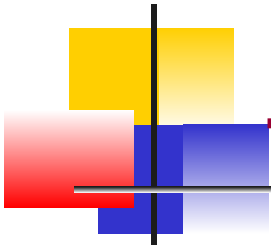


第2章

微处理器





主要内容：

- 微处理器的一般构成及工作原理；
- 8088微处理器的结构；
- 8088微处理器的内部寄存器；
- 8088微处理器的引脚；
- 8088微处理器对内存的管理；

§2.1 微型机概述

- 微处理器的功能；
- 微处理器的基本组成。

功 能

是计算机系统的核心
根据指令实现各种相应的运算
实现数据的暂存
实现与存储器和接口的信息通信

.....

组 成

运算器
控制器
内部寄存器组



§2.2 8088CPU的内部编程结构

- 8088内部由两部分组成：

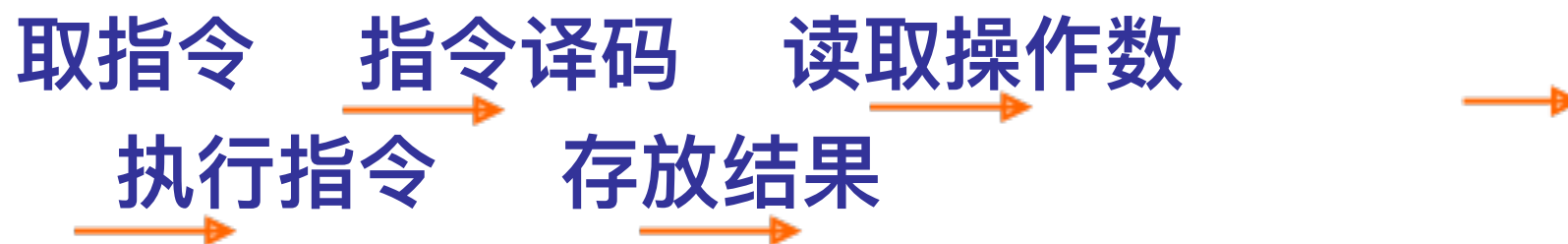
执行单元 (EU)

功能上

总线接口单元 (BIU)



指令执行的一般过程



EU：指令译码、指令执行。

BIU：CPU与存储器和I/O设备间传递数据。

段首地址

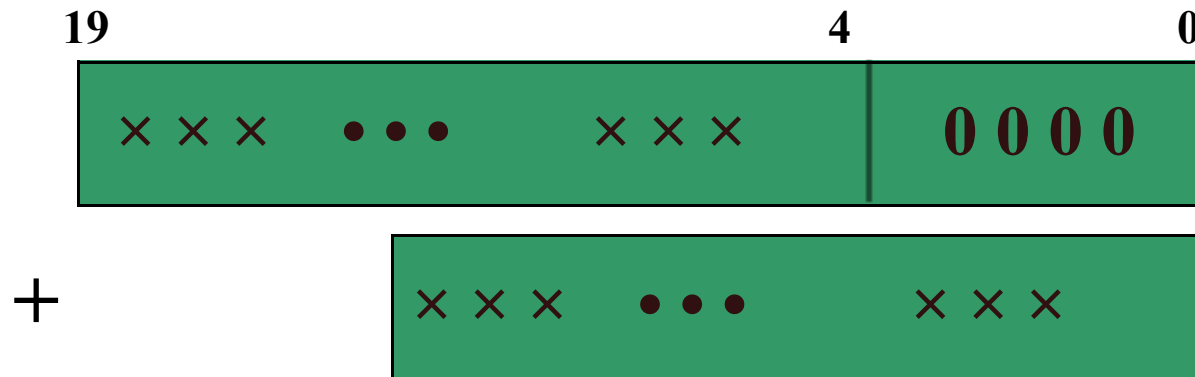
段首地址



段基地址 (16位)

- 物理地址由段基地址和偏移地址组成

段首地址



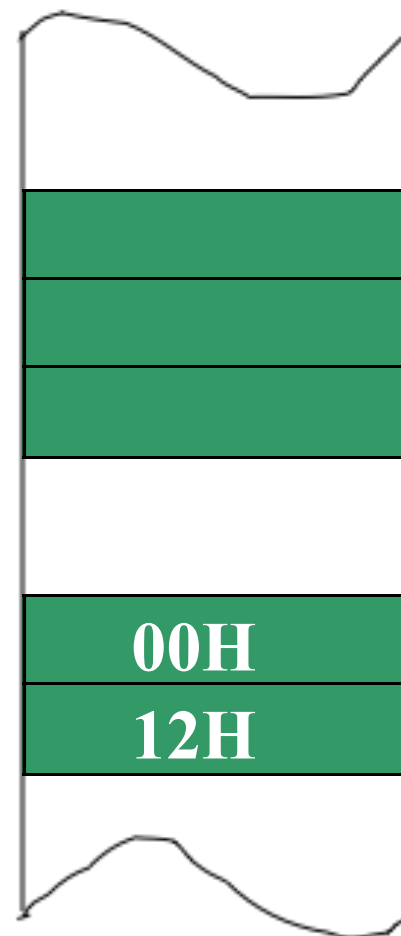
物理地址

偏移地址

物理地址=段基地址×16+偏移地址

物理地址

- 段基地址 = 6000H
- 段首地址 \longrightarrow 60000H
- 偏移地址 \longrightarrow 0009H
- 物理地址 \longrightarrow 60009H
- 逻辑地址
(6000H: 0009H)



数据段

总线接口单元 BIU

功能:

- 从内存中取指令到指令预取队列
- 负责与内存或输入/输出接口之间的数据传送

组成:

- (1) 4个段地址寄存器
CS代码段 DS数据段 ES扩展段 SS堆栈段
- (2) 指令指针寄存器IP
- (3) 20位物理地址加法器和总线控制电路
- (4) 6个字节的指令队列缓冲器

44字节

(8086)
(8088)



串行和并行方式的指令流水线

串行工作方式:

EU和BIU交替工作，按顺序完成上述指令执行过程。

并行工作方式:

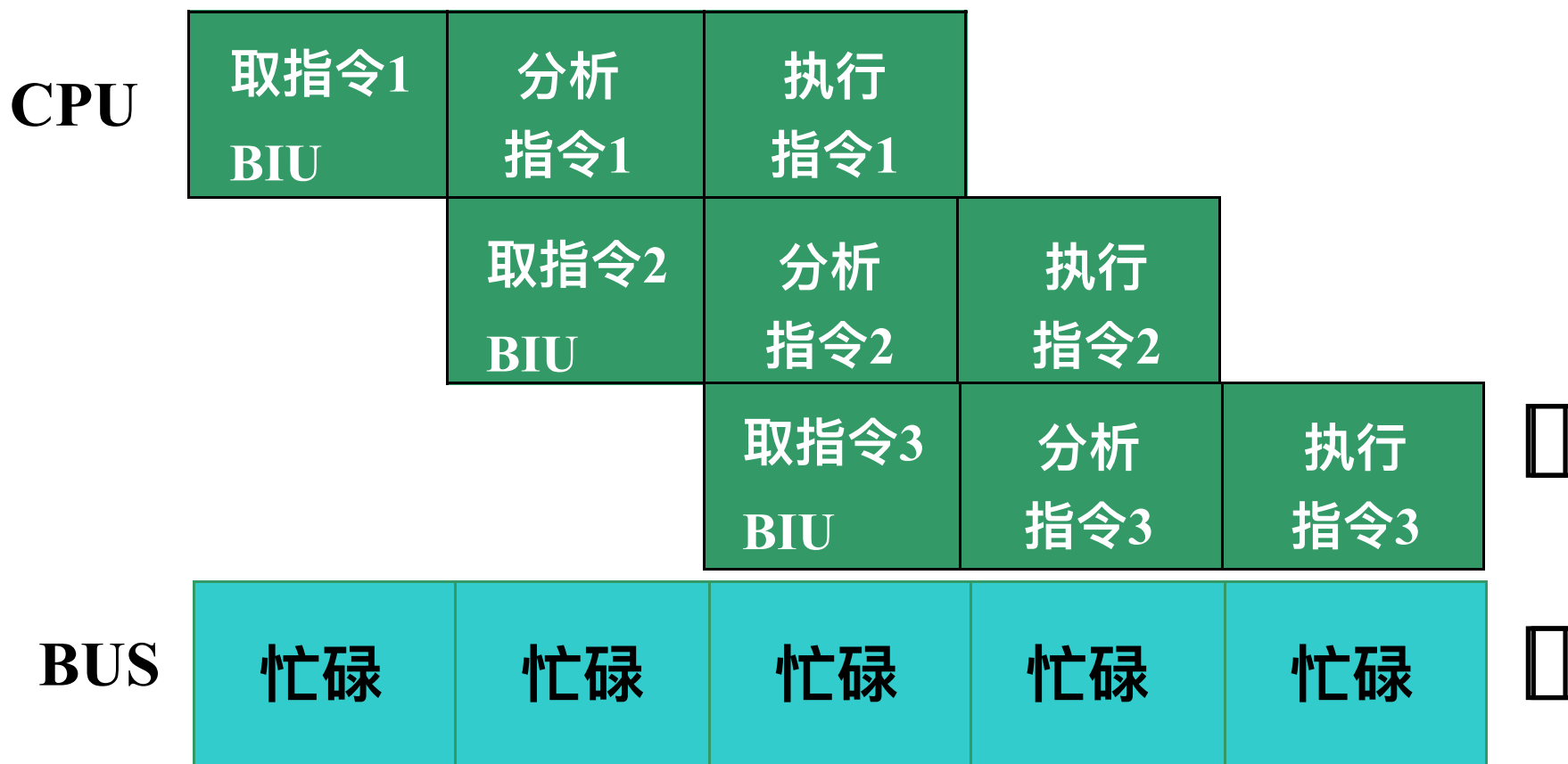
EU和BIU可同时工作

8088以前的CPU采用串行工作方式:

CPU	取指令1	分析指令1	执行指令1	取指令2	分析指令2	执行指令2
BUS	忙碌			忙碌		

并行工作方式

- 8088CPU采用并行工作方式





结论

- 指令预取队列的存在使EU和BIU两个部分可同时进行工作，从而
- ① 提高了CPU的效率；
- ② 降低了对存储器存取速度的要求

段首地址

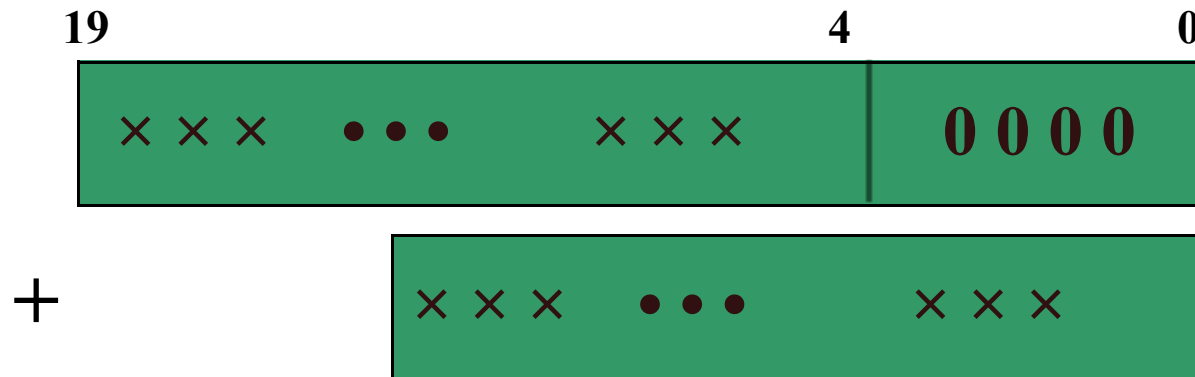
段首地址



段基地址 (16位)

- 物理地址由段基地址和偏移地址组成

段首地址



物理地址

偏移地址

物理地址=段基地址×16+偏移地址

执行单元 EU

功能 → 指令的执行

- 指令译码
- 指令执行
- 暂存中间运算结果 → 在ALU中完成
- 保存运算结果特征 → 在通用寄存器中
→ 在标志寄存器FLAGS

中

组成：

- (1) 16位ALU
- (2) 通用寄存器组：AX BX CX DX SP BP SI DI
- (3) 16位标志寄存器 FLAGS
- (4) EU控制电路



§2.3 8088的内部寄存器

- 含14个16位寄存器，按功能可分为三类

8个通用寄存器

4个段寄存器

2个控制寄存器

深入理解：每个寄存器中数据的含义



1. 通用寄存器

数据寄存器 (AX, BX, CX, DX)

地址指针寄存器 (SP, BP)

变址寄存器 (SI, DI)

- 仅4个16位数据寄存器可分为8个8位寄存器：
- AX → AH, AL
- BX → BH, BL
- CX → CH, CL
- DX → DH, DL



数据寄存器特有的习惯用法

- **AX：累加器。**所有I/O指令都通过AX与接口传送信息，中间运算结果也多放于AX中；
- **BX：基址寄存器。**在间接寻址中用于存放基地址；
- **CX：计数寄存器。**用于在循环或串操作指令中存放计数值；
- **DX：数据寄存器。**在间接寻址的I/O指令中存放I/O端口地址；在32位乘除法运算时，存放高16位数。

可能是数据
也可能是地址



1. 通用寄存器

数据寄存器 (AX, BX, CX, DX)

地址指针寄存器 (SP, BP)

变址寄存器 (SI, DI)

- SP: 堆栈指针寄存器, 其内容为栈顶的偏移地址; *没有堆栈操作时, SP也可作通用寄存器存放运算结果。*
- BP: 基址指针寄存器, 存放堆栈段内某一内存单元的偏移地址。 *也有可能是堆栈运算的中间结果*

堆栈及堆栈段的使用

- **堆栈**：内存中一个特殊区域，用于存放暂时不用或需要保护的数据。常用于响应中断或子程序调用。

例：若已知 $(SS) = 1000H$
 $(SP) = 0100H$

- 则堆栈段的段首地址

= ? $1000H$

- 栈顶地址=? $0100H$

- 若该段最后一个单元

地址为 $10200H$ ，则栈底=? $0200H$

段首地址
栈顶 = 段首
栈顶 = 栈底

段首

栈顶

栈底



堆栈区



1.通用寄存器

数据寄存器 (AX, **BX**, CX, DX)

地址指针寄存器 (SP, **BP**)

变址寄存器 (SI, DI)

BX与BP在应用上的区别

- 作为通用寄存器，二者均可用于存放数据；
- 作为基址寄存器，**BX**表示所寻找的数据在**数据段**；**BP**则表示数据在**堆栈段**。

数据寄存器 (AX, BX, CX, DX)

地址指针寄存器 (SP, BP)

变址寄存器 (SI, DI)

变址寄存器--与数据段相关

- SI: 源变址寄存器
- DI: 目标变址寄存器
- 变址寄存器常用于指令的间接寻址或变址寻址。特别是在串操作指令中，用SI存放源操作数的偏移地址，而用DI存放目标操作数的偏移地址。



