

微型机通用接口电路和应用

(下)

周明德 白晓笛 田开亮 编著

中国计算机技术服务公司培训部
山西分公司承印

一九八五年三月

73.931
12/3

微型机通用接口电路和应用

(下)

周明德 白晓笛 田开亮 编著



中国计算机技术服务公司培训部
山西分公司 承印

一九八五年三月

第三节 8257—DMAC

Intel公司的8257是一个四通道直接存储器存取控制器(DMAC)。这是一片使用单一+5V电源,单相时钟,40引脚双列直插式的集成电路,它是专为满足80/85微型计算机系统中高速数据传送的要求而设计的,其中8257—5与MCS—85兼容。

一、8257—DMAC的基本功能

8257必须和一片8212I/O芯片或类似的8位锁存器(如74LS363或74LS373)一起使用,才能形成一个完整的四通道DMA控制器。在用软件进行初始化之后,8257就可以控制每一个通道在存储器和外部设备之间以最高0.5M波特的速率直接传送最多16K(16,384)字节的数据块,而不需要CPU介入。8257并不能进行Z80—DMA所能够完成的存储器→存储器或I/O设备→I/O设备之间的传送。

当8257接收到外部设备的DMA传送请求之后,它进行下述操作:

- 1、取得对系统总线的控制。这是通过向CPU发出总线保持请求信号HRQ,而CPU以保持响应信号HLDA回答完成的。

- 2、对连接到最高优先级通道上的外部设备的请求进行响应。8257内部具有DMA请求优先权控制逻辑。

- 3、输出存储器地址低8位到系统地址总线A₀—A₇上,并通过数据总线引脚输出存储器高8位地址到8212I/O端口,再由8212把该地址置于地址总线高8位A₈—A₁₅上。

- 4、产生适当的存储器和I/O设备读/写信号,控制外部设备与所寻址的存储单元之间传送数据。

只要外部设备保持它的DMA请求,8257就保留其对系统总线的控制权并重复传送的步骤。传送的起始地址和传送的字节数都是由程序指定的。每传送128个字节,8257可输出一个MARK信号给外设,当指定字节数的数据传送完毕,8257能产生一个终点计数输出信号TC,表示操作已完成。

8257提供了三种不同的操作方式:

- 1、DMA读(存储器→外部设备):即将数据从存储器读出传送到外部设备。

- 2、DMA写(外部设备→存储器):即将外部设备传送的数据写入存储器。

8257的DMA读或写都是对存储器而言的,从存储器读出就是DMA读操作,写入存储器就是DMA写操作。上述两种操作方式只不过决定了存储器与外设之间数据传送的不同方向。

- 3、DMA校验:这种方式实际上并不进行数据传送。当一个8257通道处于DMA校验方式时,它会像上述的传送操作一样进行响应,只是不产生存储器或I/O设备的读/写控制信号,这样就阻止了数据的传送。但是8257仍将保持着它对系统总线的控制

权,并且每个DMA周期都将响应外部设备的DMA请求。外部设备可以使用这些响应信号,在外部设备内部对一个数据块的每个字节进行存取,以便进行诸如CRC校验码的累加一类的校验过程。例如,一组DMA校验周期可以跟随在一组DMA读周期(存储器→外部设备)的后面,使外部设备能够校验它新得到的数据。

8257并没有Z80—DMA那样的字符检索功能。在8257的任何一种操作方式中,数据都不进入8257内部。8257本身并没有任何校验能力。所谓DMA校验方式,也只是由8257控制总线,应外设的请求,在每个DMA周期向外设输出一个DMA响应 $\overline{\text{DACK}}$,由外设利用这些信号自己去进行某种校验,直到达到规定的字节数,或外设不再请求时为止。

8257—DMAC与Z80—DMAC比较,有几个主要特点:

1、8257的初始化比Z80—DMA的初始化工作简单得多。8257的一个通道在初始化时,通常只需要写入5个字节就能够工作,而Z80—DMA则需要写入大约35个字节。因此,8257比Z80—DMA更容易使用。

2、8257有四个独立的DMA通道,而Z80—DMA只有一个通道。如果在存储器与四个磁盘驱动器之间进行DMA传送,用一片8257可以实现(见图3—61),而用一片Z80—DMA则不能实现。

3、8257的四个DMA通道可以具有固定的或旋转的优先权控制。Z80—DMA因为是单通道器件,所以没有这种控制逻辑。但是Z80—DMA提供了在外部与其它Z80—DMA芯片形成链形优先权控制的结构,这却是8257所没有的。

值得注意的是,8257不具备可程序的中断控制功能。不管在什么情况下,都不能由程序允许或禁止8257产生中断,因为它内部根本没有中断控制电路。

二、8257—DMAC内部结构

8257的框图如图3—51所示。

1、DMA通道

8257有四个独立的DMA通道(CH0~CH3)。每个通道包括两个16位寄存器,即地址寄存器和终点计数(Terminal Count)寄存器。在允许DMA通道工作以前,必须对这两个寄存器进行初始化。

①地址寄存器:要存取的第一个存储器单元的地址装入这个寄存器。8257每传送一个字节数据,地址寄存器的内容都自动加1。与Z80—DMA和MC6844—DMAC都不同,8257的地址寄存器只能加1不能减1。它在初始化时没有从指定的起始地址开始递增或递减的选择。因此8257只能以从低地址单元开始向高地址单元递增的顺序传送,而不能以高地址到低地址递减的顺序传送。

②终点计数寄存器:初始化时,DMA周期数加1装入终点计数寄存器的低14位。如果所要求的DMA周期数为N,通常取N—1作为终点计数值装入终点计数寄存器。

所谓DMA周期实际上是由4个系统时钟周期组成(无等待状态),每个DMA周期传送一个字节。终点计数寄存器在每个DMA周期减1。因此,它实际上是一个字节计数器,每传送一个字节计数值减1,直到减到0以后输出终点计数(TC)信号。

