微型計算机

Microcomputer

微型计算机开发系统

（上 册）

1984—3

上海交通大学

（总 38 期）

前　言

微型计算机开发系统除了具有一般微型计算机系统的性能和配置外，还具有开发研究微型机的各种硬件和软件，诸如 EPROM 编程器，ICE 在线仿真器等，成为研究和开发微型机的一种有效的工具，目前在我国微电脑技术，日益普及与广泛应用的情况下，广大从事微型计算机开发、研究和应用的人员，均迫切希望了解微型计算机开发系统的组成结构和操作原理。

上海交大微机研究室于 1980 年自行设计和研制成功的 DJS-053 微型计算机开发系统，它以 8085A 为微处理机，采用操作简便，性能稳定的编程器和在线仿真器，其操作系统软件与 Intel 公司的 MDS-230 微型机开发系统完全兼容。本资料就是以 DJS-053 微型机开发系统为实例，较系统地阐述微型机开发系统的组成结构和操作原理，故本书是目前国内较为完整一本叙述微型机开发系统的资料。可供高等院校开设“微型计算机开发系统”课程的试用教材，又可用作广大科研工作者和工程技术人员学习，了解微型机开发系统的参考书。

参加本书编写的有杜毅仁、赵正校、朱煜清、韩朔瞭、徐子亮、李世祥、王洪澄等同志。

本书不免有错误之处，敬请广大读者批评指正。

微型计算机开发系统(上册)

目　　录

第一章 绪 论

微处理机的出现，开创了微电子技术和计算机科学技术的新纪元，大规模集成电路(LSI)和超大规模集成电路(VLSI)技术，目前已经可以把数以万计的MOS晶体管做在一枚分币那样大小的硅片上，因而计算机的主要功能部件，均可装入其中。微处理机就是这种硅片上计算机的统称。它是LSI和VLSI与计算机技术的高度结合的产物，由于微处理机内含的功能很强，而引脚数目却是有限的，8位处理机的引脚一般只有40条，高档16位微处理机的引脚也只有64条左右，就是32位超级微处理机的引脚也不会超过100条，这就意味着，要把微处理机内含的功能发生作用，必须从片外编制程序，时序地送入微处理机片内，才能达到目的，因此，微处理机虽然是属于硬件范围，但要它工作又必须编制程序，程序则属于软件范围，这就不难看出，微处理机的出现和广泛应用，它首先订破了过去硬件和软件间的界限，其次是打破了元件与系统的界限，这就要求从事微型机的科技工作者，必须软硬结合，并且要有系统方面的知识，否则就很难适应工作了。微型机做好以后，对各行各业，均是一种解放脑力劳动的有力工具，但它首先应该成为改造计算机科学技术领域内长期存在着的手工作业的老传统。因而各种微型机开发系统(MDS)也就应运而生，对于从事微型机系统开发和应用研究的科技部门，(如果没有MDS的支持，势必沿用以前的手工作业方式，每件事情都得人去直接参与，其工作效率很低)MDS是一个有力的辅助设计工具，如果要求设计一台新的微型机，只要选用的微处理机芯片是该MDS能开发的，就可以在新机未完成之前，利用MDS开发基本软件和应用软件，一旦新机造好，就可以把MDS上开发的软件装入，从而加快了研制进度。实际上光具有上述软件开发的系统还只是初级的开发系统，只有加上在线仿真器的系统才是真正的高级开发系统，如DJS-053就是属于这种。如果新机器的CPU板在设计制造中有问题，加电以后发生故障，MDS还可以利用在线仿真器，对它进行追踪测试，把故障位置找出来。说得简单一点，由于MDS内的微处理器，一般与新设计的微处理机板上的CPU一致，因而可以把其上的微处理器拔去，利用MDS中的微处理器来代替，在ICE仿真软件的支持下，进行检查，发现故障后，指出故障性质或部位，这样就可以一步步地调试下去，直到全部故障消除为止。

可以这样说，MDS使我们摆脱老三件(万用表、示波器和专用指示器)的手工研制局面，我们可以多快好省地研制和制造出以微处理器为基础的各种微型机系统，特别是对专用系统来说，更是如此。随着微型机在各行各业的推广应用，这种强有力的开发工具，必将获得各研究开发单位的重视。就目前的情况看来，开发系统可分成三大类：

一、通用型开发系统，这种开发系统可以开发多种微处理器为CPU的微型机，当然也配有多种在线仿真器(ICE)，例如：

1. HP公司的HP6400和Tektronix公司的8550开发系统，可开发8080/8085 Z-80和MC6800为CPU的微型机。

2. GR2300开发系统可开发8080/8085，Z-80，MC6800，MC6802和Intel 8086等

为 CPU 的微型机。

二、专用型开发系统，这种开发系列只能开发一家公司的系列产品，专用性强，但比通用型花钱少，使用方便，硬件和软件都比较统一，各微处理机的生产部门都推出自己的开发系统。因而是一种广泛被用户接受的开发系统。例如：

1. Intel 公司的 MDS 210,MDS 220,MDS230,MDS 240, MDS 287,MDS-Ⅰ, MDS-Ⅱ，等都是专供开发 Intel 公司的微处理机的开发系统，也是生产最多、起步最早，影响最大的开发系统。

2. MotoRola的EXORCISER Ⅰ,Ia, Ⅱ, EXORterm 200, 220, M68ADSZA, 68000 MDS 等则可以开发 MC6800～MC68000 为 CPU 的开发系统。

三，交叉式开发系统

这种开发系统是另一种计算机为主机，通过交叉软件来研究开发另一种微处理器为 CPU 的系统，例如，NPDS23 就是用，PDP—11/23计算机来定义开发8085的开发系统。这种系统是颇有前途的，因为可利用现存的小型机，例如 PDP-11 或NOVA型计算机，配上交叉汇编和在线仿真程序以后就成了开发系统。目前 Intel 432 系列的高档微处理机，就是利用 VAX-11 系列作为交叉开发系统的，由于可以享用小型机和超小型机的资源，而且一机多用故交叉式开发系统受到人们，特别那些拥有小型机的单位的兴趣。

