船用微型计算机原理与接口技术
第二章知识整理

前言

个人整理总结，仅作为复习参考使用，并不权威，如有错误纰漏，欢迎指正，请联系 y0r4h21@whut.edu.cn 或 2124436512@qq.com。此文档完全免费。

可先自行了解程序与指令的定义，指令执行的一般过程，顺序执行和并行流水线工作方式，此处略过。

这章有大量的概念，很有可能我有整理掉的部分，如果可以的话自行翻翻 PPT 和书，本人先在这里道个歉

第一节 微处理器

微处理器，即 CPU，分为运算器、控制器、内部寄存器三部分

运算器

运算器一般可以分为三部分——算术逻辑单元（即 ALU），通用或专用寄存器，内部总线。顾名思义，运算器核心功能为数据的算术运算和逻辑运算。其中 ALU 负责“算”，寄存器负责暂存中间运算结果以及结果特征，CPU 内部总线负责传送数据和指令。

控制器

控制器负责控制程序执行，有指令控制、时序控制、操作控制三项基本功能。控制器由一下几部分组成

程序计数器 PC，PC 初始存放第一条指令地址，执行过程中自加 1 来存放下一条指令地址。

指令寄存器 IR，存放待执行指令。

指令译码器 ID。

时序控制部件。

微操作控制部件（核心部件）。一条指令由多个基本操作完成，基本操作被称为微操作，同时执行的一组微操作称为微指令，微操作控制部件便负责产生与各条指令相对应的微操作。

第二节 8088/8086CPU

8088CPU 和 8086CPU 都是 Intel 公司早期生产的 CPU，它们比较具有代表性。8088/8086 具有相同的指令系统与基本相同的硬件结构，此章若无特别指出，所述内容对两者均成立。