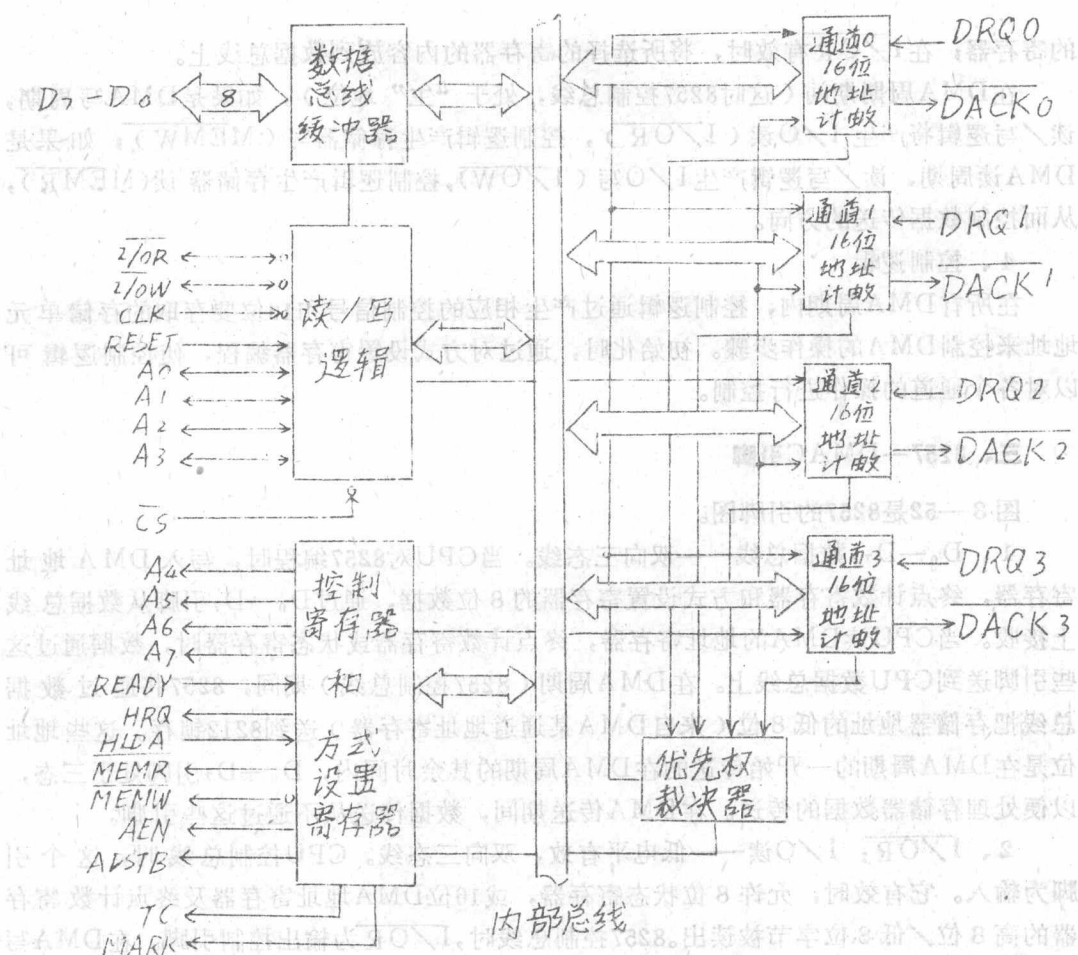


图 3—51 8257—DMAC框图

终点计数寄存器的最高两位用于确定该通道DMA操作的类型。这两位 在 DMA 周期内是不能修改的，但是可在每个DMA数据块传送之间改变。

由于终点计数值最大只能为 2^{14} ，因此限制了8257的数据块传送最多为 16,384 个字节，即为Z80—DMA和MC6844—DMAC最大数据块传送长度的四分之一。

2、数据总线缓冲器

这是双向三态 8 位的缓冲器，用于与系统的数据总线接口。当CPU控制总线时，它对8257进行读/写的所有信息都经过这个缓冲器。当8257控制总线时，在DMA周期的一开始，8257将某通道地址寄存器的高 8 位地址经这个缓冲器输出到8212锁存，然后处于三态。

3、读/写逻辑

当CPU对8257编程或读8257寄存器时（这时CPU管理总线，而8257在系统中处于“从”地位），读/写逻辑接受I/O读（I/OR）或I/O写（I/OW）信号，对地址总线的低 4 位（ A_0-A_3 ）译码。在I/OW有效时，将数据总线的内容写入所寻址

的寄存器；在 $\overline{I/O\overline{R}}$ 有效时，将所选择的寄存器的内容放到数据总线上。

在DMA周期期间（这时8257控制总线，处于“主”地位），如果是DMA写周期，读/写逻辑将产生 $\overline{I/O\overline{R}}$ （ $\overline{I/O\overline{R}}$ ），控制逻辑产生存储器写（ $\overline{MEMW}$ ）；如果是DMA读周期，读/写逻辑产生 $\overline{I/O\overline{W}}$ （ $\overline{I/O\overline{W}}$ ），控制逻辑产生存储器读（ $\overline{MEMR}$ ），从而控制数据传送的方向。

4、控制逻辑

在所有DMA周期内，控制逻辑通过产生相应的控制信号和16位要存取的存储单元地址来控制DMA的操作步骤。初始化时，通过对方式设置寄存器编程，使控制逻辑可以对各个通道的操作进行控制。

