

目 录

(上 册)

第一章 Z80单板机的基本结构及电路分析

1.1 Z80—CPU	(1)
1.2 Z80—PIO	(2)
1.3 80—CTC	(3)
1.4 时钟电路与复位电路	(5)
1.5 存储空间分配及译码	(6)
1.6 I/O接口的硬件连接	(8)
1.7 显示部分的硬件连接	(10)
1.8 键盘扫描电路	(13)
1.9 EPROM 写入部分的电路	(15)
1.10 磁带转储部分的电路	(18)

第二章 JXBUG监控程序简介

2.1 JXBUG监控程序的基本情况	(21)
2.2 JXBUG 监控程序的工作状态及键盘布置	(23)

第三章 JXBUG的键盘操作说明

3.1 复位操作—RESET 键	(25)
3.2 送数操作—十六进制数字键	(26)
3.3 返回待命状态—MON键	(26)
3.4 读写存储器—MEM、NEXT、LAST键	(27)
3.5 连续送数及删除、插入—AUT、DEL、INS键	(28)
3.6 读写寄存器—REG、NEXT、LAST、RMEM、NXDG键	(30)
3.7 读写输入输出接口—PORT、NEXT、LAST键	(31)
3.8 设置与清除断点—STBP _i 键	(32)
3.9 查找断点—GTBP键	(33)
3.10 单步执行及单步调用—STEP、SCA	(34)
3.11 连续执行—EXEC、USRO、USRE、USRF键	(36)
3.12 字符串搜索—SEEK键	(37)
3.13 求算并存放相对寻址偏量—OFST键	(38)
3.14 信息转贮磁带—DUMP 键	(40)
3.15 磁带信息校验与装入—CHEK、LOAD、1/FUNC键	(41)
3.16 EPROM检查及编程—EPRG键	(43)
3.17 十进制数与十六进制数的互相转换—DEC、HEX键	(45)
3.18 派生255档可调数字式低频信号发生器—OXXX/FUNC键	(46)

3.19 直接转移执行—2 XXX/FUNC键	(47)
3.20 成块置数—3/FUNC键	(48)
3.21 数据块传送—4/FUNC键	(48)
3.22 数据块比较—5/FUNC键	(49)
3.23 循环考机功能	(49)

第四章 单板机应用实验

第一部分 熟悉基本功能, 掌握编制程序和调试程序的技巧

实验一 简单算术运算程序	(51)
实验二 分支程序	(54)
实验三 循环程序	(55)
实验四 堆栈与子程序	(57)

第二部 分接口实验

实验五 显示器实验	(59)
实验六 键盘扫描程序	(61)
实验七 Z80—CTC应用	(63)
实验八 时钟程序	(66)
实验九 Z80—PIO的中断试验	(68)
实验十 A/D转换实验	(70)
实验十一 D/A转换实验	(71)

(下 册)

第五章 JXBUG监控程序及调用举例

5.1 与显示有关的子程序	(73)
5.2 与键盘(包括扫描显示)有关的子程序	(74)
5.3 保留的ZBUG监控(或TP801—A)的子程序及功能入口	(75)
5.4 十进制数(BCD数)←→十六进制数转换子程序	(76)
5.5 其它子程序及命令入口	(76)
5.6 JXBUG程序调用举例	(77)

第六章 趣味程序及非屏蔽中断使用方法

6.1 单板机乒乓球游戏程序	(80)
6.2 简易电子琴	(85)
6.3 非屏蔽中断使用方法	(92)

附录一 JXBUG监控程序清单	(95)
-----------------------	------

附录二 未正式发表的Z—80指令	(132)
------------------------	-------

附录三 常用集成电路引脚图	(134)
---------------------	-------

附录四 Z80单板机电路图	
---------------------	--

第一章 Z80单板机的基本结构及电路分析

Z80单板计算机的原型是美国的Z80 STARTER KIT,它以Z80-CPU为核心,配上固化于EPROM之中2K字节的监控程序ZBUG,具有28键的键盘用于输入数字和命令,6个LED数码管作为输出显示、磁带机接口用于建立外存,此外它还具有设置断点、单步执行、EPROM编程等开发功能,因而可广泛用于工业控制、智能仪器、数据处理等领域。由于该机结构简单,操作方便,使它尤其适合于初学者使用。在学习使用该机的过程中,可具体地了解微机的硬件构造,熟悉Z80的指令系统并进行汇编语言程序设计的练习。初学者还能从学习监控程序之中了解系统软件的一般构造方法和许多编程技巧。

在Z80单板机引入我国之后,许多大专院校和科研部门纷纷对它的硬、软件进行了分析和改进,推出了种种功能更强的机型。其中,江西工学院在未改变硬件结构的前提下,研制了新监控程序JXBUG,使Z80单板机的功能得到大幅度的提高。