微机开发系统正在受到各研究开发单位重视，这是值得可喜的地方，但是应该看到，开发系统不久前还是封锁产品，而且从国外引进的价格来看也是偏高的，几乎比一台同性能的通用微型机系统高出一倍，为了打破国外的封锁和高价，上海交通大学微型机研究室在1979年集中一批力量，对 Intel 公司当时的名牌产品 MDS-230 进行透彻的分析和研究，在融会贯通的基础上，在保证与 MDS-230 完全兼容的前提下，通过自行设计，自己研制成功了 DJS-053 微型机开发系统，该系统于 1980 年通过了鉴定，获得代表们的一致好评，目前 DJS-053 已经小批量生产，产品已在许多研究所和学校、工厂企业中顺利地使用。应该说，我们这次国产化还是比较彻底的，无论在硬件和软件方面，DJS-053相对MDS-230来说均有了比较大的改进。例如，我们只用一片8085，就代替了MDS-230 中的两片 8080 和一片 8041；而且省去一块 I/O 控制板；把存储器板合并成一块独立的印制板；用软件处理为主的 EPROM 写入器代替专用的 UPP 写入装置；在线仿真软件的改动就更大了。所有这些不同点，充分说明了我们完全有能力，也完全应该用性能更好的硬件去代替那些老的东西，因为我们是后来者。显然 DJS-053 是属于专用开发系统这种类型的，从实际工作中我们体会到专用开发系统针对性强，而且直接了当，确实能够帮助用户在开发工作中事半功倍，事实上实际使用的开发系统中，专用系统占了最大的比重，这一点不是没有道理的。

这本书就是我们在研制、设计、调试和使用 DJS-053 工作的基础上编写的，可以说是我们工作的一个总结。由于是从自己体会出发编写的，可能比翻译容易阅读和理解，这也是我们的希望。本书编写时均先从一般原理出发，但最后都落实到我们研制的 DJS-053 微机开发系统。限于编者水平，书中定有错误和不足之处，请读者批评指正。

第二章　微型计算机系统结构

本章以 DJS-053 微型机开发系统为例，讨论微型计算机的系统结构。

§2-1　概述

§2-1-1 微型计算机的一般结构

微型计算机是以微处理器为中心组织起来的。微处理器的种类繁多，不胜枚举，但按字长来分有 4 位、8 位，16 位和 32 位四大类。字长不同，其性能也相差甚远、大体说来，可以这样认为：4 位和早期的少数 8 位机属第一代微机产品，8 位属第二代，16 位属第三代，32 位则可列为第四代。

所有微计算机都有四个基本功能块组成，即输入设备，输出设备、存贮器和中央处理部件 CPU 或微处理器。输入设备用以送入数据、指令；输出设备用以输出结果；存贮器用以存贮指令和数据；最后 CPU 作算逻运算和控制。这几乎与其它计算机一样。

常用的输入设备有键盘，盒式磁带机，磁盘机，A/D 变换器、纸带输入器等。它们均需要经过一定的输入接口将输入信息变换成二进制代码。

输出设备主要有 CRT，打印机、磁带机、磁盘机、D/A 变换器、穿孔机、XY 记录仪等。同样需要用输出接口将二进制数变换成设备可接受的信息。

有的输入输出接口已有 LSI 可用，有的则还需要系统专门为之设计。

在微型计算机系统中，存贮器的器件多数采用半导体存贮器，也有磁泡存贮器。按照存取的内容能不能改变有只读存贮器 ROM 和随机存贮器 RAM 之分，前者用作存贮程序(由输入设备得到)。

在开电源或者系统总清时，微处理器都使控制转到从某个固定单元开始执行程序，即使处理器的程序计数器 PC 置成某个固定值，(例如 8085A，Z80 在总清后使 PC 位置零。8086 中这个初值为 FFFF0)或者从某个固定单元取得 PC 的初值后执行程序(例如 Z8000 从 2 号字单元中得到这初值)。因此开启电源或系统总清后，CPU 就开始执行程序。这样，为了使微型计算机在加电后就有可执行的程序，这个程序一般作系统初态的预置，引导系统软件等，微型计算机中 ROM 是少不了的。因此，微型计算机特点之一必定有 ROM 或 EPROM。在许多专用微计算机系统中，软件均存贮在 ROM 或 EPROM 中。

微处理机由一块到几块 LSI 电路组成，字长为 4～32 位，它是微计算机的核心，所有指令均在这里执行。

各功能块之间，一般均通过总线相连接。微机系统采用总线连接的目的是为了系列化和减少 LSI 的引出线端子，例如 8085A，8086Z8000 等，都将 I/O、MEMORY 的数据输入/输出引线与地址输出引脚合用，分时传送。各功能块之间往往采用一公共总线相连，功能块内部一般都有各自的分总线。总线结构是微机系统的又一特点。

综上所述，微机系统的结构是通过总线将 ROM 或 EPROM、RAM、CPU 以及 I/O 接口连接起来。图 2-1 为微计算机的一般结构框图。8085A 的最小系统配置就是这种结构的

典型例子(见《微型计算机》1983-1，2 两期，上海交通大学)。一般的单板计算机结构，例如 MIC-85，MIC-80，SBC80/20 等都可归结为这种形式。