三、8257—DMAC引脚

图3—52是8257的引脚图。

1、 D_0-D_7 ：数据总线——双向三态线。当CPU对8257编程时，写入DMA地址寄存器，终点计数寄存器和方式设置寄存器的8位数据，通过 D_0-D_7 引脚从数据总线上接收。当CPU读DMA的地址寄存器，终点计数寄存器或状态寄存器时，数据通过这些引脚送到CPU数据总线上。在DMA周期（8257控制总线）期间，8257将通过数据总线把存储器地址的低8位（来自DMA某通道地址寄存器）送到8212锁存。这些地址位是在DMA周期的一开始传送的在DMA周期的其余时间内， D_0-D_7 引脚处于三态，以便处理存储器数据的传送。在DMA传送期间，数据传送从不通过这些引脚。

2、 $\overline{I/O\overline{R}}$ ： $\overline{I/O\overline{R}}$ 读——低电平有效，双向三态线。CPU控制总线时，这个引脚为输入。它有效时，允许8位状态寄存器，或16位DMA地址寄存器及终点计数寄存器的高8位/低8位字节被读出。8257控制总线时， $\overline{I/O\overline{R}}$ 为输出控制引脚。在DMA写周期它有效时，允许从外设读出数据。

3、 $\overline{I/O\overline{W}}$ ： $\overline{I/O\overline{W}}$ 写——低电平有效，双向三态线。与 $\overline{I/O\overline{R}}$ 引脚相似，8257处于“从”地位时，这个引脚为输入。它有效时，允许将数据总线的内容装入8位方式设置寄存器或16位DMA地址寄存器及终点计数寄存器的高8位/低8位字节。8257控制总线处于“主”地位时， $\overline{I/O\overline{W}}$ 为输出引脚，在DMA读周期它有效时，允许将数据写入外设。

4、CLK：时钟——输入，通常接到8224时钟发生器的 $\phi 2$ TTL输出引脚，或8085A CLK输出引脚。

5、RESET：复位——高电平有效，异步输入。这个引脚通常接到8224或8085A的RESET引脚。当这个信号有效时，方式寄存器被清除并清除所有可编程序寄存器的内容，所有的控制线都为三态，从而禁止所有DMA通道操作。复位之后，必须重新初始化，8257才能工作。

6、 A_0-A_3 ：地址线——地址线的这最低4位是双向线。CPU控制总线时，它们是输入线，选择CPU要读或要进行编程的某一个寄存器。8257控制总线时，它们是输出线，构成由8257所产生的16位存储器地址的最低4位。

7、 $\overline{CS}$ ：芯片选择——低电平有效，输入。在CPU控制总线对8257进行读或编程时，

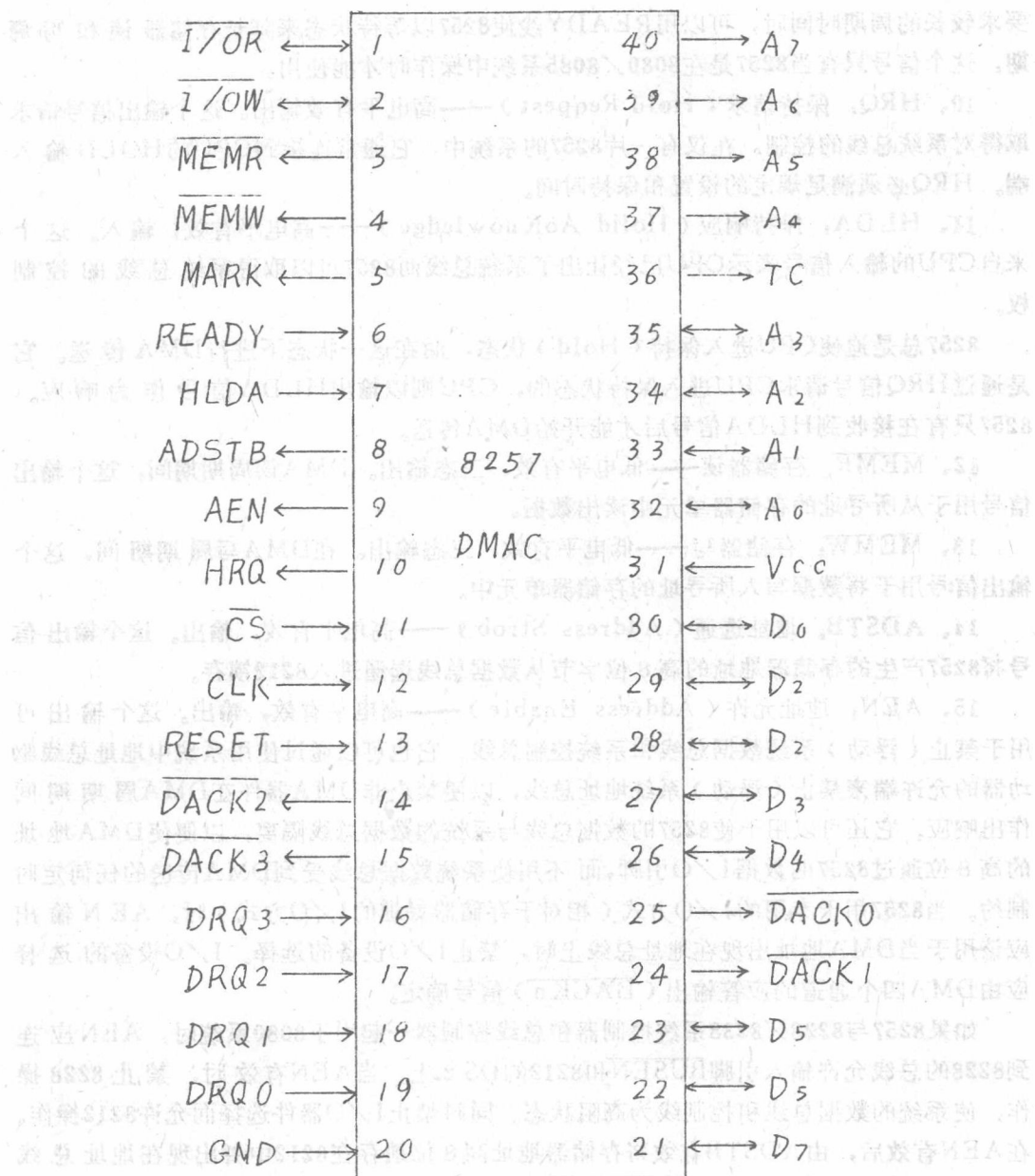


图 3—52 8257—DMAC引脚图

$\overline{CS}$ 有效将允许 $\overline{I/OR}$ 和 $\overline{I/OW}$ 输入，该芯片才能够工作。 $\overline{CS}$ 必须由地址总线 $A_7 \sim A_1$ 译码产生。当8257控制总线时， $\overline{CS}$ 输入自动被禁止，以防止在进行DMA操作期间，芯片自己选择自己。

