第二章8086体系结构

- 8086CPU结构
- 8086系统的结构和配置

2.1 8086CPU结构

- 8086CPU的内部结构
- 8086CPU的寄存器结构
- 8086CPU的管脚及功能

一、 8086CPU的内部结构

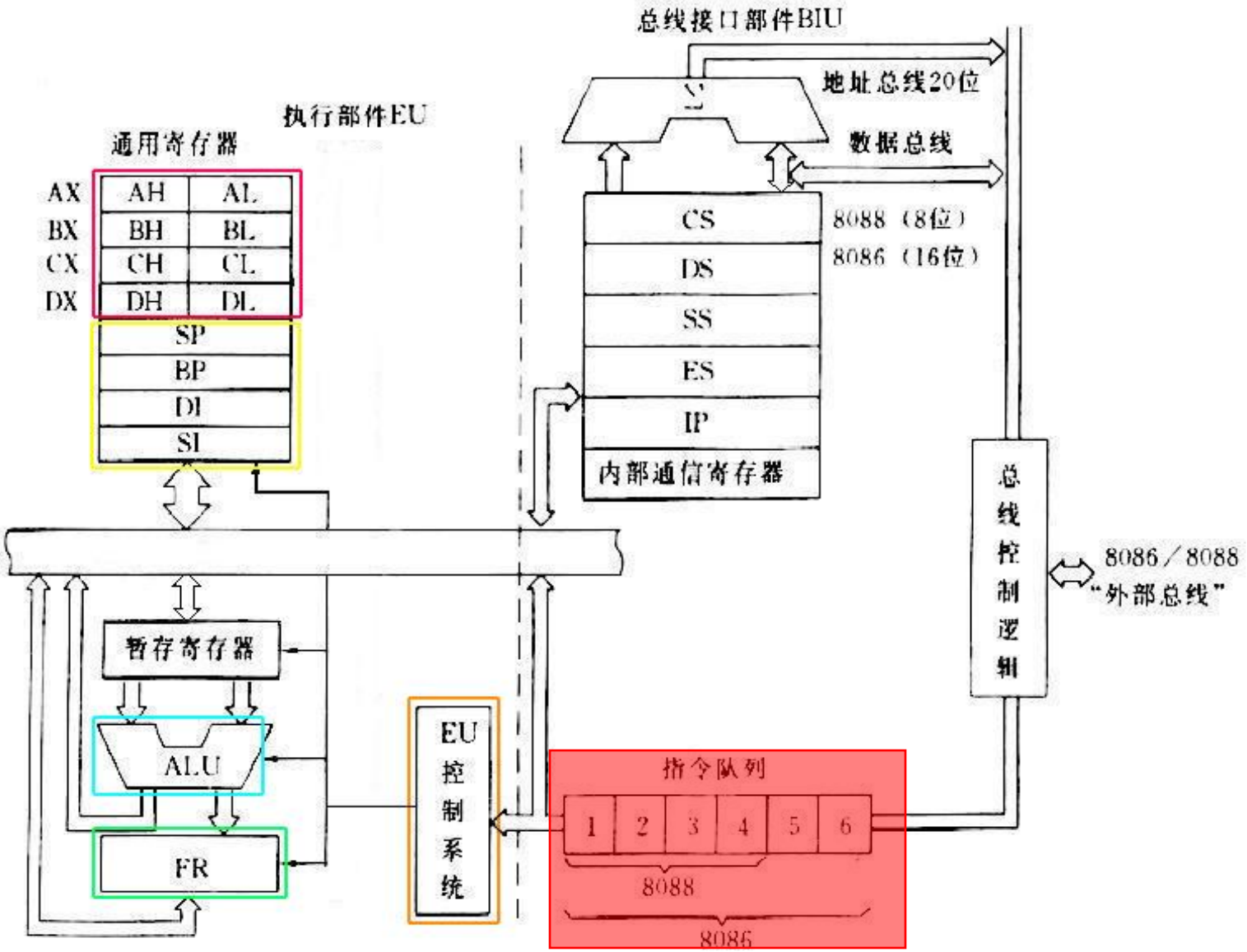
- 8086CPU的内部结构组成

8086CPU由两部分组成：

指令执行部件(EU，Execution Unit)

总线接口部件(BIU，Bus Interface Unit)

