一次上述现象。若将$\overline{ASTB}$脚碰一下地的操作得不好时，字符“8”会循环移动两次才停留在左端不动。这时，可以用一个10KΩ电阻把$\overline{ASTB}$脚与地连接，然后输入程序，用EXEC键执行程序，然后通过一个3KΩ电阻碰一下+5V端，字符立即从显示器右端移到左端并停留在左端不动。再碰一次，再重复上述现象一次，若条件许可，用TTL电平的单脉冲触发就更好些。

§4.6 求十进制的算术和

下列程序是将两个5位十进制数（或和数不大于6位的6位数）进行算术加法运算，两数长度是相同的，和数贮存在指定存储单元，并将和数在显示器上显示出来。

；数据

3030——贮存字节数（两位十进制数为1字节，5位十进制数为3字节）；

3031、3032、3033——贮存被加数，按顺序先贮存低字节数；

3041、3042、3043——贮存加数，按顺序先贮存低字节数；

3051、3052、3053——贮存和数，按顺序先贮存低字节数。

例子 895867＋91787＝987654

字节数：（3030）＝03

被加数：（3031）＝67　　加　数：（3041）＝87　　和　数：（3051）＝

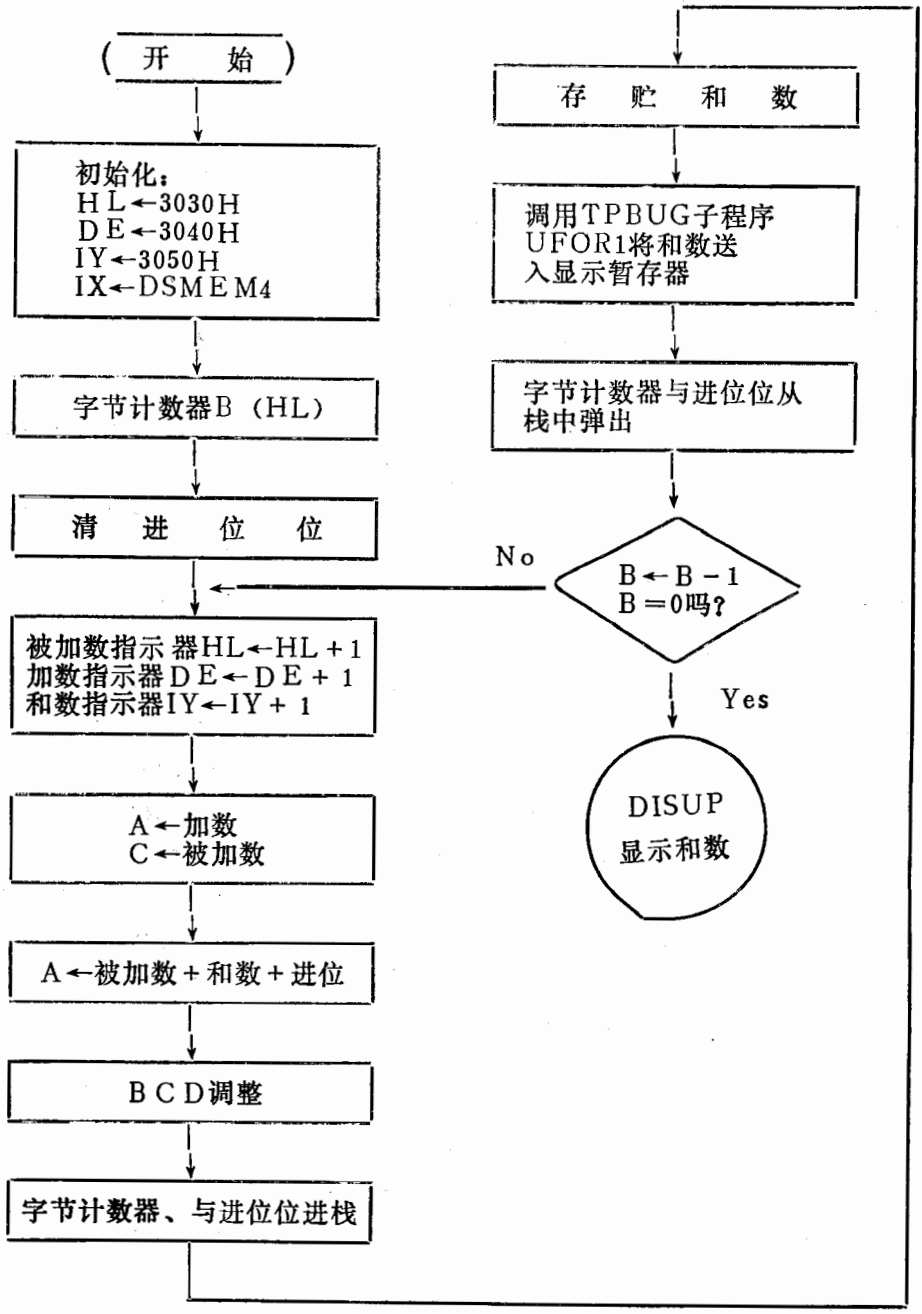
（3032）＝58　　（3042）＝17　　（3052）＝

（3033）＝89　　（3043）＝09　　（3053）＝

；程序

```
3000    213030          LD      HL, 3030H       ；初始化
3003    114030          LD      DE, 3040H
3006    FD215030        LD      IY, 3050H
300A    DD21FB3F        LD      IX, (DSMEM4)
300E    46              LD      B, (HL)         ；计数器←字节数
300F    AF              XOR     A               ；清进位位
3010    23      LOOP:   INC     HL              ；指向被加数存储单元
                                                地址
3011    13              INC     DE              ；指向加数存储单元
                                                地址
3012    FD23            INC     IY              ；指向和数存储单元
                                                地址
3014    1A              LD      A, (DE)         ；取出加数
3015    4E              LD      C, (HL)         ；取出被加数
3016    89              ADC     A, C            ；带进位加
3017    27              DAA                     ；BCD调整