1.1 Z80-CPU

Z80-CPU是一个40条引脚封装的大规模集成电路芯片。它的结构框图参见图1.1。

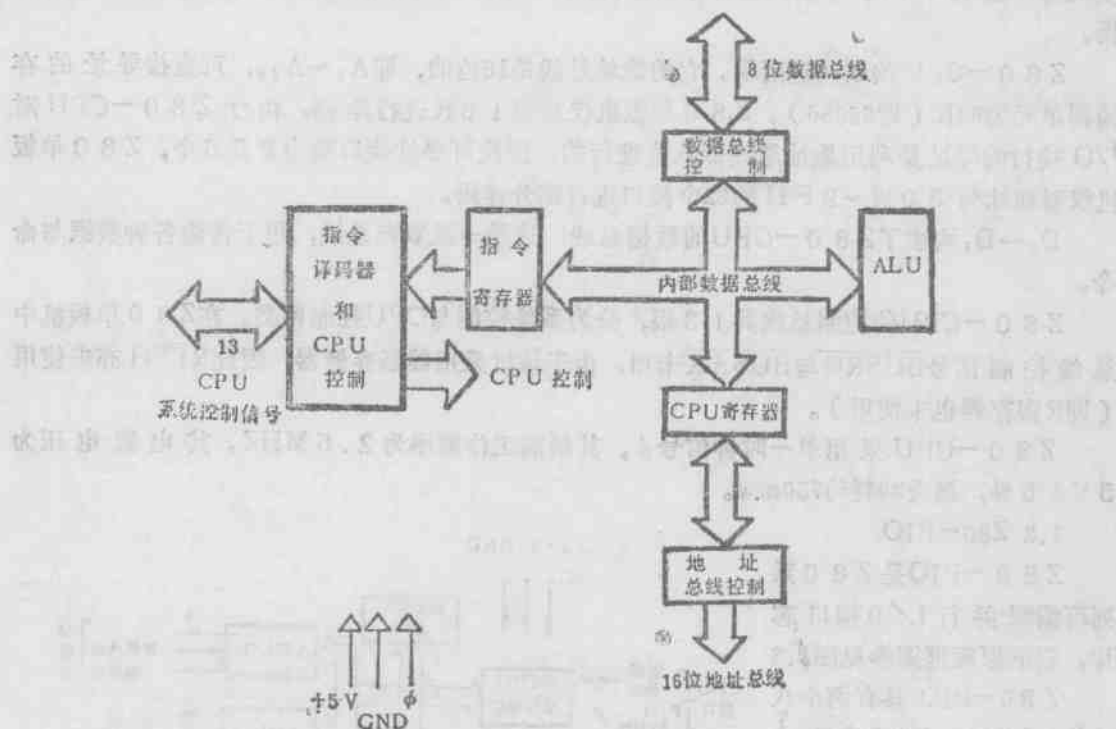


图 1.1 Z80-CPU 结构框图

Z80-CPU由CPU寄存器、算术逻辑运算单元(ALU)、指令寄存器(IR)、指令译码与CPU控制部分、数据总线控制部分和地址总线控制部分组成。从用户的角度看,CPU寄存器配置如图1.2。



图1.2 Z80—CPU寄存器配置

从图1.2中可以看到，CPU中有八个八位的通用主寄存器，包括累加器A，标志寄存器F以及寄存器B、C、D、E、H、L这八个寄存器，还各有辅助寄存器与之一一对应，在需要时，它们还可以构成十六位的寄存器对：AF、BC、DE、HL及相应的辅助寄存器对。此外，CPU中还有四个十六位专用寄存器，即堆栈指示器SP、程序计数器PC及变址寄存器IX、IY。中断向量寄存器I及动态RAM刷新寄存器R则是两个八位专用寄存器。

Z80—CPU的引脚见附录。它的地址总线是16位的，即 $A_0 \sim A_{15}$ ，可直接寻址的存储器单元为64K（即65536）。Z80单板机仅对前16K进行译码。由于Z80—CPU对I/O接口的寻址是利用地址总线低八位进行的，因此可寻址接口数为256个，Z80单板机仅对地址为80H~9FH的32个接口进行部分译码。

$D_0 \sim D_7$ 构成了Z80—CPU的数据总线，这是一组双向总线，用于传输各种数据与命令。

Z80—CPU的控制总线共13根，分为系统控制与CPU控制两类。在Z80单板机中总线控制信号 $\overline{BUSERQ}$ 与 $\overline{BUSERQ}$ 未用。由于该机采用静态存储器，因此 $\overline{RFSH}$ 亦未使用（即R寄存器也未使用）。

Z80—CPU采用单一时钟信号 ϕ ，其最高工作频率为2.5MHz。其电源电压为 $5V \pm 5\%$ ，最大功耗约750mw。

1.2 Z80—PIO

Z80—PIO是Z80系列可编程并行I/O接口芯片。它的原理框图参见图1.3

Z80—PIO具有两个八位的I/O口：A口与B口。它们的全部输入、输出电平均与TTL兼容，其中B口的八根输出线都能驱动复合晶体管（1.5mA、1.5V），A口的

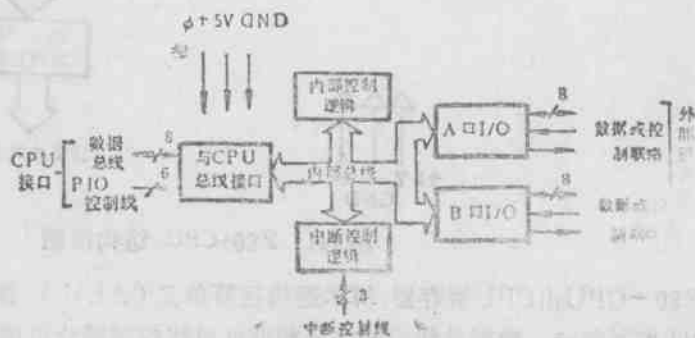


图1.3 PIO框图

输出能驱动一个标准的TTL负载。

PIO的任一个口可以在程序设定下处于下列四种方式之一：字节输入、字节输出、双向工作（仅A口）和位输入/输出（或称位控）。

PIO具有实现链式优先权的中断逻辑。并可在不另加硬件情况下，按中断方式2（IM2）工作。其 $\overline{INT}$ 端直接与CPU的同名端相连。当CPU响应它的中断请求时，它能自动提供中断矢量低字节。