8、 $A_4 \sim A_7$ ：地址线——三态输出。在所有DMA周期期间，这4条地址线构成了由8257产生的16位存储器地址的第4～第7位。

9、READY：准备就绪——高电平有效，异步输入。如果所选择的存储器或外设

要求较长的周期时间时,可以用READY线使8257以等待状态来延长存储器读和写周期。这个信号只有当8257是在8080/8085系统中操作时才能使用。

10、HRQ:保持请求(Hold Request)——高电平有效输出。这个输出信号请求取得对系统总线的控制。在仅有一片8257的系统中,它通常连接到CPU的HOLD输入端。HRQ必须满足规定的设置和保持时间。

11、HLDA:保持响应(Hold Acknowledge)——高电平有效,输入。这个来自CPU的输入信号表示CPU已经让出了系统总线而8257可以取得系统总线的控制权。

8257总是迫使CPU进入保持(Hold)状态,而在这一状态下进行DMA传送。它是通过HRQ信号请求CPU进入保持状态的,CPU则以输出HLDA信号作为响应。8257只有在接收到HLDA信号后才能开始DMA传送。

12、MEMR:存储器读——低电平有效,三态输出。DMA读周期期间,这个输出信号用于从所寻址的存储器单元中读出数据。

13、MEMW:存储器写——低电平有效,三态输出。在DMA写周期期间,这个输出信号用于将数据写入所寻址的存储器单元中。

14、ADSTB:地址选通(Address Strob)——高电平有效,输出。这个输出信号将8257产生的存储器地址的高8位字节从数据总线选通进入8212锁存。

15、AEN:地址允许(Address Enable)——高电平有效,输出。这个输出可用于禁止(浮动)系统数据总线和系统控制总线。它也可以通过使用系统中地址总线驱动器的允许端来禁止(浮动)系统地址总线,以便禁止非DMA器件在DMA周期期间作出响应。它还可以用于使8257的数据总线与系统的数据总线隔离,以便使DMA地址的高8位通过8257的数据I/O引脚,而不用使系统数据总线受到DMA传送的任何定时制约。当8257用于专用的I/O方式(相对于存储器寻址的I/O方式)时,AEN输出应该用于当DMA地址出现在地址总线上时,禁止I/O设备的选择。I/O设备的选择应由DMA四个通道的应答输出(DACK_n)信号确定。

如果8257与8228/8238系统控制器和总线控制器一起用于8080系统时,AEN应连接到8228的总线允许输入引脚BUSEN和8212的DS2上。当AEN有效时,禁止8228操作,使系统的数据总线和控制线为高阻状态。同时禁止I/O器件选择而允许8212操作。在AEN有效后,由ADSTB有效将存储器地址高8位锁存在8212中并出现在地址总线上,如图3-53所示。

16、TC:终点计数(Terminal Count)——高电平有效,输出。这个输出用于通知外部设备,现在的DMA周期已是这个数据块的最后一个周期。如果方式设置寄存器的TC停止位为置位,所选择的通道在这个周期结束时将自动被禁止。当所选择通道的终点计数寄存器的14位计数值递减到零时,TC输出有效。前面已经说过,当需要的DMA周期数为N(字节数为N)时,终点计数寄存器的低14位装入的计数值应当为N-1。TC信号可由外部设备使用进行数据块传送结束操作。例如在字组方式中,外设可在接收到TC信号时使DMA请求DRQ无效而结束DMA传送。

