考点一：掌握 8086/8088CPU 的功能构成及流水线技术，理解流水线管理规则。

考点二：掌握 8086/8088CPU 寄存器的组成及其应用。

考点三：理解 8086/8088CPU 的内存分配，掌握实地址模式下的存储器地址变换方法。

考点四：掌握 8086/8088CPU 的引脚构成，理解其引脚复用的特性。

2.1　8086/8088 CPU 的功能构成

1、8086/8088 是 Inter 公司的第三代位处理器芯片。

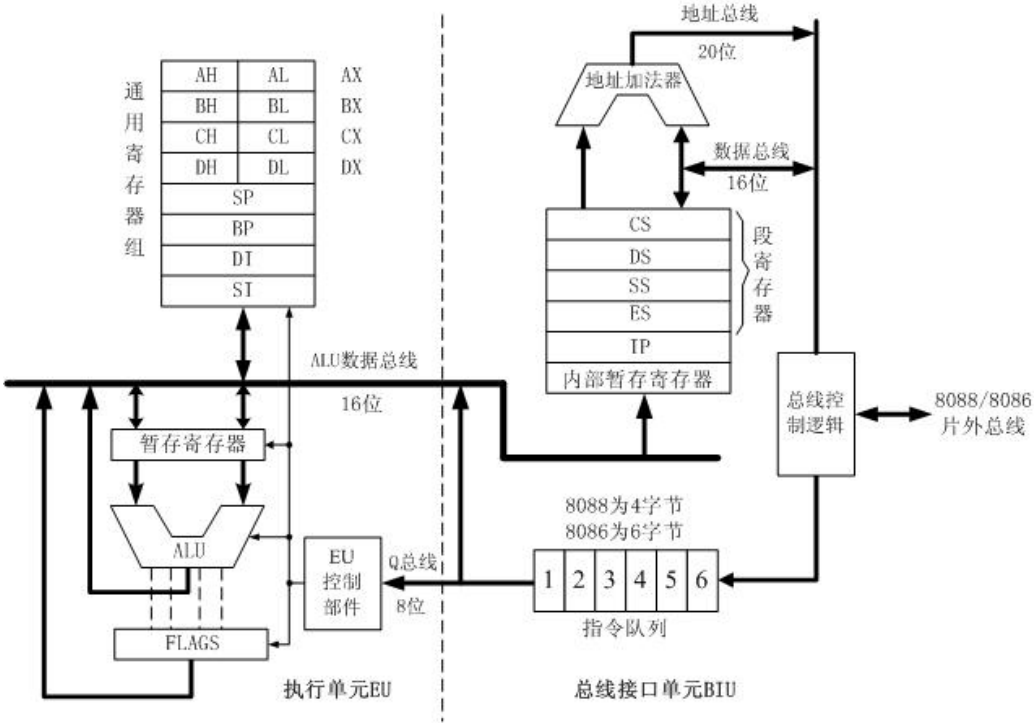
其特点：

（1）具有 20 条地址总线，直接寻址能力为 1MB。

（2）8086 有 16 条数据总线，为 16 位微处理器；8088 有 8 条数据总线，为准 16 位微处理器。

（3）片内总线和 ALU 均为 16 位，可进行 8 位和 16 位操作。

（4）8086/8088 片内均由两个独立的裸机单元组成，即总线接口单元（BIU）和执行单元（EU）



2、总线接口单元 BIU

（1）组成部件

① 4 个 16 位段寄存器（CS、DS、SS、ES）；

② 16 位指令偏移地址寄存器（IP）；

③ 指令队列寄存器（8086CPU:6 字节；8088CPU：4 字节）；

④ 形成 20 位物理地址的加法器

⑤ 与 EU 通讯的内部寄存器；

⑥ 总线控制逻辑；

（2）功能：实现 CPU 与存储器或 I/O 口之间的数据传送

① 自动按 CS 值和 IP 值组成 20 位实际地址的存储器中去取指令，一次取两个字节指令存放到指令队列中。

② 由 EU 从指令队列中取指令，并根据 EU 请求，BIU 将 20 位操作地址传送给存储器；

③ 取来操作数经总线控制逻辑传送到内部 EU 数据总线，由 EU 完成内部操作；

④ 操作结果：若 EU 提出请求，则由 BIU 负责产生 20 位实际目的地址，将结果存入存储器里；

3、执行单元 EU

（1）组成部分：

① 16 位算术逻辑单元（ALU）；

② 16 位状态标志寄存器 FLAG；

③ 8 个 16 位通用寄存器组（AX,BX,CX,DX,SP,BP,SI,DI）；

④ 16 位数据暂存器；

⑤ EU 控制电路；

