

实验指导书

周明德 主编

微型计算机 硬件软件及其应用

清华大学出版社

1-5-00000-1

微型计算机

硬件软件及其应用实验指导书

周明德 主编

清华大学出版社

内 容 简 介

本实验指导书是与教材《微型计算机硬件软件及其应用》配套使用的。共分为三部分，第一部分是单板机编程实验，按照由浅入深的原则安排了九个实验。前几个结合实验介绍了单板机的使用和调试程序的方法，安排了数据块传送与搜索、字符变换、算术运算以及子程序调用等编程。打*号的可作为选作。第二部分是微型计算机系统实验，介绍了如何在CP/M操作系统支持下做实验。亦介绍了常用的编辑命令、汇编命令、连接命令，存盘和执行程序的方法。前几个是基本的编程实验，打*号的编程的要求较高，可根据情况选作。第三部分是硬件和接口实验，前几个是基本逻辑电路以及Z80-CTC和Z80-PIO的实验，打*号的是一些应用性质的实验。

在附录中给出了主要实验的参考程序清单，这是一些典型的实用程序，可作为今后工作的参考。

本实验指导书内容丰富、系统，可作高校本科各专业学习微型计算机的实验指导书，也可作为非计算机专业的研究生学习微型计算机的实验指导书。亦可作为各种微型计算机技术培训班的实验指导书。

微型计算机硬件软件及其应用

实验指导书

周明德 主编

清华大学出版社 出版

北京 清华园

北京市联华印刷厂印装

新华书店总店科技发行所发行

开本：787×1092 1/16 印张：14 字数：335千字

1984年10月第1版 1990年9月第5次印刷

印数：401001—426000 定价：3.90元

ISBN 7-302-00305-X/TP·113

前 言

本实验指导书是与教材《微型计算机硬件软件及其应用》一书配套的。

计算机是一种实践性很强的学科，学习计算机必须理论与实践紧密结合。理论学习要与一定数量的习题练习相结合，更重要的是上机练习。凡是具有上机条件的必须尽量争取多上机多练习。如果能够创造条件，做到上机练习时间为理论学习时间的1/2，甚至能做到1:1或更多，那一定会大大提高学习的效果，真正能提高分析问题和解决问题的能力。

本实验指导书紧密结合教材的内容，安排了三部分实验。

第一部分是单板机编程实验。由于限于条件，不是所有的单位都能具备足够数量的微型计算机系统的；而且对于初学者说来，在单板机上做编程实验也简单、方便。这一部分我们按由浅入深循序渐进的原则安排了九个实验。打*号的实验要求较高一些，可以作为学时较多的专业选用，或作为学习程度较高的同学选作。

第二部分是微型计算机系统实验。凡有条件的单位，应在单板机实验的基础上选作其中一部分。这样，有利于学生加深对微型计算机系统的了解；也可以初步掌握如何使用磁盘操作系统的命令和系统调用。在这一部分从机型上我们简要地介绍了一下目前国内较普遍的Cromemco系统和Apple II系统，而主要是介绍如何使用CP/M(或其变型)操作系统，在CP/M操作系统的支持下做实验。此部分中打*的几个实验编程要求较高，可以根据情况选作。

第三部分是硬件和接口实验，按照目前国内的实际情况，我们是以单板机加实验板来做这部分实验的。前面几个实验是硬件的基本逻辑电路实验和CTC及PIO的基本实验。对于加深教材中硬件和接口片子的了解是有好处的。打*的实验是一些带应用性质的实验，也可以根据条件加以选择。

在附录中，我们给出了主要实验的参考程序清单。我们的目的，一方面是为学生自己编程序时作参考。我们坚决主张学生在做实验前，必须要充分预习，充分准备，要依靠自己在实验前编出程序，经过实验调试改正程序，得出正确的结果。这样做实验，才真正有收获，真正能加深对理论的理解，也才真正能提高分析问题和解决问题的能力。我们坚决反对学生事先没有准备，实验时照参考程序敲键的做法，这样时间化得再多，收效也是不大的。我们所以要提供参考程序，只是给学生自己编了程序以后作为对照用的，以提高学生编程序的能力。另一方面的目的也是给大家提供一些实用程序。这些程序是有一定的典型性和实用性的。可以作为将来工作中编制更复杂程序的基础。

附录中提供了两种调试程序的主要命令，可帮助我们在实验中调试程序。

附录中也收集了一些常用集成电路的引脚图，对于实验，对于将来的工作都有一定的参考价值。

本实验指导书内容较丰富、系统，可以作为大学本科各专业知识学习微型计算机的实验指导书，也可以作为非计算机专业的研究生学习微型计算机的实验指导书。也可作为各

种类型的微型计算机培训班的实验指导书。

本实验指导书的第一部分内容是张淑玲同志编写的；第三部分中的电子秒表实验和交通信号灯实验是白晓笛同志准备的；A/D实验和A/D与D/A正逆变换实验是田开亮同志准备的；8251A串行通讯实验是付强同志准备的。全书由周明德同志主编。由于编者的水平有限，实践经验还不够丰富，缺点错误在所难免。敬请读者批评指正。