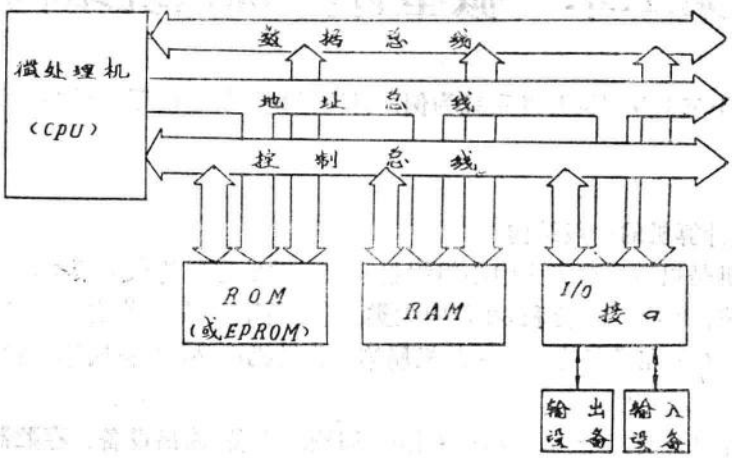

图 2-1 微型计算机的一般结构框图

§2-1-2 DJS-053 系统的结构

图 2-2 为 DJS-053 系统的主微型计算机的结构。如果将图中点划线内部分看作一个微处理机的话，则和图 2-1 相同。

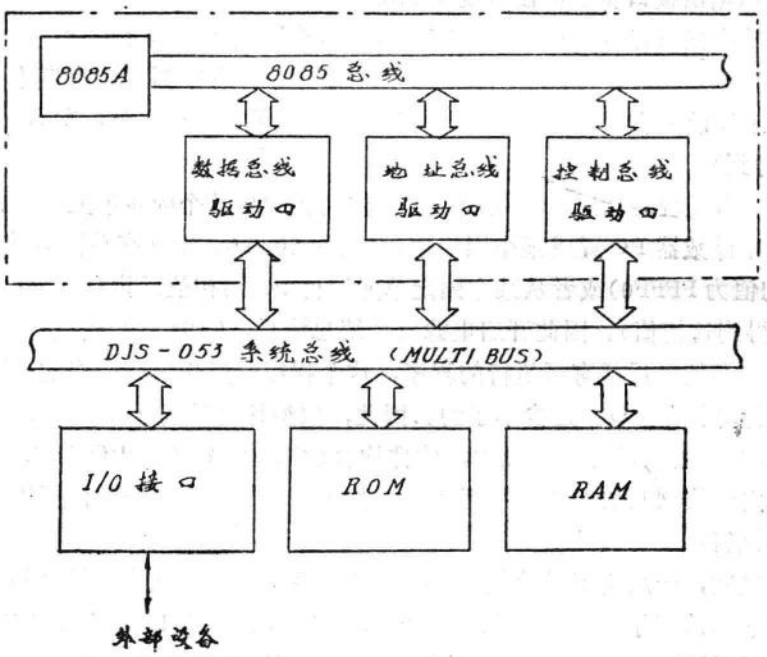

图 2-2 DJS-053 系统中的主微机结构

很多情况下，在微处理器与其它功能解块之间要增加某些东西，正如图中 2-2 中所示那样。其作用主要有：

• 提供一些外部控制，便于启动、暂停、调试

• 增加驱动能力。一般 MOS 门只能带一个 TTL 门。

• 扩大中断处理能力。大多数微处理器只能直接响应一个到 n 个中断，往往不够满足系统要求，而需附加硬件扩大中断处理能力。有的微处理机还要求在响应中断后由外部接口提供一条指令或向量地址(详见中断系统一节)。

• 扩大对系统总线和系统资源的控制、管理能力。

从而使处理机的能力得到进一步扩充。

DJS-053 系统的软磁盘分系统和仿真器中都有各自的微计算机作为控制中心，因而 DJS-053 系统实际上是一个多机系统。图 2-3 为它的框图。

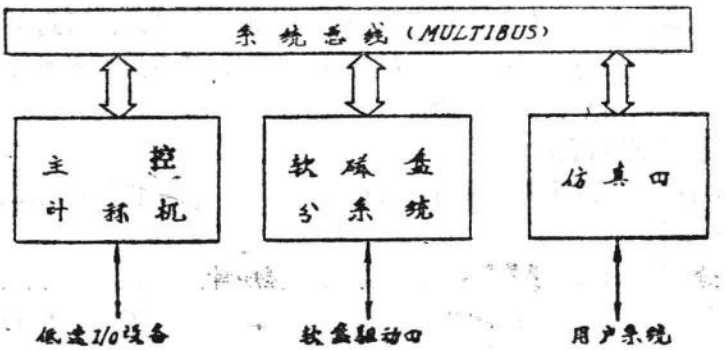

图 2-3 DJS-053 系统框图

三个微计算机通过系统总线(MULTIBUS)相互连接起来。若用图 2-2 代入图 2-3 的主微计算机框，则将发现在 DJS-053 系统中，RAM、ROM 和 I/O 接口都直接挂在系统总线上，因而这些资源可供这三台微机共享；同时软盘分系统则由主机和仿真器共享。

§2-1-3 微处理机的选择。

常有人提出这样一个问题，选用哪种微处理机比较好？回答常常不能令人满意。因为目前微处理机的种类已经很多，而且特性各不相同，故要对各种微处理机的性能作出比较，评价已经是困难的事情。要作全面的评价更难，并且选择微处理机不仅要根据其性能价格比指标，更重要的是要重视选择一种最适合于具体应用的处理机。微处理机的应用目前已深入各领域，所以这里只列出有关微处理机性能的几方面。

