微处理器系统設计基础

（下）

仇仪傑编

南京工学院工业自动化教研组
江苏省自动化协会微机学习班

下册目录

第五章 微处理口系统的接口技术

§5—1 概述

在微处理口系统中，由于微处理口和外部设备都有它们自己固有的工作方式和特征，因此要使它们结合在一起组成一个能协调工作的整体就必然需要一些用来沟通信息的硬设备，这些设备就称为接口（interface），而构成接口的原理和实践就称为接口技术。下面我们先介绍接口所担负的功能和它们的类型，然后再来分析信息在接口中传递和交换的情况。

一、接口的功能和类型

1．功能

1）．实现信息有效转换

(1)．缓冲和同步

在系统中各部分的信息速度往往是不一致的，例如MPU的速度一般总要比外部设备高，为了实现信息同步，速度高的信息就必须进行缓冲，一般这种功能是由三态门来完成。

(2)暂存

为了使信息能在需要使用时仍保持有效，就必须设置暂存寄存口来锁存那些存在时间较短的信息。例如MPU送到数据总线的信息保持的时间一般为几百个毫微秒（nS），这对外部设备来说是太短了，因此接口必须担负起锁存的作用。

(3)匹配

由于MPU的输入／输出信息无论在信号型式、电平值和驱动能力上都是一定的，而外设在一般情况下其输入／输出信息的情况也不能随意改变，因此这些协调工作必须由接口来完成。例如为实现信号型式的匹配要有数字—模拟相互转换和电压—电流相互转换的能力，为实现电平值匹配需要有电平转换电路，为扩大输出能力还需要增加驱动电路等。。

2)进行设备选择

一个微处理口系统一般都有好几个外设，严格一点说是一定有一个以上的受控单元（一个外设可以不止有一个受控单元），为了在它们中间寻找需要与之交换信息的那一个单元，接口就必须具有地址线译码能力以实现设备选择。

目前微处理四实现设备选择有二种方法，第一种是单独设置输入/输出指令，例如8080系统采用IN〔port〕，OVT[port]二个指令，它们都是双字节指令，IN和OUT为输入和输出操作码，〔port〕表示输入或输出的地址又称"口"号，因为它占一个字节，因此不论输入或输出设备数都可多达256个。另一种称为存储四映象（memory mapping）法，它是把处设和内存实现统一编址（在0—65535之间），不需再设新的输入/输出指令而直接借用存储四指令，因此指令丰富，用得比较广泛，6800系统就是采用这一种方法。但前一种也有它的优点，例如它不占用内存地址、字节短、执行时间要快些，另外接口电路中的译码部分也要相应简单些。

3）发布命令信息

在系统工作过程中，外部设备在有些特况下需要的是控制命令而不是数据，这时接口就要担任起发布命令信息的作用。此外为了使系统能协调工作，接口还必须向处理四发回一些外设的状态信息，这些功能也必须由接口来完成。

4）具有时序控制能力

要合理有效地完成上述这些功能，一般都要求接口本身具有时序控制能力以便能按要求顺序逐步地完成。

2．类型

接口当然可以由中小规模等成电路组合而成，但目前随着大规模集成电路的日趋成熟，接口也已进入大规模集成化阶段。由于对接口的要求是多种多样的，不可能做一种接口来满足，因此就必然有各种不同类型。从它的发展特况来看比较有代表性是具有可程编能力的通用接口，这些接口芯片不但具有很丰富的功能，也就是有一定的通用性，而且它的某些功能还可以通过MPU的指令来变更。一般特况下只需要附加少量硬件（中、小规模集成电路）甚至不需要任何附加硬件的配合就能实现外设和系统的协调联接。这种接口还可以"接管"MPU的一部分工作以提高MPU的时间效率。接口芯片另一个发展方向则是考虑到某些常用外围设备例如CRT、软盘（floppy）等，它们在各种系统中都要使用，如果采用通用接口那不但要附加许多

硬件以致体积庞大，而且通用接口的许多功能它们又不需要，因此从降低价格（提高经济指标）、缩小体积和提高工作可靠性出发还专门生产了这类外围设备的专用接口电路。近年来为方便用户还将一些接口和 RAM 或 MPU 组合在一块芯片上制造，这也是一种有效的发展趋向。

现有的大规模集成电路接口芯片可以分成三大类：

1）通用接口芯片

属于这一类的有这样几种

（1）并行输入/输出（parallel input – output）接口

它们的输入和输出信息都是并行传送的，位数一般都是八位，有可程编的特性，比较有代表性的产品为 MC6820 和 intel 8255。前者是 Motolora 公司的产品，称为可程编接口适配口 PIA（pragrammable interface adapter），后者是 Intel 公司的产品，称为外围并行接口 ppI（periphral parallel interface），它们可用来控制光电读入机、打印输出、键盘—显示装置等。