3018    FD7700          LD      (IY+0), A       ；贮存和数
301B    C5              PUSH    BC              ；记忆计数器内字节数
301C    F5              PUSH    AF              ；记忆进位位
301D    CD6605          CALL    UFOR1           ；显示初始化
3020    DD2B            DEC     IX
3022    DD2B            DEC     IX
3024    F1              POP     AF
3025    C1              POP     BC
3026    10E8            DJNZ    LOOP—$
3028    C3BE00          JP      DISUP           ；显示和数
```

程 序 流 程 图


开 始
初始化：
HL←3030H
DE←3040H
IY←3050H
IX←DSMEM4
字节计数器B（HL）
清 进 位 位
被加数指示器HL←HL+1
加数指示器DE←DE+1
和数指示器IY←IY+1
A←加数
C←被加数
A←被加数+和数+进位
BCD调整
字节计数器、与进位位进栈
存 贮 和 数
调用TPBUG子程序
UFOR1将和数送
入显示暂存器
字节计数器与进位位从
栈中弹出
B←B-1
B=0吗?
No
Yes
DISUP
显示和数

附　录

实 验 指 示 书

实验一　熟悉键盘操作

一、实验目的：熟悉按键操作及CMCBUG

二、实验设备：CMC－80微型电脑一台

三、实验准备工作：

①启动稳压电源，将其输出电压调到5V±5%；

②将计算机的电源线“＋”端、“地”端分别接到稳压电源的“＋”“－”端。

四、实验内容：

本实验通过执行下面的程序，一方面熟悉通过键盘命令输入程序的操作，另一方面通过显示器的显示，部分地观察程序的执行，并熟悉STEP，MON，BRPT……等命令键的使用，简单了解调试程序的过程。

```
                        ORG     3000H
3000        3EAA        LD      A, 0AAH
3002        06BB        LD      B, 0BBH
3004        78          LD      A, B
3005        0ECC        LD      C, 0CCH
3007        79          LD      A, C
3008        3EAA        LD      A, 0AAH
300A        76          HALT
```

五、实验步骤

1、通过键盘操作，将ORG 3000H程序送入双板机。（被操作的按键均标以横线CMCBUG响应均以方框表示显示器显示。）

按　键	显　示
RESET (S1)	—
3 0 0 0 DISP/MEM	3 0 0 0 × ×
3 E	3 0 0 0 3 E
LOAD/NEXT A A	3 0 0 1 A A
LOAD/NEXT 0 6	3 0 0 2 0 6
LOAD/NEXT B B	3 0 0 3 b b