8088/8086CPU 的特点

1：采用并行流水线工作方式（通过设置指令预取队列实现）

2：对内存空间实行分段管理（将内存分为 4 个段并且设置地址寄存器，以实现对 1MB 空间的寻址）

3：支持多处理器系统

两种工作模式

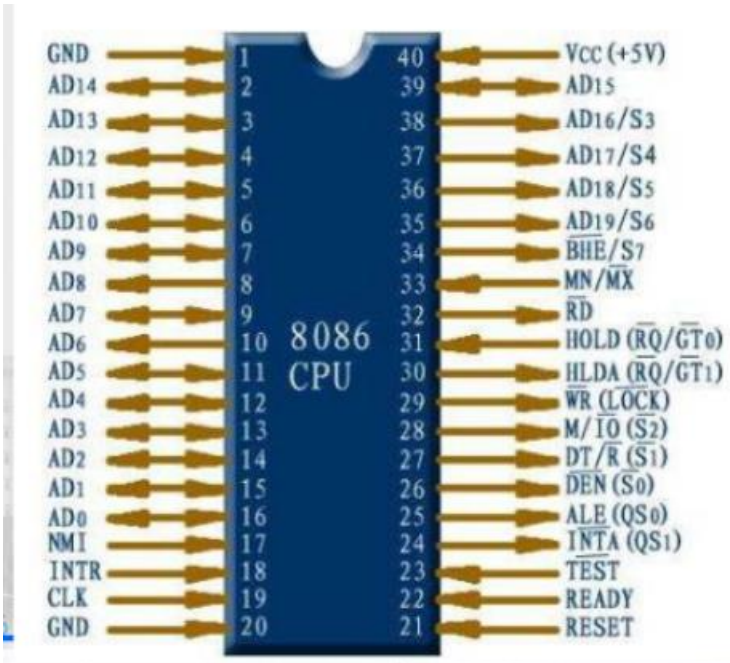
8088CPU 可以工作于最大、最小两种模式之下。

最小模式为单处理器模式，所有控制信号由微处理器产生；最大模式为多处理器模式，部分控制信号由外部总线控制器产生。

8086CPU 的外部引脚及其功能

如图所示，8086 有 40 条引出线，许多引脚具有双重功能，采用分时复用方式工作（分时复用即在不同时刻，引脚上的信号不同）（功能部分繁琐量大）

PS：由于符号上一横不好输入，这里把类似 33 号引脚符号上有一横的都写作例如 MN/MXh 形式

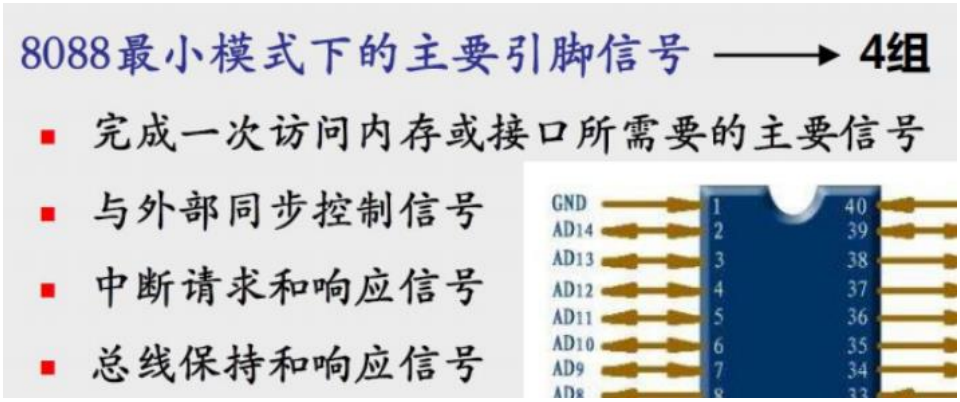


最大/最小工作模式由 33 号引脚 MN/MXh 控制，它等于 1 时处于最小工作模式，等于 0 时处于最大工作模式

引脚 24~34 根据 33 号引脚的状态存在不同

注：引脚符号上带一横其实是说它在低电平的时候起效，0 为低电平，1 为高电平（电平当做一种判断标识即可）。大部分信号在低电平时起效的目的是防干扰。

我们将引线分为 4 组，课程内容只有最小状态下的，最大状态下的可以自行了解。



第一组：地址线与数据线

1：16~9 引脚：AD0——AD7，低 8 位地址与低 8 为数据信号分时复用——当引脚 26 DENh=0 时传数据信号，当引脚 25 ALE（地址锁存信号）=1 时传地址信号