（2）串并行转换接口

这种接口除了并行接口的一般功能外，它的主要特点是能实现串并行仪息的转换并自动对信息实现同步和检错，还可以进行发送控制等。比较有代表性的产品有 MC 6850 和 intel 8251。前者称为异步通信接口适配口 ACIA（asynchronous communication interface adapter），后者叫做同步—异步接收发送口 UART（universal asynchronous receiver transitter），它不但可用异步的方式进行信息传送，这时工作速率约为 9.6K 波特（baud —每秒传送的二进制数位），也可工作在同步方式，这时信息速率可达 56K 波特。

这种接口芯片可用于电传机、CRT、调制—解调口上。

（3）调制—解调通信接口

它的基本功能是可将串行的数字信息转换成适合于共载波线路（例如电话线）上传送的模拟信息，或是反过来将这种模拟信息转换成串行的数字信息。从实质上来看它就是一种数模/

模数转换装置，只不过模拟信息是带有载波的而已。按照调制的方法可以有调幅、调频和调相式三种，调幅式是利用某一载波频率作为基准发出二种不同幅值的信息亚用某一种幅值的载波表示"0"（称为空格SPACE）、另一种表示"1"（称为标记mark），由于这种形式的抗干扰能力较差所以虽然简单但不太使用。调频式则是使用较多的一种形式，它是将某一频率的载波代表"0"、另一频率的载波代表"1"。调相式则可以用有绝对方式和相对方式二种，绝对方式是以某一个相位作基准，和它同相的载波为"0"、反相（差180°）的载波为"1"。相对方式则以二个载波之间的相位差值作为表示"0"和"1"的依据。调制—解调通信接口比较有代表性的是MC6860，一般称它为调制—解调口 MOD—DEM（modulator-demodulator）。

2）调度接口芯片

（1）可程编中断控制口 PIC（programmable interrupt controller）

又称优先度中断控制口（priority interrupt controller）利用这种芯片可使MPU实现多重中断功能。它可以进行中断的优先度管理，即对要求中断的设备按其重要程度确定其响应的次序。还能进行自动矢向，即不需经过判断程序而可直接转到应服务设备的处理程序去。这种芯片的代表产品有MC6828和intel8259。

（2）可程编间隔定时口 PIT（programmable interval timer）

在前一章中已经说过可以利用软件来实现定时，但显然这样会浪费MPU的时间，而且如果系统有中断级的话就会使定时不平，因此在实时系统中PIT这种芯片具有相当重要的地位，它不但能发出定时信息还能用来完成测量等功能，用处较大。比较有代表性的产品为MC6840和intel8253。

（3）直接对存储口存取控制口 DMAC（direct memory access controller）

利用这种芯片可以使外部设备不需通过MPU而直接以较高

的速度和存储口交换信息，一般用于交换信息量较大的情况。比较有代表性的产品有 MC6844 和 intel8257 。

3)专用外围设备控制接口芯片

这类芯片的形式繁多，常用的有：

(1)触盘/显示（keybroad/display）控制口典型产品有 intel8279 。

(2) CRT（cathode—ray tube）控制口

典型产品有 MC6845 和 intel8275 。

(3)盒式磁带（cassette tape）控制口

(4)打印机（printer）控制口

典型产品有 intel8295 。

(5)软磁盘（floppy）控制口

典型产品有 MC6843 和 intel8271 。

(6)动态 RAM（dynamic RAM）控制口

典型产品有 MC3480 。

(7)模—数转换（analog to digital convertor）控制口

典型产品有 M68MM05A/B 。

(8)数—模转换（digital to analog Convertor）控制口

典型产品有 MC1406 。

二．MC6800 的控制信息

要了解接口的功能首先要对 MPU 的控制信息有较深入的认识，因此虽然我们在第二章硬件中约略介绍了 MPU 的引出脚，但光有这些知识是不够的，为此这里再对 MC6800 的控制信息作进一步的分析。

前面已经介绍过 MC6800 除了空脚（2），电源（1），地（2）时钟（2），地址线（16）和数据线（8）外共有控制信息线 9 条，它们分别为读/写 R/W，存储口地址有效 VMA，暂停 $\overline{HALT}$，总线有效 BA，三态控制 TSC，数据总线可用 DBE，外部中断请求 $\overline{IRQ}$，非屏蔽中断请求 $\overline{NMI}$ 和复位 $\overline{RESET}$，下面我们把