17、MARK:模128标记(Modulo 128 Mark)——高电平有效,输出。这个输

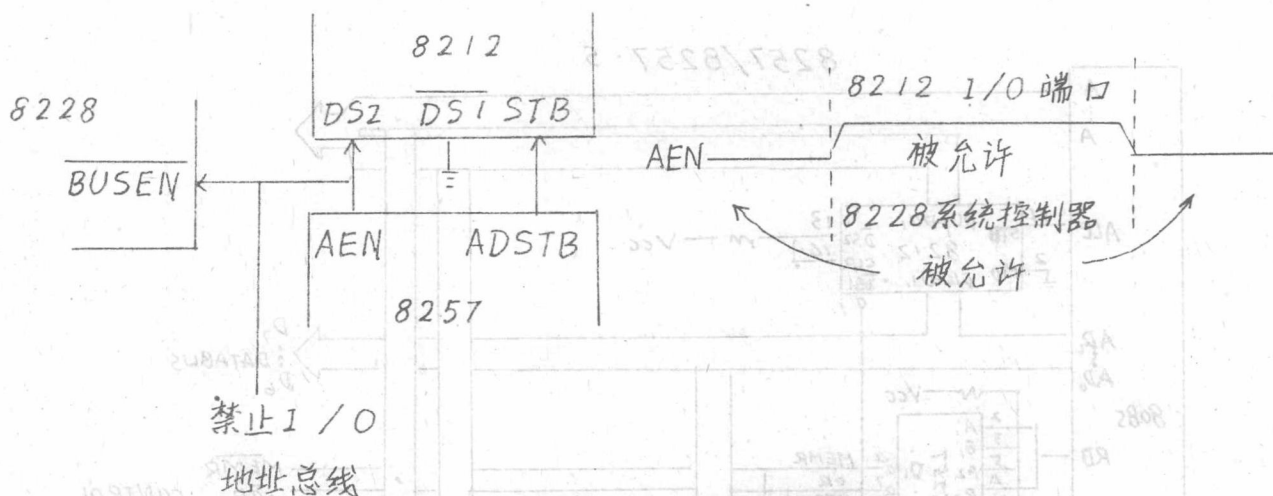


图 3—53 8257 AEN信号的使用

出用于通知所选择的外部设备，现在的DMA周期是自前次的MARK输出以后的第128个周期，即传送的第128个字节。MARK总是在距数据块结束每间隔128个周期时产生。只有在DMA周期总数（终止计数寄存器装入N—1）能够被128整除时，MARK才在距数据块开始的每间隔128个周期时产生。MARK信号可由外设使用记录已传送的字节数。这个功能在Z80—DMA中是以256个字节为间隔输出脉冲，而MC6844—DMAC没有这个功能。

18、DRQ 0—DRQ 3：DMA请求（DMA Request）——高电平有效，输入。外部设备使用这些异步的通道请求输入信号来获得DMA周期。在没有选择旋转优先权方式时，DRQ 0的优先权最高，DRQ 3最低。外设设备以请求线升高作为请求，并保持高直到DMA响应为止。对于执行多个DMA周期（字组方式）的情况，请求线应保持高直至最后一个周期的DMA响应信号到来。

19、DACK 0—DACK 3：DMA响应（DMA Acknowledge）——低电平有效，输出。这些输出信号用于通知与通道相连接的外部设备已被选中获得DMA周期。DACK输出可以作为请求服务的外部设备的“芯片选择”信号。DACK线在每传送一个字节时总是变为有效（低）和无效（高）一次，即使在传送一个字节组数据时也是这样。

8257—DMAC与8085的连接如图 3—54所示。

四、8257—DMAC寄存器选择

每个8257有四对“通道寄存器”，每一对由一个16位DMA地址寄存器和一个16位终点计数寄存器组成，每个通道各一对。8257还有两个“通用寄存器”，一个8位方式设置寄存器和一个8位状态寄存器。这样，一片8257共有10个可编程和使用的内部寄存器，两个8位和八个16位寄存器。当CPU执行一个写或读指令对8257寻址时，8257内部相应的寄存器被写入或读出。8228系统控制器和总线驱动器可产生读或写控制信号。地址

总线高12位 (A_4-A_{15}) 的全部或一部分通常经译码产生芯片选择信号 $\overline{CS}$ 送到 8257。
 I/O写信号 (或存储器对应的I/O方式用存储器写信号) 确定对所寻址的寄存器进行编程。而I/O读信号 (或存储器读信号) 确定对所寻址的寄存器读出。地址位 A_3 用于确定或者“通道寄存器组” ($A_3 = 0$) 或者“通用寄存器组” ($A_3 = 1$) 的方式设置 (只编程) 及状态 (只读) 寄存器将被存取, 如表 3—16所示。

表 3—16 8257寄存器组选则

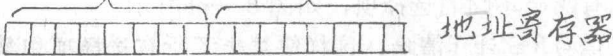
寄存器组	操 作	CS	I/O $\overline{W}$	I/O $\overline{R}$	A_3
通 道	对某通道寄存器的一半编程	0	0	1	0
寄存器组	读某通道寄存器的一半	0	1	0	0
通 用	方式设置寄存器编程	0	0	1	1
寄存器组	读 状 态 寄 存 器	0	1	0	1

每个16位的寄存器都需要两次读或写操作。所有通道寄存器 (16位) 都是可读/写的。方式设置寄存器是只写的, 而状态寄存器只读。

一个8257共占10个地址单元。地址的最低位 A_0-A_2 三位用于确定要被存取的具体寄存器。当选择“通用寄存器” (方式设置/状态寄存器) 时, A_0-A_2 都为 0, $A_3 = 1$ 。当寻址通道寄存器时, $A_0 = 0$ 选择DMA地址寄存器; $A_0 = 1$ 选择终点计数寄存器。 A_2 和 A_1 用于确定四个通道之一。

由于“通道寄存器”都是16位的, 需要两次读/写操作来读或写一个完整的寄存器。例如, 向通道 2 写入一个16位起始地址, 编程如下:

```
LD    A, DMALO    ; 取起始地址低字节
OUT   (DMA2), A   ; 写入DMA通道2
LD    A, DMAHI    ; 取起始地址高字节
OUT   (DMA2), A   ; 写入DMA通道2
```



即使要传送的数据块小于256个字节, 也要先将小于256的字节数写入终点计数寄存器的低8位, 然后将 0 0 H写入高8位, 否则就会出错。

8257内部有一个先/后 (First/Last) 触发器 F/L, 在每次通道读或写完成时改变一次状态。F/L触发器用于确定读或写16位寄存器的高或低8位字节。F/L触发器由RESET信号或任意时刻写方式设置寄存器时复位。在存取“通道寄存器”时, 若要保证正确地同步, 所有的通道命令指令操作应该成对产生, 并首先存取寄存器的低

8 位字节。在 $\overline{I/O\overline{R}}$ 或 $\overline{I/O\overline{W}}$ 有效期间，确实不要让 $\overline{CS}$ 波动，因为这将造成 F/L 触发器处于错误状态。如果在系统中使用中断，在任何成对编程操作之前应禁止中断，以避免一个中断使成对的操作分开。这种分开操作的结果也会使 F/L 触发器处于错误状态。当其它 DMA 通道是由中断程序编程时，这触发器处于错误状态。当其它 DMA 通道是由中断程序编程时，这个问题就特别明显。