2.段寄存器

用于存放相应逻辑段的段基地址

64KB

CS: 代码段寄存器。代码段存放指令代码

DS: 数据段寄存器

ES: 附加段寄存器

SS: 堆栈段寄存器

这两个段存放操作数

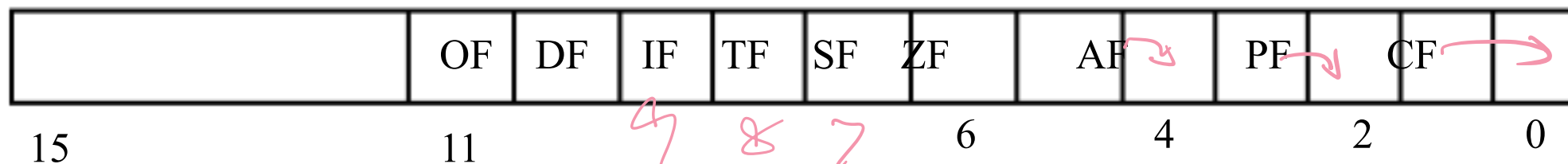
3. 控制寄存器

- **IP**: 指令指针寄存器, 其内容为下一条要执行指令的偏移地址

- **FLAGS**: 标志寄存器, 存放运算结果的特征

6个状态标志位 (CF, SF, AF, PF, OF, ZF)

3个控制标志位 (IF, TF, DF)



0 状态标志位

CF: 进位标志位, 最高位进(借)位, CF=1

OF: 溢出标志位, 算术运算超出有符号数的可表示范围, OF=1 $C_6 \oplus C_7$

ZF: 零标志位, 结果为0时, ZF=1

SF: 符号标志位, 当结果最高位为1, SF=1

PF: 奇偶标志位: 运算结果中低8位中1的个数为偶, 则PF=1

AF: 辅助进位标志位。加(减)操作中, 若

Bit3向Bit4有进位(借位), AF=1

控制标志位

TF: 陷阱标志位, TF=1时, CPU处于单步执行指令的工作方式

IF: 中断允许标志位, IF=1使CPU可以响应可屏蔽中断请求。

DF: 方向标志位, 在数据串操作时确定操作方向

§2.4 8088CPU的引线及功能

GND	1	40	$V_{cc}(+5V)$
A_{14}	2	39	A_{15}
A_{13}	3	38	A_{16} / S_3
A_{12}	4	37	A_{17} / S_4
A_{11}	5	36	A_{18} / S_5
A_{10}	6	35	A_{19} / S_6
A_9	7	34	$SS_0 / (HIGH)$
A_8	8	33	MN / MX
AD_7	9	32	\overline{RD}
AD_6	10	31	$HOLD(\overline{RC})$
AD_5	11	30	$HLDA(\overline{RC})$
AD_4	12	29	$\overline{WR}(\overline{LOCK})$
AD_3	13	28	$\overline{MIO}(\overline{S_2})$
AD_2	14	27	$DT / \overline{R}(\overline{S_1})$
AD_1	15	26	$\overline{DEN}(S_0)$
AD_0	16	25	$ALE(QS_0)$
NMI	17	24	$\overline{INTA}(QS_1)$
INTR	18	23	TEST
CLK	19	22	READY
GND	20	21	RESET

引脚定义的方法可大致分为：

地址/数据分时复用引脚（ AD_7 —— AD_0 等）；

地址/状态分时复用引脚（ A_{19} —— A_{16} / S_3 —— S_6 等）；

控制总线：

每个引脚只传送一种信息（ \overline{RD} 等）；

引脚电平的高低不同的信号（ IO/M 等）；

CPU工作于不同方式有不同的名称和定义（ $\overline{WR}/\overline{LOCK}$ 等）；

引脚的输入和输出分别传送不

§2.4 8088CPU的引线及功能

GND	1	40	$V_{cc}(+5V)$
A_{14}	2	39	A_{15}
A_{13}	3	38	A_{16} / S_3
A_{12}	4	37	A_{17} / S_4
A_{11}	5	36	A_{18} / S_5
A_{10}	6	35	A_{19} / S_6
A_9	7	34	$SS_0 / (\overline{HIGH})$
A_8	8	33	MN / MX
AD_7	9	32	\overline{RD}
AD_6	10	31	$HOLD(\overline{RQ} / \overline{GT_0})$
AD_5	11	30	$HLDA(\overline{RQ} / \overline{GT_1})$
AD_4	12	29	$\overline{WR}(\overline{LOCK})$
AD_3	13	28	$\overline{MIO}(\overline{S_2})$
AD_2	14	27	$DT / \overline{R}(\overline{S_1})$
AD_1	15	26	$\overline{DEN}(\overline{S_0})$
AD_0	16	25	$ALE(QS_0)$
NMI	17	24	$\overline{INTA}(QS_1)$
INTR	18	23	\overline{TEST}
CLK	19	22	READY
GND	20	21	RESET

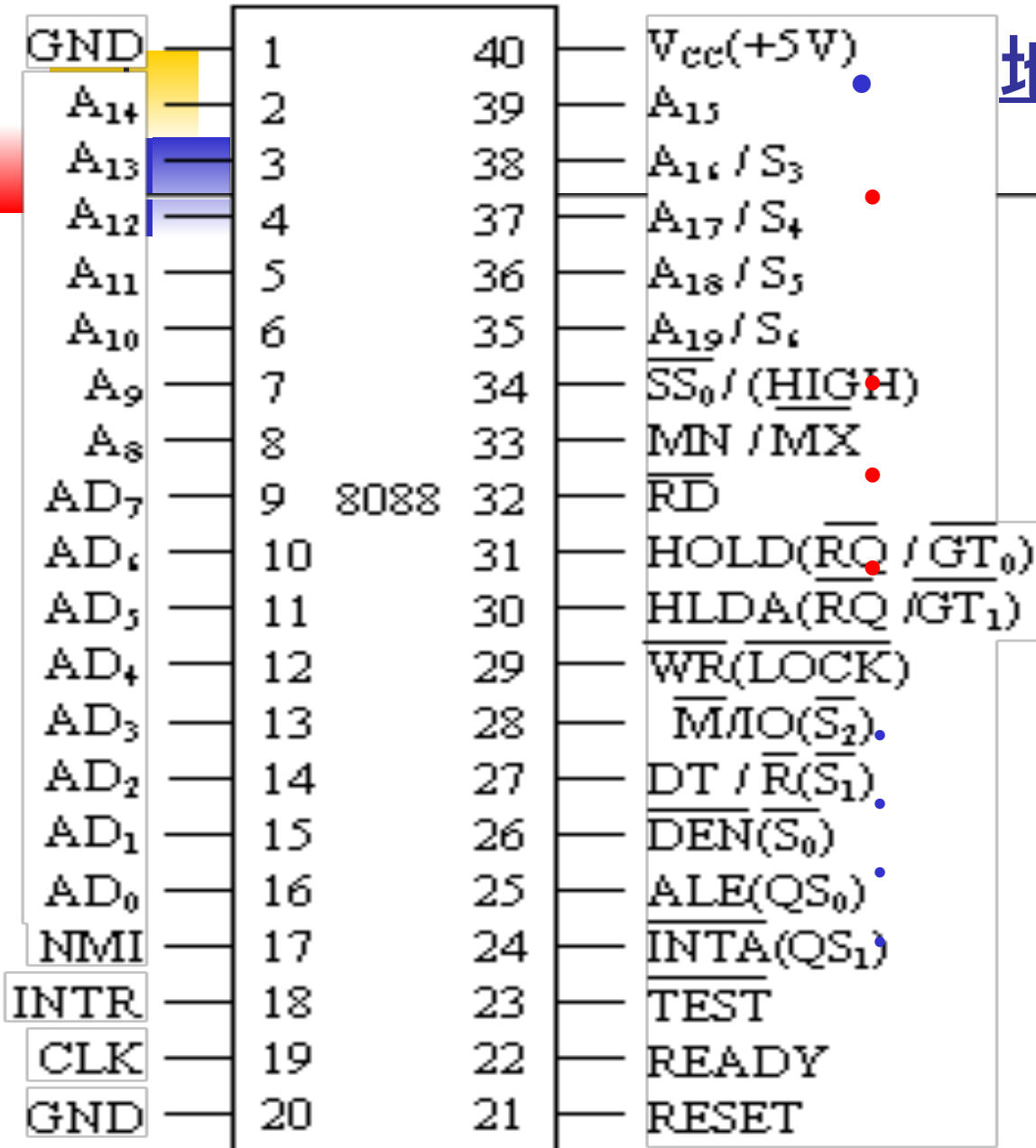
地址线和数据线：

AD7--AD0：低8位地址和数据信号分时复用。在传送地址信号时为单向，传送数据信号时为双向。

A19--A16：高4位地址信号，分时复用。

A15--A8：输出8位地址信号。

§2.4 8088CPU的引线及功能



地址线和状态线:

A₁₉—A₁₆ / S₃—S₆

分时复用、三态输出

S₆=0, 8088连在总线上;

S₅=1, 允许屏蔽中断;

S₄ S₃组合表示正在使用的段寄存器;

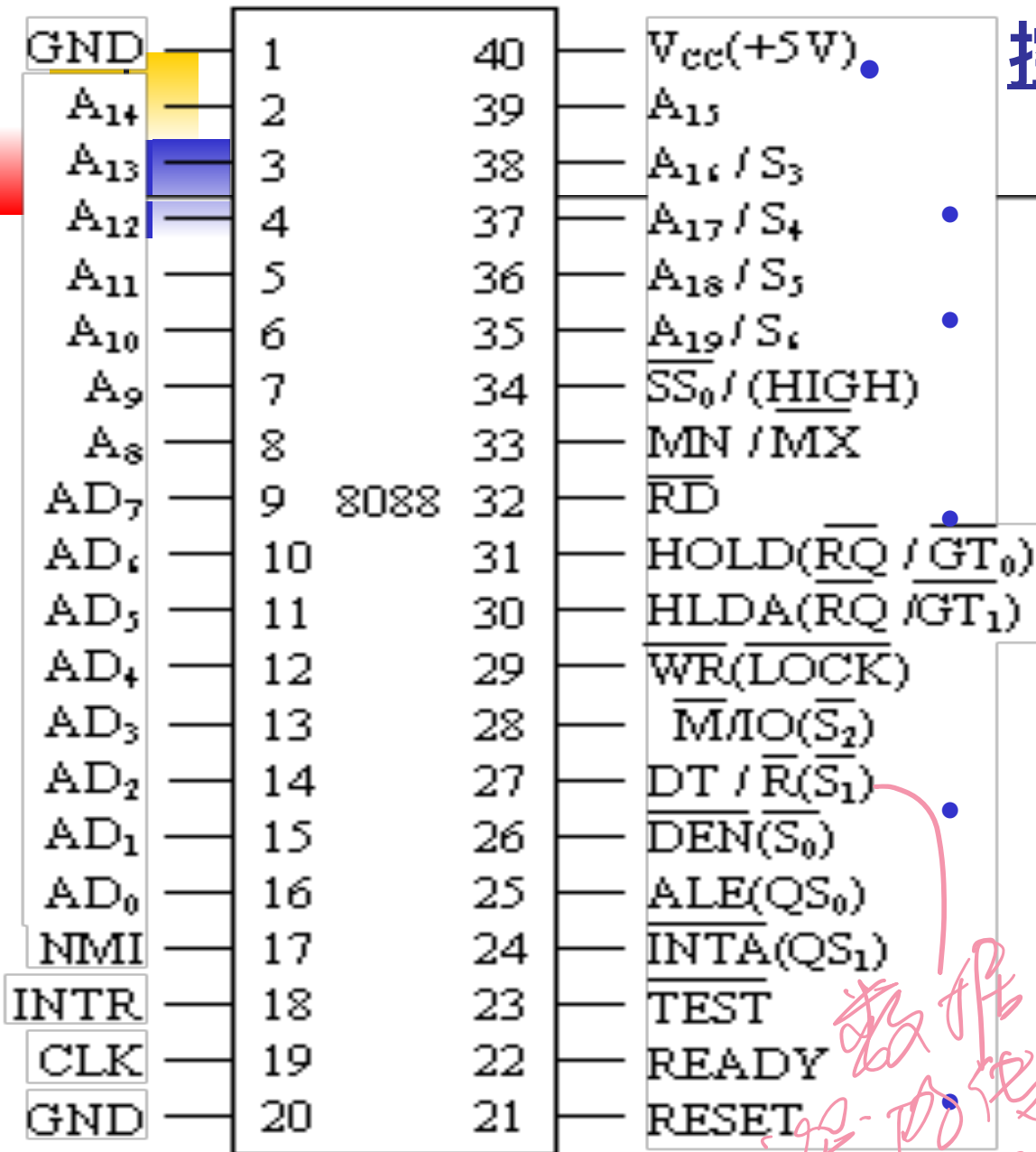
S₄=0, S₃=0, ES;

S₄=0, S₃=1, SS;

S₄=1, S₃=0, CS或未使用;

S₄=1, S₃=1, DS;

§2.4 8088CPU的引线及功能



控制总线—地址数据相关：

WR：写信号；三态输出

RD：读控制信号、三态输出；—

IO/M：为“0”表示访问内存，为“1”表示访问接口，三态输出；

DEN：低电平有效输出时，表示数据总线具有有效数据；

ALE：高电平有效输出时，表明CPU地址总线上具

数据收发器接口

地址总线

- 
- 当 $\overline{WR}=1$, $\overline{RD}=0$, $\overline{IO}/M=0$ 时,
表示CPU当前正在进行读存储器操作
-

- 当RESET高电平持续时间大于4个时钟周期, CPU产生复位状态;
- 恢复起始状态并重新启动, $CS=FFFFH$, DS 、 ES 、 SS 、 $FLAGS$ 、 IP 及其余寄存器清零, 指令队列清空

§2.4 8088CPU的引线及功能

GND	1	40	$V_{cc}(+5V)$
A_{14}	2	39	A_{15}
A_{13}	3	38	A_{14} / S_3
A_{12}	4	37	A_{17} / S_4
A_{11}	5	36	A_{18} / S_5
A_{10}	6	35	A_{19} / S_6
A_9	7	34	$SS_0 / (\overline{HIGH})$
A_8	8	33	MN / MX
AD_7	9	32	\overline{RD}
AD_6	10	31	$HOLD(\overline{RQ} / \overline{GT_0})$
AD_5	11	30	$HLDA(\overline{RQ} / \overline{GT_1})$
AD_4	12	29	$\overline{WR}(\overline{LOCK})$
AD_3	13	28	$\overline{MIO}(\overline{S_2})$
AD_2	14	27	$DT / \overline{R}(\overline{S_1})$
AD_1	15	26	$\overline{DEN}(S_0)$
AD_0	16	25	$ALE(QS_0)$
NMI	17	24	$\overline{INTA}(QS_1)$
INTR	18	23	\overline{TEST}
CLK	19	22	READY
GND	20	21	RESET

控制总线—中断相关：

INTR: 可屏蔽中断
请求输入、高电平有效

INTR=1, 外设提出
中断请求

NMI: 非屏蔽中断
请求输入、上升沿触发
与IF无关

INTA: 中断响应
输出, 低电平有效

§2.4 8088CPU的引线及功能

GND	1	40	$V_{cc}(+5V)$
A_{14}	2	39	A_{15}
A_{13}	3	38	A_{16} / S_3
A_{12}	4	37	A_{17} / S_4
A_{11}	5	36	A_{18} / S_5
A_{10}	6	35	A_{19} / S_6
A_9	7	34	$SS_0 / (\overline{HIGH})$
A_8	8	33	$\overline{MN} / \overline{MX}$
AD_7	9	32	\overline{RD}
AD_6	10	31	$\overline{HOLD}(\overline{RQ} / \overline{GT_0})$
AD_5	11	30	$\overline{HLDA}(\overline{RQ} / \overline{GT_1})$
AD_4	12	29	$\overline{WR}(\overline{LOCK})$
AD_3	13	28	$\overline{MIO}(\overline{S_2})$
AD_2	14	27	$\overline{DT} / \overline{R}(\overline{S_1})$
AD_1	15	26	$\overline{DEN}(S_0)$
AD_0	16	25	$\overline{ALE}(QS_0)$
NMI	17	24	$\overline{INTA}(QS_1)$
INTR	18	23	\overline{TEST}
CLK	19	22	READY
GND	20	21	RESET