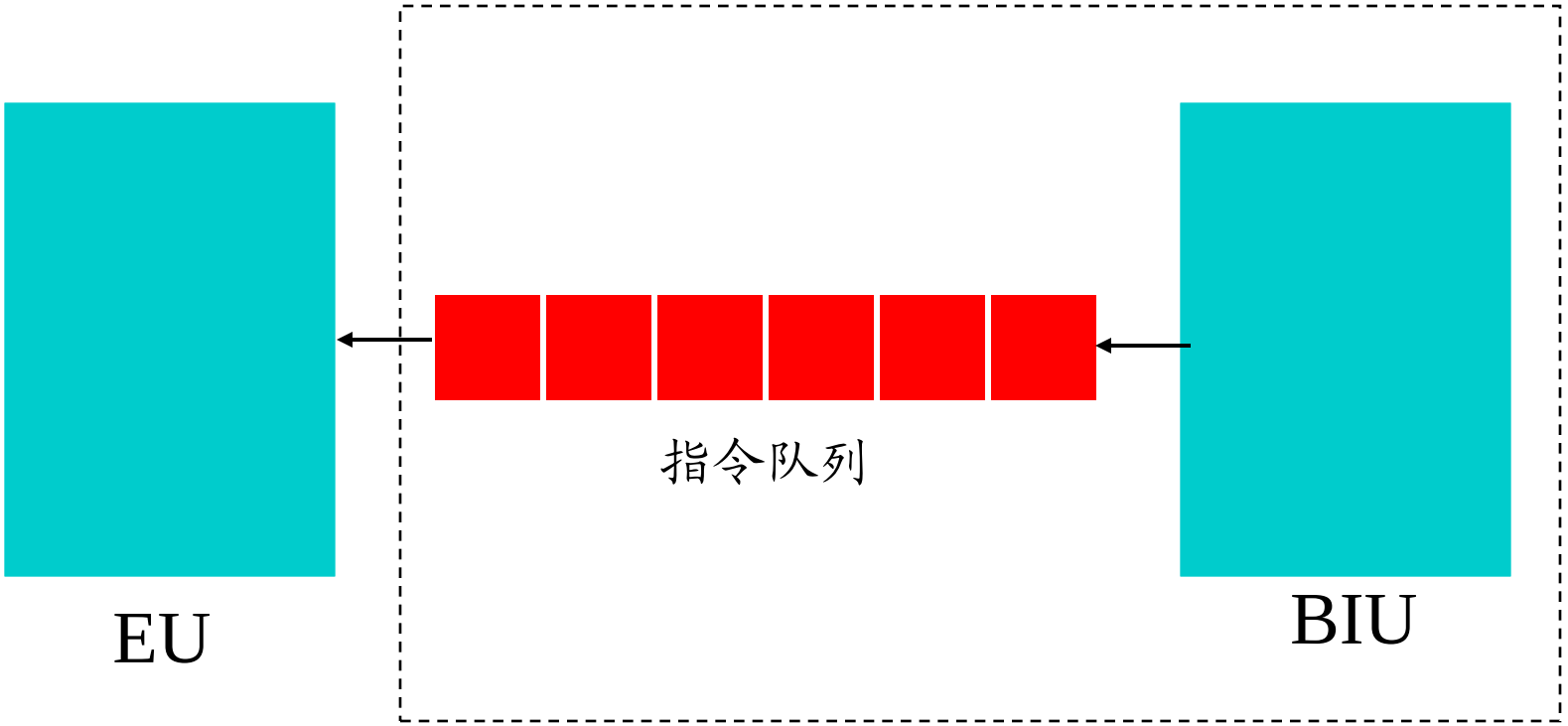
8086CPU的内部结构



执行部件（EU）主要由算术逻辑运算单元(ALU)、标志寄存器FR、通用寄存器组和EU控制器等4个部件组成，其主要功能是执行指令。

总线接口部件(BIU)主要由地址加法器、专用寄存器组、指令队列和总线控制电路等4个部件组成，其主要功能是形成访问存储器的物理地址、访问存储器并取指令暂存到指令队列中等待执行，访问存储器或I／O端口读取操作数参加EU运算或存放运算结果等。


USTC


指令队列
EU
BIU

EU和BIU的操作原则

- **BIU中的指令队列有2个或2个以上字节为空时，BIU自动启动总线周期，取指填充指令队列。直至队列满，进入空闲状态。**
- **EU每执行完一条指令，从指令队列队首取指。系统初始化后，指令队列为空，EU等待BIU从内存取指，填充指令队列。**

- EU取得指令，译码并执行指令。若指令需要取操作数或存操作结果，需访问存储器或I/O，EU向BIU发出访问总线请求。

  当BIU接到EU的总线请求，若正忙（正在执行取指总线周期），则必须等待BIU执行完当前的总线周期，方能响应EU请求；若BIU空闲，则立即执行EU申请总线的请求。

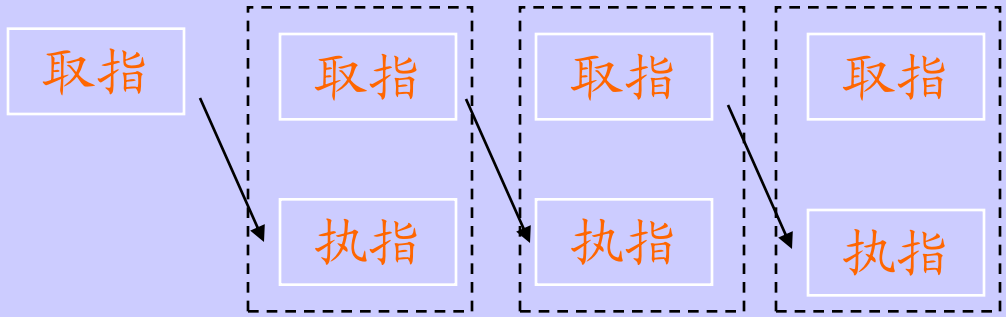
- EU执行转移、调用和返回指令时，若下一条指令不在指令队列中，则队列被自动清除，BIU根据本条指令执行情况重新取指和填充指令队列。