（2）功能：

① 从 BIU 指令队列中取指令；

② 由 EU 控制电路对指令进行译码分析，指出操作性质及对象；

③ 在 EU 中 计算出操作数的 16 位地址偏移量送给 BIU，由 BIU 的加法器形成 20 位绝对地址；

④ 将取来的操作数经系统数据总线送 ALU 进行制定的操作；

⑤ 运算结果经内部总线送到指定位置；

4、EU 和 BIU 单元执行过程中，应该满足的规则

（1）当指令队列寄存器中无指令时，EU 处于等待状态；

（2）当指令队列中存满指令，而 EU 又没有访问存储器或 I/O 端口的需要，则 BIU 进入空闲状态；

（3）当指令队列中有两个空闲字节，则 BIU 自动执行取指令的总线周期；

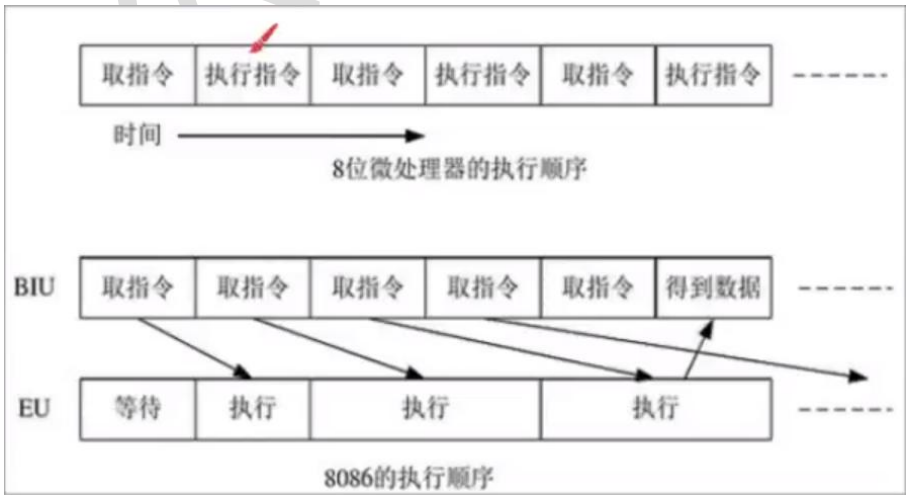
（4）在 EU 执行指令时，需要访问存储器或 I/O 端口，如果这时 BIU 正在取指令，则应等待 BIU 完成取指令周期，然后 BIU 进入存储器和 I/O 端口访问周期；

（5）在 EU 执行转移，子程序调用或返回等指令时，自动清除指令队列的内容。

2.2 8086/8088CPU 的流水线技术

1、流水线技术即指令执行的顺序

BIU 和 EU 分开，取指令和执行指令可以重叠，大大减少了等待区赤岭所需要的时间，提高 CPU 的利用率。



2、BIU 和 EU 的动作协调原则

BIU 和 EU 按以下流水线技术原则协调工作，共同完成所需要的任务。

（1）每当 8086 的指令队列中有两个空字节，BIU 就会自动把指令取到指令队列中。其取值的顺序是按指令在程序中出现的前后顺序。

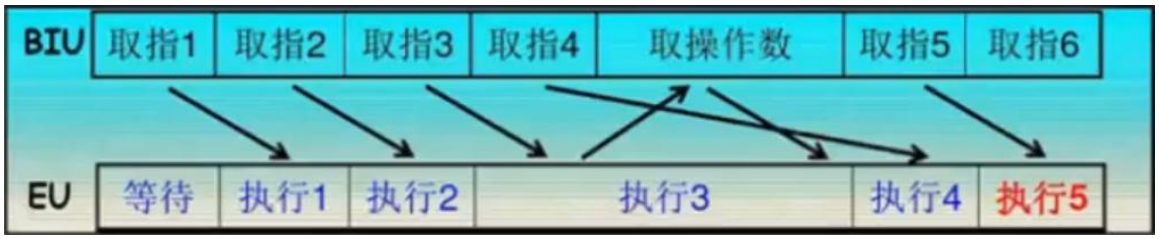
（2）每当 EU 准备执行一条指令时，它会从 BIU 部件的指令队列前部取出指令的代码，然后用几个时钟周期去执行指令。在执行指令的过程中，如果必须访问存储器或者 I/O 端口，那么 EU 就会请求 BIU，进入总线周期，完成访问内存或 I/O 端口的操作；如果此时 BIU 正好处于空闲状态，会立即响应 EU 的总线请求。如 BIU 正将某个指令字节取到指令队列中，则 BIU 将首先完成这个取指令的总线周期，然后再去响应 EU 发出的访问总线的请求。