控制总线—总线保持：

HOLD： 总线保持请求
信号输入，高电平有效。

当CPU以外的其他设备要求占用总线时，通过该引脚向CPU发出请求。

HLDA： 总线保持响应
信号输出，高电平有效。

CPU对HOLD信号的响



8088CPU的两种工作模式

- 8088可工作于两种模式下

最小模式

最大模式

- 最小模式为单处理器模式，控制信号较少，一般可不接总线控制器。
- 最大模式为多处理器模式，控制信号较多，须通过总线控制器与总线相连。

§2.4 8088CPU的引线及功能

GND	1	40	$V_{cc}(+5V)$
A_{14}	2	39	A_{15}
A_{13}	3	38	A_{16} / S_3
A_{12}	4	37	A_{17} / S_4
A_{11}	5	36	A_{18} / S_5
A_{10}	6	35	A_{19} / S_6
A_9	7	34	$SS_0 / (\overline{HIGH})$
A_8	8	33	MN / \overline{MX}
AD_7	9	32	\overline{RD}
AD_6	10	31	$HOLD(\overline{RQ} / \overline{GT_0})$
AD_5	11	30	$HLDA(\overline{RQ} / \overline{GT_1})$
AD_4	12	29	$\overline{WR}(\overline{LOCK})$
AD_3	13	28	$\overline{MIO}(\overline{S_2})$
AD_2	14	27	$DT / \overline{R}(\overline{S_1})$
AD_1	15	26	$\overline{DEN}(S_0)$
AD_0	16	25	$ALE(QS_0)$
NMI	17	24	$\overline{INTA}(QS_1)$
INTR	18	23	\overline{TEST}
CLK	19	22	READY
GND	20	21	RESET