按　键	示　显	
LOAD/NEXT 7 8	3 0 0 4	7 8
LOAD/NEXT 0 E	3 0 0 5	0 E
LOAD/NEXT C C	3 0 0 6	C C
LOAD/NEXT 7 9	3 0 0 7	7 9
LOAD/NEXT 3 E	3 0 0 8	3 E
LOAD/NEXT A A	3 0 0 9	A A
LOAD/NEXT 7 6	3 0 0 A	7 6
MON/MON′ MON/MON′	—	

2.在未执行程序前，检查有关寄存器内容：

2 REG′/REG	2 × ×	× ×
LOAD/NEXT	3 × ×	× ×
MON/MON′	′	

（注：在××符号边上记上这时数据，以便和步骤3对比）

3.用STEP键，单步执行程序。首先调整程序计数器PC，使PC指向程序首地址3000H，然后才按动STEP′/STEP键，注意观察显示器的变化。

当显示器显示地址为300AH时，按下MON键，检查并记录有关寄存器内容，与步骤2进行比较。

MON/MON′ MON/MON′	—	
2 REG′/REG	2	
LOAD/NEXT	3	

MON/MON′ MON/MON′	—	
1 REG′/REG	1	
MON/MON′	′	

4.用BRPT键，设置断点。

在3002、3005、3008、300A单元设置断点，并用EXEC键执行程序，注意观察显示器的变化，比较与STEP方式执行程序有什么不同。

MON/MON′	—	
3 0 0 2	3 0 0 2	
LOAD?/BRPT	—	
3 0 0 5	3 0 0 5	
LOAD?/BRPT	—	
3 0 0 8	3 0 0 8	
LOAD?/BRPT	—	
3 0 0 A	3 0 0 A	
LOAD?/BRPT	—	

然后输入首地址3000H，连续按动EXEC键执行程序。

当执行程序到显示器显示300AH地址时，停止按动EXEC键，与步骤3一样，检查有关寄存器内容：

2	2	
3	3	
1	1	

5.清除断点。用EXEC键连续执行程序。

MON/MON′	—	
3 0 0 0 SECH/EXEC		

问题：

（1）为什么显示器不显示？若要显示器自动显示标志“一”，应如何修改程序？

（2）检查PC内容，对比三种方式执行程序，PC内容为什么有异、同？

（3）清除断点有几种方式？

（4）思考：当采用STEP方式执行程序，若要显示DATA的显示器循环不止的显示AA BB，CC，如何修改本程序？

实验二　相对转移中偏移量的计算

一、实验目的：熟悉微型机的性能，了解Z80的指令。

在Z80系统中，有许多指令需要计算相对转移的偏移量，而这些偏移量是以二进制的补码表示的。在本系统中，具有自动计算并填写相对偏移量的功能。

二、实验设备：同实验一

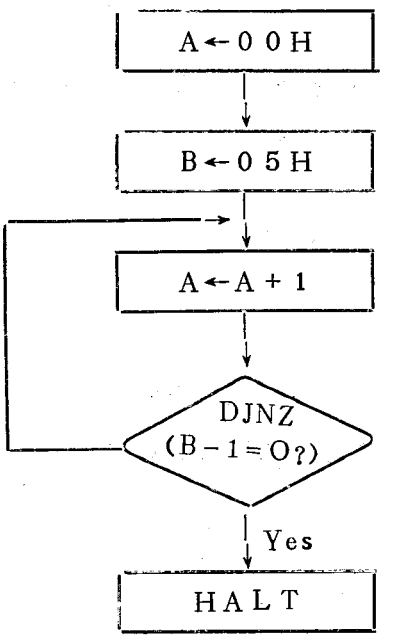
三、实验准备工作：同实验一

四、实验内容：

相对地址的计算

地　址	机器码	操作码
3000	3E00	LD　A，00H；
3002	0605	LD　B，05H；
3004	3C	→LOOP：INC　A
3005	10?	——DJNZ　LOOP—$
3007	76	HALT