周明德

1984.2

目 录

前言	1
第一部分 单板机实验	1
第一节 TP801-A 单板计算机的结构和原理	1
第二节 TPBUG-A 介绍	6
第三节 TP801-A 使用说明	19
第四节 实验	22
第二部分 微型计算机系统实验	50
第一节 CROMEMCO 系统介绍	50
第二节 磁盘操作系统命令介绍	52
第三节 建立和运行汇编语言源程序的过程	65
第四节 微型计算机系统实验	79
第三部分 硬件和接口实验	102
第一节 TP80TS 实验板介绍	102
第二节 实验	103
附录	141
附录 1 第一部分 实验中的参考程序	141
附录 2 第二部分 实验中的参考程序	155
附录 3 硬件与接口实验的参考程序	175
附录 4	191
一、调试程序 (DEBUGGER) 使用介绍	191
二、动态调试工具 DDT (Dynamic Debugging Tool) 使用介绍	202
附录 5 常用集成电路引脚介绍	207

第一部分 单板机实验

第一节 TP801-A单板计算机的结构和原理

TP801-A 是以Z80-CPU为核心的单板微型计算机，其结构如图 1.1 所示。

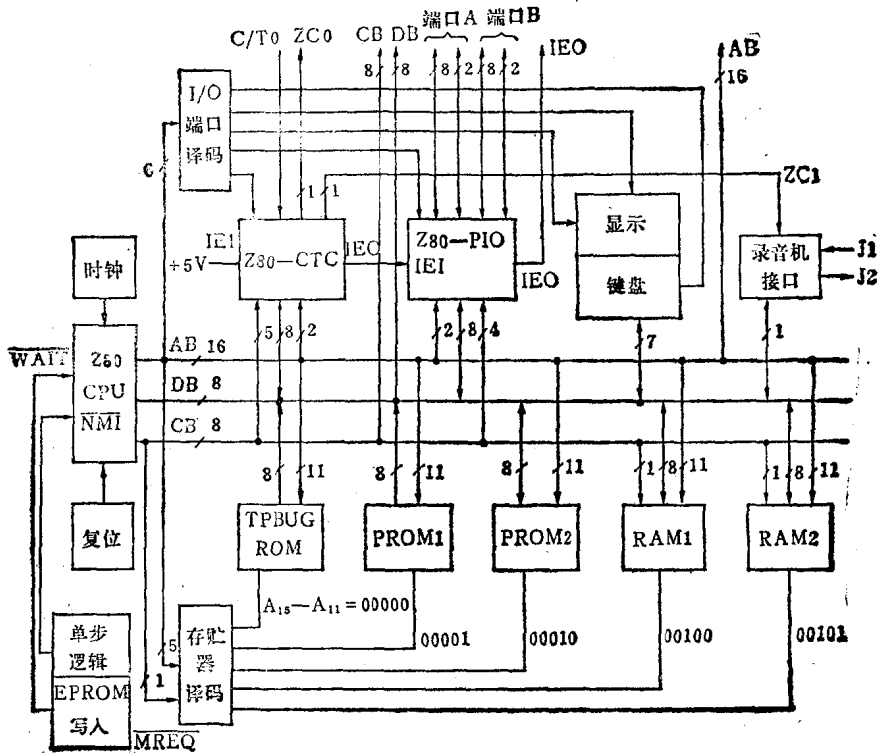


图1.1 TP801-A单板计算机原理框图

它以Z80-CPU为微处理器，时钟频率为2MHz。单板机上没有汇编程序，以机器语言进行操作，凡要在TP801上运行的用户程序，必须经过手工汇编，把它们变为机器码才能在单板机上运行。

TP801的存储器分成两大部分：ROM和RAM。

ROM又分为：ROM(其中驻有TPBUG-A)，PROM1和PROM2。每一种都为2K×8Bit的片子，构成2KROM。其中，ROM中存放系统的监控和调试程序TPBUG-A，它用于管理TP801的工作，也可用于调试用户的程序。

PROM1中可以存放固化的用户程序。若TP801用于一个固定的工作对象，则可将用户程序经过调试后就固化在PROM1中。若单板机的开关S₂合向下边(详见后面介绍)，则当系统复位(上电即自动复位，或因按RESET按钮引起的复位)后，在经过适当初始化后，就可直接进入和执行固化在PROM1中的用户程序，而不进入TPBUG-A。

PROM2 是一个 EPROM 片子，典型的为 Intel 2716。在 TP801 上，可以把在 RAM 中并经过调试的用户程序，写入到位于 PROM2 上的 EPROM 中。然后可以将此 EPROM 片子插至 PROM1 位置，令单板机执行固化在其中的用户程序。

RAM 又可分为 RAM1 和 RAM2。由于上述的三种 ROM 片子都是 $2K \times 8$ Bit 的，所以，RAM 也以 2K 字节作为一组，分别叫做 RAM1 和 RAM2。TP801-A 单板计算机的 RAM 容量基本配置是 4K 字节，但尚可以按需要扩展，再扩展 4K 字节是比较方便的。

RAM 可作用用户的程序区和数据区，以及系统和用户的堆栈。RAM 中有一部分空间用作系统的数据区。

Z80-CPU 有 16 条地址线，可寻址 64K 存储空间。而在 TP801-A 中的存储器，ROM 占 6K 存储空间，RAM 为 4-8K。那么，这些 ROM 和 RAM 的地址是如何分配的呢？RAM 中的用户区是在什么地址范围内呢？这就必须分析一下存储器的地址连线。TP801-A 的存储器结构如图 1.2 所示。