它们分成四组来讨论。

1.读/写 R/W (read/write)和存储口地址有效 VMA (valid memory address)

读写输出信息 R/W 线具有三态缓冲结构，它在高电平时能驱动一个标准 T.T.L 负载和 90Pf 的电容负载。

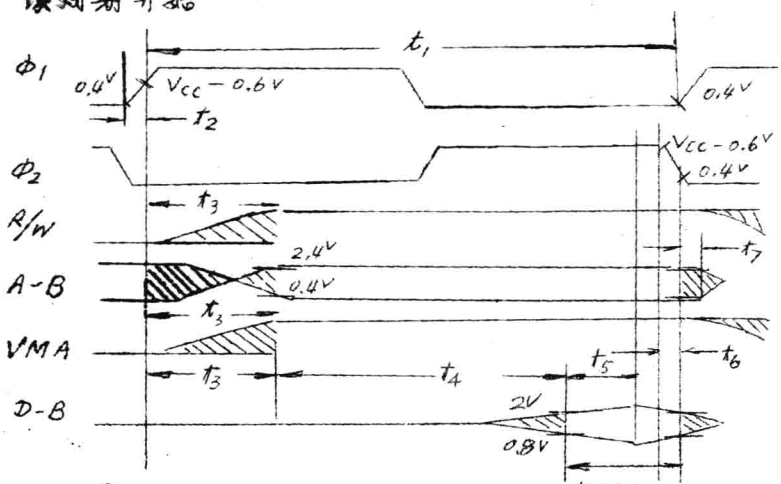
这个信息可以用来指示数据总线上信息的流向，只有当 MPU 处于写状态时 R/W = 0 为低电平，表示数据总线上的信息是由 MPU 流向外部设备。在读状态和其它情况 R/W = 1，这时如果数据总线有效，那表示信息是从外部设备流向 MPU。当 MPU 受外部控制信息的控制而停止工作时，R/W 信息线可输出高阻抗三态。

存储口地址有效输出信息 VMA = 1 可用来指出当前地址总线送出的代码是有效的，否则 (VMA = 0) 就是无效的，它的输出不具有三态结构，当它在高电平时可驱动一个标准的 T.T.L 负载和 90Pf 的电容负载。

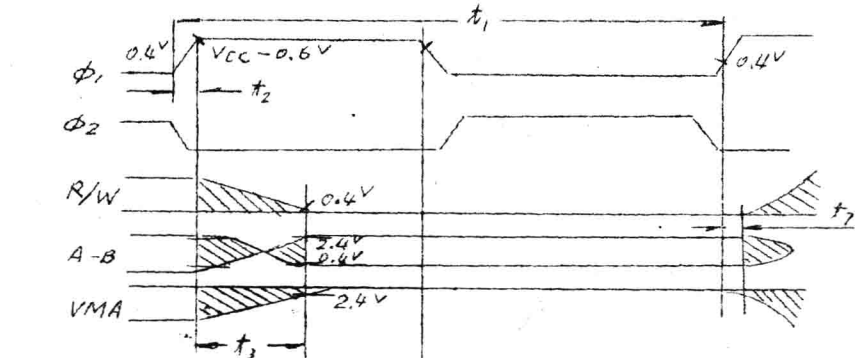
在具体进行读/写操作时，这二个信息和地址总线 A—B、数据总线 D—B 上信息之间的关系可分别见 <图 5.1.1> 和 <图 5.1.2> 所示。在 <图 5.1.1> 中画出了 MPU 从存储口或外设中读取数据的情况，当 ϕ_1 上升到 0.4V 时表示读机口周期 ($t_1 = 1^{MS}$) 开始，经过 $t_2 = 25^{nS}$ 的时间 ϕ_1 上升到 Vcc − 0.6V，这时如果 R/W 信息原来处于写状态 (R/W = 0) 这时就开始上升，在此同时地址总线 A—B 上的地址信息和 VMA 信息也同时上升，约经过 $t_3 = 200 - 300^{nS}$ 的地址延时使地址线上信息有效，这时，VMA = 1 向外部表示地址有效。然后被选中地址的外设将数据取出到数据总线 D—B 上，它最大不得超过 $t_4 = 530^{nS}$，这一点实际上是比较有保证的。从这时开始 MPU 就从 D—B 取入数据，为保证可靠从这时到 ϕ_2 下降到 Vcc − 0.6V 这段时间 t_5 不得小于 100^{nS}。另外还希望在 ϕ_2 下降到 0.4V 之后数据还需保持一段时间 $t_6 = 10^{nS}$，而 R/W、VMA 和地址信息保持的时间为 $t_7 = 30 - 50^{nS}$。