框　图



五、实验步骤：

1.操作键盘，输入程序：

RESET	—	
3 0 0 0 LOAD/NEXT	3 0 0 0	× ×
3 E	3 0 0 1	× ×
0 0	3 0 0 2	× ×
0 6	3 0 0 3	× ×
0 5	3 0 0 4	× ×
3 C	3 0 0 5	× ×
1 0	3 0 0 6	× ×
LOAD/NEXT 7 6	3 0 0 8	× ×
MON/MON′ MON/NON′	—	

注意：在3006单元不输入内容，因为要将计算的相对偏移量填入此单元。

7 REG′/REG 3 0 0 4	7 3 0	0 4

（说明：标号LOOP单元目的地址3004H装入IY寄存器。）

DUMP/LAST 3 0 0 5	6 3 0	0 5

（说明：现行$单元源地址3005H装入IX寄存器。）

MON/MON′ DISP/MEM	′ 0 0	F D

（执行相对地址计算命令，00表示偏移未超越范围，Fd为偏移值。）

MON/MON′ 3 0 0 6 DISP/MEM	3 0 0 6	F d

（检查3006，看偏移量是否已装入存储器。）

2.用STEP单步键来检查程序的执行过程：

MON/MON′	′	
MON/MON′ 1 REC′/REG	1 × ×	× ×
	使PC为3000H	
3 0 0 0	1 3 0	0 0
STEP′/STEP	3 0 0 2	0 0
STEP′/STEP	3 0 0 4	0 0
STEP′/STEP	3 0 0 5	0 1
	第一次增A	
STEP′/STEP	3 0 0 4	0 1
	循环返回	
STEP′/STEP	3 0 0 5	0 2
	第二次增A	
STEP′/STEP	3 0 0 4	0 2
STEP′/STEP	3 0 0 5	0 3
	第三次增A	
STEP′/STEP	3 0 0 4	0 3
STEP′/STEP	3 0 0 5	0 4
	第四次增A	
STEP′/STEP	3 0 0 4	0 4
STEP′/STEP	3 0 0 5	0 5
	第五次增A	
MON/MON′		B被减到1
MON/MON′ 3 REG′/REG	3 0 1	× ×
STEP′/STEP	3 0 0 7	0 5

（B减量，若非零则转移，现在B为零，所以PC→3007H）

选作题：

下列程序使字符“8”在显示器内从右方移到左方

```
3000    3E00              LD      A, 00H
3002    D390              OUT     (90H), A
3004    3E01              LD      A, 01H
3006    D394        LOOP: OUT     (94H), A
3008    CDD706            CALL    D20MS
300B    CDD706            CALL    D20MS
300E    CDD706            CALL    D20MS
3011    CDD706            CALL    D20MS
3014    CDD706            CALL    D20MS
3017    07                RLC  A
3018    18EC              JR      LOOP-$
```

注：“CALL D20MS”：调用管理程序中D20MS子程序，“D20MS”表示延时20ms意思。

思考：

①能否将上面选作题稍作修改，可以很方便地控制字符在显示器上停留时间？

②若要显示别的字符，应如何修改程序？

实验三 程序设计(一)