- 空闲状态

8086CPU与一般CPU区别

- 一般CPU工作方式

- 8086CPU工作方式



二、 8086CPU的寄存器结构

- **通用寄存器**
- **段寄存器**
- **标志寄存器FR**
- **指令指针寄存器IP**

1、 通用寄存器

指令执行部件（EU）设有8个通用寄器

AX BX CX DX

SP BP SI DI

AX	AH	AL
BX	BH	BL
CX	CH	CL
DX	DH	DL
SI		
DI		
BP		
SP		

通用寄存器

- AX（Accumulator Register）

  累加器一般用来存放参加运算的数据和结果，在乘、除法运算、I/O操作、BCD数运算中有不可替代的作用。

- BX（Base Register）

  基址寄存器除可作数据寄存器外，还可放内存的逻辑偏移地址，而AX，CX，DX则不能。

- **CX（Counter）**

  将它称作计数寄存器，是因为它既可作数据寄存器，又可在串指令和移位指令中作计数用。

- **DX（Data Register）**

  DX除可作通用数据寄存器外，还在乘、除法运算、带符号数的扩展指令中有特殊用途。

- **SI（Source Index）**

  源变址寄存器多用于存放内存的逻辑偏移地址，隐含的逻辑段地址在DS寄存器中，也可放数据。

- **DI（Destination Index）**

  目标变址寄存器多用于存放内存的逻辑偏移地址，隐含的逻辑段地址在DS寄存器中也可放数据。

- **BP（Base Pointer）**

  基址指针用于存放内存的逻辑偏移地址，隐含的逻辑段地址在SS寄存器中。

- **SP（Stack Pointer ）**

  堆栈指针用于存放栈顶的逻辑偏移地址，隐含的逻辑段地址在SS寄存器中。

- 寄存器的特殊用途和隐含性质

  在指令中没有明显的标出，而这些寄存器参加操作，称之为“隐含寻址”。

  具体的：在某类指令中，某些通用寄存器有指定的特殊用法，编程时需遵循这些规定，将某些特殊数据放在特定的寄存器中，这样才能正确的执行这些指令。采用“隐含”的方式，能有效地缩短指令代码的长度。

寄存器名	特殊用途	隐含性质
AX, AL	在输入输出指令中作数据寄存器用	不能隐含
	在乘法指令中存放被乘数或乘积，在除法指令中存放被除数或商	隐　　含
AH	在LAHF指令中，作目标寄存器用	隐　　含
AL	在十进制运算指令中作累加器用	隐　　含
	在XLAT指令中作累加器用	隐　　含
BX	在间接寻址中作基址寄存器用	不能隐含
	在XLAT指令中作基址寄存器用	隐　　含
CX	在串操作指令和LOOP指令中作计数器用	隐　　含
CL	在移位/循环移位指令中作移位次数计数器用	不能隐含
DX	在字乘法/除法指令中存放乘积高位或被除数高位或余数	隐　　含
	在间接寻址的输入输出指令中作地址寄存器用	不能隐含
SI	在字符串运算指令中作源变址寄存器用	隐　　含
	在间接寻址中作变址寄存器用	不能隐含
DI	在字符串运算指令中作目标变址寄存器用	隐　　含
	在间接寻址中作变址寄存器用	不能隐含
BP	在间接寻址中作基址指针用	不能隐含
SP	在堆栈操作中作堆栈指针用	隐　　含

2、段寄存器

总线接口部件BIU设有4个16位段寄存器

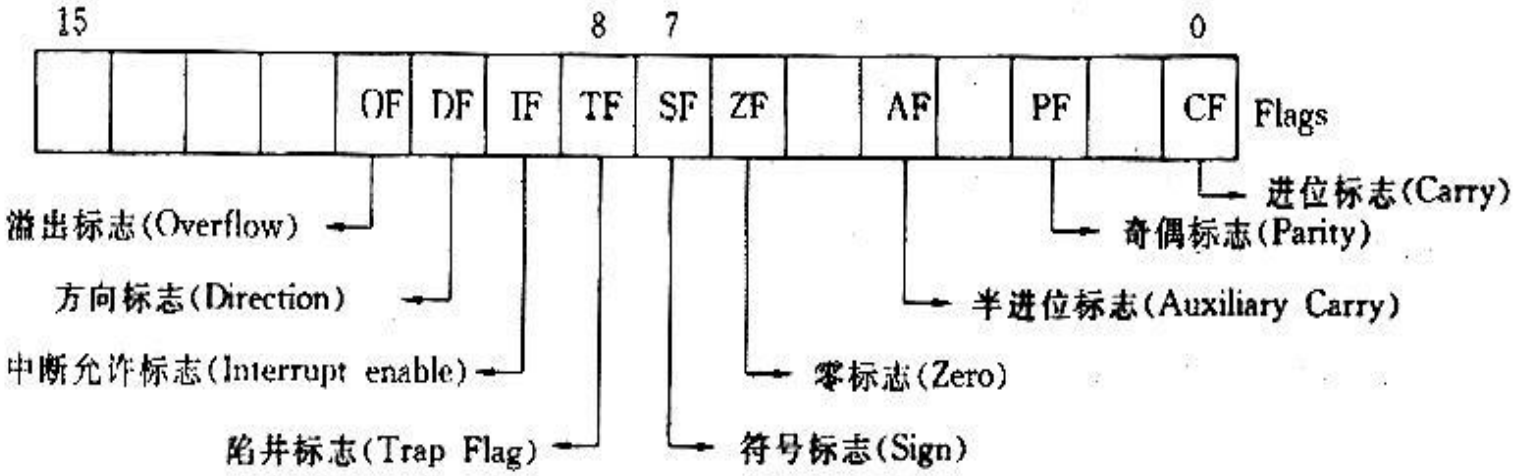
- CS（Code Segment），代码段寄存器中存放程序代码段起始地址的高16位。
- DS（Data Segment），数据段寄存器中存放数据段起始地址的高16位。
- SS（Stack Segment），堆栈段寄存器中存放堆栈段起始地址的高16位。
- ES（Extended Segment），扩展段寄存器中存放扩展数据段起始地址的高16位。