（3）当指令队列已满，且 EU 又没有总线访问请求时，BIU 便进入空闲状态；

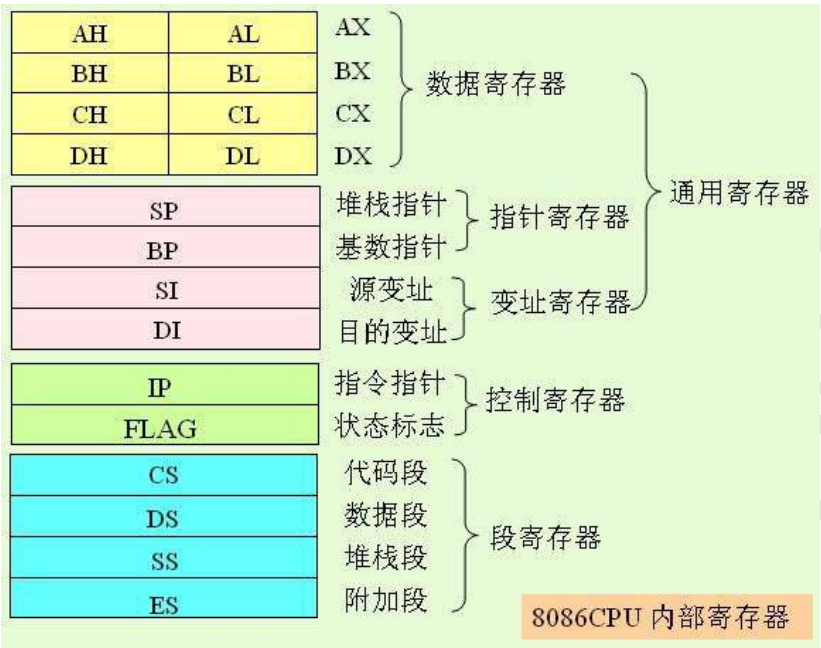
（4）在执行转移指令、调用指令和返回指令时，由于待执行指令的顺序发生了变化。则指令队列中已经装入的字节被自动消除，BIU 会接着往指令队列转入转向的另一程序段中的指令代码。

将 8086/8088CPU 分成两个独立的功能部件使二者能够并行工作，把取指令工作和分析指令、执行指令工作重叠进行，从而提高 CPU 的工作效率，加快指令的执行速度。

指令队列可以被看成是个特殊的 RAM，他的工作原理是“先进先出”，写入的指令只能存放在队列尾，读出的指令是队列头存放的指令。EU 和 BIU 之间就是通过指令队列联系起来，多数情况下，BIU 在不停的向队列写入指令，而 EU 每执行完一条指令后，就向队列读取下一条指令。二者的动作即独立，又协调。



2.3 8086/8088CPU 寄存器的组成及应用



1、通用寄存器

8086/8088 有 4 个 16 位的通用寄存器（AX、BX、CX、DX），可以存放 16 位的操作数，也可以分为 8 个 8 位的寄存器（AL,AH;BL,BH;CL,CH;DL,DH）来使用。