PIO是40个引脚双列直插封装的芯片，其引脚布置参见附录。图中， $PA_0 \sim PA_7$ 、 $PB_0 \sim PB_7$ 分别为A口和B口的双向总线； $ARDY$ 、 $\overline{ASTB}$ 、 $BRDY$ 、 $\overline{BSTB}$ 分别为其联络信号； $D_0 \sim D_7$ 与CPU数据总线直接相连，用于传输数据或命令。PIO的六根控制线可分为地址形成线（ $\overline{CE}$ 、 $B/\overline{A}$ 与 $C/\overline{D}$ ）和功能控制线（ $\overline{M_i}$ 、 $\overline{IORQ}$ 与 $\overline{RD}$ ）两组，其作用参见表1.1与表1.2。

表1.1 PIO地址形成

引 $\overline{CE}$	脚		所选定的寄存器
	$C/\overline{D}(A_1)$	$B/\overline{A}(A_0)$	
0	0	0	A口的数据寄存器
0	0	1	B口的数据寄存器
0	1	0	A口的控制寄存器
0	1	1	B口的控制寄存器
1	×	×	本芯片不被选中

注：×表示为任意值。

表1.2

引		脚	功 能
$\overline{M_i}$	$\overline{IORQ}$	$\overline{RD}$	
0	0	1	中断响应
0	1	0	执行RETI指令时，使IEO恢复
0	1	1	复位
1	0	0	CPU从PIO读出
1	0	1	从CPU写入PIO

和Z80-CPU一样，PIO采用单一的+5V_i电源和单相时钟信号 ϕ ，其最高工作频率亦为2.5MHZ。

在本机中，PIO的两个通道均未被系统占用，各口的双向总线及联络线均引至布线区，供用户使用。

1.3 Z80-CTC

Z80-CTC是Z80系列可编程定时器/计数器接口芯片。其原理框图参见图1.4。

Z80-CTC具有四个独立的，功能近似的通道，每个通道都可由程序定为按计数器或定时器方式工作，其中0、1和2通道，无论处于何种工作方式，当到达程序设定的值时，产生的正脉冲输出（宽度约1.5倍时钟周期）可驱动复合晶体管。芯片的所有输入、输出电平均与TTL兼容。

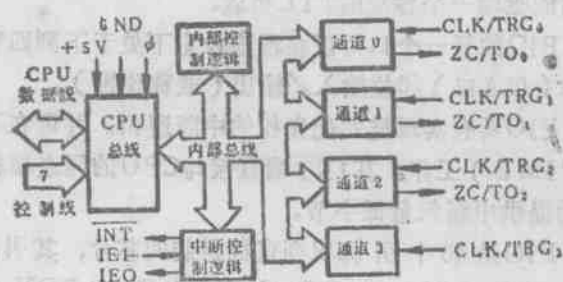


图 1.4 CTC的方框图

CTC的中断工作特点与PIO相似，只是它内部四个通道的中断优先级已固定设置成0通道最高，3通道最低。

CTC是28个引脚的双列直插封装芯片，其引脚布置参见附录。图中，CLK/TRG为各通道定时器启动外触发脉冲或计数脉冲输入端；ZC/TO为各通道减1计数器回零时正脉冲输出端（通道3无此引脚）； $D_0 \sim D_7$ 与CPU数据总线直接相连。其六根控制线如同PIO一样可分为地址形成线（ $\overline{CE}$ 、 CS_0 和 CS_1 ）和功能控制线（ $\overline{Mi}$ 、 $\overline{IORQ}$ 与 $\overline{RD}$ ）两组，其作用参见表1.3与表1.4。

表 1. 3

引		脚		所寻址的通道号
$\overline{CE}$ (16)	CS_1 (19) A_1	CS_0 (18) A_0		
0	0	0		0
0	0	1		1
0	1	0		2
0	1	1		3
1	任意	任意		芯片未被选中

表 1. 4

引		脚		功能说明
$\overline{Mi}$	$\overline{IORQ}$	$\overline{RD}$		
0	0	1		中断响应
0	1	0		在执行RETI时，使IEO恢复
1	0	0		CPU从CTC读出
1	0	1		从CPU写入CTC