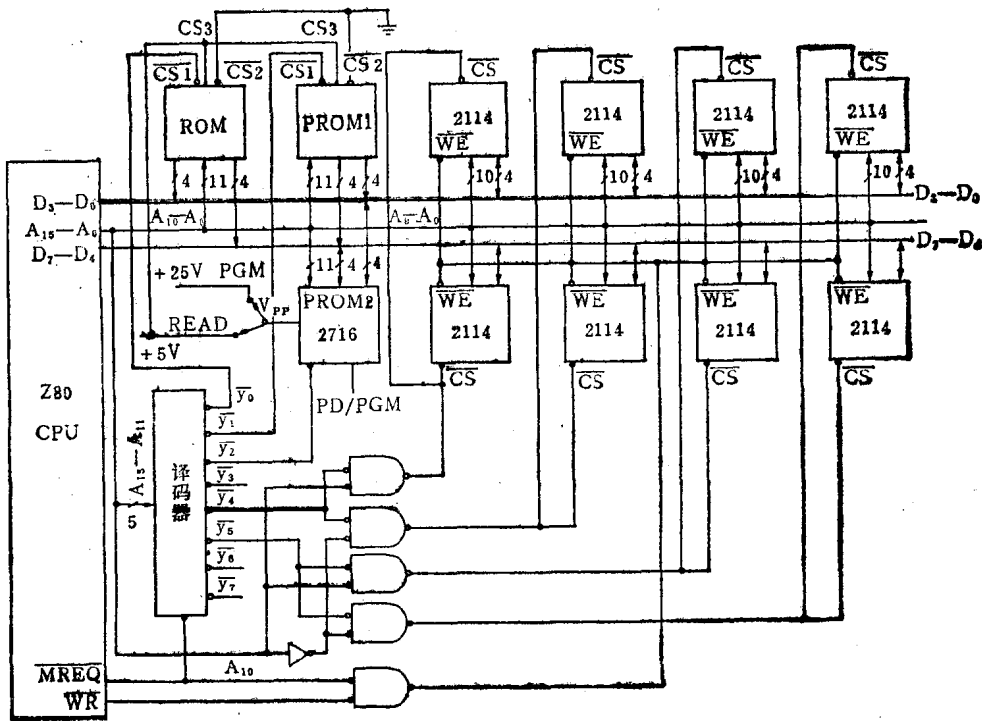


图 1.2 TP801-A 存储器结构

ROM、PROM1 和 PROM2 都是 $2K \times 8$ Bit 的片子，每个片子上有 11 条地址线，可直接区分片内的 $2^{11} = 2K$ 地址单元，它们可直接连至 Z80-CPU 地址总线的 A_0-A_{10} 。ROM 和 PROM1 的选片端都有 3 个，其中两个中一个为低电平有效的那端直接接地，而另一个高电平有效的那端直接接至 +5V；另一个选片端连至高位地址线的译码输出。

CPU 的高位地址线 $A_{11}-A_{15}$ ，连至译码器，如图 1.3 所示。

\bar{A}_{15} 连至选片端 G_1 ，而 A_{14} 连至选片端 G_2A 。故要选中此译码器片子，必须 $A_{15} = A_{14} = 0$ 。由于 A_{13} 、 A_{12} 、 A_{11} 的不同组合，译码器的输出有效的情况如图 1.3 中的真

A_{11} —A	1	16	$-V_{CC}$	G_1	$\overline{G_{2A}}$	$\overline{G_{2B}}$	C	B	A	输出	
A_{12} —B	2	15	$-\overline{Y}_0$	\overline{MREQ} — $\overline{G_{2B}}$	5	12	$-\overline{Y}_3$	1	0	0	$\overline{Y}_0=0$ 其余为 1
A_{13} —C	3	14	$-\overline{Y}_1$	A_{15} —G1	6	11	$-\overline{Y}_4$	1	0	0	$\overline{Y}_1=0$ 其余为 1
A_{14} — $\overline{G_{2A}}$	4	13	$-\overline{Y}_2$	\overline{Y}_7	7	10	$-\overline{Y}_5$	1	0	0	$\overline{Y}_2=0$ 其余为 1
\overline{MREQ} — $\overline{G_{2B}}$	5	12	$-\overline{Y}_3$	G_{ND}	8	9	$-\overline{Y}_6$	1	0	0	$\overline{Y}_3=0$ 其余为 1
A_{15} —G1	6	11	$-\overline{Y}_4$					1	0	0	$\overline{Y}_4=0$ 其余为 1
\overline{Y}_7	7	10	$-\overline{Y}_5$					1	0	0	$\overline{Y}_5=0$ 其余为 1
G_{ND}	8	9	$-\overline{Y}_6$					1	1	0	$\overline{Y}_6=0$ 其余为 1
								1	1	1	$\overline{Y}_7=0$ 其余为 1
								不是上述情况	× × ×		输出全为 1