2：38~35 引脚：A16~A19/S3~S6，高 4 位地址信号/4 位状态信号，与状态信号分时复用

3：8~2 与 39 引脚：A8~A14 与 A15（我也不知道为什么 A15 要被分开），8 位地址信号

第二组：主要控制信号

1：21 引脚 RESET：复位信号

2：22 引脚 READY：外部同步控制输入信号，为“1”有效，有效时表示存储器或 I/O 设备准备好了

3：25 引脚 ALE：地址锁存信号

4：26 引脚 DENh：低电平有效，允许进行读或写操作

5：27 引脚 DT/Rh：数据收发器的传送方向控制，为“0”时 CPU 从存储器或 I/O 接口接受数据，为“1”时 CPU 向存储器或 I/O 接口发送数据

6：28 引脚 IO/Mh：输入输出/存储器控制信号，为“0”表示访问内存，反之表示访问接口

7：29 引脚 WRh：表示写入的信号

8：32 引脚 RDh：表示读取的信号

第三组：中断请求与响应信号（其实是第六章相关内容，因为不考所以理论上可以不看）

1：17 引脚 NMI：非屏蔽中断请求输入端

2：18 引脚 INTR：可屏蔽中断请求输入端

3：24 引脚 INTAh：中断输入响应端

第四组：总线保持信号

1：30 引脚 HLDA

2：31 引脚 HOLD

8088 与 8086CPU 之间引线功能略有差异。具体表现为：

1：数据总线宽度不同：8088 外部总线宽度为 8 位而后者为 16 位

2：28 引脚 IO/Mh 含义不同：8088 的 IO/Mh=0 表示访问内存，而后者的 IO/Mh=1 表示访问内存

3：其他部分引线功能不同：略

8088 内部结构

虽然前文所述微处理器由三部分组成，但 8088 内部其实多按两部分分，即 EU（执行单元）和 BIU（总线接口单元）两大部分

EU 由运算器、8 个通用寄存器、1 个标志寄存器、EU 部分控制电路组成，具有指令译码、指令执行（这两块在 ALU 中完成）、暂存中间运算结果（存在通用寄存器中）、保存运算结果特征（存在标志寄存器 FLAGS 中）四个功能。

BIU 有三个功能，分别是：从内存中取指令到指令预取队列（为并行流水线工作打下基础）、负责与内存或输入/输出接口之间的数据传送、执行转移程序时，BIU 使指令预取队列复位，从指定的新地址取指令并立即传给 EU 执行

结论：指令预取队列的存在使得 EU 和 BIU 两部分可同时工作，提高 CPU

效率、降低对存储器存取速度的要求。

内部寄存器（重要）

8086/8088CPU 的内部共有 14 个 16 位寄存器，按功能分为三类：8 个通用寄存器、4 个段寄存器、2 个控制寄存器（以下很多内容初次看可能会比较陌生，记得在接触到对应知识时再返回来看）

通用寄存器

通用寄存器有 4 个数据寄存器 AX、BX、CX、DX，2 个地址指针寄存器 SP、BP，2 个变址寄存器 SI、DI。不管如何，它们的通用功能都是暂存运算结果（即存放数据），接下来介绍它们的特殊功能。

AX：累加器。所有 I/O 指令都通过 AX 与接口传递信息，中间运算结果也多存放于 AX 中。

BX（base）：基址寄存器。在间接寻址中用于存放基地址。

CX（count）：计数寄存器。用于在循环或串操作指令中存放计数值。

DX（data）：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 位乘除运算中存放高 16 位数。

4 个数据寄存器由 8 个 8 位寄存器组合而成。

即 AX→AH、AL（high，low）；BX→BH、BL……

SP（stack pointer）：堆栈指针寄存器，其内容为栈顶偏移地址。

BP（base pointer）：基址指针寄存器。常用于在访问内存时存放内存单元偏移地址。

SI（source）：源变址寄存器。

DI（destination）：目标变址寄存器。　SI 和 DI 常用于存放数据在内存中的地址。

注：1：除 AX 和 CX 外，所有通用寄存器其实都可以存放地址。但是任何 8 位寄存器都无法存放地址，地址一定是 16 位的

2：BP，BX 作为通用寄存器，二者均可用于存放数据。但二者发挥属于基址寄存器的特殊用途时，用 BX 表示所寻找的数据在数据段，用 BP 表示数据在堆栈段

段寄存器

作用：用于存放相应逻辑段的段基地址

在 8088/8086 内存中，逻辑段类型有四种（有且仅有四种）

1：代码段——存放指令代码

2：数据段——存放操作的数据

3：附加段——存放操作的数据（你没看错，这个和上面的一样）

4：堆栈段——存放暂时不用但需保存的数据

段寄存器有 4 个：

CS（code segment）：代码段寄存器，用于存放代码段段基地址

DS（data segment）：数据段寄存器，用于存放数据段段基地址

ES：附加段寄存器，用于存放附加段段基地址

SS（stack segment）：堆栈段寄存器，用于存放堆栈段段基地址

在一个模块中，逻辑段个数可以很多（64K 个），但是逻辑段种类有且仅有 4 种。4 个段寄存器都是 16 位的，段寄存器的值表明相应逻辑段在内存中的位置。

控制寄存器

IP：指令指针寄存器，其内容为下一条要取的指令的偏移地址。

简单地说，8086/8088 中的 IP 就是程序计数器 PC（不知道这个东西的请回顾第一章内容）

FLAGS：标志寄存器，用于存放运算结果的特征。

FLAGS 整体上看是一个 16 位寄存器，但是它是按位工作，只有 9 位是有效的，剩下 7 位空闲。这 9 个有效位中有 6 个状态标志位 CF、OF、ZF、SF、PF、AF 和 3 个控制标志位 TF、IF、DF