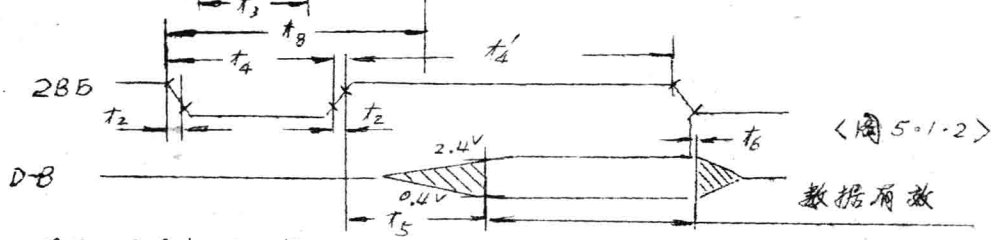
将数据从 MPU 写入存储口或外设的情况见 <图 5.1.2> 所示，一般说来数据总线可用信息 DBE 都和时钟 ϕ_2 接在一起，这

读周期开始

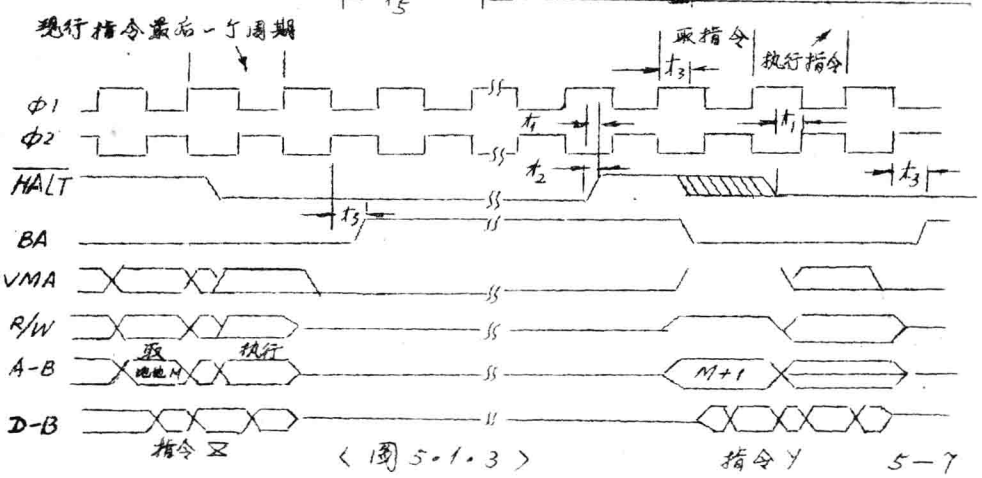


〈图 5·1·1〉





〈图 5·1·2〉



〈图 5·1·3〉

里则画出了更一般的情况，也就是DBE有它自己本身时序时的情况，虽然这时D—B上的信息不再由ϕ_2控制而直接由DBE信息控制了。在写周期开始一直到地址A—B建立的过程和读周期情况相似，不同的只是R/W信息是由高变低而已。之后我们假设DBE经由$t_4=150^{nS}$低电平阶段转入$t_4'=450^{nS}$的高电平阶段，当DBE上升到2^V后就进入数据延时阶段。为保证数据有足够的有效时间，延时t_5最大不能大于225^{nS}保持时间t_6不能小于10^{nS}。同样为保证工作可靠，R/W、VMA和地址信息的保持时间为$t_7=30-50^{nS}$。有关对DBE信息的其它时间要求（例如t_8）将在后面介绍。

2．暂停$\overline{HALT}$和总线有效BA（bus available）

利用暂停输入信息$\overline{HALT}$可使MPU的工作受外部控制，当$\overline{HALT}=1$为高电平时MPU正常工作，这时总线有效输出信息$BA=0$。在$\overline{HALT}=0$为低电平时MPU停机，（但它对外部中断请求仍能记忆，一旦在MPU脱离停机状态后仍能为这些中断服务）使地址和数据线、R/W信息线对总线呈高阻抗三态并强迫$VMA=0$，接着才使总线有效信息$BA=1$，表示MPU已经让出总线（有效），系统其它部分可开始利用总线（例如进DMA操作等），其波形关系可见<图5·1·3>所示。这里有几点应该说明的是：