图1.3 RAM的译码电路

值表所示。

由此可确定各个 ROM 的地址范围：

ROM:	$A_{15}A_{14}A_{13}A_{12}A_{11}$	A_{10} — A_0	
最低地址	0 0 0 0, 0	000, 0000, 0000	0000H
最高地址	0 0 0 0, 0	111, 1111, 1111	07FFH
PROM1:	$A_{15}A_{14}A_{13}A_{12}A_{11}$	A_{10} — A_0	
最低地址	0 0 0 0, 1	000, 0000, 0000	0800H
最高地址	0 0 0 0, 1	111, 1111, 1111	0FFFH
PROM2:	$A_{15}A_{14}A_{13}A_{12}A_{11}$	A_{10} — A_0	
最低地址	0 0 0 1, 0	000, 0000, 0000	1000H
最高地址	0 0 0 1, 0	111, 1111, 1111	17FFH

RAM1 和 RAM2 都是由 2114 片子组成。2114 是 $1K \times 4\text{Bit}$ 的片子，两片构成 1K 字节。故，RAM1 和 RAM2 各由 4 片 2114 组成。每一个片子上有 10 条地址线，它们直接连至地址总线的 A_0 — A_9 ，可以区分 1K 地址。RAM1 和 RAM2 由译码器的输出线 \overline{Y}_4 和 \overline{Y}_5 来区分。而 RAM1 和 RAM2 的两个 1K 字节就要由地址线 A_{10} 来区分。故， \overline{Y}_4 与 A_{10} 选择 RAM1 中的 2K 地址， \overline{Y}_5 与 A_{10} 选择 RAM2 中的 2K 地址。由此可以确定 RAM 的地址范围。

RAM1:

前 1 K 地址	$A_{15}A_{14}A_{13}A_{12}A_{11}$	A_{10}	A_9 — A_0	
最低地址	0 0 1 0, 0	0	00, 0000, 0000	2000H
最高地址	0 0 1 0, 0	0	11, 1111, 1111	23FFH

后 1 K 地址	$A_{15}A_{14}A_{13}A_{12}A_{11}$	A_{10}	A_9 — A_0	
最低地址	0 0 1 0, 0	1	00, 0000, 0000	2400H
最高地址	0 0 1 0, 0	1	11, 1111, 1111	27FFH

RAM2:	$A_{15}A_{14}A_{13}A_{12}A_{11}$	A_{10}	A_9 — A_0	
前 1 K 地址				
最低地址	0 0 1 0, 1	0	00, 0000, 0000	2800H
最高地址	0 0 1 0, 1	0	11, 1111, 1111	2BFFH
后 1 K 地址				

最低地址 0 0 1 0, 1 1 00, 0000, 0000 2C00H
 最高地址 0 0 1 0, 1 1 11, 1111, 1111 2FFFH

综合以上讨论, TP801-A 中存储器地址分配如表 1-1 所示。

2114 片子与 2MHz 的 Z80-CPU 相连接, 则即使在取指周期 2114 也能满足 CPU 的要求, 不需要在 CPU 的基本时序中插入 T_w 状态。

TP801-A 的监控程序需要利用 RAM 的一些空间作为它的数据区和堆栈。故 RAM 区域的分配如表 1-2 所示。

表 1-1

地 址	器 件	A ₁₅ —A ₁₁	A ₁₀ —A ₀	译码器的有效输出
0000—07FFH	2K ROM	0 0 0 0 0	可变	$\overline{Y}_0 = \overline{CS0} = \text{MON SEL}$
0800—0FFFH	2K PROM1	0 0 0 0 1	可变	$\overline{Y}_1 = \overline{CS1} = \text{PROM1 SEL}$
1000—17FFH	2K PROM2	0 0 0 1 0	可变	$\overline{Y}_2 = \overline{CS2} = \text{PROM2 SEL}$
1800—1FFFH	没用	0 0 0 1 1	可变	$\overline{Y}_3 = \overline{CS3}$ 没用
2000—27FFH	2K RAM1	0 0 1 0 0	可变	$\overline{Y}_4 = \overline{CS4} = \text{RAM1 SEL}$
2800—2FFFH	2K RAM2	0 0 1 0 1	可变	$\overline{Y}_5 = \overline{CS5} = \text{RAM2 SEL}$
3000—37FFH	备用	0 0 1 1 0	可变	$\overline{Y}_6 = \overline{CS6}$ 没用
3800—3FFFH	备用	0 0 1 1 1	可变	$\overline{Y}_7 = \overline{CS7}$ 没用

表 1-2

地 址 空 间	用 途	字 节 数
2000—23FFH	RAM1 的用户区	1K
2400—27FFH	RAM1 的用户区	1K
2800—2BFFH	RAM2 的用户区	1K
2C00—2F87H	RAM2 的用户区	904
2F88—2F9FH	监控程序堆栈工作区	24
2FA0—2FB7H	用户程序寄存器存放区, 用户程序堆栈区	24
2FB8—2FBFH	TP801-A 4 个用户程序的入口地址	8
2FC0—2FFFH	TPBUG-A 使用的 RAM 暂存区和断点表	64

TP801-A 的 I/O 设备及接口电路如图 1.4 所示。

接有一个 Z80-CTC 片子, 其中通道 1、2 和 3 用于监控程序, 通道 0 留给用户用。用户可根据需要把它编程为定时器或计数器用。

有一个 PIO 片子, TP801-A 没有使用, 故它的两个通道都可留给用户用。

TP801-A 的典型输入是键盘, 有 16 个 16 进制数字键和 12 个功能键, 如图 1.5 所示。

这些键的功能和使用方法在后面介绍。

TP801-A 的典型输出设备是 6 个 7 段显示管, 可以显示机器码。通常左面 4 个显示管显示内存单元的地址或 CPU 内部寄存器号, 或端口地址; 而右面两个, 显示该单元或寄存器或外设端口的内容。