一、实验目的：利用已掌握的Z80指令系统，进行一些简单程序设计，并通过实验，熟悉调试程序的过程。

二、实验准备：同实验一

三、实验准备工作：同实验一

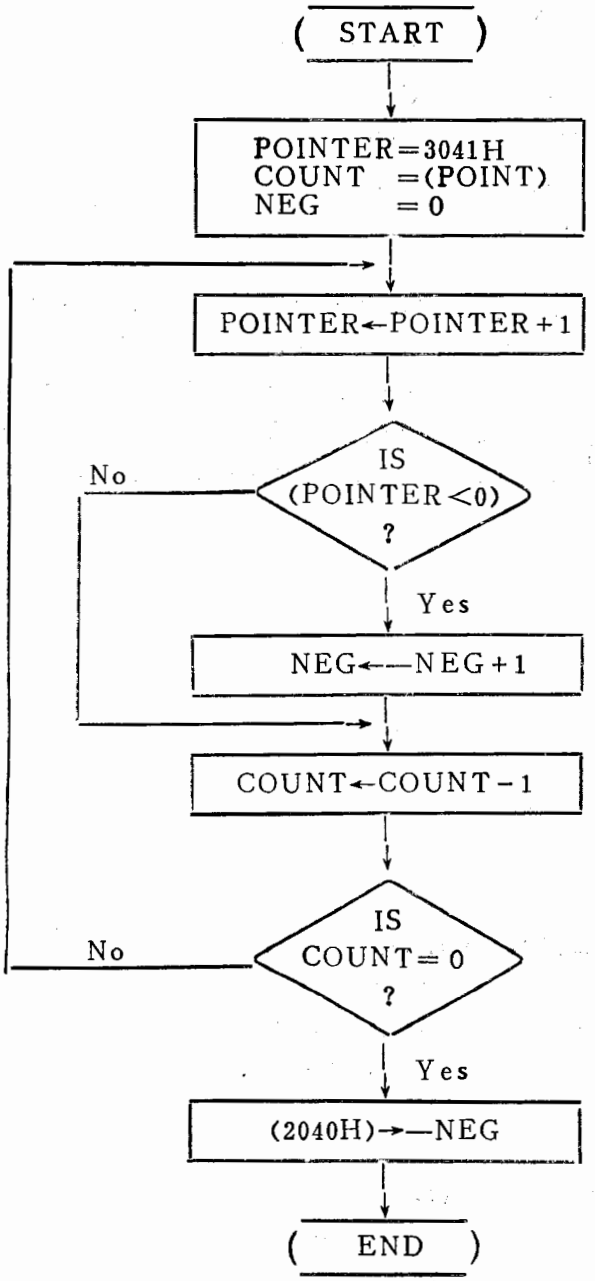
四、实验内容：

1、在一个有正、负数的数据块中，找出负数的个数。假定每个数都是8位二进制数，并将每个数的第7位（b7）作为符号位。数据块的长度存放在3041H单元，数据从3042H单元开始存放，负数的个数存放在3040H单元。

例如：

(3040H)=?
(3041H)=06H
(3042H)=53H
(3043H)=F4H
(3044H)=85H
(3045H)=30H
(3046H)=58H
(3047H)=3AH

流　　程　　图



请大家查找指令表，将机器码填入。

```
                    ORG     3000
3000                LD      HL，3041H
                    LD      B，(HL)
                    LD      C，00
              LOOP1:INC     HL
```

```
        LD      A, (HL)
        AND     A
        JP      P, LOOP2
        INC     C
LOOP2:  DJNZ    LOOP1
        LD      A, C
        LD      (3040H),A
        HALT
```

注：AND A这条指令是用来判别A的内容的第7位B_7是0还是1(即A的内容是正数还是负数)，当$B_7=1$时，标志寄存器F的S=1；反之，S=0，请问：能完成这种功能的指令还有那些？请按你的意见，修改上面程序，并通过实验，验证计算机执行程序结果如何？

2、在一个数据块中找出最大数。假定这些数都是8位二进制数，且不带符号位的正整数。数据块的长度存放在3041H单元，数据块从3042H单元开始存放，结果找出的最大数存放在3040H单元。