• 微处理机的硬件结构。包括字长、寄存器(累加器、通用寄存器、变址寄存器)的数目、堆栈类型、中断处理能力等。

字长直接与计算精度有关；较多的寄存器可以减少访问存贮器的次数，从而提高处理效率；硬件堆栈的速度较高，软件堆栈的容量较大；处理向量中断和查询中断的能力以及能处理多少个中断。

• 速度。它有许多评价方法，例如时钟周期、最短指令执行时间、寄存器之间加法时间、中断响应时间。由于几乎每种微处理机都有加法指令，故寄存器间的加法常常作为一种比较依据。

• 指令系统。它的编程灵活性如何，寻址方式；算逻运算能力(例如是否有乘除指令、

浮点指令等)；输入输出控制指令，能管理多少外围设备；指令系统是否可由用户更改。

- 硬件后援是否完整。例如时钟发生器，数据及地址缓冲器、中断控制器、I/O接口等。
- 软件后援指操作系统、高级语言、汇编语言、实用程序等。

当然，在选用微处理机，尚应考虑它的价格、货源及其维修等方面的问题。

§2-2 微处理机的外部控制

微处理机的外部控制就是通过改变某个或某几个引出端的电平来影响它的工作方式。控制用输入端主要有再启动、暂停、准备好，中断等。

再启动是所有微处理器必备的，也是唯一的一个能使微处理机回到原始状态的外部控制端，“暂停”“保持”都能暂停程序计数器的计数。“准备好”系用来实现与低速器件或外部器件同步的，“中断”则用来改变程序的执行流程。

§2-2-1 微处理机的再启动

“再启动”的目的是使微处理机回到初始状态，然后以初始状态开始工作。每种微处理机都设有这个输入端。

不同的微处理机的再启动操作不同；一般均使程序计数器和某些寄存器回到初态。程序计数器的初值随机器而异，例如8085A，Z80的初值为0，MC6800的初值为FFFE，而8086为FFFF0。为了在再启动后，让微处理机从该单元获得指令(8085A，Z80，8086等)或向量地址(MC6800，Z8000等)，故至少要固化少量软件，例如系统的引导程序或监控程序，这些程序起点即为再启动时程序计数器的初值。

何时需要产生再启动输入信号呢？有两种情况，一是在接通电源时，再则是人工按动再启动键时，

再启动信号应有一定宽度(或延续时间)，例如8085A要求该讯号至少要持续500ns以上，MC6800则要求持续八个时钟周期以上。手动产生的再启动信号，其持续时间显然是不成问题的，但要有消除机振措施；接通电源时的再启动信号应有措施，保证有一定的时延。

综上所述，在接通电源和人工再启动时，要求再启动电路应发出一个具有一定宽度的再启动信号给微处理机。图2-4中为再启动电路系列，它由RC网络，一个反相器和一个施密特电路组成。在刚通电时，C_1两端的电压为0，经R_2对C_1充电，在C_1上的电平达到反相器的开门电平之前，施密特的输出$\overline{\text{Reset}}$为低电平。根据微机对时延的要求选择R_1，C_1的数值。图中电阻R_2和两极管D_1为断电时加速C_1上电荷的释放过程。

在DJS-053中，再启动电路比图2-4要简单得多。这是由于8085A的再启动输入端($\overline{\text{RESET IN}}$)内已设置了施密特电路，因而只需要外加阻容网络就可以了。$\overline{\text{RESET IN}}$信号使8085A的程序计数器、指令寄存器、中断允许触发器、RST7.5锁存器、SOD锁存器请零，使RST7.5、RST6.5、RST5.5的屏蔽触发器置“1”。在$\overline{\text{RESET IN}}$有效时，8085A的RESET OUT端为高电平，经反相后得到系统复位信号$\overline{\text{INIT}}$(MULTIBUS的信号线之一)，使系统回到初始状态。图2-5为DJS-053系统中的再启动电路。