此外, TP801-A 还可以与盒式磁带机相连。可以把内存中的内容转贮到磁带上, 或

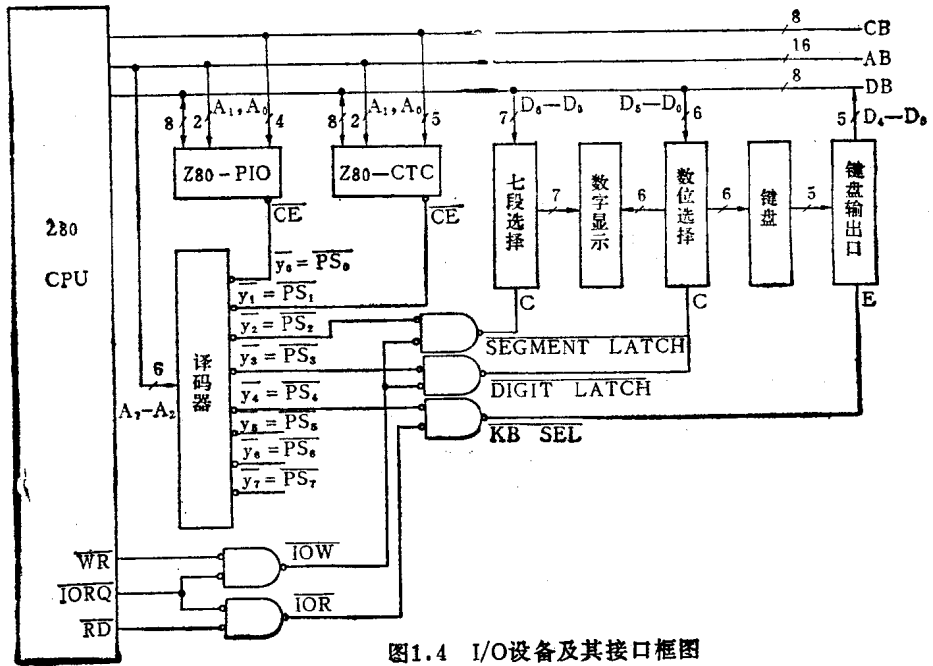


图1.4 I/O设备及其接口框图

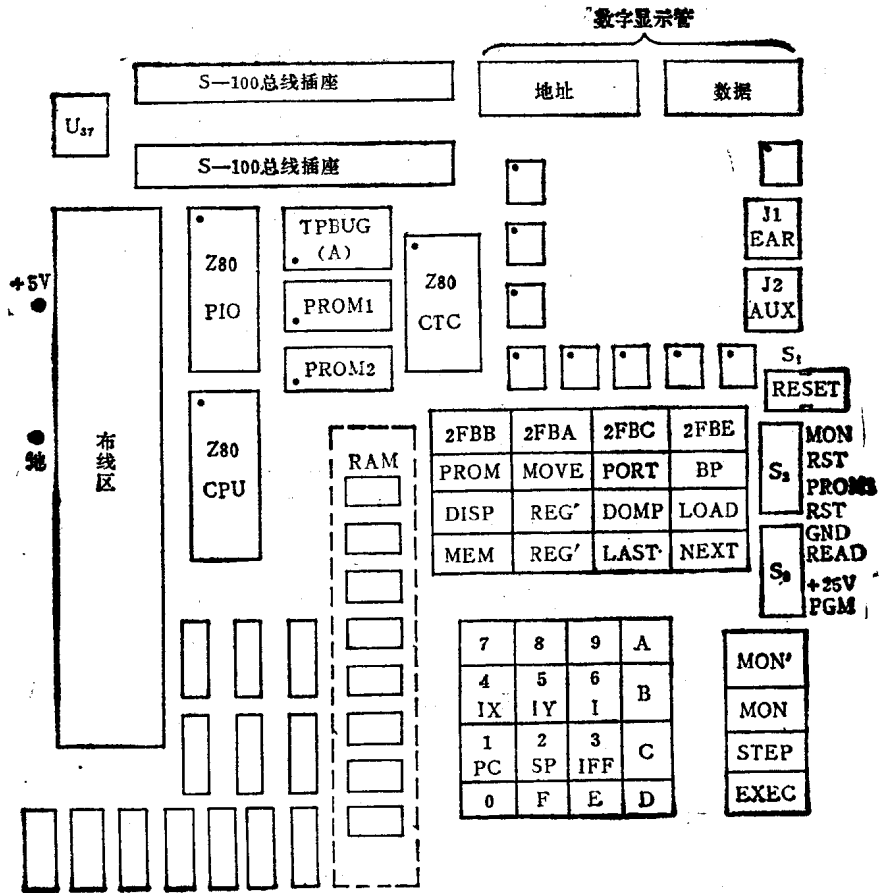


图1.5 TP801-A面板布置图

把磁带上记录的信息输入内存中。

所有上述这些 I/O 设备和接口电路，都要由地址总线的低 8 位来加以区别和选择。

Z80-PIO 以及 Z80-CTC，都要用地址总线的最低两位 A_1 和 A_0 来选择片内的不同端口。所以，要由地址总线的高 6 位—— A_7-A_2 经过译码器后作为选片信号。于是 TP801-A 的 I/O 设备和接口电路的端口地址如表 1-3 所示。