（1）AX 称为累加器，多用于存放中间运算结果。所以 I/O 指令都通用 AX 与接口传送信息，中间运算结果也多放于 AX 中；

（2）BX 称为基址寄存器，在间接寻址中用于存放基地址；

（3）CX 称为计数寄存器，用于再循环或串操作指令中存放计数值，即存放循环次数或重复次数。

（4）DX 称为数据寄存器，在间接寻址的 I/O 指令中存放 I/O 端口地址

2、指针寄存器

系统中有两个 16 位的指针寄存器 SP 和 BP

(1)SP 是堆栈指针寄存器，由它的堆栈段寄存器 SS 一起来确定堆栈在内存中的位置，其内容为栈顶的偏移地址；

(2)BP 是基数指针寄存器，通常用于存放基地址。

BP 和 BX 在应用上的区别：作为通用寄存器，二者都可以用于存放数据，作为基址寄存器，用 BX 表示所寻找的数据在数据段，用 BP 表示的数据在堆栈段。

3、变址寄存器

系统中有两个 16 位的变址寄存器 SI 和 DI，其中 SI 是源变址寄存器，DI 是目的变址寄存器，都用于指令的变址寻址方式。

4、控制寄存器

IP、标志寄存器是系统中的两个 16 位控制寄存器

（1）IP 是指令指针寄存器，用来控制 CPU 的指令执行顺序，它和代码寄存器 CS 一起可以确定当前所要取的指令的内存地址。顺序执行程序时，CPU 每取一个指令字节，IP 自动加 1.指向下一个要读取的字节；当 IP 单独改变时，会发生段内的程序转移；当 CS 和 IP 同时改变时，会产生段间的程序转移。

（2）标志寄存器的内容被称为处理器状态字 PSW，用来存放 8086CPU 在工作过程中的状态。

5、段寄存器

系统中有 4 个 16 位段寄存器，即代码段寄存器 CS、数据段寄存器 DS、堆栈段寄存器 SS 和附加段寄存器 ES。

这些段寄存器的内容与有效地址编译量一起，可确定内存的物理地址。通常 CS 划定并控制程序区，DS 和 ES 控制数据区，SS 控制堆栈区。

6、标志寄存器

8086/8088 内部标志寄存器的内容，又称为处理器状态字（PSW），共有 9 个标志位。可分为两类：一类为状态标志，一类为控制标志。

状态标志表示前一步操作（加、减等）执行以后，ALU 所处的状态，后续操作可以根据这些状态标志进行判断，实现转移；

控制标志则可以通过指令人为设置，用以对某一种特定的功能起控制作用（如中断屏蔽等），反映了对微机系统工作方式的可控制性。

6 个状态标志：表示处理器当前运行状态

CF ： 进位标志，做加法时最高位出现进位或做减法是最高位出现借位，CF=1

PF ： 奇偶标志，低 8 为中 1 的个数为偶数个，PF=1

AF ： 辅助进位，低 4 位向前有进（借）位，AF=1

ZF ： 零标志位，结果为 0，ZF=1

SF ： 符号标志，与最高位一致

OF ： 溢出标志，双高位判别法

3 个控制标志：控制处理器某一特定功能

IF ： 可屏蔽中段允许标志，IF=1 表示允许

TF ： 陷阱标志（单步执行）

DF ： 方向标志，DF = 0 地址增量变化，DF = 1 地址减量变化

2.4 8086/8088 的内存分配

1、存储单元的地址和内容

（1）存储器按字节编址

①一个字节单元占用一个地址码

②一个字节单元可存放 8 位数据。如字节单元（0002）=34H

（2）用 MOV 指令访问字节单元。

如 MOV AL，[0002H];（AL）<--（0002H）

（3）一个字占用相继的两个单元。低字节在前，高字节在后。

（4）同一地址即可以看作字节单元地址，又可看作字单元地址，需要根据使用情况确定。

①字节单元：（0002H）=34H

字节访问：MOV AL,[0002H]

②字单元：（0002H）=1234H

字访问：MOV BX,[0002H]

（5）字单元又分为两种情况

①对准字：字的起始地址是偶地址字单元。如（0002H）=1234H

②非对准字：字的起始地址是奇地址。如（0003H）=7812H

（6）规则存放和非规则存放

①只有 8086 存储器才有规则存放和非规则存放的概念。

②多字节数据从偶数地址开始存放，称为规则存放；从奇数地址开始存放，称为非规则存放。

③字节数据只占 1 个存储单元，不存在规则存放与非规则存放。

2、地址码宽度与存储容量

（1）地址码宽度为 n 位，则存储器容量为 2^n 个单元。如

① 地址码宽度为 10 位（A9~A0），则存储器容量为：2^10 个单元=1024 个单元=1K 个单元。

② 16 位地址吗，访问内存 64K，0000H~FFFFH

③ 20 位地址码，访问内存 1M，0000H~FFFFH

（2）8086/8088 系统：地址总线 20 条：A19~A0；物理地址范围：00000H~FFFFFH；存储容量：1MB

3、8086/8088 存储器分段使用

（1）为什么要存储器地址分段使用？

如果要把 20000H 单元的字节数据送入 AL 中，指令似乎可以写成 MOV AL,[20000H];8086 有 20 位地址码，但 CPU 中的总线是 16 位，无法直接处理 20 位地址码，即指令中的地址信息不允许超过 16 位宽度。