1）停机一定要等现行指令执行完才实现。

2）停机信息$\overline{HALT}=0$必须出现在现行指令最后一个周期ϕ_1下降沿前$t_1=200^{nS}$，否则将再执行一条指令后才实现停机。

3）HALT的上升和下降沿时间$t_2<100^{nS}$

4）在让出所有总线而且$VMA=0$之后由ϕ_2的上升沿触发使总线有效信息BA变高，其延时t_3应不大于250^{nS}。

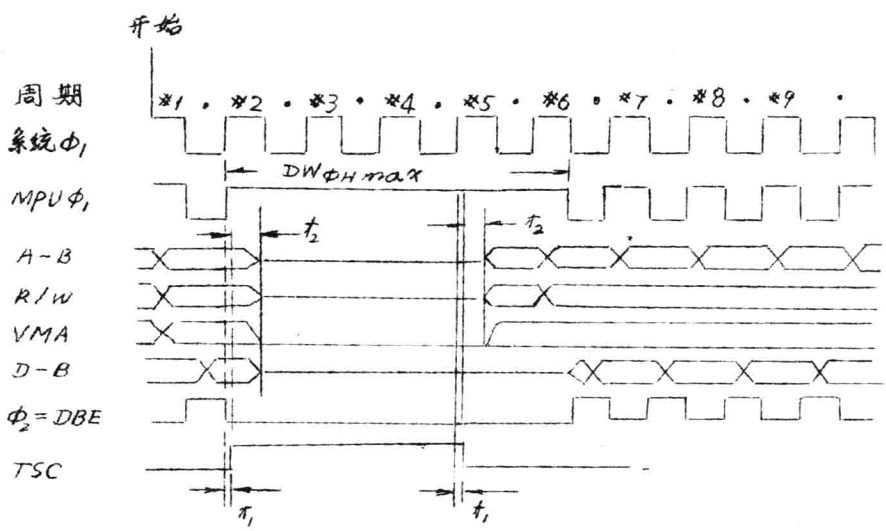
5）停机时间可任意由外界确定，当外界仅使$\overline{HALT}$在一个指令周期内为高电平，那MPU每次只执行一条指令，程序调试时就经常采用这种所谓"单步"的工作方式（见图5·1·3>阴影部分）。

6）当暂停信息$\overline{HALT}=1$如在时钟周期ϕ_1下降沿前$t_1=200^{nS}$

之前产生，那么下一个ϕ_1上升沿恢复全部总线並使 VMA＝1，总线有效 BA 信息也及时变低（BA＝0），其时延$t_3<250^{nS}$。

3．三状态控制 TSC（three State Control）和数据总线可用 DBE（data bus enable）

三状态控制输入信息 TSC是用来提供外界对地址总线和R/W实现控制的能力的，这样就便于实现 DMA 传送。当 TSC＝0为低电平时，MPU正常工作，而当 TSC＝1时使MPU打止工作。一般应该使 TSC在系统时钟ϕ_1上跳时触发，经过$t_1<100^{nS}$之后 TSC就变高而 MPU时钟就始终保持高电平（打止工作），再经过一段延时$t_2<270^{nS}$使地址线和 R/W 呈高阻抗三态（MPU让出总线给系统使用）並强迫 VMA＝0 以避免外设误读/写。它的工作情况可见<图 5·1·4>所示。应该指出 TSC信息只能



（图 5·1·4）

控制地址线，R/W线和 VMA，而对数据总线却无控制能力。因此一般它还需和数据总线可用输入信息 DBE 配合使用。此外由于 TSC信息迫使 MPU 打止工作，使 MPU时钟$\phi1=1$始终保持高电平，$\phi2=0$始终保持低电平，这样就会使 MPU 内部准静态触发回所保持的信息得不到再生而造成丢失现象，所以$\phi1=1$保持高电平时间不能超过 MPU所规定的最大值，一般为 4—5 倍

时钟周期（约 4500 nS），目前改进型可达 9500 nS 之久。TSC信息恢复的情况也可从<图 5·1·4> 中看到，它也在系统时钟 $\phi 1$ 上跳时触发，经过 $t_1 < 100$ nS 后变低，再经 $t_2 < 270$ nS 使地址线、R/W 线和 VMA 恢复。由于要在这个周期的 $\phi 2$ 上跳时恢复工作时间不够，因此这个周期实际上只成作为同步用（图中周期 * 5），一直到下一个周期（图中周期 * 6）的 $\phi 2$ 上跳（即 $\phi 1$ 下跳）时才恢复 MPU 正常工作。