表 3—17 是 8257 的寄存器选择表。8257 寄存器寻址的简单规律是：地址低 4 位小于 8 的是“通道寄存器”地址；等于 8 为“通用寄存器”地址。“通用寄存器”是用 $\overline{I/O\overline{W}}$ 来区别状态和方式设置寄存器的。“通道寄存器”中，地址双数是 DMA 地址寄存器；地址单数是终点计数寄存器。通道号 $\times 2$ 和通道号 $\times 2 + 1$ 为该通道的两个地址。16 位寄存器必须先读/写低 8 位接着读/写高 8 位，其低 8 位和高 8 位在内部是由 F/L 触发器区别的。

C_0-C_{15} ：终点计数值 ($N-1$)

Rd 和 Wr ：DMA 校验 (0 0)，写 (0 1) 或读 (1 0) 周期选择

AL ：自动装入

TCS ：TC 停止

EW ：扩展写

P_0 ：旋转优先权

EN_3-EN_0 ：通道允许

UP ：修改标志

TC_3-TC_0 ：终点计数状态位

五、8257—DMAC 编程

1、8257 初始化编程的步骤

8257 的初始化编程是各种典型的 DMAC 中最简单的，其步骤为：

- ① 将存储器的起始地址写入 DMA 地址寄存器。
- ② 将要传送的字节数写入终点计数寄存器，并同时确定 DMA 传送方向。
- ③ 写入方式设置寄存器。

写入方式设置寄存器作为初始化的最后一步，只在允许开始 DMA 传送时写入。

2、方式设置寄存器

向方式设置寄存器写入一个方式设置控制字可以分别允许四个 DMA 通道操作。同时，对 8257 的操作可以有四个不同的选择项，如图 8—56 所示。

该寄存器可由复位信号 $\overline{RESET}$ 清除，这样便禁止了所有选择项和禁止所有通道操作，在加电时可避免造成总线冲突。方式设置寄存器通常是在各通道的 DMA 地址寄存器和终点计数寄存器已包含有效数值时，才能允许一个通道的操作。否则，从外设来的一个偶然的 DMA 请求 ($\overline{DRQ_n}$) 信号将使 DMA 周期开始而破坏存储器的数据。

方式设置寄存器可能允许的四种选择项如下：

- ① 旋转优先权 (Rotating Priority) 位 4
- 在旋转优先权方式 ($D_4 = 1$) 中，各通道的优先权是按通道号递增的顺序循环的。

表 3—17 8257 寄存器选择表

寄 存 器	字 节	地 址 输 入				HEX	F/L	双 向 数 据 总 线*							
		A ₃	A ₂	A ₁	A ₀			D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
通道0DMA地址	低8位 高8位	0	0	0	0	0	0 1	A ₇ A ₁₅	A ₆ A ₁₄	A ₅ A ₁₃	A ₄ A ₁₂	A ₃ A ₁₁	A ₂ A ₁₀	A ₁ A ₉	A ₀ A ₈
通道0终点计数	低8位 高8位	0	0	0	1	1	0 1	C ₇ Rd	C ₆ Wr	C ₅ C ₁₃	C ₄ C ₁₂	C ₃ C ₁₁	C ₂ C ₁₀	C ₁ C ₉	C ₀ C ₈
通道1DMA地址	低8位 高8位	0	0	1	0	2	0 1	与	通	道	0	相	同	同	同
通道1终点计数	低8位 高8位	0	0	1	1	3	0 1	与	通	道	0	相	同	同	同
通道2DMA地址	低8位 高8位	0	1	0	0	4	0 1	与	通	道	0	相	同	同	同
通道2终点计数	低8位 高8位	0	1	0	1	5	0 1	与	通	道	0	相	同	同	同
通道3DMA地址	低8位 高8位	0	1	1	0	6	0 1	与	通	道	0	相	同	同	同
通道3终点计数	低8位 高8位	0	1	1	1	7	0 1	与	通	道	0	相	同	同	同
方式设置(只 写)	—	1	0	0	0	8	0	AL	TOS	EW	RP	EN3	EN2	EN1	EN0
状 态(只 读)	—	1	0	0	0	8	0	0	0	0	UP	TC3	TC2	TC1	TC0

*A₀—A₁₅: DMA 起始地址

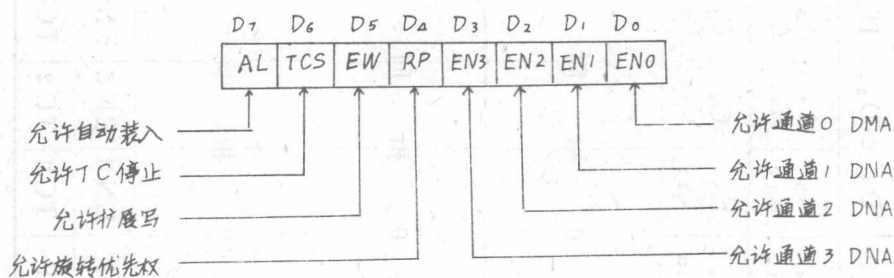


图 3—55 8257方式设置寄存器

在每个DMA周期（不是每个DMA请求）之后，各个通道的优先权都将发生变化：刚刚服务过的通道其优先权变为最低，而将最高优先权按顺序让给下一个通道，如图 3—56和表 3—18所示。

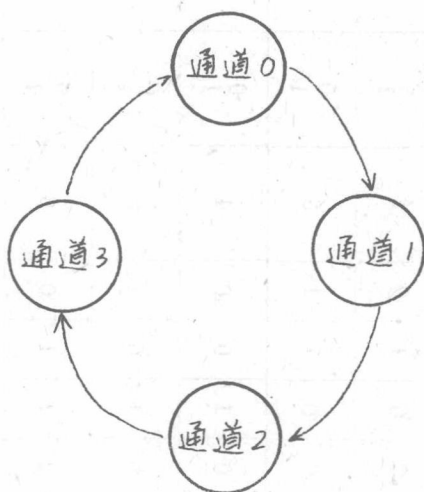


图 3—56 8257旋转优先权方式

表 4—18 8257旋转优先权的变化

刚服务过的通道→	CH - 0	CH - 1	CH - 2	CH - 3
最 高 优 先 权	CH - 1	CH - 2	CH - 3	CH - 0
	CH - 2	CH - 3	CH - 0	CH - 1
	CH - 3	CH - 0	CH - 1	CH - 2
最 低 优 先 权	CH - 0	CH - 1	CH - 2	CH - 3