与PIO不同的是CTC有专门的复位输入端RESET，故不需另行组合逻辑电路。

在本机中，CTC的通道0与通道3均供用户使用。通道1供磁带转储使用，通道2供单步、EPROM编程及MON键动作程序使用。

1.4 时钟电路与复位电路

一、时钟电路

Z80单板机的时钟电路由晶体振荡器与分频器两部分组成，参见图1.5

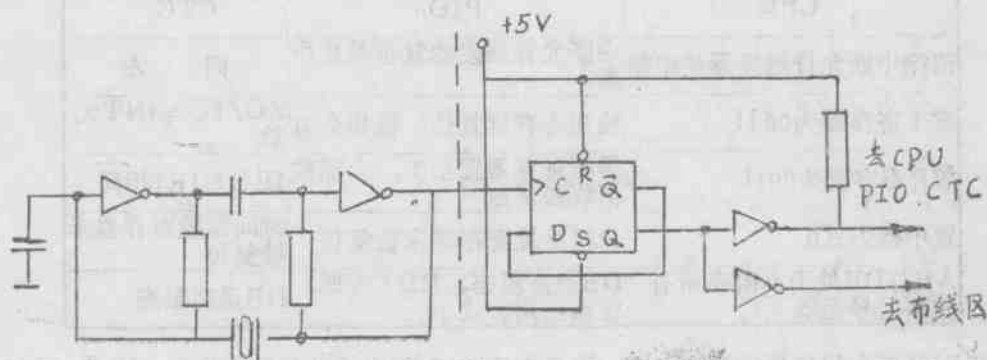


图1-5 时钟电路

图中，虚线左边为晶振电路，由它送出的信号通过D触发器74LS74 (1/2 U30) 分频后，自两个反相器分别输出，一路送往CPU，PIO与CTC，另一路送至布线区，供用户扩展时使用。

如前所述，Z80系列器件的最高工作频率为2.5MHZ。该机采用的晶振为3.9936MHZ，经分频后，送出的时钟频率为1.9968MHZ。

值得说明的是：该机的晶振电路是普及型电路，所采用的晶体等级和晶振的环境控制要求均较低，因此一般说来，在上电后的数十分钟内，振荡频率会略有漂移，最终稳定值与标称频率亦略有偏差。用户在编制对时间精度要求较高的程序时，应加以注意。

二、复位电路

复位电路是在上电或人工复位（按RESET按钮）时，产生一个宽度大于三个时钟周期的低电平给CPU、PIO、CTC并送至布线区，使相应器件产生复位动作。其原理图参见图1.6。

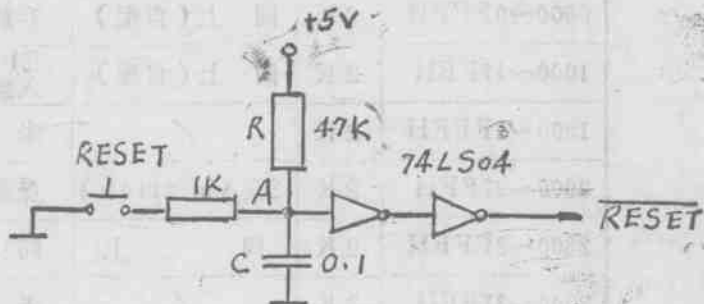


图1.6 复位电路

如图所示, 由于电容器 C 的存在, 因此在上电时 A 点初始电位为低电位, 并需经过一段时间延迟方上升至高电位; 同样, 在按下 RESET 按钮时 A 点降至低电位, 此时 C 放电, RESET 按钮弹起后, A 点亦需经过一段时间延迟再上升至高电位。因 A 点的复位脉冲经过两个反相器整形驱动后输出并且其宽度远大于三个时钟周期, 故能可靠复位。

复位脉冲送至 CPU、PIO 与 CTC 后产生一系列复位操作, 见表 1.5。

表 1.5 Z80 主要器件的复位操作

CPU	PIO	CTC
清除中断允许触发器关中断	中断允许触发器复位禁止中断	同 左
置 I 寄存器为 00H	输出寄存器复位, 输出全 0	ZG/TO 与 INT 无效
置 R 寄存器为 00H	屏蔽寄存器置全 1, 全部数据线被屏蔽	IEO 与 IEI 相同
置中断方式 0	矢量地址寄存器未被复位	时间常数寄存器未被复位
AB 与 DB 处于高阻态所有控制信号无效	DB 呈高阻态, RDY 为低, 自动选用方式 1	DB 呈高阻态

由于 PIO 无专门复位信号输入端, 该机为使 PIO 能自动进行复位状态, 特将 CPU 的 RESET 端和 $\overline{M}_1$ 端相“与”后引入 PIO 的 $\overline{M}_1$ 端。这样, 当 RESET 有效时, 将使 PIO 的 $\overline{M}_1$ 引脚有效, 此时因 $\overline{RD}$ 及 $\overline{IORQ}$ 均无效, 从而使 PIO 复位。

1.5 存储空间分配及译码

一、存储空间的分配

如前所述, 该机仅对可直接寻址存储器的前 16 K 进行译码, 其分配见表 1.6 与 1.7。

表 1.6

地址范围	字节数	器 件	内 容	插座号
0000~07FFH	2 K	EPROM 2716	存放监控程序 TPBUG	U ₇
0800~0FFFH	2 K	同 上 (自配)	存放用户程序	U ₈
1000~17FFH	2 K	同 上 (自配)	EPROM 写 入或用户程序	U ₉
1800~1FFFH	2 K	/	未 用	
2000~27FFH	2 K	SRAM 2114×4	见表 1.7	U ₁₀ ~U ₁₂
2800~2FFFH	2 K	同 上	同 上	U ₂₀ ~U ₂₃
3000~37FFH	2 K	/	未 用	
3800~3FFFH	2 K	/	未 用	