现在再来介绍数据总线可用输入信息 DBE，当 DBE = 1 激动输出缓冲口使 MPU 控制数据总线，在 DBE = 0 时禁止输出缓冲口使 MPU 对数据总线呈高阻抗三态（让出数据总线给系统）另外当 MPU 处于读状态（R/W = 1）时也禁止输出缓冲口，MPU 输出三态。为了使用方便一般都将它和 $\phi 2$ 相连，这样当 $\phi 1$ = 1、$\phi 2$ = 0 时 DBE = 0 MPU 让出数据总线进行内部操作，而当 $\phi 1$ = 0、$\phi 2$ = 1 时 DBE = 1 MPU 就占用数据总线和外部设备交换信息。在 TSC = 1 时由于 $\phi 1$ = 1（$\phi 2$ = 0），DBE = 0，MPU 也让出数据总线以配合 TSC 信息使系统能借用全部总线进行 DMA 等操作。当外设希望数据有效时间较长时 DBE 信息可利用特有的时序，即减少 DBE = 0 的时间、增加 DBE = 1 的时间，这种情况可参见<图 5·1·2>。这时有二点还是要求满足以保证可靠工作的，一是 DBE 的下跳沿和系统时钟 $\phi 1$ 的下跳沿之间的最小时间 t_3 不得小于 300 nS，二是 DBE 的低电平保持时间 t_4（DBE = 0）不能小于 150 nS，而且一定要出现在 $\phi 1$ = 1 阶段。

到现在为止我们已经将 MC6800 的内部工作控制信息介绍完了，为明确它们的动能在这里列出<表 5·1·1>以助记忆。

信息	电平情况	
	0	1
R/W	表示MPU为非读状态	表示MPU为读状态
VMA	表示MPU地址线信息无效	表示MPU地址线信息有效
$\overline{HALT}$	停止MPU工作(仍记忆中断) 让出所有总线和R/W线	并使 VAM=0 BA=1 MPU正常工作 BA=0
BA	表示MPU处于工作状态	表示MPU处于暂停或等待状态已让出全部总线.
TSC	MPU正常工作	MPU让出地址和R/W线, VMA=0 但BA=0
DBE	MPU让出数据总线	MPU占用数据总线