控制总线—工作模式选择：

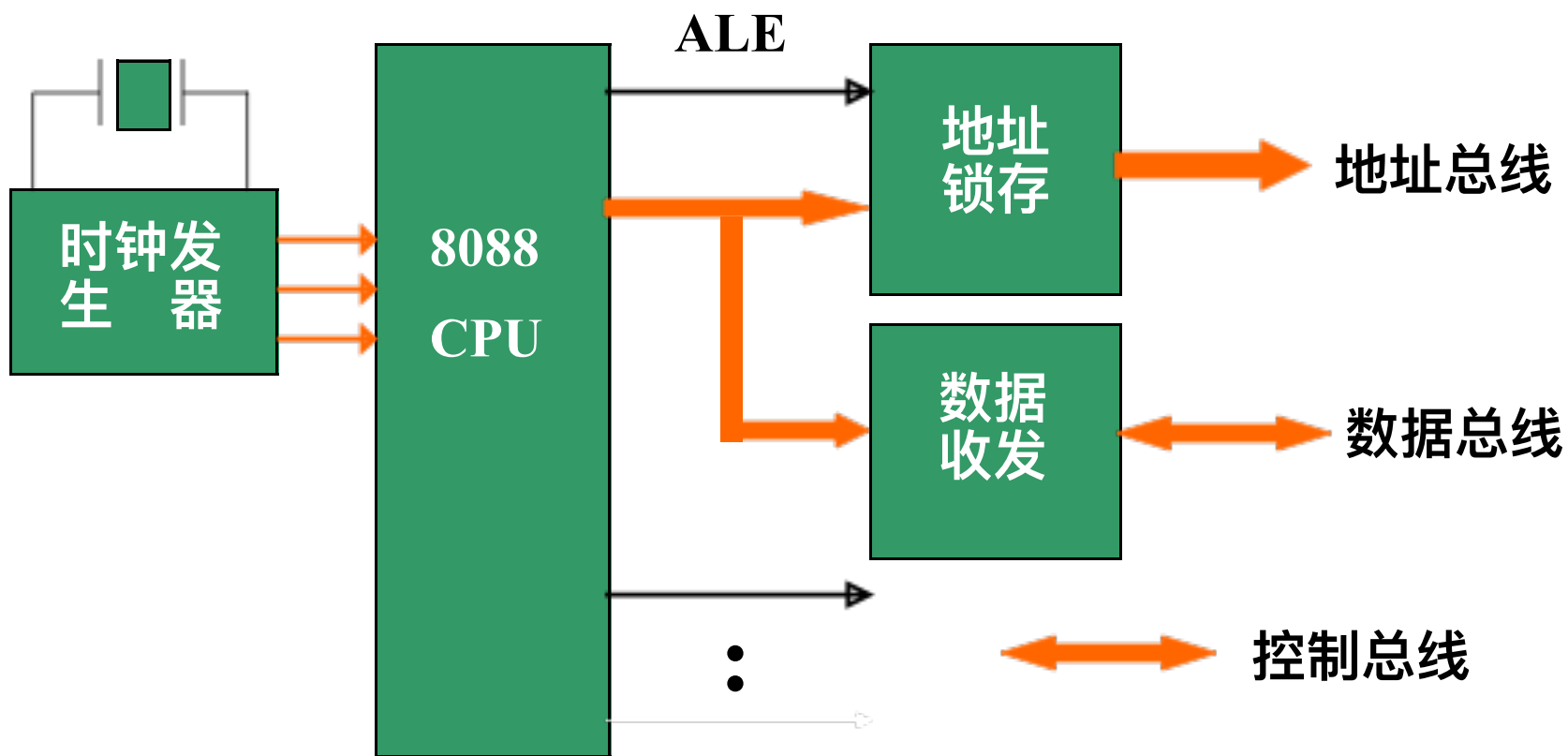
8088是工作在最小模式还是最大模式

由 MN/\overline{MX} 端状态决定。

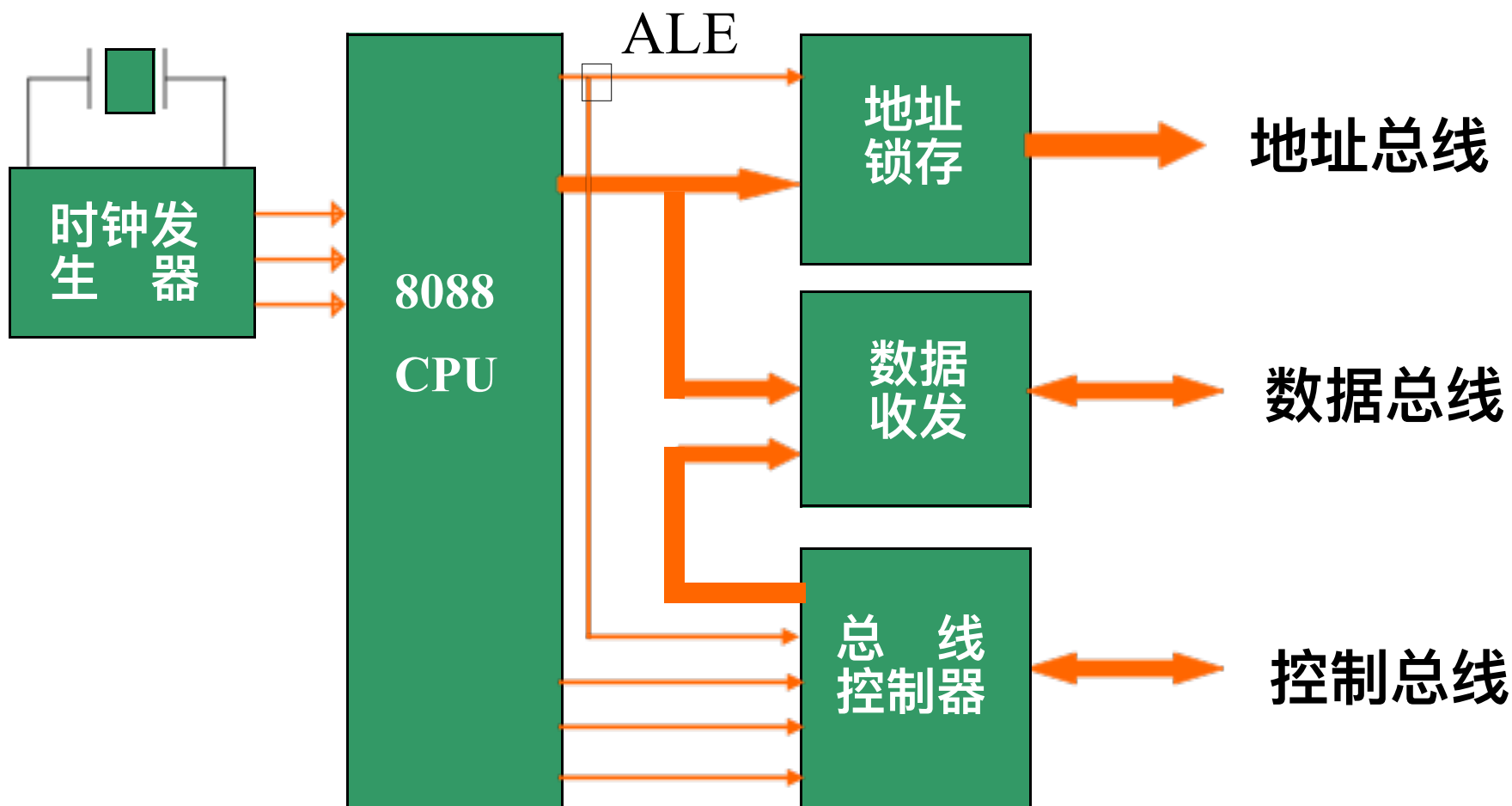
$MN/\overline{MX}=0$ 工作于最大模式，反之工作于最小模式。

最大最小只区别于24-34引脚括号部分

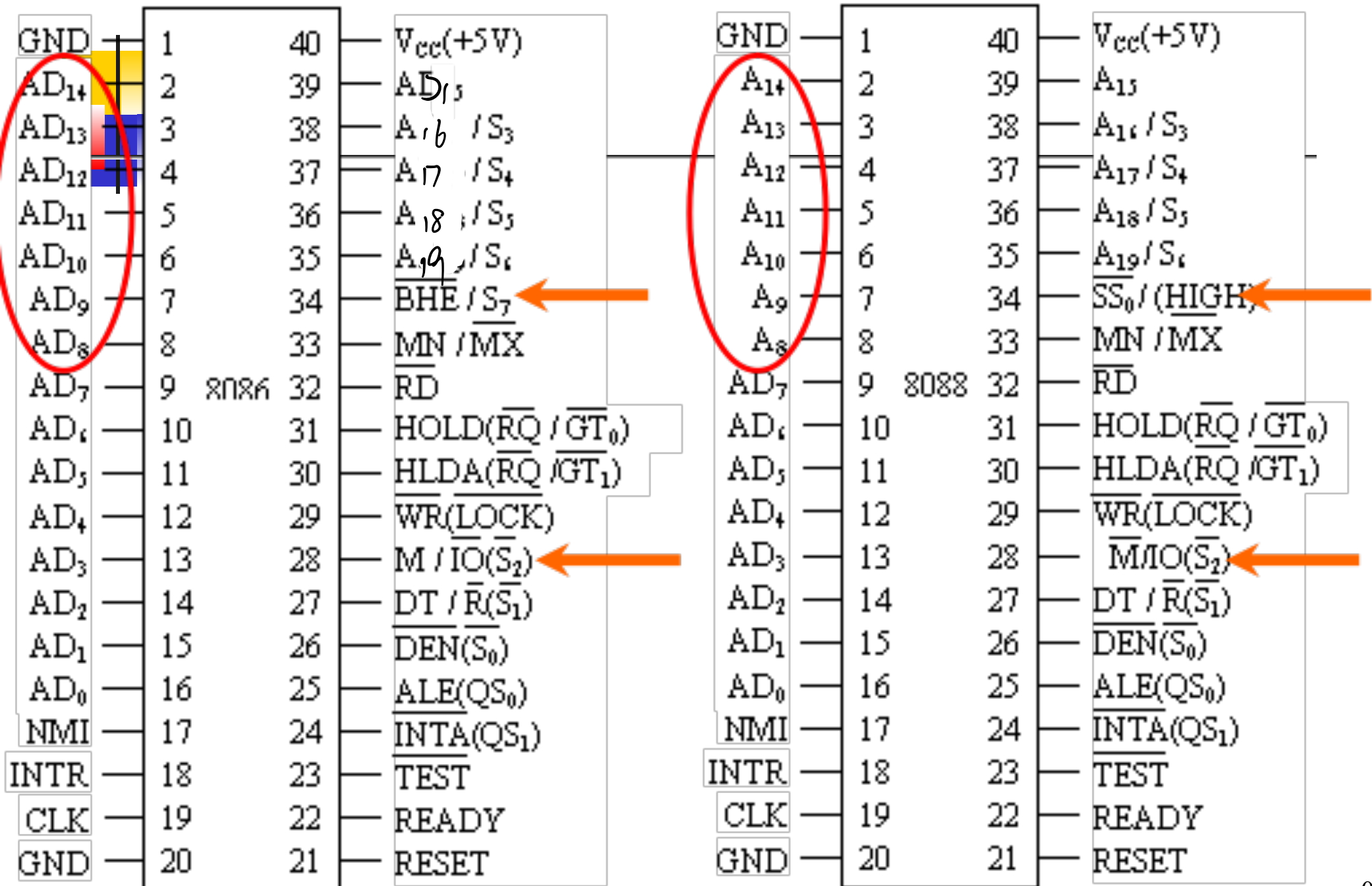
最小模式下的连接示意图



最大模式下的连接示意图



8086 v.s. 8088



8086 和 8086 CPU 引线功能引脚数

8088 外部总线宽度 8 位, 8086 是 16 位

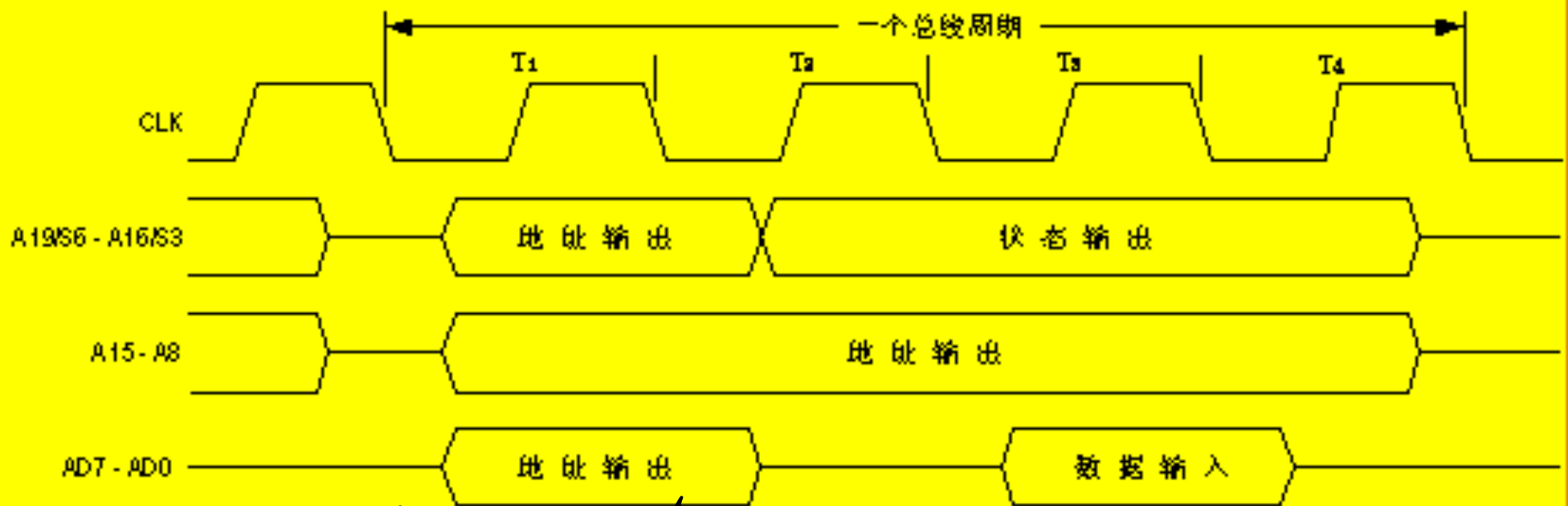
8088 — $IO/M = 0$ 表示访问内存

8086 — $IO/M = 1$ 表示访问内存



§2.5 8088总线操作时序

- **时序的概念**：CPU各引脚信号在时间上的关系
- **指令周期**：从取指令到执行完毕指令所需要的时间。
- **总线周期**：CPU从内存（或接口）存取一个字节操作所需要的时间。
- **时钟周期**：CPU的基本时间计量单位，由CPU主频决定。
- 一个**总线周期**至少包括4个时钟周期，每个时钟周期叫做一个T状态，T1、T2、T3、T4