（2）解决问题的思路：把 20 位物理地址设法写成逻辑地址（即两个 16 位地址码）。

① 物理地址计算公式：物理地址=段地址 x16+偏移地址

② 引入逻辑地址后，指令中只能使用逻辑地址；CPU 会自动计算物理地址；物理地址直接送至 A19~A0 访问内存。

③ 内存单元的物理地址是唯一的，但逻辑地址不是唯一的。

④ 当段地址不变时，改变偏移地址，可访问 16KB 区域，该区域称为一个段。

⑤ 当段地址确定后，改变偏移地址，可访问不同的内存单元。

⑥ 一般情况下，访问一个连续的内存区域时，段地址值是不同的，故引入

段地址寄存器来存放段地址的值。

⑦ 8086/8088 中 4 个专用的段地址寄存器，用来分别存放 4 个段的段地址值。

a) DS：数据段寄存器，用来存放数据段地址值；

b) CS：代码段寄存器，用来存放代码段段地址值；

c) SS：堆栈段寄存器，用来存放堆栈段段地址值；

d) ES：附加段寄存器，用来存放附加段段地址值；

⑧ 段寄存器和其他寄存器组合指向不同的存储单元

⑨ 段的分配情况说明：

a) 操作系统会根据内存的使用情况为每个段分配地址

b) 每个段最大占用 64K 存储区，最小根据需要而定

c) 各段之间可以不连续，各段之间允许重叠

d) 段寄存器的初始化：系统会自动完成 CS 的初始化，程序员要在程序的首部初始化 DS,ES 和 SS

2.6 8086/8088CPU 的引脚构成

1、8086CPU 是 40 引脚双列直插式芯片，微处理器通过这些引脚可以和存储器、I/O 接口、外部控制管理部件，以及其他微处理器相互交换信息。

2、采用了分时复用地址/数据总线技术，减少了芯片的引脚。

3、最小模式，在系统中只有一个处理器，所有总线控制信号都是直接由该处理器产生，此时系统中的总线接口电路被减到最少。

4、最大模式，是相对最小模式而言的，系统中包含两个或两个以上的微处理器，其中一个是主处理器，其他的处理器是协处理器，协助主处理器工作。

5、8086/8088CPU 的最大或最小模式是由硬件决定的。是通过第 33 号引脚（MN/MX#）控制的。

6、8086CPU 的引脚按其作用可分为 5 类

（1）地址/数据总线 AD15~AD0（双向，三态）

（2）地址/状态线 A19/S6 ~ A16/S3（单向输出，三态）

（3）控制总线，包括

① 总线高字节允许/状态（34 引脚）

② 读控制信号线（32 引脚）

③ 准备就绪信号（22 引脚）

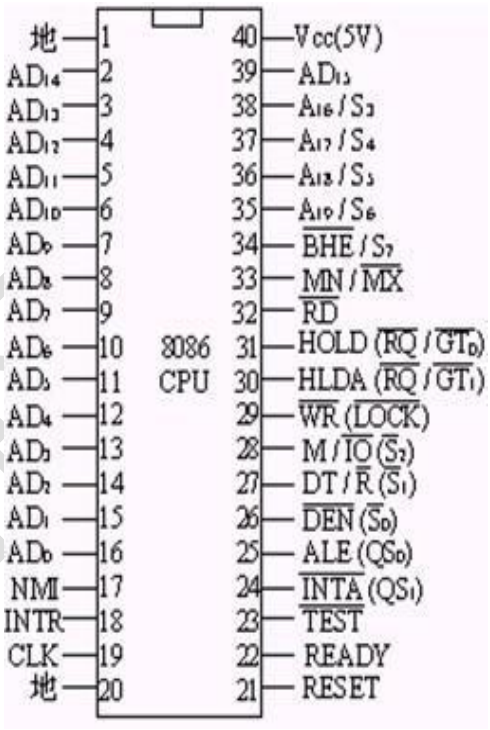
④ 测试信号（23 引脚）

⑤ 可屏蔽中段请求信号（24 引脚）

⑥ 非屏蔽中段请求信号（17 引脚）

⑦ 复位信号（21 引脚）

⑧ 系统时钟（19 引脚）



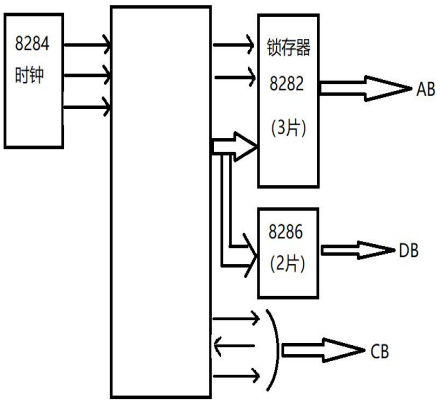