〈表 5·1·1〉

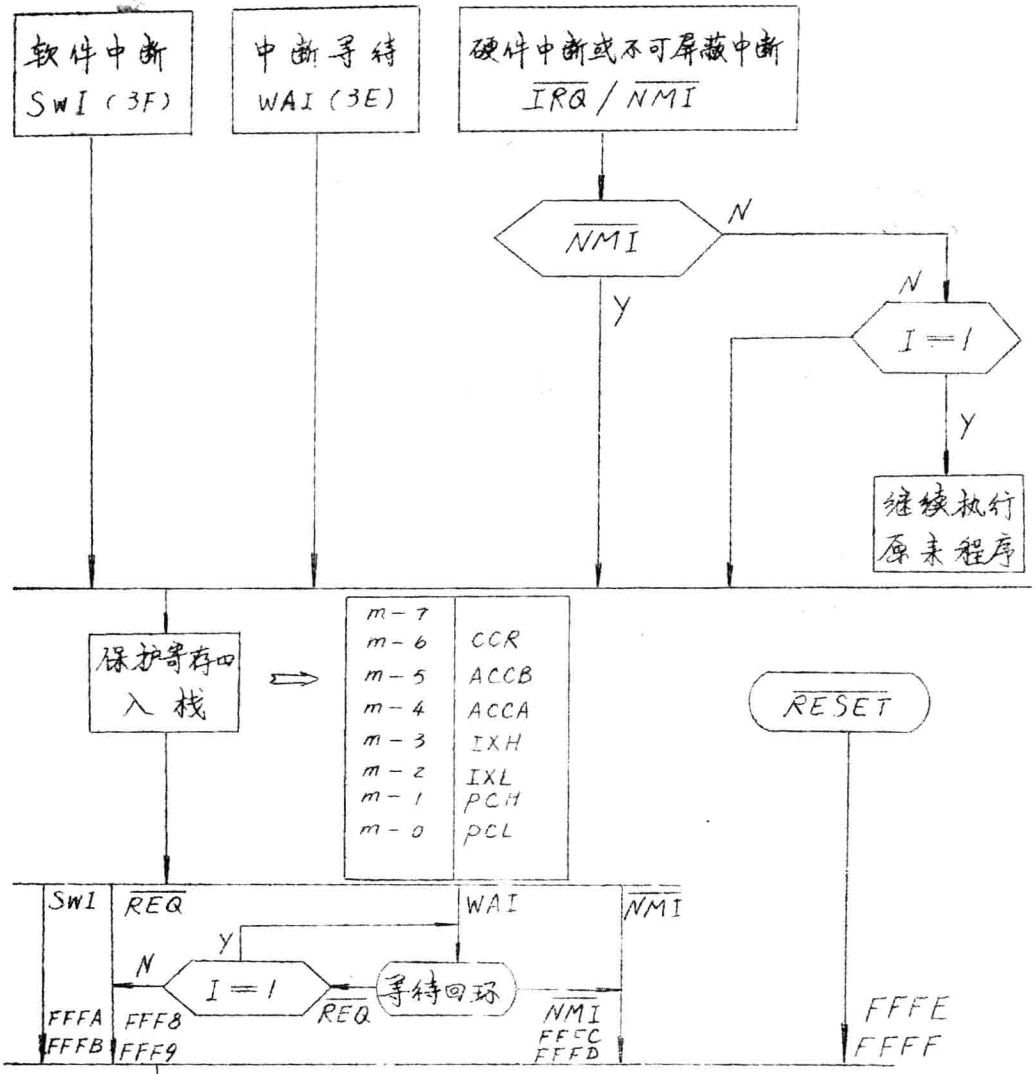
4. 中断控制信息

在MC6800中有三条中断控制信息线，它们是中断请求 $\overline{IRQ}$ (interrupt request)信息、不可屏蔽中断 $\overline{NMI}$(non-maskable interrupt)信息和复位中断 $\overline{RESET}$信息，其中前二条输入线在芯片内部已接有高阻抗的上拉(pull-up)电阻，因此允许外界以线"或"形式输入，但为达到电路的最佳匹配最好在线"或"形式输入时再接以3K的上拉电阻，而 $\overline{RESET}$输入线一般不希望采用线"或"输入形式。

为了能更清楚地了解这些中断控制信息的机理，还得从MC6800的中断结构谈起。在〈图 5·1·5〉中画出了MC6800的中断结构特性，它一共有四种中断级，按MPU对它们的响应优先程度(优先度)依次为复位 $\overline{RESET}$、不可屏蔽中断 $\overline{NMI}$、软件中断SWI和中断请求 $\overline{IRQ}$(又称硬件中断)。其中 $\overline{RESET}$用作对系统进行复位和初始化，SWI可用来模拟硬件中断实行单步、断点等功能，这在调试程序时相当有用。而 $\overline{IRQ}$是作为一般外部设备的中断请求信息它可以利用软件来实现屏蔽，如有多个需中断处理的外部设备可接成线"或"输入形式。 $\overline{NMI}$是不能

由软件实现屏蔽的外部设备中断请求信息，一般用在比较重要的中断要求例如断电事故等。为实现快速中断处理，对应每一种中断都规定一个用来存放相应处理子程序首址的中断矢量地址，从〈图5·1·5〉中可以看出它们分别为FFFE—FFFF、FFFC—FFFD、FFFA—FFFB和FFF8—FFF9。如果我们没有用足全部16根地址线，那这些中断矢量地址分别为（n—0）—（n—1）、（n—2）—（n—3）、（n—4）—（n—5）和（n—6）—（n—7），其中n为全部地址线为1时所对应的地址，这是由MC6800结构所确定的。例如我们用了12根地址，则$n=(1111'1111'1111)_2=0FFF$，相应中断矢量地址就为0FFF—0FFE、0FFC—0FFD、0FFA—0FFB和0FF8—0FF9。为保护中断时原执行程序中各工作寄存器的内容应设定堆栈区并将栈底位置m在系统初始化时送入堆栈指针SP中保存。另外为实现中断屏蔽作用还采用状态寄存器的第四位作用中断屏蔽位I，该位可由程序设定，当I=0表示无屏蔽、I=1表示要屏蔽中断。在〈图5·1·5〉中还涉及二个指令，即软件中断SWI（操作码为3F）和中断等待WAI（操作码为3E），这二条指令的内容和操作已经在指令一节中作了介绍，这里就不再重复，如果不熟悉可参阅之。