TP801-A 是一个单板计算机，但任何一个计算机都必须在一个系统程序的管理下才能正常工作，在单板机中此程序固化在 ROM 中，称为监控程序，在 TP801-A 中即为固化在 ROM 中的 TPBUG-A。

表 1-3

A_7-A_2	译码器输出	器 件	A_1A_0	端 口	端口地址
1 0 0 0 0 0	$\bar{Y}_0 = \overline{PS0} = \text{PIO SEL}$	Z80-PIO	0 0	端口 A 数据	80H
			0 1	端口 B 数据	81H
			1 0	端口 A 控制寄存器	82H
			1 1	端口 B 控制寄存器	83H
1 0 0 0 0 1	$\bar{Y}_1 = \overline{PS1} = \text{CTC SEL}$	Z80-CTC	0 0	通道 (Ch) 0	84H
			0 1	通道 1	85H
			1 0	通道 2	86H
			1 1	通道 3	87H
1 0 0 0 1 0	$\bar{Y}_2 = \overline{PS2} = \text{SEGLH}$	74LS273 (八锁存器)	× ×	七段选择 (只写)	88—8BH
1 0 0 0 1 1	$\bar{Y}_3 = \overline{PS3} = \text{DIGLH}$	74LS273	× ×	数位选择 (只写)	8C—8FH
1 0 0 1 0 0	$\bar{Y}_4 = \overline{PS4} = \text{KB SEL}$	74LS244 (八缓冲器)	× ×	读键值 (只读)	90—93H
1 0 0 1 0 1	$\bar{Y}_5 = \overline{PS5}$	没使用			94—97H
1 0 0 1 1 0	$\bar{Y}_6 = \overline{PS6}$	没使用			98—9BH
1 0 0 1 1 1	$\bar{Y}_7 = \overline{PS7}$	没使用			9C—9FH

第二节 TPBUG-A 介绍

TPBUG-A 是一个 2K 字节的监控和调试程序，它主要是由以下几部分程序组成。

1. 初始化程序；
2. 键盘输入及处理程序；
3. 显示程序；
4. 各个功能键的处理程序；
5. 公用子程序。

这些程序在教材的第十章中做了较详细的分析。

为了实验中和以后工作中的需要，我们在这儿详细介绍一下显示程序与键盘输入程序。

一、显示程序

(一) 数码显示管

TP801-A 中所用的是 5V 电源的七段数字显示管，每一段为一发光二极管，其结构及原理如图 1.6 所示。

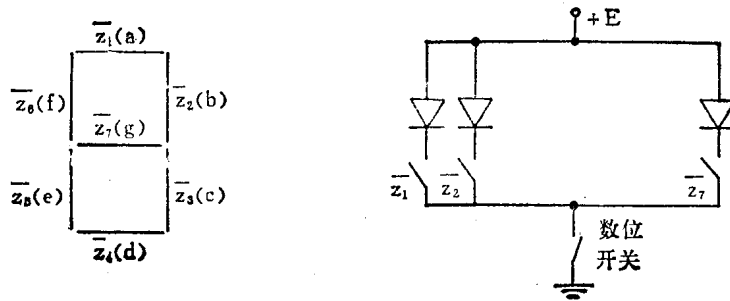


图1.6 显示管原理图

七段，共有七个段控制信号，以控制所显示的字形。当某一段的控制信号为低电平时，则相当于相应的触点闭合，这一段发光；整个管还有一个数位控制信号，也是低电平闭合，只有此触点闭合时，数字管才能发光。