例如：

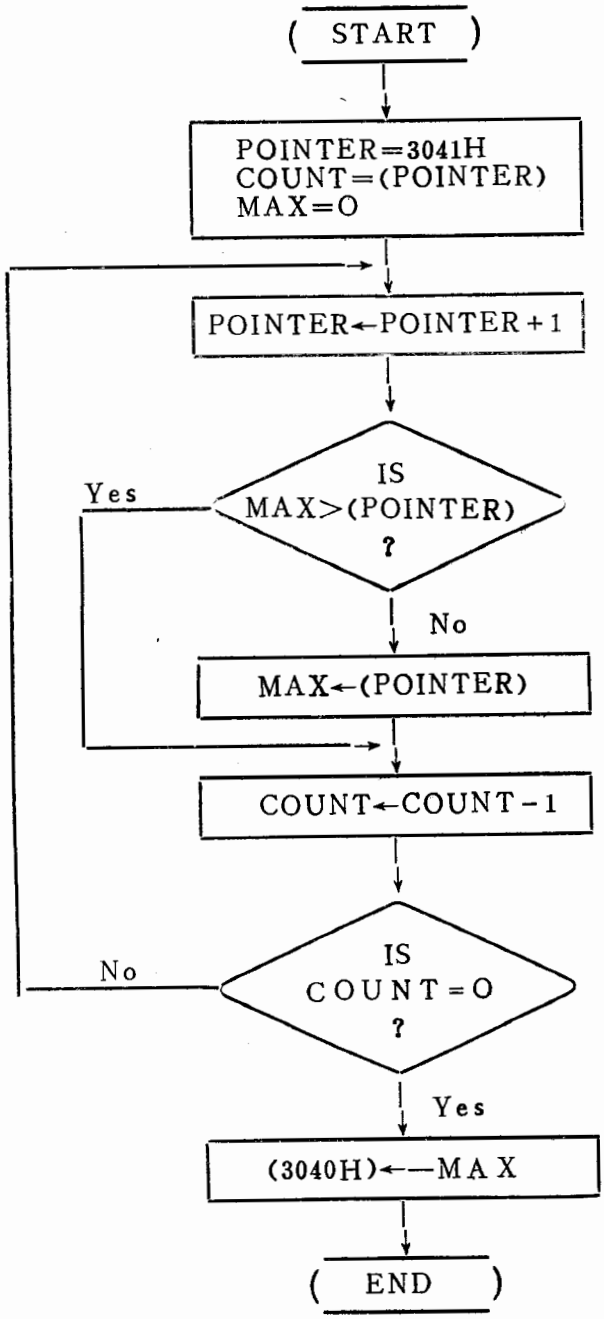
(3040H)=?
(3041H)=06H
(3042H)=57H
(3043H)=89H
(3044H)=A7H
(3045H)=17H
(3046H)=E8H
(3047H)=95H

请大家查找指令表，将对应机器码填入下面程序

```
        ORG     3000H
3000    LD      HL, 3041H
        LD      B, (HL)
        SUB     A
LOOP1:  INC     HL
        CP      (HL)
        JR      NC, LOOP2
        LD      A, (HL)
LOOR2:  DJNZ    LOOP1
        LD      (3040H),A
        HALT
```

按下MON，检查3040H单元内容。

流　程　图


START
POINTER=3041H
COUNT=(POINTER)
MAX=O
POINTER←POINTER+1
IS
MAX>(POINTER)
?
Yes
No
MAX←(POINTER)
COUNT←COUNT-1
IS
COUNT=O
?
No
Yes
(3040H)←—MAX
END

实验四　程序设计（二）

一、实验目的：同实验三

二、实验设备：同实验一

三、实验准备工作：同实验一

四、实验内容：

1、8位二进制数求和　38H＋55H＋26H＋12H＋23H＝？

5个八位数相加，这组数的个数放在（3041H）单元，和存在（3040H）单元。这组数从（3042H）单元开始存放（假定和数也是一个八位数，暂不进行和数大于八位数的计算）