3、标志寄存器FR

标志寄存器FR中共有9个标志位，可分成两类：

- **状态标志** 表示运算结果的特征，它们是CF、PF、AF、ZF、SF和OF
- **控制标志** 控制CPU的操作，它们是IF、DF和TF。

标志寄存器FR


15
8
7
0
OF
DF
IF
TF
SF
ZF
AF
PF
CF
Flags
溢出标志(Overflow)
方向标志(Direction)
中断允许标志(Interrupt enable)
陷井标志(Trap Flag)
符号标志(Sign)
零标志(Zero)
半进位标志(Auxiliary Carry)
奇偶标志(Parity)
进位标志(Carry)

FR中的状态标志

- CF(Carry Flag)：进位标志位

CF＝l，表示本次运算中最高位(D15或D7)有进位(加法运算时)或有借位(减法运算时)。CF标志可通过STC指令置位，通过CLC指令复位(清除进位标志)，还可通过CMC指令将当前CF标志取反。

- PF(Parity Flag)：奇偶校验标志位

PF＝1，表示本次运算结果中有偶数个“l”，PF＝表示本次运算结果中有奇数个“1”。

- AF(Auxiliary Carry Flag)：辅助进位标志位。AF＝l，表示运算结果的8位数据中，低4位向高4位有进位(加法运算时)或有借位(减法运算时)，这个标志位只在十进制运算中有用。
- ZF(Zero Flag)：零标志位

　ZF＝1，表示本次运算结果为零，否则即运算结果非零时，ZF＝0。

- SF(Sign Flag)：符号标志

  SF＝1，表示本次运算结果的最高位(第7位或第15位)为“l”，否则SF＝0。

- OF(Overflow F1ag)：溢出标志

FR寄存器的控制标志

- IF(Interrupt Flag)：中断标志位

IF＝1，表示允许CPU响应可屏蔽中断。IF标志可通过STI指令置位，也可通过CLI指令复位。

- DF(Direction Flag)：方向标志位

在串操作指令中，若DF＝0，表示串操作指令地址指针自动增量；DF＝1，表示地址指针自动减量。DF标志位可通过STD指令置位，也可通过CLD指令复位。

- TF(Trap Flag)：单步标志位

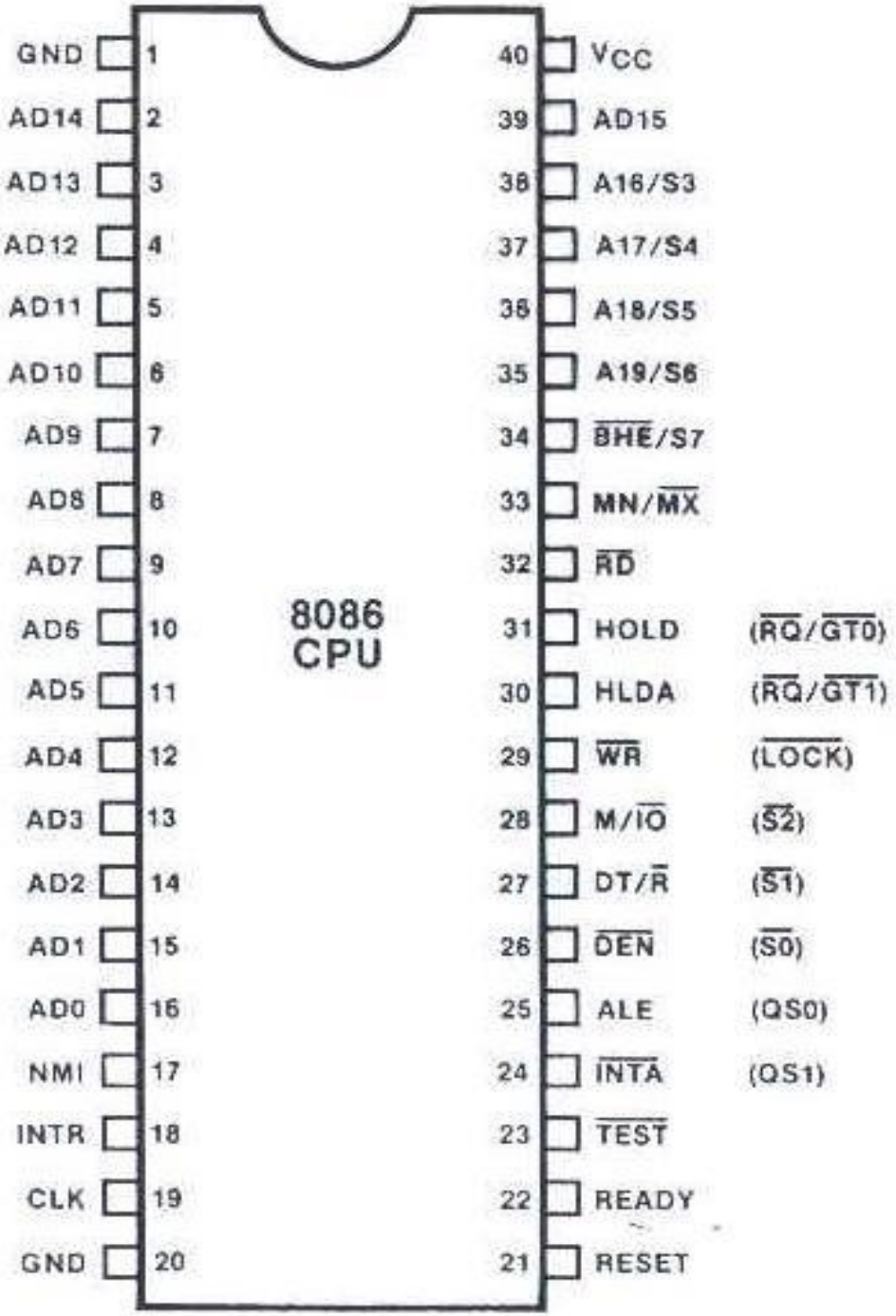
4、指令指针寄存器 IP

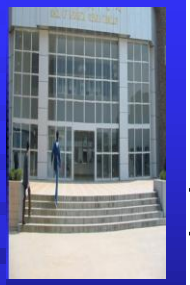
IP ： BIU要取指令的地址。

三、8086CPU的管脚及功能

8086是16位CPU。它采用高性能的N—沟道，耗尽型负载的硅栅工艺(HMOS)制造。由于受当时制造工艺的限制，部分管脚采用了分时复用的方式，构成了40条管脚的双列直插式封装