§2-2-2 再启动地址的修改

再启动地址指的是再启动(或总清)时CPU取得信息(指令或向量地址)的第一个存贮

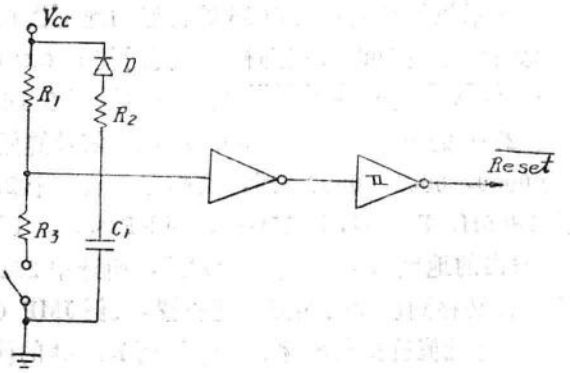

图 2-4 再启动电路

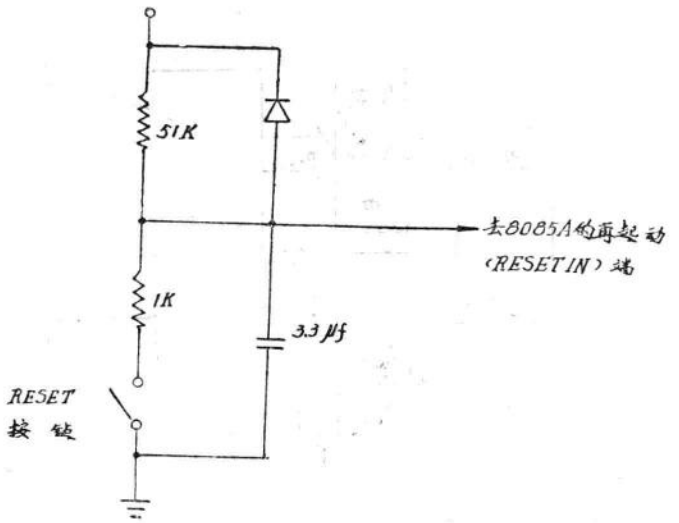

图 2-5 DJS-053 的再启动电路

器地址。对于给定的微处理机而言，再启动时它向外送出的这个地址是固定不变的，例如8085 和 Z80 为 0000H，而 8086 则为 FFFFOH 等等。该地址在某些微处理机中与某个中断程序入口地址或某条再启动指令的启动地址相同，例如 8085A 的再启动指令 RST0 的启动地址和总清时的启动地址均为 0000H。但在一些系统中又往往要求这两类再启动完成不同的任务，例如 DJS-053 中要在总清后先将系统置初态，然后等待命令(监控命令或操作系统命令)；而在程序中执行 RST0 指令或者 0 级中断时，则进入监控程序，等待监控命令。

众所周知，微处理机总清后，它送出的第一个地址是由掩模确定的，使用者无法改变这个地址。总清后，微处理机就企图从这个单元开始取指令(或向量地址)，代码由数据总线进入处理机。然而处理机却不管进入数据总线的代码实际上来自何处，因而使用者可以附加一些硬件，使得处理机在总清后所得到的头几条指令(或向量地址)实际上不是来自处理机指定的单元，而是来自设计者指定的单元就可以了。

图 2-6 为再启动地址修改逻辑示例。图中处理机用 8085A，2716 中为固化的引导程序。设其地址为 F800H。图示逻辑可实现将 8085 的再启动地址由 0000H 改为 F800H。工作过程是这样的：外部总请信号使 8085A 回到初态，同时使 Rreset 触发器置位，由于2716 (ROM)的选片讯号如 $\overline{A_{15}A_{14}A_{13}A_{12}A_{11}+Rr_{eset}}$，所以在总清后 2716 被选中而不管这时处理机的输出地址是什么。2716 的 $A_0 \sim A_{10}$ 来自 8085A 地址信号的低八位。这样在总清后，当 8085A 送出的地址为 0000，0001，0002…时，实际上却从这个 2716 的头几个单元读出，它们的地址分别为 F800H，F801H，F802H…。即 RReset 置"1"后，它对该 2716 选片端的影响等效于 8085A 送出的地址 A_{11} 到 A_{15} 均为"1"，而选中 2716。所以只要从该片取到的第一条指令为无条件转移到该片的实际物理位置，(例 JMP 0F806H)，则8085A在执行完首条指令后它的 PC 地址值就指向该片 2716，因而 Rreset 的使命也完成了。值得指出的是接着就应使 $R_R$eset 触发器复位，否则指令只能在该片 2716 取得。若硬件设计中，将 Rreset 触发器为 8085A 的某个输出接口中的一位，则只要执行一条适当的输出指令，就可以将其复位了。

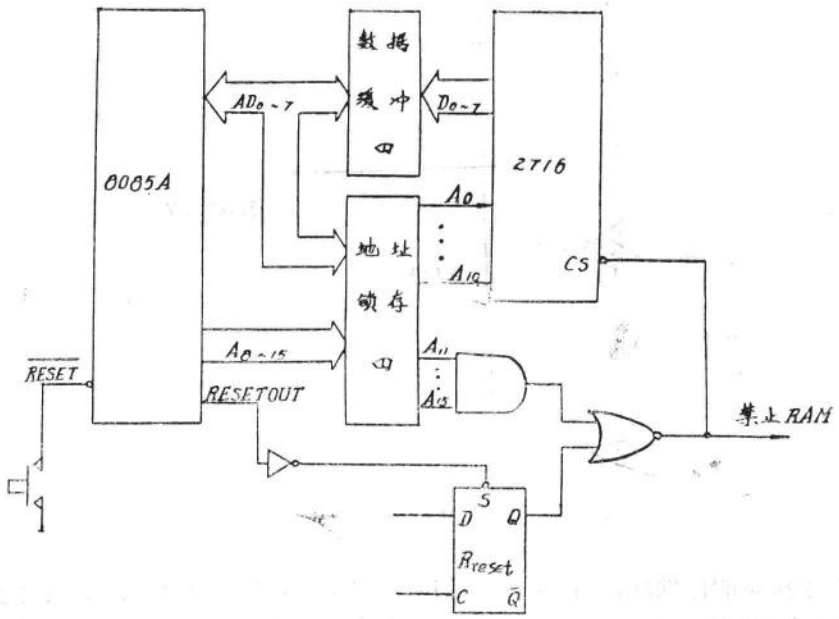

图 2-6 再启动地址修改逻辑

请注意，除了上述以外，在选中该 2716 的同时，还应该阻止从其它的内存单元读出，否则，在 RESET 后，8085A 会同时从 F800H 和 0000 单元同时进行读出，显然这是不允许的。这里可以方便地利用该 2716 的选片信号作为禁止信号来实现。

推而广之，利用上述触发器实现选片和禁止技术，还可方便地实现影子 RAM 和影子 ROM，以扩大内存地址空间。

§2-2-3 单周期和单指令操作

单步操作常常是调试硬件和软件的有效手段。单步操作时，机器操作受某按键控制，每按一次键往下走一步。对于单周期操作，每步执行一个机器周期；而单指令操作则每步执行一条指令。微处理机的某些引脚指示状态和控制内部工作，微算机设计者可利用这些引