如果旋转优先权位没有置位 ($D_4 = 0$), 则每个DMA通道的优先权都是固定的, 在固定优先权方式, 通道0的优先权最高, 通道1次之, 通道3的优先权最低。

旋转优先权将防止任何一个通道独占DMA方式。如果一个以上的通道被允许并请求服务的话, 连续的DMA周期将为不同的通道服务 (这一点与MC6844有所不同)。所有DMA操作最初都指定从通道0开始具有最高优先权。

②扩展写 (Extended Write) 位5

如果扩展写位置位 ($D_5 = 1$), $\overline{\text{MEMW}}$ 和 $\overline{\text{I/O}}$ 两个写信号的宽度都因在DMA周期中提前有效而扩展。微型计算机系统异步处理的数据传送允许使用各种类型存取时间不同的存储器和I/O设备。如果一个设备不能在一段确定的时间内进行存取, 它将以“未准备就绪”信号通知8257, 使8257在其内部时序中插入一个或多个等待状态。有些设备的存取相当快, 不需要使用等待状态。但是如果它们用准备就绪Ready信号响应 $\overline{\text{I/O}}$ 或 $\overline{\text{MEMW}}$ 信号 (通常在传送序列的后期产生) 的前沿, 将使8257进入一个等待状态, 因为8257并没有及时接收到READY信号。对于使用这种设备的系统, 扩展选择为I/O和存储器写信号提供了另外一种定时, 允许外设提前回答READY信号, 而防止在8257内产生不必要的等待状态, 这样可提高系统的吞吐量。

③终点计数停止 (TC Stop) 位6

该位置位 ($D_6 = 1$) 时, 在终点计数TC输出有效后, 通道被禁止 (即该通道的允许位被复位), 从而自动阻止该通道继续进行DMA操作。为继续或开始另一次DMA操作, 必须对该通道的允许位重新编程。如果TC停止位没有置位 ($D_6 = 0$), 则TC输出并不影响该通道的允许位。在这种情况下, 通常由外部停止DMA请求而结束DMA操作。

④自动装入 (Atto Load) 位7

自动装入方式使通道2可用于重复数据块或数据块链接操作, 从而在数据块传送之间, 不需要软件马上干预。通道2寄存器作为第一个数据块初始化, 而通道3用于保存重新初始化数据块的参数 (DMA起始地址, 终点计数值和DMA传送方式)。在通道2完成第一个数据块DMA周期 (即TC输出有效) 后, 保存在通道3寄存器的参数在一个修改周期期间传送给通道2相应的寄存器。在自动装入方式下, 通道3的作用相当通道2的暂存器。注意, 当自动装入位置位时, 前边所说的终点计数停止特性对通道2将不起作用。

如果自动装入位置位 ($D_7 = 1$), 在通道2编程时, 通道2的初始化参数自动复制到通道3中, 这就允许只对通道2一个通道编程来实现数据块重复操作。这种数据块重复传送可应用在诸如CRT刷新等操作中。如果通道2编程之后再对通道3编程, 则通道2和通道3仍然可以用不同的参数装入, 这就可以实现两个不同的数据块链接传送操作。值得注意的是, 在自动装入方式下, 如果通道3的允许位置位 ($D_3 = 1$), 用户仍可使用通道3。但是这个通道的使用将改变在修改周期内要自动装入通道2的参数。在一般情况下, 通道2和通道3或者是作为两个独立的通道分别使用 (不使用自动装入功能), 或者只用通道2的自动装入功能, 禁止通道3, 而使通道3作为通道2的缓冲器使用。作为链接操作而使用的自动装入特性, 有必要在每个修改周期结束时用下个数据块传送的新的参数重新装入通道3寄存器。自动装入定时如图3—57所示。

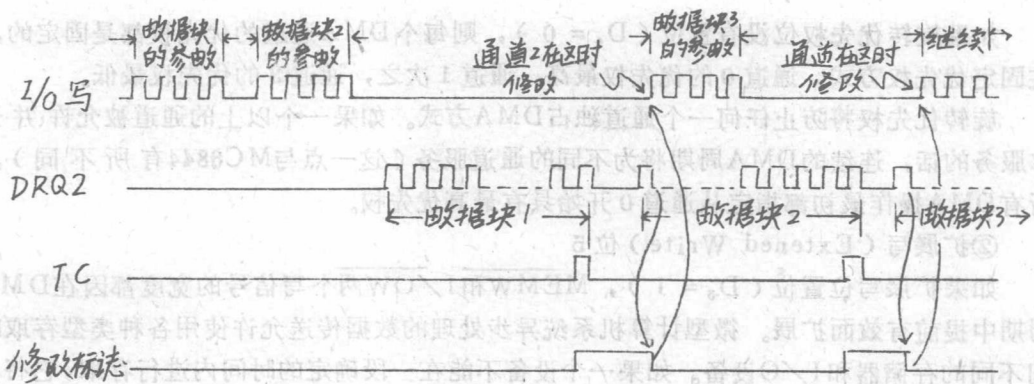


图 3—57 8257 自动装入定时

每次8257进入修改周期，状态寄存器的修改标志置位。同时，通道3的参数传送到通道2，而通道3的内容并不破坏。通道2实际的重新初始化发生在TC周期以后下一个通道2 DMA周期开始时，这将是通道2新的数据块传送的第一个DMA周期。在这个DMA周期的末尾，修改标志被清除。对于链接操作，状态寄存器的修改标志可由CPU监测，以便确定通道2重新初始化处理何时完成，从而可以将下一个数据块的参数安全地写入通道3中，而不至于在尚未修改时由于写入新的参数而将上一个数据块的参数冲掉。