USTC


GND 1
AD14 2
AD13 3
AD12 4
AD11 5
AD10 6
AD9 7
AD8 8
AD7 9
AD6 10
AD5 11
AD4 12
AD3 13
AD2 14
AD1 15
AD0 16
NMI 17
INTR 18
CLK 19
GND 20
40 VCC
39 AD15
38 A16/S3
37 A17/S4
36 A18/S5
35 A19/S6
34 BHE/S7
33 MN/MX
32 RD
31 HOLD (RQ/GT0)
30 HLDA (RQ/GT1)
29 WR (LOCK)
28 M/IO (S2)
27 DT/R (S1)
26 DEN (S0)
25 ALE (QS0)
24 INTA (QS1)
23 TEST
22 READY
21 RESET
8086
CPU



1、 8086的两种工作方式

最小模式： 系统中只有8086一个处理器，所有的控制信号都是由8086CPU产生(MN/MX=1)。

最大模式： 系统中可包含一个以上的处理器，比如包含协处理器8087。在系统规模比较大的情况下，系统控制信号不是由8086直接产生，而是通过与8086配套的总线控制器等形成(MN/MX=0)。

- **最小模式下的引脚说明**

（1） AD15～AD0 (Address Data Bus)：

地址/数据复用信号，双向，三态。在T1状态（地址周期）AD15～AD0上为地址信号的低16位A15～A0；在T2 ～ T3状态（数据周期）AD15～AD0 上是数据信号D15～D0。

- 三总线结构
  - 数据线DB
  - 地址线AB
  - 控制线CD

（2） A19/S6～A16/S3 (Address/Status)：

地址/状态复用信号，输出。在总周期的T1状态A19/S6～A16/S3上是地址的高4位。在T2～T4状态，A19/S6～A16/S3上输出状态信息。

$\overline{BHE}$

S_4	S_3	当前正在使用的段寄存器
0	0	ES
0	1	SS
1	0	CS或未使用任何段寄存器
1	1	DS

（3）BHE# /S7 (Bus High Enable/Status)：

数据总线高8位使能和状态复用信号，输出。在总线周期T1状态，BHE#有效，表示数据线上高8位数据有效。在T2～T4状态BHE #/S7 输出状态信息S7。S7在8086中未定义。

（4） RD# (Read)

读信号， 三态输出，低电平有效，表示当前CPU正在读存储器或I／O端口。

（5） WR# (Write)

写信号，三态输出，低电平有效，表示当前CPU正在写存储器或I／O端口。

（6）M／IO# (Memory／IO)

存储器或I／O端口访问信号。三态输出，M／IO#为高电平时，表示当前CPU正在访问存储器，M／IO# 为低电平时，表示当前CPU正在访问I／O端口。

$\overline{BHE}$

（7）READY

准备就绪信号。由外部输入，高电平有效，表示CPU访问的存储器或I／O端口已准备好传送数据。当READY无效时，要求CPU插入一个或多个等待周期Tw，直到READY信号有效为止。

（8）INTR(Interrupt Request)

中断请求信号，由外部输入，电平触发，高电平有效。INTR有效时，表示外部设备向CPU发出中断请求，CPU在每条指令的最后一个时钟周期对INTR进行测试，一旦测试到有中断请求，并且当中断允许标志IF＝1时，则暂停执行下条指令转入中断响应周期。

（9）INTA# (Interrupt Acknowledge)

中断响应信号。向外部输出，低电平有效，表示CPU响应了外部发来的INTR信号。

（10） NMI(Non—Maskable Interrupt Request)

不可屏蔽中断请求信号。由外部输入，边沿触发，正跳沿有效。CPU一旦测试到NMI请求信号，待当前指令执行完就自动从中断入口地址表中找到类型2中断服务程序的入口地址并转去执行。

（11）TEST#

测试信号。由外部输入，低电平有效。当CPU执行WAIT指令时(WAIT指令是用来使处理器与外部硬件同步)，每隔5个时钟周期对TEST进行一次测试，若测试到该信号无效，则CPU继续执行WAIT指令，即处于空闲等待状态；当CPU测到TEST输入为低电平时，则转而执行WAIT的下一条指令。由此可见，TEST对WAIT指令起到了监视的作用。

（12）RESET

复位信号。由外部输入，高电平有效。RESET信号至少要保持4个时钟周期，CPU接收到该信号后，停止进行操作，并对标志寄存器(FR)、IP、DS、SS、ES及指令队列清零，而将CS设置为FFFFH。当复位信号变为低电平时，CPU从FFFF0H开始执行程序，由此可见，采用8086CPU计算机系统的启动程序就保持在开始的存储器中。

（13）ALE(Address Latch Enable)

地址锁存使能信号，输出，高电平有效。用来作为地址锁存器的锁存控制信号。

（14）DEN# (Data Enable)