表1.7 RAM存储分配

地 址	字节数	插 座 号	用 途
2000~27FFH	2K	U ₁₀ ~U ₁₉	RAM (1) 用户工作区
2800~2F8FH	1936		RAM (2) " "
2F90~2FA7H	24	U ₂₀ ~U ₂₃	监控程序栈工作区
2FA8~2FBFH	24		用户栈工作区
2FC0~2FFFH	64		监控的暂存区与断点表

二、片选信号的获得

为了获得如表 1.2 所示的八个地址区, 该机采用一块 74LS138 对地址进行译码, 其输入与输出参见图 1.7

对图 1.10 可分析如下:

1、由于把地址线 A₁₅、A₁₄ 和控制线 $\overline{\text{MREQ}}$ 均当作译码器的控制信号, 即当 CPU 发出 $\overline{\text{MREQ}}$ 时, 仅当 16 根地址总线中的最高两位 A₁₅、A₁₄ 为 “0” 方开放译码器。此时选中的内存为 0000~3FFFH。

2、当 A₁₃、A₁₂、A₁₁ 自 000~111 变化时, 把上述 16K 的内存划分为八块, 每块占 2K 字节。如:

A₁₃、A₁₂、A₁₁ 均为 0 时, $\overline{\text{Y}}_0$ 输出 “0”, 该信号直接送至 U₇ 插座上 EPROM 2716 的片选端, 因此该片被选中; 又因加在该芯片上的地址线 A₀~A₁₀ 可在全 “0” 至全 “1” 间变化, 故 U₇ 插座存储器占用的地址区为 0000~07FFH。

依次类推, 可知其它各片选信号 $\overline{\text{Y}}_0$ ~ $\overline{\text{Y}}_7$ 与地址区的关系是和表 1.2 一一相符的。

三、ROM 的连接。

如表 1.2 所列, 该机的 ROM 插座有三个, 即 U₇ (MONITOR)、U₈ (PROM 1) 和 U₉ (PROM 2)。在这些插座中, 可插入 2K×8 的 EPROM 2716 或其同类芯片。其中, U₇ 和 U₈ 的片选端分别与 $\overline{\text{Y}}_0$ 、 $\overline{\text{Y}}_1$ 相连, V_{pp} 接至 +5V, PD/PGM 接地, D₀~D₇ 与 A₀~A₇ 分别接至数据总线与地址总线 (参见附录)。故 U₇ 或 U₈ 被选中时, ROM 均处于读出状态。U₉ 插座则不仅可用于存放用户程序, 还可用于对 EPROM 进行写入操作, 与之有关的硬件在后面再行介绍。

四、RAM 的连接

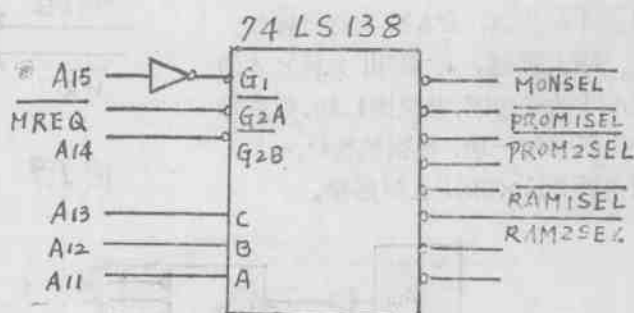


图 1.7 内存片选信号译码

本机由于采用了静态RAM2114, 因此不需要刷新逻辑, 但在连接方式上, 与ROM有二个区别:

1. SRAM2114容量为 $1K \times 4$, 因此要2块2114, 一块取高四位, 一块取低四位才能构成 $1K \times 8$ 的八位内存; 又因其地址线只有 $A_0 \sim A_9$, 所以 A_{10} 用于区别2K一组中的低1K字节与高1K字节, 参见图1.8。

当 $\overline{Y_4}$ (或 $\overline{Y_5}$) 出现后, 如 A_{10} 为“0”则 $\overline{CS_1}$ 有效, A_{10} 为“1”则 $\overline{CS_2}$ 有效。

2. RAM是可读可写的器件, 因此还需增加读写控制逻辑, 参见图1.9。

由图中可见, 当CPU发出MREQ信号后, 如 $\overline{WR}$ 有效 (为“0”) 则 $\overline{MEMW}$ 为真, RAM执行写操作。

综上所述, 可画出 $2K \times 4$ 的RAM的连接图, 参见图1.10。如在每块上再并接一块, 数据线接 $D_4 \sim D_7$, 则得到 $2K \times 8$ 的RAM连接。