CF（carry flag）：进位标志位，若（无符号数）最高位有进（借）位则 CF=1

OF（overflow flag）：溢出标志位，若（有符号数）运算结果超出有符号数可表达范围则 OF=1

ZF（zero flag）：零标志位，运算结果为 0 则 ZF=1

SF（sign flag）：符号标志位，运算结果最高位为 1 则 SF=1

PF（parity flag）：奇偶标志位，运算结果中的低 8 位中“1”的个数为偶数则 PF=1

AF（auxiliary carry flag）：辅助进位标志位，（8 位数之间）加减操作时若 D_3 位向 D_4 位有进（借）位则 AF=1

注：1：OF=1 也就代表符号数运算时最高位和次高位进（借）位状态不同（第一章内容）

2：CF,OF,ZF,SF 对 8 位数或 16 位数运算均适用。例如在 8 位数运算中，若 D_7=1 则 SF=1；在 16 位数运算中，D_{15}=1 则 SF=1

3：PF、AF 只对 8 位数运算有效，就算在 16 位运算中它俩也只看低 8 位

4：注意!!!!!! Dx 位不是第 x 位，一定要注意!!!!!! Dx 位其实代表的是第（x+1）位，因为数字的第一位是 D_0 位，就跟 C 语言里的数组从 0 开始一个道理

5：这里提到“有符号数”“无符号数”，但是其实是研究哪个标志位就按它的标准来。举例如下

10110110+11110100

研究 CF 时，就把这俩数都当成无符号数，发现最高位有进位，则 CF=1。

研究 OF 时，就把这俩数都当成有符号数，发现最高位（第八位）有进位，次高位（第七位）也有进位，则 OF=0（虽然看起来是溢出了一位但以计算结果为准）

TF（trap flag）：陷阱标志位，也称为跟踪标志位，TF=1 则使 CPU 处于单步执行指令的工作方式。

IF（interrupt enable flag）：中断允许标志位，IF=1 则使 CPU 可响应可屏蔽中断请求（“可屏蔽中断请求”是一个完整的名词）

DF（direction flag）：方向标志位，用以在数据串操作中确定方向。DF=1 则从高地址开始按减地址方式进行（每进行一次操作，地址指针自动减 1 或 2）；DF=0 则从低地址方式按增地址方式进行。

存储器寻址

内存储器管理

8088CPU 是 16 位体系结构的微处理器，可以同时处理（产生）16 位二进制码。理论上只能直接产生 64K 个（种）编码（注：16 个二进制位则有 2^16 种组合方式，1K=1024,2^16=64K），也就是直接管理 64K 个内存单元（也就是每种编码都分得一块地，CPU 能分出去多少块地也就是能直接管理多少个内存单元）

但实际上，8088CPU 管理 1MB 内存，这是用内存分段管理的方式实现的。

我们很容易想到，只要我们把 16 位改成 20 位，不就可以做到管理 1MB，所以一开始会想到一种暴力的方法——我们人为的把一大块内存划分，每一小块都对应 64K，这样只要划出 16 块就可以成 1MB 了，但这样做不符合 16 位 CPU 的设计（虽然我们确实是人为的划分了，但是 CPU 没法一次产生 20 位二进制位），也会造成内存分块零散（并不是每一小块内存都会被填满），内存利用效率低下。我们希望内存分块紧凑。