数据使能信号，输出，三态，低电平有效。用于数据总线驱动器的控制信号。

（15） DT/R#(Data Transmit/Receive)：

数据驱动器数据流向控制信号，输出，三态。在8086系统中，通常采用8286或8287作为数据总线的驱动器，用DT/R#信号来控制数据驱动器的数据传送方向。当DT/R#＝1时，进行数据发送；DT/R#＝0时，进行数据接收。

（16）HOLD(Hold Request)

总线请求信号。由外部输入，高电平有效器向CPU请求使用总线。

（17）HLDA(Hold Acknowledge)

共享总线的处理总线请求响应信号。向外部输出，高电平有效。

（18）MN/MX＃(Minimum/Maximum Mode Control)：
最大最小模式控制信号，输入。MN/MX＃＝1（＋5V），CPU工作在最小模式。MN/MX＃＝0（接地），CPU则工作在最大模式。

（19） GND 地。

（20) VCC 电源，接＋5V。

➢ 最大模式下的引脚说明

当8086CPU工作在最大模式系统时，有8个管脚重新定义。

（1）S2#、S1#、S0#（Bus Cycle Status，最小模式为M/IO#、D/TR#、DEN#）：

总线周期状态信号，输出。这三个信号的组合表示当前总线周期的类型。在最大模式下，由这三个信号输入给总线控制器8288，用来产生存储器、I/O的读写等相关控制信号。如下表：

S2#	S1#	S0#	CPU状态	8288命令
0	0	0	中断响应	INTA#
0	0	1	读I/O端口	IORC#
0	1	0	写I/O端口	IOWC# AIOWC#
0	1	1	暂停	无
1	0	0	取指令	MRDC#
1	0	1	读存储器	MRDC#
1	1	0	写存储器	MWTC# AMWC#
1	1	1	无作用	无

（2）LOCK# 封锁信号。

三态输出，低电平有效。LOCK有效时表示CPU不允许其它总线主控者占用总线。这个信号由软件设置。当在指令前加上LOCK前缀时，则在执行这条指令期间LOCK保持有效，即在此指令执行期间，CPU封锁其它主控者使用总线。

（3）QS1、QS0(Instruction Queue Status，最小模式为ALE、INTA#)：

指令队列状态信号，输出。QS1,QS0组合起来表示前一个时钟周期中指令队列的状态，以便从外部对芯片的测试。

（4）RQ#/GT0# ，RQ#/GT1# (Request / Grant)

总线请求信号请求 / 同意信号。双向，低电平有效，当该信号为输入时表示其它主控者向CPU请求使用总线；当为输出时表示CPU对总线请求的响应信号。两条线可同时与两个主控者相连，同时,RQ#/GT0#优先级高于RQ#/GT1#。

（5）QS1、QS0(Instruction Queue Status)：

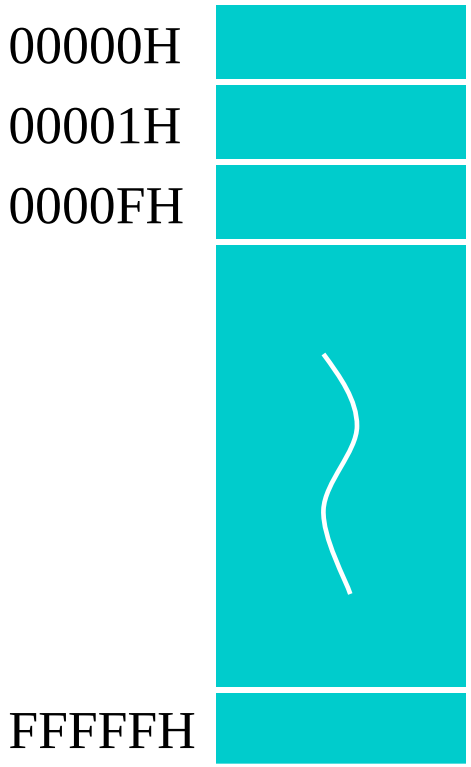
指令队列状态信号，输出。QS1,QS0组合起来表示前一个时钟周期中指令队列的状态，以便从外部对芯片的测试。

QS1	QS0	编码含义
0	0	无操作
0	1	从队列中取第一个字节
1	0	队列已空
1	1	从队列中取后续字节

2.2 8086系统的储存器组织

一、8086存储器结构

8086系统中的存储器是一个最多1M个8位数量的字节序列，即可寻址的存储空间为1M字节，系统为每个字节分配一个20位的物理地址(对应16进制的地址范围从00000H～FFFFFH)。