脚实现单步操作。

下面以8085A为例，说明单步操作的实现方法。8085A可资利用的引脚有：$IO/\overline{M}$、S_1、S_0、READY。

一、单周期操作的实现

众所周知，8085A的READY(准备好)引脚是一条外部控制端，由存贮器或I/O逻辑提供信息，当被访问的存贮器或I/O已准备好收/发数据，则使其变高电平。8085A在读/写机器周期T_2状态内对它进行测试，如果测得为高电平，则进入T_3状态；否则表示未准备好就进入T_{WAIT}状态。处在T_{wait}状态时，仍继续测试READY端的电平，并在T_{wait}状态踏步，直至READY端变高后才脱离T_{wait}状态，进入T_3成态。

可提出很多方案来实现上述要求。其中用手动按键产生一个单脉冲信号接到READY端是一种简便有效的方法(图2-7示)。脉冲应得保证使8085A脱离现行T_{Wait}周期，而进入下一个机器周期的T_{wait}状态。因此对脉冲宽度应有一定要求

$$T \leqslant \text{脉冲宽度} \leqslant M_{mim}$$

式中M_{min}为最小机器周期，在8085A中，$M_{min}=3T$。

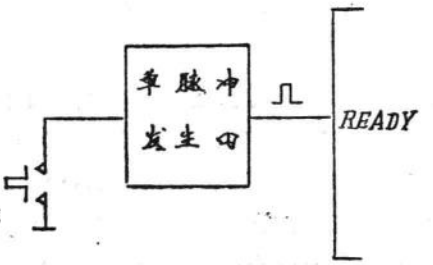

图 2-7 单周期操作的实现逻辑

单周期操作，很适合于硬件的调试。这是因为在等待状态时，处理机的控制讯号，传送的数据，各种译码选片讯号，均保持不变，故可用常规仪表，例如一只万用电表就可判断各点信号电平的正确性。调试时，可先准备一个放有调试程序的ROM(称调试ROM)，调试程序可由几条或几十条指令组成，使它能访遍机器的各个角落；然后用单周期操作对各条指令执行的正确性作鉴别即可。

在8085A系统中，指令步进的硬件实现方法是处理机每执行一条后，就保持在下条指令的取指周期的T_{wait}状态。8085A的取指周期信息可以从它的两条状态线s_0、s_1得到。众所周知S_1、S_0各组合的意义是：

S_1	S_0	
0	0	停机(HALT)
0	1	写(WRITE)
1	0	读(READ)
1	1	取指(FETCH)

请注意，上表中取指与读的区别在于取指仅仅表示读取指令的第一字节(或操作码)，而读则表示读取指令的地址部分或立即数部分，或者操作数。如果设法在每条指令的FETCH取操作码周期时，使它处于WAIT状态踏步，然后用手控单脉冲脱离，就可实现指令的步进运行。

在图 2-8 示出了一个既考虑了单周期和单指操作的逻辑。它由 R-C 电路、按键和一些门组成。M_1 和 M_2 系开关 K 的消振电路；M_3、M_4 和 RC 组成一微分型单稳，其脉宽大于一个时钟周期而小于三个时钟周期。当单周期/单指令切换开关 K_2 位于单周期位置时，M_5 作反相器用。平常由于 M_5 输出低电平，使 8085A 停留在 WAIT 状态；当按一次 K_1 时，M_1 送出一个正脉冲，使 8085A 脱离现行机器周期进入下一机器周期的 T_{wait} 状态。当 K_2 位于单指令位置时，M_5、M_6 组成 R-S 触发器。FETCH 周期的 ALE 脉冲使 M_5 变低电平，使 8085A 停留FETCH 周期的 T_{wait} 状态。这时按 K_1，从 M_4 输出的负脉冲将使 M_5、M_6 组成的 R-S 触发器翻转，M_5 变高电平，使 8085A 脱离 WAIT 状态，直到下条指令的 FETCH 指令时，又再次进入 WAIT 状态。

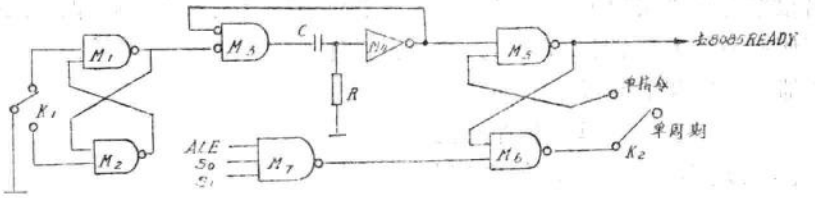

图 2-8　实现单步操作逻辑示例

§2-2-4 读写回答和定时回答

微处理机为了能与任意速度的存贮器和 I/O 设备同步工作，一般都用外部控制它的 1～2 条引脚来达到同步。例如 8085A 的 READY 引脚、Z80 的 WAIT 引脚等。

在 8085A 为 CPU 的系统中，如果各种存贮器和 I/O 接口的速度均与它相匹配，则将 READY 线直接到 +5V，正如 MIC-85 单板微型计算机那样。