七段用七位数即用 CPU 输出的 D_0 - D_7 控制， D_7 始终为 0，因而 16 进制数的显示编码如表 1-4 所示。

TP801-A 中用 6 个这样的数字管，可显示 6 个 16 进制数，其硬件电路如图 1.7 所示。

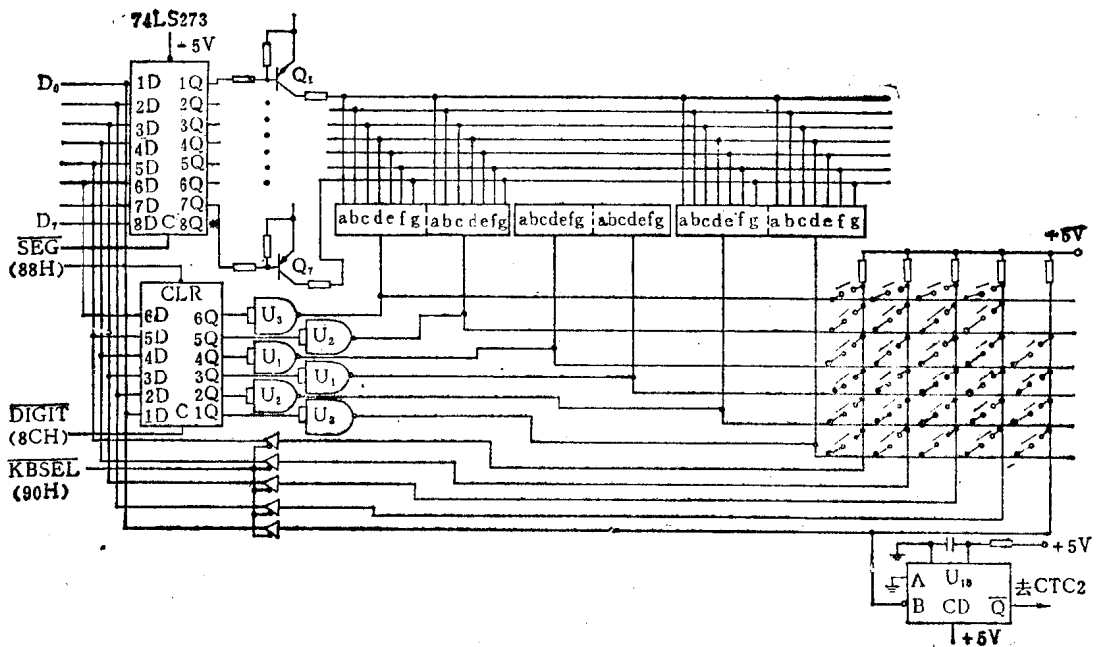


图1.7 键盘输入和显示硬件电路图

表 1-4

数 符	Z ₈	Z ₇	Z ₆	Z ₅	Z ₄	Z ₃	Z ₂	Z ₁	16进制字模编码
0	0	1	0	0	0	0	0	0	40H
1	0	1	1	1	1	0	0	1	79H
2	0	0	1	0	0	1	0	0	24H
3	0	0	1	1	0	0	0	0	30H
4	0	0	0	1	1	0	0	1	19H
5	0	0	0	1	0	0	1	0	12H
6	0	0	0	0	0	0	1	0	02H
7	0	1	1	1	1	0	0	0	78H
8	0	0	0	0	0	0	0	0	00H
9	0	0	0	1	1	0	0	0	18H
A	0	0	0	0	1	0	0	0	08H
b	0	0	0	0	0	0	1	1	03H
c	0	1	0	0	0	1	1	0	46H
d	0	0	1	0	0	0	0	1	21H
E	0	0	0	0	0	1	1	0	06H
F	0	0	0	0	1	1	1	0	0EH

要显示的字符的编码由数据总线送至锁存器 $\overline{\text{SEG}}$ (端口地址 88H) 锁存, 后经三极管 Q_1-Q_7 , 供给大的驱动电流, 连到所有数字管相应的段。由于要显示的字模编码是同时连到所有显示管的, 所以要使某一个显示管显示出数码, CPU 还必须输出一个数位控制字 (只有 D_0-D_6 有用), 它由 DIGIT 锁存器锁存, 以选择哪一个显示管工作。所以, 要显示一个数码, 首先要把这个数码转换成相应的字模编码, 再输出给端口 88H ($\overline{\text{SEG}}$); 然后把控制哪一个数字管亮的数位控制字输出给端口 8CH ($\overline{\text{DIGIT}}$)。

(二) 显示程序

要显示的数放在以 DISMEM (在 TP801-A 中为 2FF7H) 单元开始的缓冲区中, 总共为 6 个单元, 由于每个数字显示管只能显示一位 16 进制数, 故这 6 个存储单元的高 4 位全为 0, 低 4 位为要显示的数的二进制数码。故若要显示一个存储单元或某一个寄存

器的内容，则必须通过一个子程序把这一字节的高4位和低4位拆开，放至显示缓冲区的两个单元中。

要显示某一个16进制数，首先要把这个数转换为显示管段形的字模编码，这个转换是由软件通过搜索一个表格（字模表）来实现的。这个表格以16进制数的次序，把它们相应的字模编码顺序存放，我们把表格的起始地址送IX，以要显示的数作为偏移量，由IX加上偏移量找到字模编码的地址，则从表格中取出来的即为相应的数的字模编码。