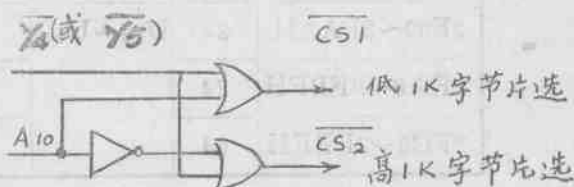


图 1.8 RAM 的片选逻辑



图 1.9 RAM 读写控制逻辑

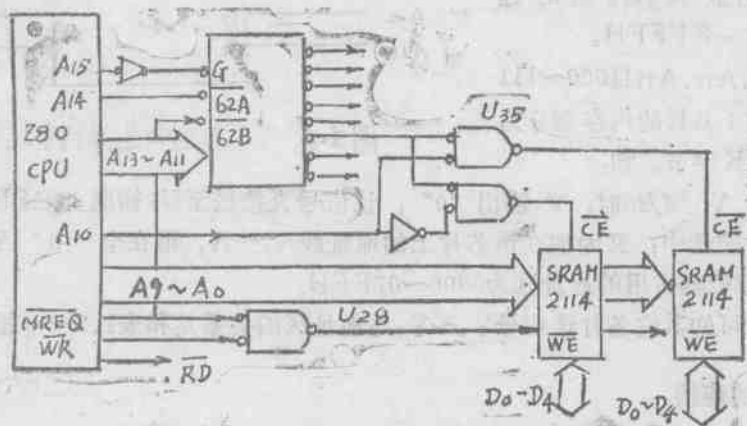


图 1.10 $2K \times 4$ RAM 的连接

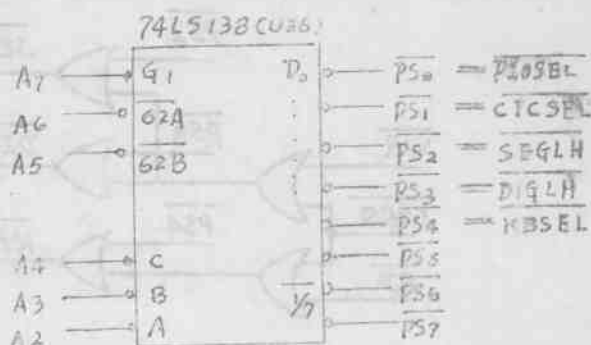
1.6 I/O 接口的硬件连接

本机的 I/O 接口包括 Z80CTC (U10), Z80PIO (U5), 提供七段数字显示的段码锁存器 74LS273 (U12), 供数字显示扫描及键盘扫描用的选通锁存器 74LS273 (U11) 和自键盘读入信息的三态门 74LS244 (U13)。

一、I/O地址的译码

Z80的I/O地址是由地址总线的低八位提供的,采用全译码方式时,可对256个I/O口进行操作。本机的I/O口少于256个,因此只要采用部分译码就可以了,又因为PIO和CTC每块芯片占用四个口地址,所以I/O地址译码器采用了图1.11所示形式:

从图1.11中,可以看出I/O地址译码器74LS138(U36)输入端用A₇、A₆、A₅作控制,即指定其值为100,再对A₄、A₃、A₂进行译码,送出八个选通信号PS₀~PS₇,A₁、A₀直接送入,这样,I/O口的空间分配如表1.8。



二、I/O控制线的连接

在上述I/O口中,PIO和CTC分别有4至5根控制线与CPU相连,而U₁₁~U₁₃仅有一根选通线(即CLK端),因此

表1.8 I/O空间分配

I/O 地址	译码器输出	器 件	口 内 容	备注
80H	PS ₀ = $\overline{\text{PIOSEL}}$	Z80— PIO (U ₆)	口A数据寄存器DRA	
81H			口B数据寄存器DRB	
82H			口A控制寄存器CRA	
83H			口B控制寄存器CRB	
84H	PS ₁ = CTCSEL	Z80— CTC (U ₁₀)	CTC ₀	
85H			CTC ₁	
86H			CTC ₂	
87H			CTC ₃	
88—8BH	PS ₂	74LS273 (U ₁₂)	段码锁存器	只写
8C—8FH	PS ₃	74LS274 (U ₁₁)	数位选择锁存器	只写
90—93H	PS ₄	74LS244 (U ₁₃)	键值读入	只读
94—97H	PS ₆	未 用		供用户扩展使用
98—9BH	PS ₆			
9C—9FH	PS ₇			

需按表 1.8 备注栏的要求, 对信号再行组合, 以区分 I/O 口或存储器, 读或写。组合方法参见图 1.12。

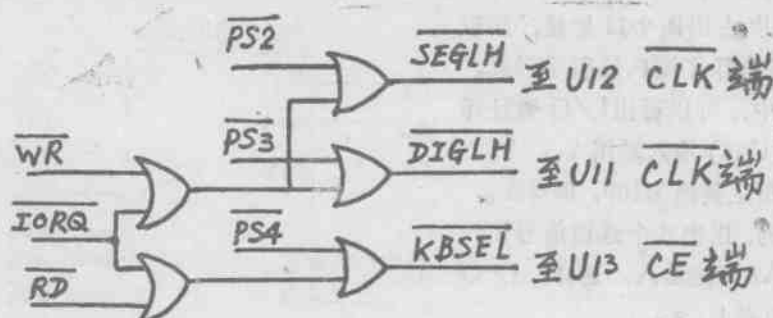


图 1.12 I/O 控制信号译码

图中, $\overline{IORQ}$ 、 $\overline{WR}$ 和 $\overline{RD}$ 信号经两个或门组合后, 取得 $\overline{IOW}$ (I/O 口写) 与 $\overline{IOR}$ (I/O 口读) 两个信号, 再分别与 $\overline{PS_2}$ 、 $\overline{PS_3}$ 、 $\overline{PS_4}$ 组合, 获得各口的选通信号。