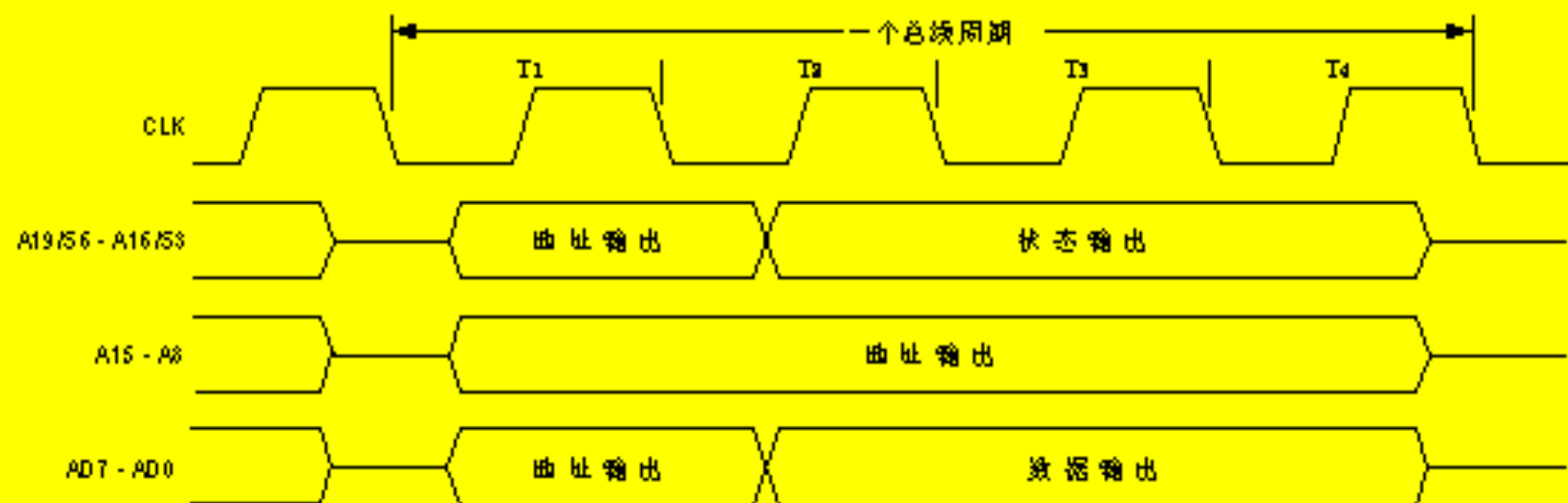


总线的主要性能指标

总线带宽 (B/s): 单位时间内总线上可传递的数据量

$$= \text{位宽} \times \text{工作频率}$$

位宽 (bit): 能同时传递的数据位数





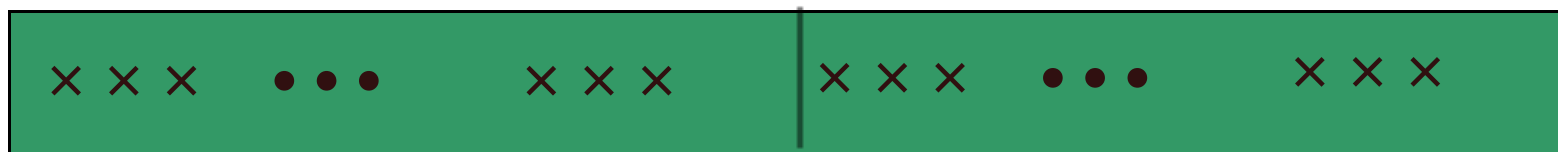
§2.6、存储器的组织与管理

- 存储器按**字节**组织；
- 每个存储单元存放**一个字节**的信息；
- 每个存储单元都有一个唯一的**20位地址**
编号，这个地址被称为内存单元的**物理**
地址。

1. 管理原则

8086 CPU 需要管理 1MB 内存

- (1) 分段管理 每逻辑段 64KB
- (2) 每段的段首地址能够被 16 整除
- (3) 每个内存单元的地址用逻辑地址来表示，由段基地址和段内偏移地址两部分构成，可以写成 (XXXXH: YYYYYH) 的形式
- $PA = XXXXH * 16 + YYYYYH$



段基地址 (16位)

段内偏移地址 (16)

段首的偏移地址

址:

0000H

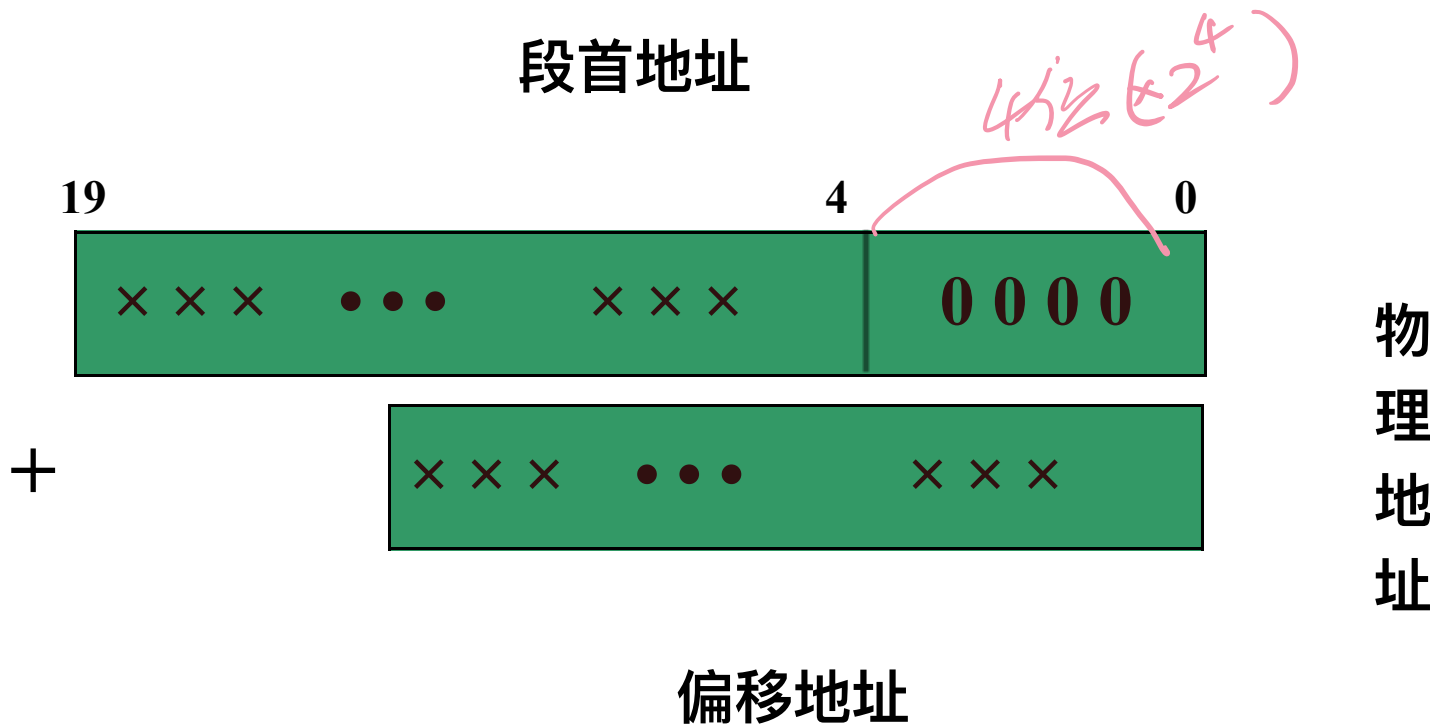
段首地址



段基地址 (16位)