显示程序的流程图如图1.8所示。

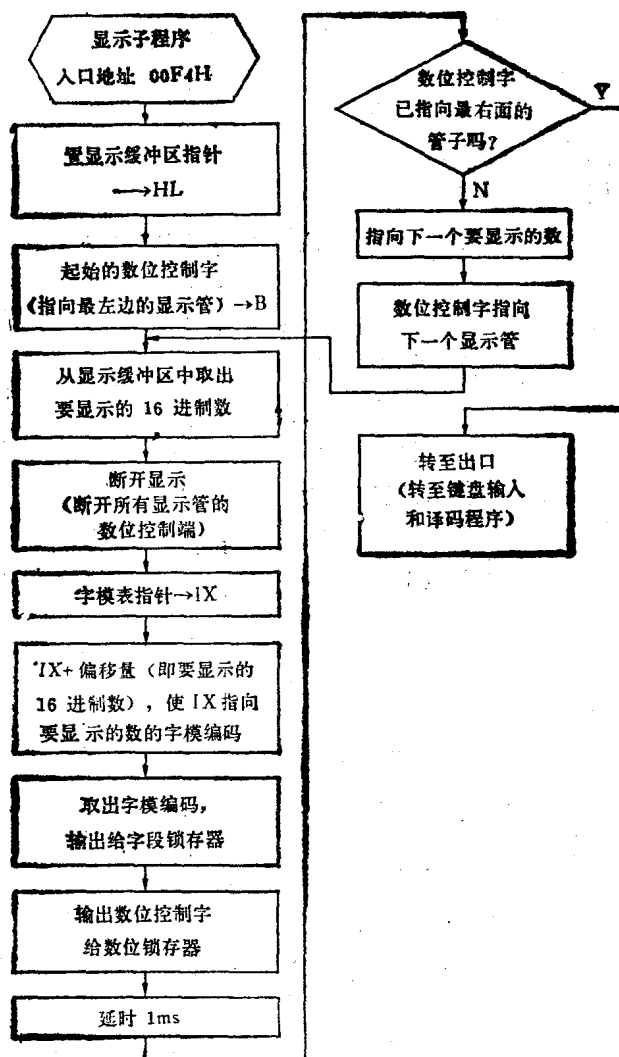


图1.8 显示程序流程图

显示程序为：

； 显示程序
； 功能：把数据从 DISMEM 开始的缓冲区送显示管。

输入: 要显示的数放在以 DISMEM 开始的 6 个单元中。

输出: 把预置的 6 个数输出以更新原有的显示。

所用的寄存器: HL 是指向 DISMEM 的指针

IX 是指向字模表的指针

B 是当前数位指示器

E 保留要被显示的数

A 和 D 是暂存寄存器

ORG 00F4H

```
DISUP:    LD    HL, DISMEM    ; 指向显示缓冲区的起点
          LD    B, 20H      ; 指向最左边的显示管
DISUP1:   LD    E, (HL)     ; 从显示缓冲区取数
          LD    D, 0
          LD    A, 0
          OUT  (DIGLH), A   ; 断开显示
          LD    IX, SEGPT   ; IX 指向字模表起点
          ADD  IX, DE       ; 指向要显示的数的字模
          LD    A, (IX+0)   ; 取出字模编码→A
          OUT  (SEGLH), A   ; 输出至段形锁存器
          LD    A, B
          OUT  (DIGLH), A   ; 输出至数位锁存器 (控制是第几个
                               ; 数字显示管亮)
          LD    E, 45       ; 置 1ms 延时
DISUP2:   DEC  E
          LD    A, 0
          CP   E
          JR   NZ, DISUP2
          LD    A, 01H
          CP   B            ; B = 1? (即是否已移至最右面的显
                               ; 示管)
          JR   Z, DISUP3   ; 是, 转出口
          INC  HL           ; 指向下一个要显示数
          SRL  B            ; 移至下一个显示管
          JR   DISUP1       ; 循环
DISUP3:   JP   DECKY       ; 转至键盘输入程序
DIGLH:   EQU  8CH
SEGLH:   EQU  88H
          ORG  07A6H
SEGPT:   DB   40H          ; 0
          DB   79H          ; 1
```


DB	24H	; 2
DB	30H	; 3
DB	19H	; 4
DB	12H	; 5
DB	02H	; 6
DB	78H	; 7
DB	00H	; 8
DB	18H	; 9
DB	08H	; A
DB	03H	; B
DB	46H	; C
DB	21H	; D
DB	06H	; E
DB	0EH	; F
DB	7FH	; 空白
DB	0CH	; 响应符 P
DB	5FH	; 撇号 ' ,'

二、键盘输入程序

在 TP801-A 中是用 16 个数字键和 12 个命令键输入的，但键盘上方的 8 个命令键又可分为上、下两档，每一个键可以有二种命令。每一个键相当于一个按钮，把所按的键转换成相应的数或命令是通过键盘输入与译码程序实现的。

(一) 键盘输入硬件电路

键盘输入和显示的硬件电路见图 1.7。

数据总线的信号 D_0-D_3 经反相后输出作为行线（键盘的输入信号线），它由端口 DIGIT（地址为 8CH）控制；5 条列线由 +5 V 电源经 10K 电阻引出（键盘的输出信号线），通过三态缓冲器引至数据总线的 D_4-D_8 ，由端口 KBSEL（地址为 90H）控制。

在每一行与每一列之间接有一个键。

在工作时，程序按行扫描键盘，检查列的输出，由行信号与列信号的组合以确定是哪一个键闭合。即先让 $D_0=1$ （反相后为低电平）， $D_1-D_3=0$ ，此数据由端口 8CH 输出，即使键盘的最下面一行为低电平，若在此行中有键闭合，则相应的列输出为 0（低电平），而其他列为 1。若这一行没有键闭合（即所有的列全为高电平），则再使上面一行为低电平，再检查列的输出，有键闭合的列输出为 0，否则为 1……，这样一行一行地扫描，由行的值与列的输出的配合，即可确定是哪一个键闭合。

例如，数码 0 闭合，则其对应的行值为 01H，而列的输出即为 0FH；数码 1 则为：行=02H，列=0FH；数码 9 则为：行=04H，列=1BH……如表 1-5 所示。

键盘输入程序首先检查是否有任一键闭合，若无任一键闭合则转至显示程序。当发现有某一键闭合时，则按行扫描键盘，每扫一行输入列值检查此行中是否有键闭合，若有键闭合，则转至键盘译码程序以确定此键值；若无键闭合，则扫描另一行……。程序