上述 I/O 口的全部连接图, 读者可参见附录三。在以下数节我们还要从硬件与软件结合的角度, 详述段码、选通与读键盘三个口的电路。

1.7 显示部分的硬件连接

一、LED 数码管的显示原理

在本机上使用的 LED 数码管是共阴极的七段数码管, 它的每一段都相当于一个发光二极管, 八个发光二极管的阴极连在一起, 如图 1.13 所示

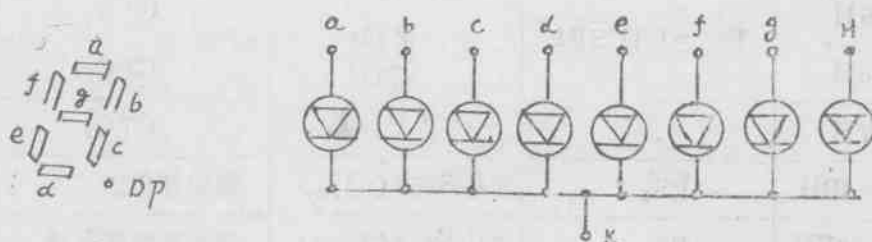


图 1.13 LED 数码管构造示意

由图 1.13 中可见, 当我们把 K 点接 0 电位 (即数字 “0”) 时, $a \sim g$ 中的那些送入高电位 (即数字 “1”) 的段均会点亮; 而当 K 点为高电位 (即数字 “1”) 时, 各段均不会点亮, 因此 K 端又叫选通端。

适当地选择 $a \sim g$ 中的某些段亮, 数码管可显示不同的数字、某些字母或符号。在本

机中, a~g七段通过驱动器和锁存器接口与数据总线的 D₀~D₆ 反相相连, 对应方式为,

D ₆				D ₀			
	g	f	e	d	c	b	a

且有 D₀~D₆ 中如“0”所对应的段点亮, D₇ 与显示无关, 可为任意值, 程序中把它当作“0”。由此可得到显示内容与数据总线输出(称作段码)的一一对应关系, 见表 1.9。

表 1.9 显示段码表

显示内容	g	f	e	d	c	b	a	Hex 数
0	1	0	0	0	0	0	0	40
1	1	1	1	1	0	0	1	79
2	0	0	1	0	1	0	0	24
3	0	1	1	0	0	0	0	30
4	0	0	1	1	0	0	1	19
5	0	0	1	0	0	1	0	12
6	0	0	0	0	0	1	0	02
7	1	1	1	1	0	0	0	78
8	0	0	0	0	0	0	0	00
9	0	0	1	1	0	0	0	18
A	0	0	0	1	0	0	0	08
B	0	0	0	0	0	1	1	03
C	1	0	0	0	1	1	0	46
D	0	1	0	0	0	0	1	21
E	0	0	0	0	1	1	0	06
F	0	0	0	1	1	1	0	0E
不显示	1	1	1	1	1	1	1	7F
提示符“—”	0	1	1	1	1	1	1	3F
提示符“P”	0	0	0	1	1	0	0	0C
提示符“,”	1	1	1	1	1	0	1	7D
字母 H	0	0	0	1	0	0	1	09
I	1	1	1	1	0	0	1	79
J	1	1	1	1	0	0	1	61
L	1	0	0	0	1	1	1	47
U	1	0	0	0	0	0	1	41
h	0	0	0	1	0	1	1	0B
n	0	1	0	1	0	1	1	2B
r	0	1	0	1	1	1	1	2F
u	1	1	0	0	0	1	1	63
y	0	0	1	1	0	0	1	19
问号“?”	0	1	0	1	1	0	0	20

二、显示部分的硬件连接与工作过程

本机显示部份的硬件包括:

(1) 段码锁存器74LS273 (U12) : $\overline{SEGLH}$ 选通, 地址为88H。

(2) 反相驱动电路: 由七组晶体管 电路组成。

(3) 数字屏 $S_1 \sim S_6$: 为6个共阴极LED数码管。

(4) 数位选择锁存器74LS273 (U11) 由: $\overline{DIGLH}$ 选通, 地址为8CH。

(5) 数位输出的反相驱动器 MC7545 (U_1 、 U_2 、 U_3)。上述硬件的连接参见图1.14。

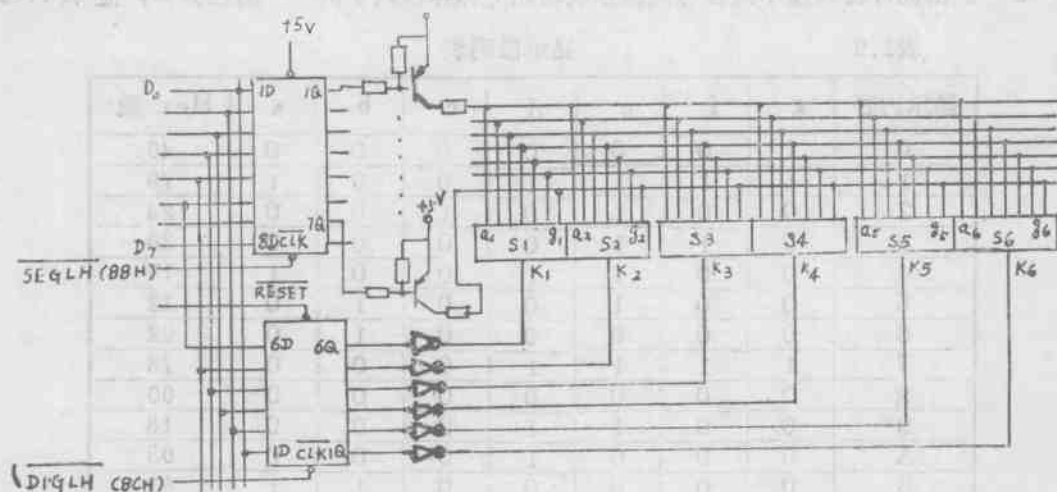


图 1.14 显示部分电路原理图

此外, 在本机的 RAM 区中, 还开辟了一个显示缓冲区, 它由 六个存储单元构成。在这六个单元内存放着准备显示的数字或符号 (不是段码), 显示缓冲区的单元地址及其与数位的对应关系见表1.10。