（4）电源线 Vcc（40 引脚）和地线 GND（1 引脚）

（5）其他控制线（最大最小模式控制字）（24 引脚~31 引脚）

（6）

7、最小模式下的专用引脚信号（MN/MX#为高电平）

（1）INTA#：可屏蔽中断响应信号，输出。CPU 通过信号对外设提出的可屏蔽中断请求做出响应。低电平表示 CPU 已经响应外设的中段请求，即将执行中断服务程序。



（2）ALE：地址锁存允许信号，输出。CPU 利用该

信号可以把 AD15~AD0 地址/数据、A19/S6~A16/S3 地址/状态线上的地址信息锁存在地址锁存器中。

（3）DEN#：数据允许信号，输出，三态。用作总线收发器的选通控制信号。有效时，表明 CPU 进行数据的读/写操作。

（4）DT/R#：数据发送/接收信号，输出，三态。用来控制数据传送的方向。高电平时，CPU 发送数据到存储器或 I/O 端口；低电平时，CPU 接收来自存储器或 I/O 端口的数据。

（5）M/IO#：存储器/IO 控制信号，输出。指明当前 CU 是选择访问存储器还是访问 IO 端口。高电平时，访问存储器，表示当前要进行 CPU 与存储器之间的数据传送；低电平时，访问 IO 端口，表示当前要进行 CPU 与 IO 端口之间的数据传送；DMA 方式，此时为高阻态。

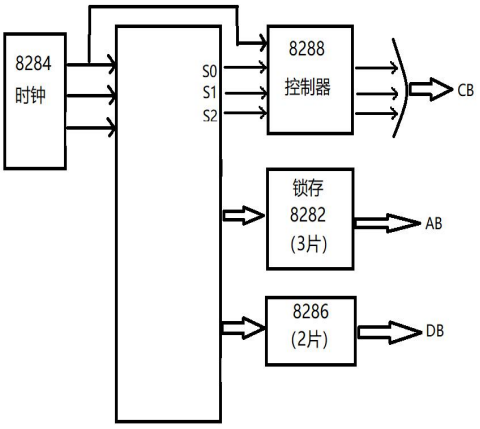
（6）WR#：写信号，输出。有效时，表示 CPU 正在进行写总线周期，同时由 M/IO# 信号决定是对存储器还是对 IO 端口执行写操作。

（7）HOLD：总线保持请求信号，输入。

（8）HLDA：总线保持响应信号，输出。

7、最大模式下的专用引脚信号（MN/MX#为低电平）

（1）QS0、QS1：指令队列状态信号，输出。QS0 和 QS1 信号的组合可以只是总线接口部件 BIU 中队列指令的状态。



QS1	QS0	含义
0	0	无操作
0	1	从指令队列的第一个字节中取走代码
1	0	队列已空
1	1	除第一个字节外，还取走了后续字节中的代码

（2）S2#、S1#、S0#：总线周期状态信号，输出。他们的组合表明当前总线周期所进行的操作类型。

S2	S1	S0	操作过程
0	0	0	发中断响应信号 (INTA)
0	0	1	读I/O端口 (IOR)
0	1	0	写I/O端口 (IOW)
0	1	1	暂停
1	0	0	取指令
1	0	1	读内存 (MEMR)
1	1	0	写内存 (MEMW)
1	1	1	无源状态

（3）LOCK#：总线封锁信号，输出。有效时，表示 8086CPU 不允许其它总线部件占用总线。

（4）RQ#/GT1#和 RQ#/GT0#：总线请求信号输入/总线请求允许信号输出。高信号用以取代最小模式时的 HOLD/HLDA 两个信号的功能，是特意为多出力气系统而设计的。