3、状态寄存器

8位状态寄存器如图3—58所示。其低4位为各通道终点计数(TC)的状态位，分别用来指示每个通道是否达到终点计数状态。另有一位(D₄)为通道2自动装入方式时的修改标志位，已在前边讨论过。状态寄存器的最高3位不用。

当某个通道终点计数(TC)输出有效时，其TC状态位置位。TC状态保持置位状态直到CPU读状态寄存器或8257复位时才被清除。但修改标志不受状态寄存器读操作的影响。它可由8257复位或通过进入非自动装入方式(即方式设置寄存器的自动装入位清0时)而复位，它也可以在修改周期完成时自动清除。修改标志的作用是为了防止

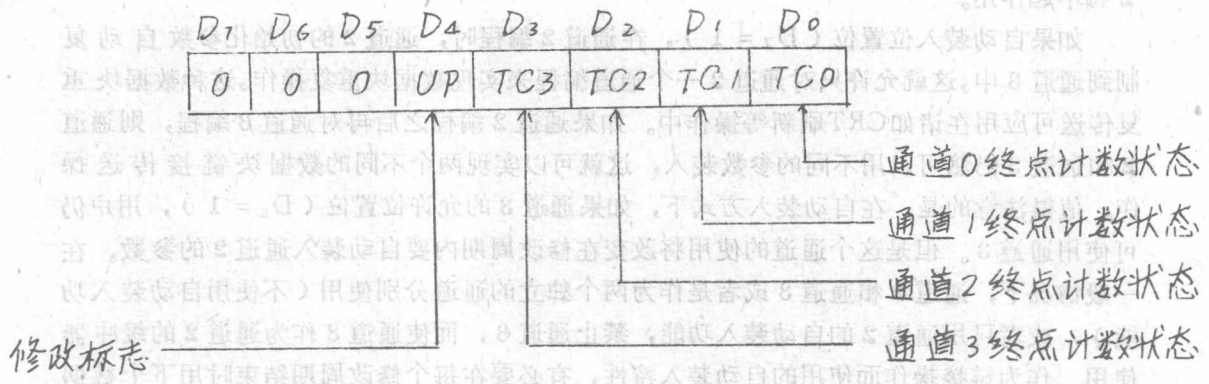


图 3—58 8257 状态寄存器

CPU在通道3的参数自动装入通道2以前,因不小心复盖了通道3寄存器中的起始地址或终点计数值而跳过了一个数据块没有传送。

六、8257—DMAC的操作和使用

1、8257—DMAC操作

8257的操作可分为几个不同的状态,见图3—59。在8257开始DMA周期前,一直处于空转状态S1,不断查询DMA请求输入线DRQ_n。当任何一个已允许的DMA通道的DMA请求输入有效(DRQ=1)时,8257向CPU输出保持请求信号HRQ,进入S0状态,在S0状态,8257等待CPU发回的保持响应信号HLDA。在得到CPU响应并判定了请求DMA的通道的优先权后,8257开始一个DMA周期。整个DMA周期至少要经过四个内部状态S1~S4。如果存储器或I/O设备的存取时间太慢,不能在指定的时间内给8257回答准备就绪信号READY,还要在S₃与S₄之间插入一个或多个等待状态SW,对于DMA检验周期,不需要READY响应信号。

2、8257DMA操作定时

8257 DMA操作定时

在8257获得系统总线控制权开始DMA周期前至少要有两个时钟周期。8257在一个时钟脉冲(S1)的下降沿对DRQ线采样(图中①处)如果检测到一个或多个DRQ线为有效(高),8257将在下一个时钟脉冲的上升沿向CPU输出保持请求信号HRQ(图中②)。CPU在接到HRQ请求后,进入保持状态,并在下一个时钟脉冲(S0)的下降沿之前向8257发出HLDA信号作为响应(图中③)。在8257接收到HLDA信号的同一个脉冲下降沿,它输出地址允许信号AEN(图中④)使系统总线处于三态,这标志着8257获得了系统控制权和DMA周期的开始。

在DMA周期开始后,DMA传送所需要的存储器地址必须出现在地址总线上。存储器地址的低8位在DMA周期状态1(S1)时钟周期的上升沿通过8257的A₀—A₇引脚输出(图中⑦)这个地址的低8位字节是直接从所选中的DMA通道的地址寄存器输出到地址总线上的。这个地址将稳定并一直保持到一个DMA周期S1的上升沿改变(图中⑥)。存储器地址的高8位字节是通过8257的数据引脚D₀—D₇输出的。在地址低8位出现在地址总线的同时,地址高8位也出现在数据总线上(图中⑧),并且就在这同一时刻(S1的上升沿),8257输出地址选通脉冲ADDSTB有效(图中⑩),将数据总线上的地址高8位选通进入8212锁存。虽然定时图上没有表示出来,但在地址选通脉冲ADDSTB有效之后,地址的高8位字节就已经从8212锁存器输出到地址总线高8位上了。至此,存储器所需要的16位地址便在地址总线上稳定下来。地址的低8位和高8位(先在数据总线上)以及地址选通脉冲ADDSTB总是同时在DMA周期的第一个状态S1的上升沿有效。在第2个状态S2的上升沿,ADDSTB变为无效(图中⑪),8257的数据引脚三态(图中⑫)。因此,地址选通脉冲ADDSTB的宽度和8257的D₀—D₇引脚保持地址高8位字节的时间都不过一个时钟周期。然后,8257的数据引脚D₀—D₇为三态,数据总线让位于存储器和I/O设备之间的DMA传送。

在上升沿产生存储器地址的S1状态,其下降沿使8257的DMA响应输出信号DACK_n