数据：　(3041H)＝05H　　(3044H)＝26H

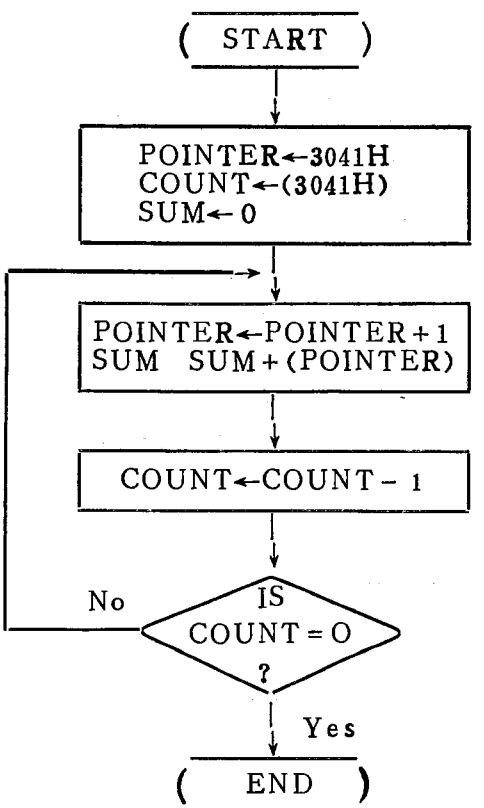
(3042H)＝38H　　(3045H)＝12H

(3043H)＝55H　　(3046H)＝23H

和数：　(3040H)＝E8H

根据下面流程图，请大家进行程序设计，然后在计算机上进行调试。

流　程　图



输入程序，从首地址3000H开始执行，按MON键，检查3040H地址单元内容，即是和数。

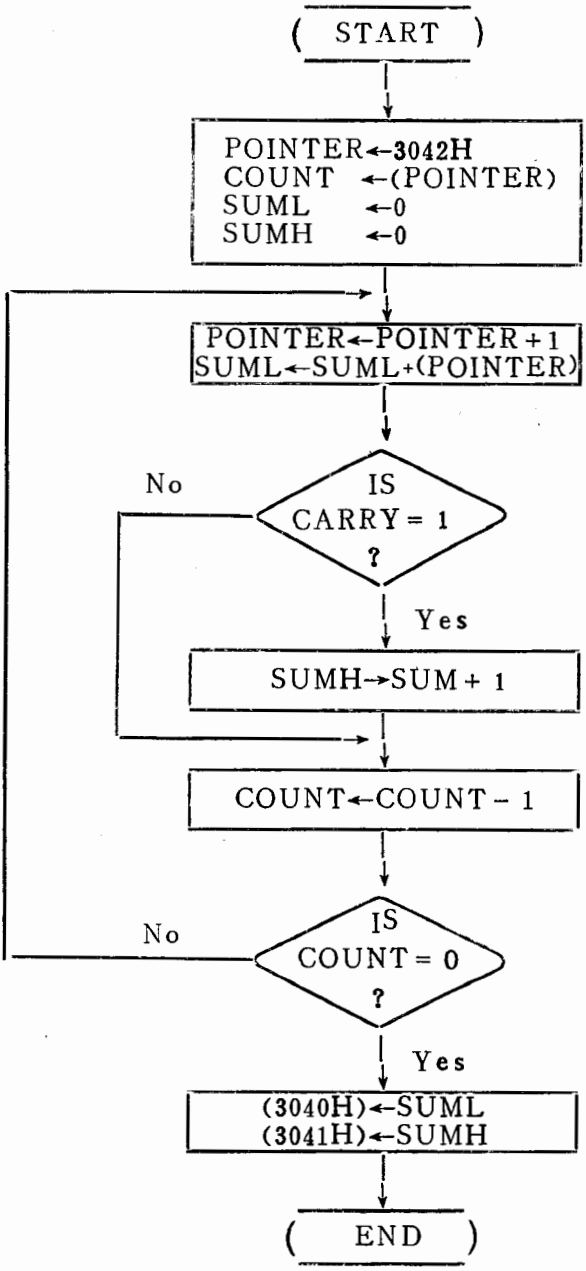
2、8位二进制数求和，和数是16位二进制数

C8H＋FAH＋96H＋F9H＋E8H＝？

和的低位存放(3040H)单元，高位存放(3041H)单元，数组的个数存放(3042H)单元，数据从(3043H)开始存放

```
；和数        (3040H)=?
              (3041H)=?
；数据        (3042H)=05H          (3045H)=96H
              (3043H)=C8H          (3046H)=F9H
              (3044H)=FAH          (3047H)=E8H
```

流　　程　　图



输入程序，从首地址3000H开始执行，按MON键，检查3040H和3041H地址单元内容，即是和数。

选作题：将BCD码转换为二进制码

本程序是将存贮在3040H和3041H单元的BCD码29D转换成二进制代码，存放在3042H单元。

```
;   数据
    (3040H)=02
    (3041H)=09
    (3042H)=1DH=29  Decimal
        ORG     3000
        LD      HL, 3040H
        LD      A, (HL)
        ADD     A, A
        LD      B, A
        ADD     A, A
        ADD     A, A
        ADD     A, B
        INC     HL
        ADD     A, (HL)
        INC     HL
        LD      (HL), A
        HALT
```