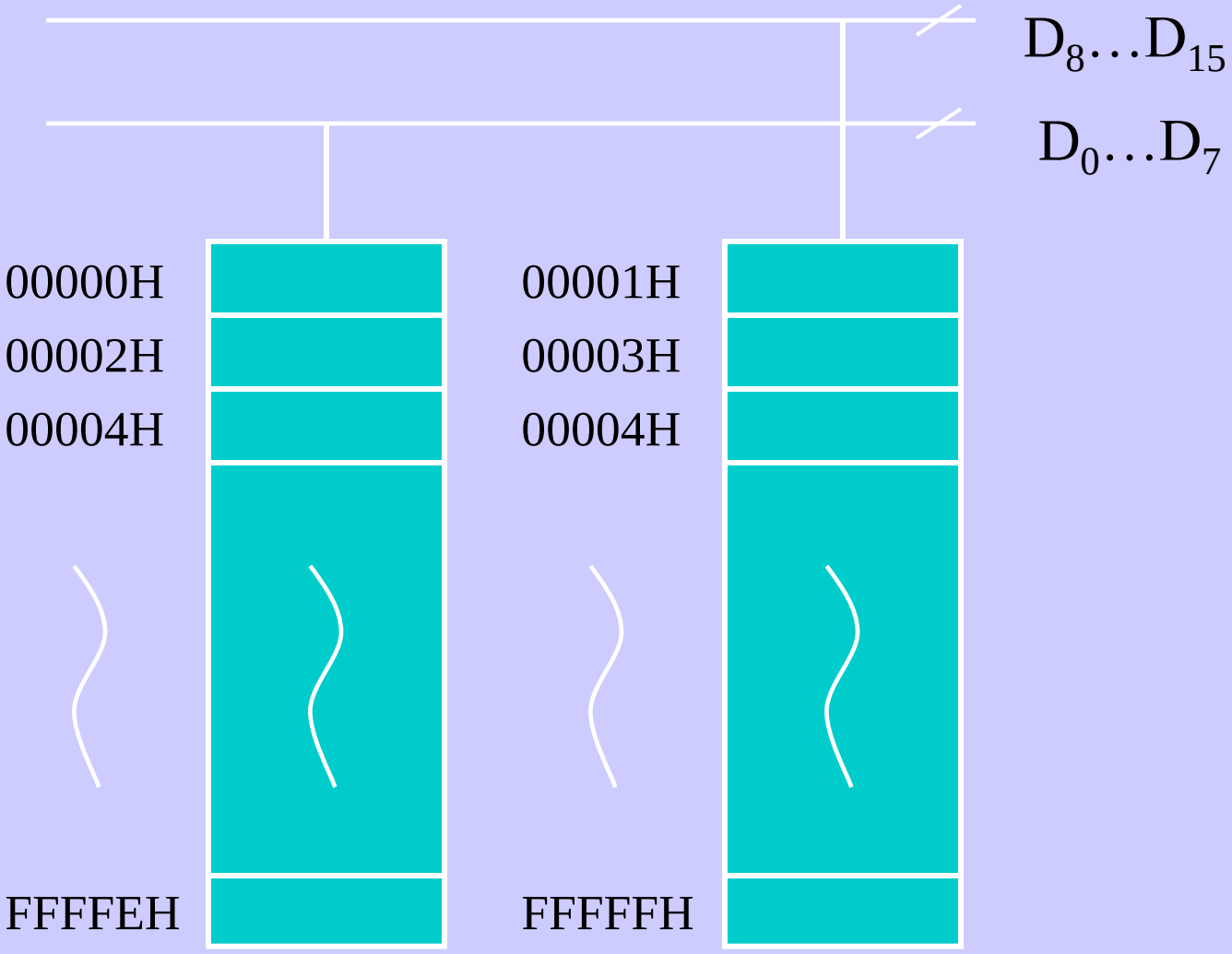
- （一）、数据在内存的位置
  字节、字、双字及其地址
- （二）、8086CPU对字/字节的读操作
  - 16位读
  - 从偶地址读

二、8086存储器的分体结构

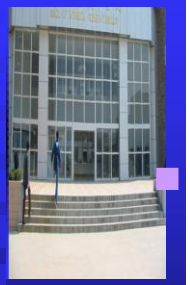
8086系统中，存储器是分体结构，1M字节的存储空间分成两个512K字节的存储体。