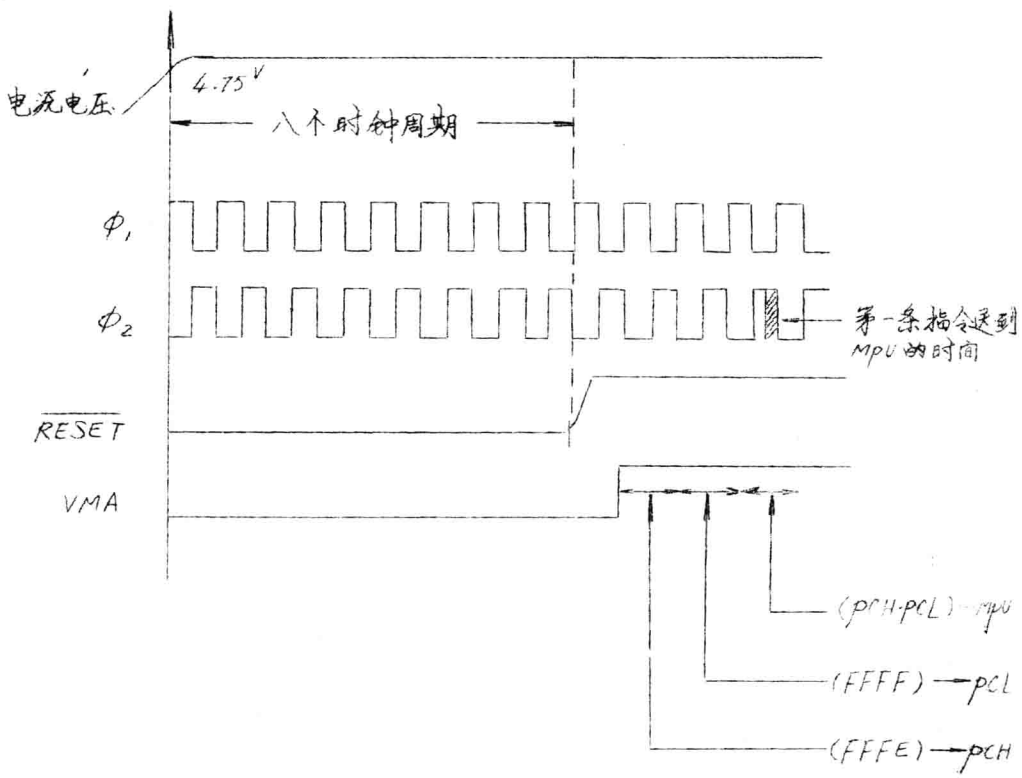
现在就可以来介绍〈图5·1·5〉所示的中断流程了，无论当指令SWI和WAI或中断信息线$\overline{IRQ}$和$\overline{NMI}$出现，都进入这个流程。可以看到除了$\overline{IRQ}$需进行判I=1之外（如I=1则继续执行原来程序，不过中断仍记忆着）都接着进行工作寄存器入栈保护，它依次将PCL、PCH、IXL、IXH、ACCA、ACCB和CCR压入堆栈，并将栈针退到m—7。之后MPU就根据信息情况自动地将相应的矢量地址送入地址线，其中如果是等待指令WAI则处于循环询问状态。然后将中断屏蔽位I置"1"以防止在中断定向过程中再来中断而造成信息丢失（之后如不需屏蔽，可由用户在中断处理子程序中开屏蔽位I=0，有关内容将在下一节中介绍），再从矢量地址所对应的单元中取出矢量（相应处理子程序的首址）送到PC，于是接下去就自动定向转入了相应处理子程序（中断程序）。如果是复位中断$\overline{RESET}$



FFF8 FFF9	硬件中断处理程序首址高位 低位
FFFA FFFB	软件中断处理程序首址高位 低位
FFFC FFFD	不可屏蔽中断处理程序首址高位 低位
FFFE FFFF	复位再启动处理程序首址高位 低位

〈图 5·1·5〉

出现，因为它的中断优先度最高，所以不管什么情况都直接转入复位处理，使 MPU 内部恢复到某一预定的初始状态以便重新启动。应该指出一点，就是在接到 $\overline{RESET}$ 信息，在地址线上自动出现 FFFE — FFFF 的地址并使中断屏蔽位置 "1" 之后到转入复位处理子程序之前，除了将矢量送入 PC 之外还要使所有工作寄存由复位（包括中断屏蔽位），这一点在〈图 5·1·5〉的流程图中没有表明，所以应予注意。它的工作过程可参见〈图 5·1·6〉所示。首先要保证起作用 $\overline{RESET}$（低电平）信息至少要大于八个时钟周期以保证 MPU 进行内部处理，而且这八



〈图 5·1·6〉

个时钟周期一定要在电源电压已经大于 4.75 V 之后计算起（对开机复位要注意此问题）。在这段时间 MPU 的数据总线呈高阻抗三态，地址总线上自动呈现 FFFE 地址码，由于这时 VMA = 0 因此地址无效。然后如果 $\overline{RESET}$ 信息在时钟 Φ2 上升沿前恢复高电平（否则再要加一个周期），使数据总线恢复并使 VMA = 1，