- 物理地址由段基地址和偏移地址组成

段首地址



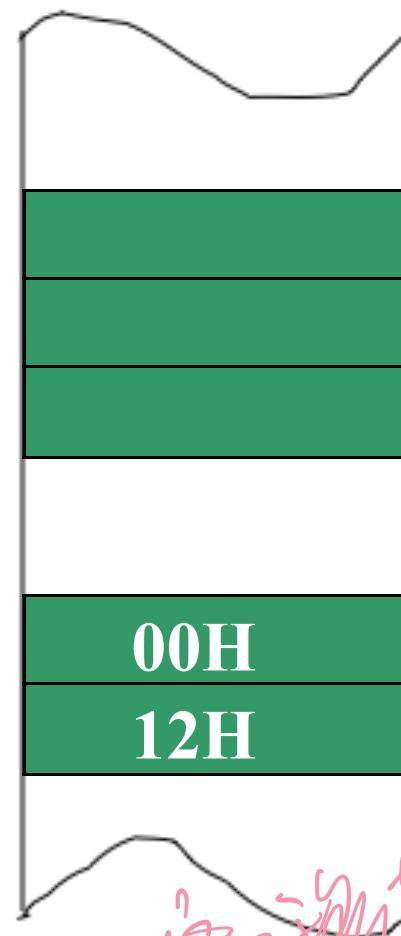
偏移地址

物理地址=段基地址×16+偏移地址

24

物理地址

- 段基地址 = 6000H
- 段首地址 \longrightarrow 60000H
段首地址的偏移地址为
- 偏移地址 \longrightarrow 0009H
- 物理地址 \longrightarrow 60009H
- 逻辑地址
(6000H: 0009H)



数据段

段基地址: 偏移地址

例

已知 CS=1055H, DS=250AH, ES=2EF0H
SS=8FF0H 某操作数偏移地址=0204H,

默认以64位

画出各段在内存中的分布、段首地址及操作数的物理地址。

设操作数在数据段, 则操作数的物理地址为:

$$250AH \times 16 + 0204H = 252A4H$$

10550H

CS

250A0H

2EF00H

DS

ES

8FF00H

SS



2. 组织原则

。两个逻辑段可先上或下设计为奇数
。两个不同程序模块装入主存时，同
一类型的逻辑段也可以装入相同
或不同的物理空间

- (1) 若存放8位字节信息，按顺序存放。
- (2) 任意两个相邻的内存单元都可以存放一个16位的数据，成为一个**字**；在一个字中，将字的低位字节存放在**低地址**上；高位字节存放在**高地址**上；每个字节都有相应的地址；低位字节的地址为**字地址**。
- (3) 若字地址为偶数，为规则存放，存放的字为规则字；反之，。。。

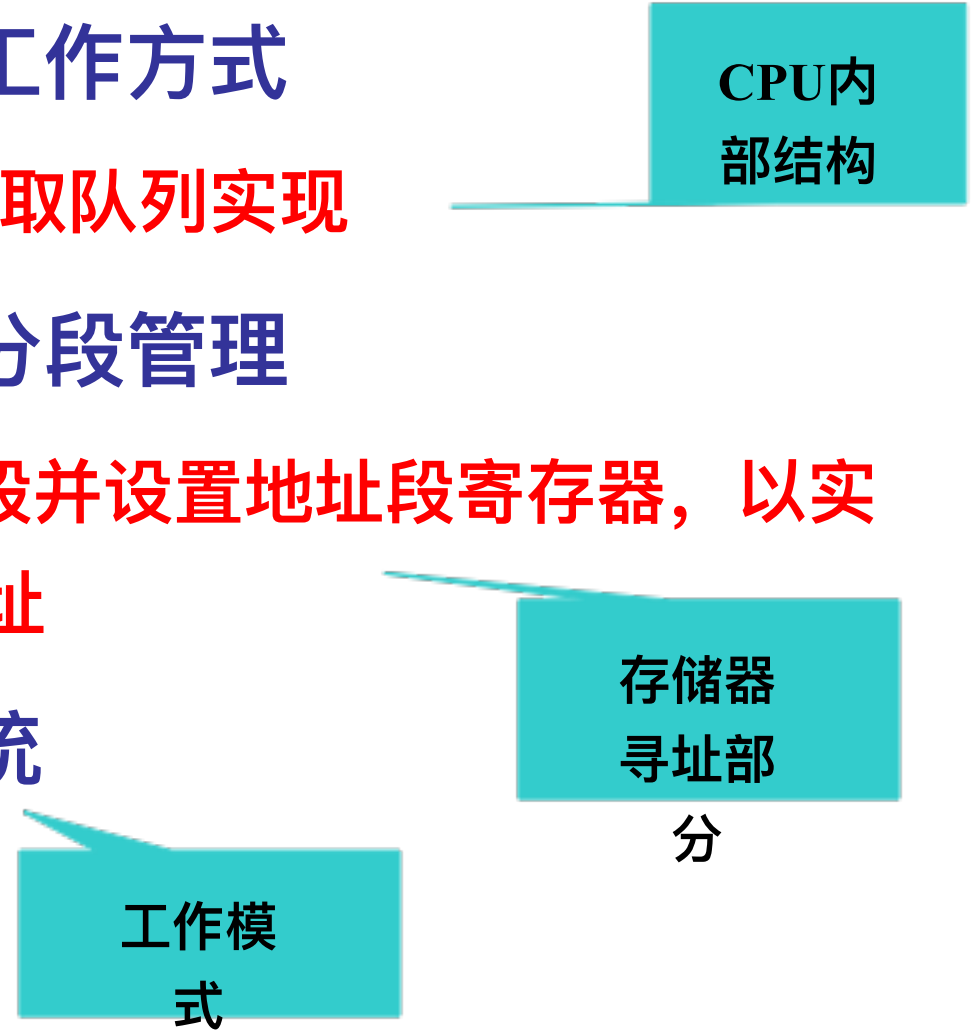


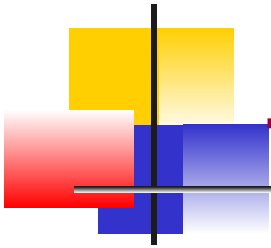
访问方式

- 对于8086-- 16位DB，均为字操作
 - (1) 访问字节，读取其所在偶数规则字的值，省去不需要的8位
 - (2) 访问字时，若为规则字，进行一次访问；若为非规则字，连续读写两个连续的偶地址字。省去不需要的两个半字信息。
- 对于8088—8位DB，均为字节操作，效率高



8088/8086 CPU的特点

- 采用并行流水线工作方式
—— 通过设置指令预取队列实现
 - 对内存空间实行分段管理
—— 将内存分为4类段并设置地址段寄存器，以实现
对1MB空间的寻址
 - 支持多处理器系统
- 
- CPU内部结构
- 存储器寻址部分
- 工作模式



主要内容：

- 微处理器的一般构成及工作原理；
- 8088微处理器的结构；
- 8088微处理器的内部寄存器；
- 8088微处理器的引脚；
- 8088微处理器对内存的管理；