如果存贮器或 I/O 器件的存取速度有的比 CPU 慢，就要用 READY 端来同步。即这时要求系统中的存贮器和 I/O 器件在完成读写操作时向 CPU 发送一个回答信号来实现同步。图 2-9 为某机器中回答讯号实例，其中 CS_1/，CS_2/，CS_3/为存贮器或 I/O 器件的选片讯号，在没有器件被选中时，门 1 输出低电平，XACK 为高电平。使有预置的四位二进制计数器 74LS161 送入初值为 0110，Q_D 为低电平，门 2 输出为高电平。当 CS_1/～CS_3/之一变低电平时，门 1 输出高电平，故一开始 0C 门的两输入端为全“1”，使 XACK 呈低电平(未准备好)。此后 74LS161 开始计数，当计入两个脉冲后 Q_D 变电平，使XACK 回到高电平。从分析知，图 2-9 电路，在选中某一器件时在XACK 端全输出一个负脉冲，其宽度可以用 A，B，C，D 端的接法得改变。8085A 用 ALE 的后沿锁存地址的低八位 A_{7-0}，实际上当地址寄存器用锁存前在 ALE 内在 A_{7-0} 上在已有正确地址存在，选片信号在 T_1 的后沿附近已经形成，故在 T_2 时XACK 为低电平。如果将 XACK 端直接与 8085A 的 READY 相联，则在 T_2 测得 READY 为低电平，而进入 T_{wait} 状态。在第二个 T_{wait} 状态时，XCAK 回到高电平，CPU 就结束 T_{wait} 状态，进入正常的机器状态。用图 2-9 回答讯号，产生电路，将插入二个 T_{wait} 状态。图 2-9 所示回答电路，使用灵活，且与 CPU 的 CLK 能同步。

在 DJS-053 系统中，回答电路用一些门和 RC 网络组成(见图 2-10)。该路的特点是简单，调节方便，缺点是插入的 T_{wait} 个数要受电源电压变动的影响。在没有访问存贮器和 I/O 时 READY 端为低电平，如果 ROM、I/O 接口或中断控制器 8259 三者之一选中，

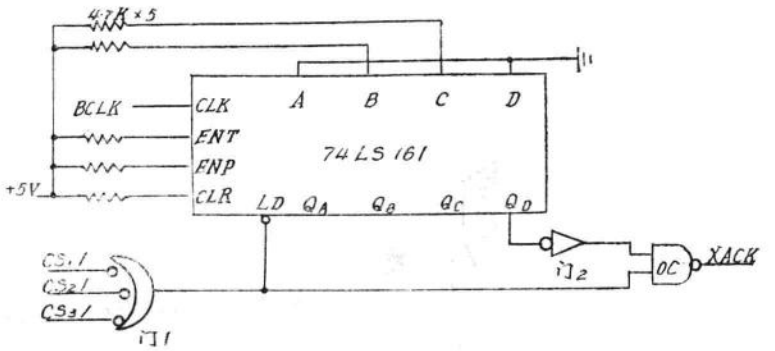

图 2-9 回答信产生器示例

M_1 为高电平，而且在访问存贮器或 I/O 时系统总线中四条读写控制线之一有效，M_2 变低电平，M_3 变高电平。由于 M_3 经 RC 网络接到下级门，故 M_4 要经一段延迟后使 READY 端才变高电平，这样就扦入了若干 T_{wait} 周期。改变 R 的值可修改扦入的 T_{wait} 的个数。DJS-053机，在正常电压时调至1～2个 T_{wait}。二极管的作用是加快 C 的放电速度，使该 RC 网络的充电速度慢，放电速度快。8085A 的中断响应周期(INA)，除了送出 INTA 而不是 RD 之外，是与取指周期(OF)是相同。图 2-10 中的 INTA/输入端是为了产生中断响应周期的回答讯号而加入的。XACK/是系统总线中一条回答讯号输入线，接收 CPU 板外(例如RAM 板)的回答讯号。

微型计算机系统中，设置了上述读写回答逻辑后，就可以与各种速度的存贮器或 I/O 相匹配。然而，系统资源是不会配足的，8085A 可配 64K 的存贮器和 256 个 I/O 接口，许多系统都没有配得这么全。这样，就会出现一个新问题，程序如果由于某种原因（例如操作错误，或功能测试），而访问尚未接入系统的存贮空间或 I/O 空间时，因没有读写回答讯号反馈到 CPU，造成 CPU 在 T_{wait} 状态踏步不前。这显然是不允许的。定时回答讯号产生器是为解决这个问题而引入的。它的作用是当 CPU 访问某存贮单元或 I/O 接口的读写命令发出后，经过一定时间(例如 100uS)后仍未收到回答讯号，则由它自动产生一个回答讯号给 CPU。图 2-11 示 DJS-053 的定时回答讯号产生电路。它由一个与非门，微分型单稳和D型触发器组成。无读写讯号时 M_1 为低电平，D 触发器被清零，单稳没有工作，M_3 为高电平。当有读写讯号时，M_1 变高即 D 触发器的 D 端为 1，M_3 变低，D 触发器仍为 0 状态，单稳电路进入暂态过程。正常情况下，读写脉冲只有几个 T 状态(或几个时钟周期)，单稳的暂态尚未结束，M_1 先变对前沿触发的 D 触发器而言，由于 D 端先变低后，C 端(即 M_3)再上升，，故打入的仍旧是 0，所以正常时不会出现定时回答讯号。在特殊情况下，例如访问未插入的系统固件时，则读／写讯号一直维持着，达到一定时间时，单稳的暂态结束，M_3 变高电平时 M_1 仍为高电平，从而使 D 触发器状态变低了，XACK／变低有效。XACK／讯号使 CPU 脱离 T_{wait} 状态，继续执行下去。

§2-2-5 DMA 传送与总线控制器

在DMA(直接存贮器存取)传送中，外围设备直接与内存联系，而不影响微处理机的内部寄存器。采用程序方式传送数据(例如中断方式传送)，处理一次 I/O 交换往往需要几百微秒。如果外部事件频频发生，比如每隔 4～5ms 发生一次，则会严重影响主程序的执