内存地址变换

由此，我们引入内存地址变换，也就是将直接产生的 16 位编码变换为 20 位的物理地址（可以说，在本课程中只要是 20 位地址一定是物理地址）

物理地址：每个内存单元在整个内存空间中具备的唯一地址。

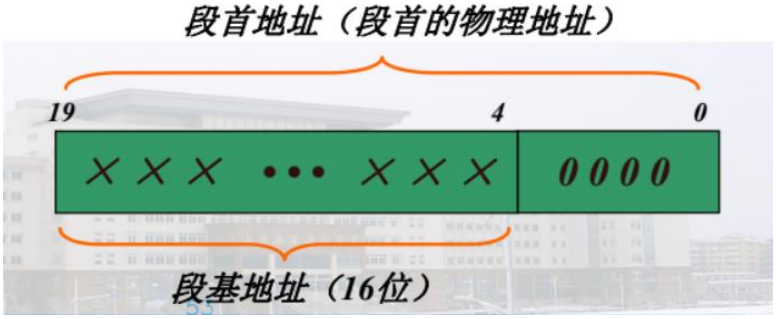
我们将每个内存单元的地址码划分为两个组成部分

即——16 位的段基地址+16 位的段内地址

段基地址：决定存储单元在内存中的位置

段内地址（相对地址\偏移地址）：决定该存储单元相对段内第一个单元的距离（为什么要这么做，自有妙用，请往下读）

我们将逻辑段的起始地址成为段首（这也正是每个逻辑段内第一个单元），显然，段首偏移地址为 0。在地址表示中，0 地址的位数是无关紧要的（也就是说 0000,00000000000000,0 这三个东西都表示 0 地址）。我们为了使用方便和符合设计要求，将段首的偏移地址取为 0000（准确的说是 0000B，可以认为这个是人为约定好的）。那么我们就将段首地址表示为了如下图



（吐槽：我怎么感觉段首地址和段基地址在某种意义上是一样的）

之后对于此逻辑段内其他内存单元，我们都是看它与段首的距离，因此可以表示为 段首地址+偏移地址 2，而段首地址=段基地址（16 位）+偏移地址 1（4 位，取 0000）这里我擅作主张，以我个人理解写为了“偏移地址 1、2”

有以下公式：

物理地址=段基地址×16+偏移地址

这里段基地址*16，其实就是在补偏移地址 1，即 0000B，由第一章中二进制的乘法，这里也就是移位加的思想。然后再加上偏移地址 2，最后得到物理地址。（这里我用我个人的理解讲了为什么要*16，仅供参考）

举例如下

设某操作数存放在数据段，DS=250AH，数据所在单元的偏移地址=0204H。则该操作数所在单元的物理地址为：

250AH ×16+0204H = 252A4H

首先复习一点，十六进制用 H 表示，在表述地址的时候，基本都是用 16 进制表示，由进制换算知识，1 个 16 进制位=4 个 2 进制位，这里的 250AH，其实就是 0010010100001010B。

段首地址便是 250AH*16=250A0H，加上偏移地址 0204H 后得物理地址 252A4H。

显然我们会发现偏移地址的范围是 0000H~FFFFH，由此有下面的例题

已知 CS=1055H，
DS=250AH
ES=2EF0H
SS=8FF0H
画出各段在内存中的分布。

段寄存器

略，见前即可

逻辑段与逻辑地址

- 内存的分段是逻辑分段，不是物理段。各个逻辑段在地址上可以不相连、可以部分重合，也可以完全重合。
- 每个内存单元具有惟一物理地址，但可能具有多个逻辑地址。即：

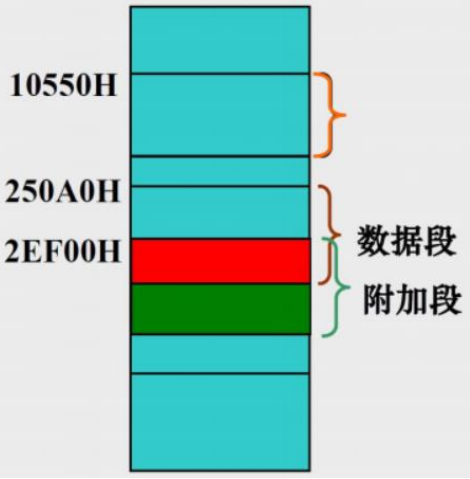
一个内存单元可以同时处于两个逻辑段

一个内存单元可以在不同的时刻属于相同（或不同）类型的段

一个内存单元在同一时刻可以属于不同类型的段

逻辑段说明

- 同一程序模块装入主存时，不同类型的段可以装入在相同/不同的物理空间
  - 两个逻辑段完全重合或部分重合
- 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中