一个是偶数地址存储体，一个是奇数地址存储体，两个存储体采用字节交叉编址方式


USTC

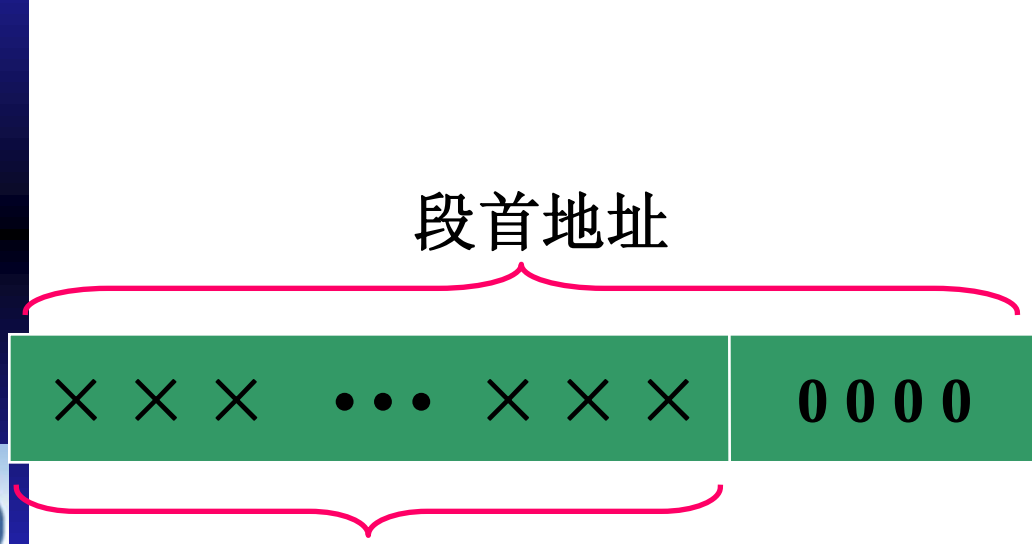

D8…D15
D0…D7
00000H
00002H
00004H
FFFFEH
00001H
00003H
00004H
FFFFFH

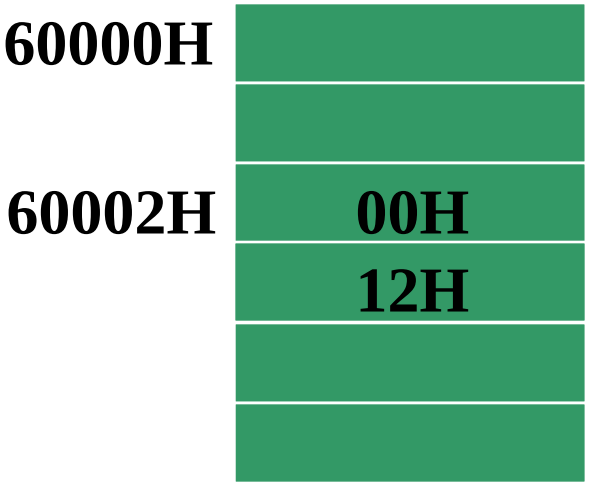
- 三、存储器的分段
  - 为什么分段
  - **8086存储器分4个段**
  - 段基地址与段寄存器
  - 偏移地址的产生
  - 逻辑地址、物理地址



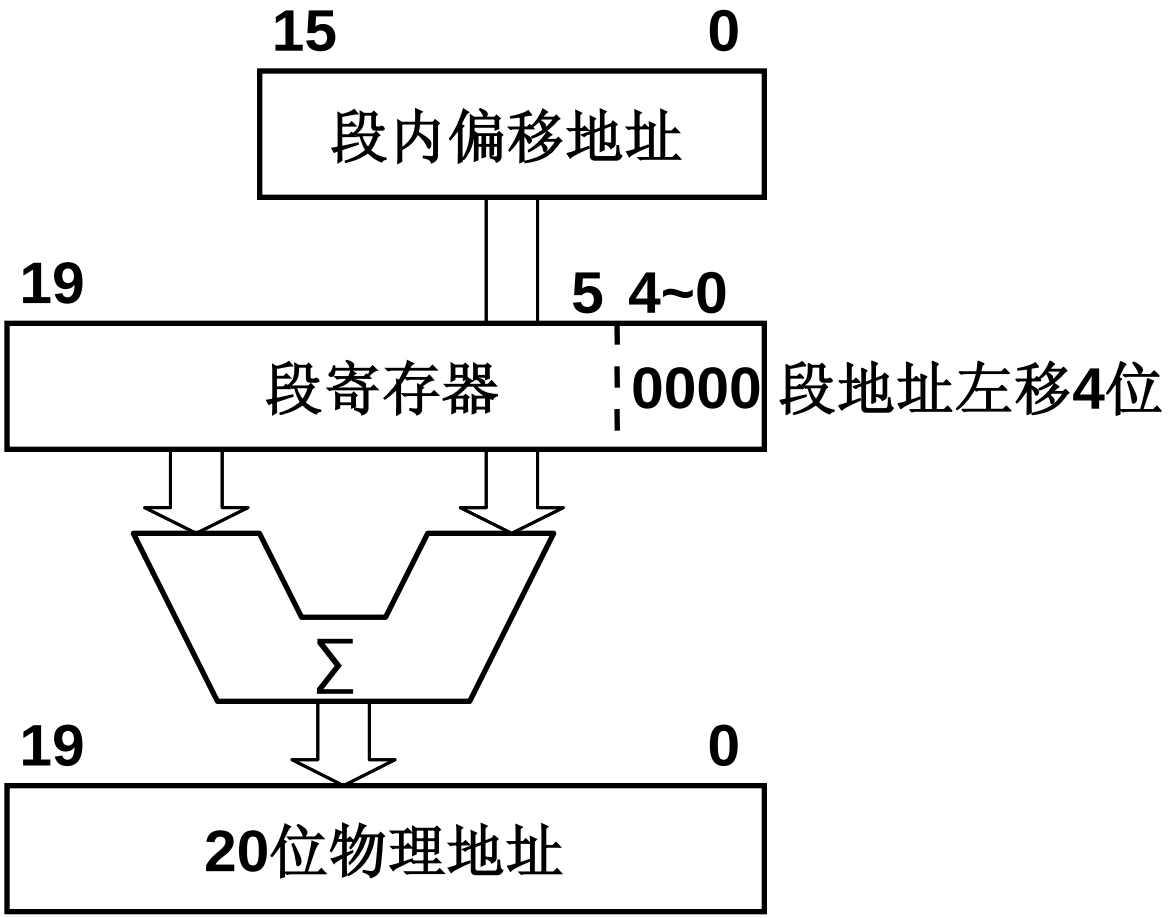
段基地址和偏移地址组成了逻辑地址

格式为：段基地址:偏移地址





物理地址=段基地址×16+偏移地址



物理地址的形成

- 8086分段的好处
  - 1、解决了16位地址寄存器对20位物理地址的寻址问题
  - 2、实现了程序代码的浮动装配
- 8086复位后程序运行的起始地址

- 段寄存器使用约定

存储器操作类型	隐含段	超越段	偏移量
取指令	CS	无	IP
堆栈	SS	无	SP
数据变量	DS	CS,ES,SS	有效地址
源串变量	DS	CS,ES,SS	SI
目的串变量	ES	无	DI
基址BP指针	SS	CS,ES,DS	有效地址

四、堆栈段的使用

所谓堆栈是在存储器中开辟一个区域，用来存放需要暂时保存的数据，其工作方式是“先进后出”或“后进先出”的方式。

8086系统中的堆栈段是由段定义语句在存储器中定义的一个段，堆栈段容量小于等于64K字节。段基址由堆栈寄存器SS指定，栈顶由堆栈指针SP指定，堆栈地址由高向低增长，栈底设在存储器的高地址区。

8086堆栈是递减型的“满”堆栈。