表1.10 显示缓冲区对应表

名 称	地 址	对 应 的 数 位
DISMEM	2FF7H	S_1
DISMEM ₁	2FF8H	S_2
DISMEM ₂	2FF9H	S_3
DISMEM ₃	2FFAH	S_4
DISMEM ₄	2FFBH	S_5
DISMEM ₅	2FFCH	S_6

如前所述, 当数码管的K端为“1”时, 无论在其输入端 $a \sim g$ 任一端上有无输入, 均

子符与奴子。

子符与奴子。



图 1.15 键盘的硬件连接

由图中可见,本机的28个键连接成一个5列6行的矩阵网络,网络的行与数字屏的扫描控制端共用数位选择接口8CH,网络的列通过三态门74LS244接至数据总线,该三态门由KBSEL选通,地址为90H。为叙述方便,以下把行自下而上称作第一行至第六行、把列自右至左称作第一列至第五列,图的右下角有一个单稳态电路,当B端(即第一列)出现负跳变时,自Q端输出一个宽度为一定的脉冲。

二、键盘信号的获得与识别

1. 按键与否的判别

在键盘扫描开始时,自8CH口送入全“1”从而使所有行为“0”(低电平),此时,若无键按下,则所有列因均通过10K电阻接连在+5V上而全为“1”(高电平),自90H口读入的数据D₀~D₄也都为“1”;若有键按下,则该键所在的列将变为“0”(低电平)自90H口读入的数据D₀~D₄中必有一个为“0”。

按上述分析,按键与否的判别方法为:使所有行为0,自90H读入数据,若D₀~D₄中有一位为0,则判为有键按下,否则无键按下。

2. 键值的获得

在得知有键按下后,紧接着的工作是判别何键被按。因矩阵网络上的每一个键都可以用它所在的行与列来定位,所以我们可以使第一行至第六行逐行为0,再通过90H口来查找,当查到有一列为0后,该键即可由对应的行和列来确定。例如当8CH口为04H时,第三行为0,此时,读得90H口为1BH(00011011B)则可知第三列有键按下,该键位于第三行和第三列为回键。

在监控程序DECKY中,把从8CH得到为0的行值存于寄存器B中而把自90H口读入的列值存入累加器A中,并令D₀~D₇为0,由此可得到键盘所应对的行列值,见图1.16。

变换值在C 行值在B

A 0	20	PROM PROG	CASS LOAD	CASS DUMP	BREAK POINT	
B 0	10	MEM EXAM	PORT EXAM	REG EXAM	REG' EXAM	
B 0	08	7 H	8 L	9	A	NEXT
D 0	04	4 I X	5 I Y	6 I	B	MON
E 0	02	1 P C	2 S P	3 IFF	C	SINGLE STEP
F 0	01	0	F	E	D	EXEC
		0F	17	18	10	1E

列值在A

图1.16 键值变换图

从图1.16中所给出的A与B的值还不能直接给出单字节的按键代码。如将A与B直

得到 0FH

接相加, 则对键 $\begin{matrix} \text{MFM} \\ \text{E} \times \text{AM} \end{matrix}$ 得到 $0\text{FH} + 10\text{H} = 1\text{FH}$, 对键 $\begin{matrix} 8 \\ \text{L} \end{matrix}$ 得到 $17\text{H} + 08\text{H} = 1\text{FH}$, 这样,

代码与按键不能唯一对应, 因此还需进行变换。变换的方法是先令寄存器 C 为 0, 然后把 B 右移一次, C 减 1, 如此重复至 B 为 0, 再把 C 补 0 左移四次, 就把 B 中的行值变换成 C 中的值, 见图 1.16 中最左一行所示。将 A 与 C 相加, 可得到按键所对应的代码, 见图 1.17。

AF	B7	BB	BD
BF	C7	CB	CD
CF	D7	B	DD
EF	E7	EB	ED
FF	F7	FB	FD
FF	07	0B	0D

DE
EE
FE
0E

图 1.17 经变换后的键代码

在得了键码后, 程序用查表法把上述键代码再行变换成键值, 见图 1.18。

1B	1A	19	18
17	16	15	14

7	8	9	A
4	5	6	B
1	2	3	C
0	F	E	D

13
12
11
10

图 1.18 经查表后得到的键值

由图 1.18 中可以看出, 在 0FH 以内的键值均为与之相同的数字键; 大于或等于 10H 的键值则为功能键。在键盘分析程序中, 功能键还需要查一次表, 从而跳转至相应的功能键执行程序。

1.9 EPROM 写入部分的电路

Z80 机作为一个简易的开发系统, 用相应的硬件与软件来实现 EPROM 的写入功能, 为用户提供了固化应用程序的手段。

一、EPROM 写入的硬件连接

本机要求把待写入的 EPROM 2716 插在 PROM₂ 插座上, 如上节所述, 它占用的地址区为 $1000 \sim 17\text{FFH}$, 其各部分的连接参见图 1.19。

參見圖