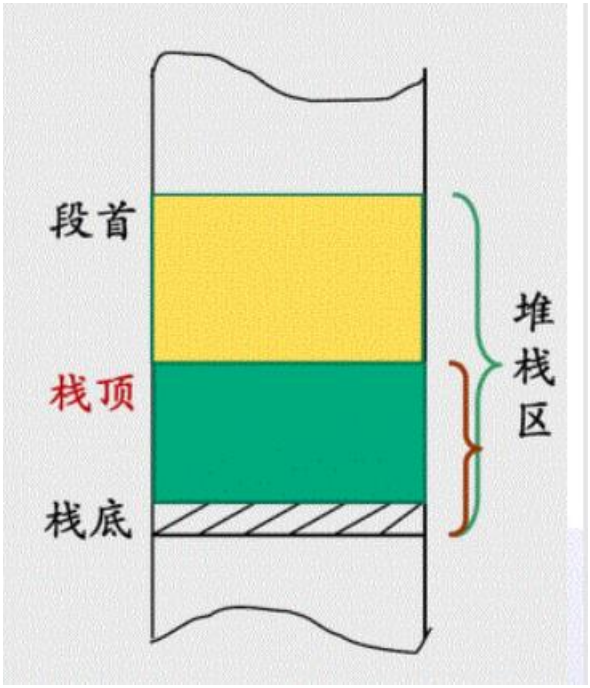


堆栈及堆栈段的使用

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

栈具有先进后出的特性（打个比方就是，你往一个空盒子里放东西，最先放的堆在了最底下，最后放的堆在了最上面，然后你往外拿的时候就是先拿最后放的东西）

一个栈的示意图如下



用 SP 寄存器，即栈顶指针来指向栈顶。（就把它当栈顶地址就好）
若栈顶=栈底则代表空栈。
若栈顶=栈首则代表满栈。（图里段首就是栈首）

8088 系统总线

总线时序

CPU 工作时序：CPU 各引脚信号在时间上的关系

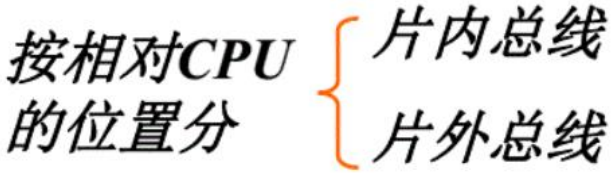
总线周期：CPU 完成一次访问内存或接口操作所需的时间。一个总线周期至少包含 4 个时钟周期

还有一些内容碍于本人能力，无法理解后表述，可自行查阅 PPT 与观看视频，在此提出抱歉。

总线

概念略，见第一章。
总线按层次结构分，可分为

总线按相对 CPU 的位置分，可分为



总线的系统结构

分为单总线结构，双总线结构（细分为面向 CPU 的双总线结构和面向主存的双总线结构），多总线结构。可观看视频进行细致了解。

吐槽一句：单总线结构双总线结构早就被淘汰了吧。。。

总线的基本功能

数据传送、仲裁控制、出错处理、总线驱动

常用系统总线

ISA（8/16位）

PCI（32/64位）

AGP（加速图形端口，用于提高图形处理能力）

PCI-E（PCI Express ）

目前最新的系统总线标准，采用串行方式传输数据，依靠高频率来获得高性能。

总线主要性能指标

总线带宽（B/S）：单位时间内总线上可传送的数据量

总线位宽（bit）：能同时传送的数据位数

总线的工作频率（MHz）

总线带宽=

（位宽/8）×（工作频率/每个存取周期的时钟数）