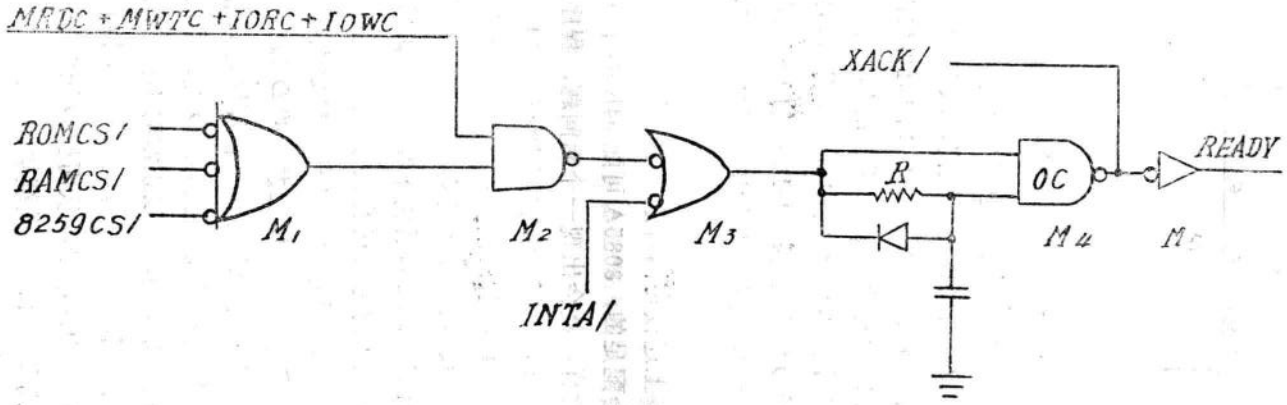

图 2-10 DJS-053CPU 板上的读写回答电路

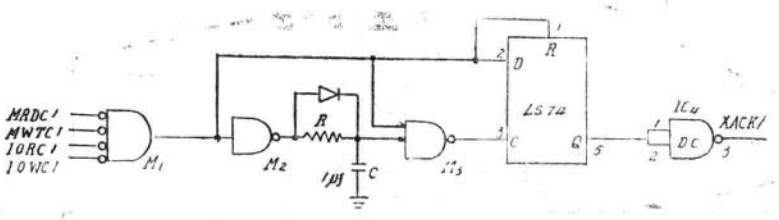

图 2-11 DJS-053 的定时回答讯号产生电路

行。DAM 传送是一种实现内存与外围之间高速交换数据，而又较少影响主程序执行的最有效方法。

大多数微处理机都设有一、二条引出端，专供 DMA 传送通讯用。例如 8085A 的 HOLD 和 HLDA 两条引出端。当外设需要 DMA 传送时，向处理机提出 DMA 请求（使 HOLD 变高电平）处理机在接收到这个请求后，内部锁存该请求继续做完当前的机器周期，而后放弃使用总线（使地址、数据、$\overline{RD}$、$\overline{WR}$ 和 IO/M 等处于三态）。只有当外设撤消 HOLD 后，处理机方能再度获得总线。8085A 的 HLDA 是给外设 DMA 请求的回答，通知外设 8085A 已放弃总线，可进行 DMA 传送了。

DJS—053 系统中，系统总线采用 MULTIBUS。DMA 传送的应答讯号，由总线请求，BREQ/，总线忙 BUSY/，总线优先出 BPRO/ 和总线优先入 BPRN/ 组成。外设的 DMA 请求由 BREQ/ 传送。BREQ/ 经一反相器反相后送到 8085A 的 HOLD 输入端。由图 2-12 可见，DJS-053 系统中，要实现 DMA 传送，要仅仅使 8085A 进入保持状态是不够的，还必须让出对系统总线的控制，使主微计算机的总线驱动器，数据总线驱动器和控制总线驱动器的输出变为三态，图 2-12 为 DJS—053 的总线控制逻辑。它由一个 D 触发器和一些门电路组成，RESET 时，RESETOUT 为高电平，INIT/ 变低电平，作系统总清，还使总线请求允许 BRE 触发器置 1。因而总线驱动器变三态而让出系统总线，BPRN/ 变低。DMA 设备在测到 BPRN/ 低电平时，还要同时判得 BUSY/ 为高电平时，才能作 DMA 传送。在图 2-12 中，BUSY/ 是由总线请求允许触发器的 $\overline{Q}$ 端经 OC 门反相后得到。显然在 RESET 时，由此得到的 BUSY/ 为高电平。

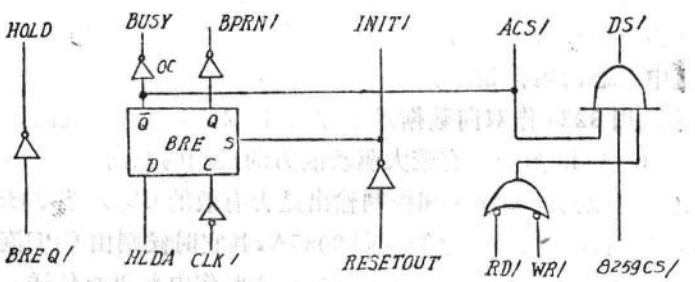

图 2-12 DJS-053 的系统总线控制器

在 RESET 后，如果没有 HOLD 请求，则 HLDA 为低电平，总线请求允许触发器为“0”状态。这时由 8085A 留用系统总线，BUSY/ 为低电平，BPRN/ 为高电平。如果有 HOLD 请求，则 8085A 进入保持后再经一个时钟周期，总线请求触发器置 1，主微计算机让出总线，直到外设使 HOLD 变低为止。图 2-13 出示了图 2-12 中的若干波形中断。也是 CPU 的一