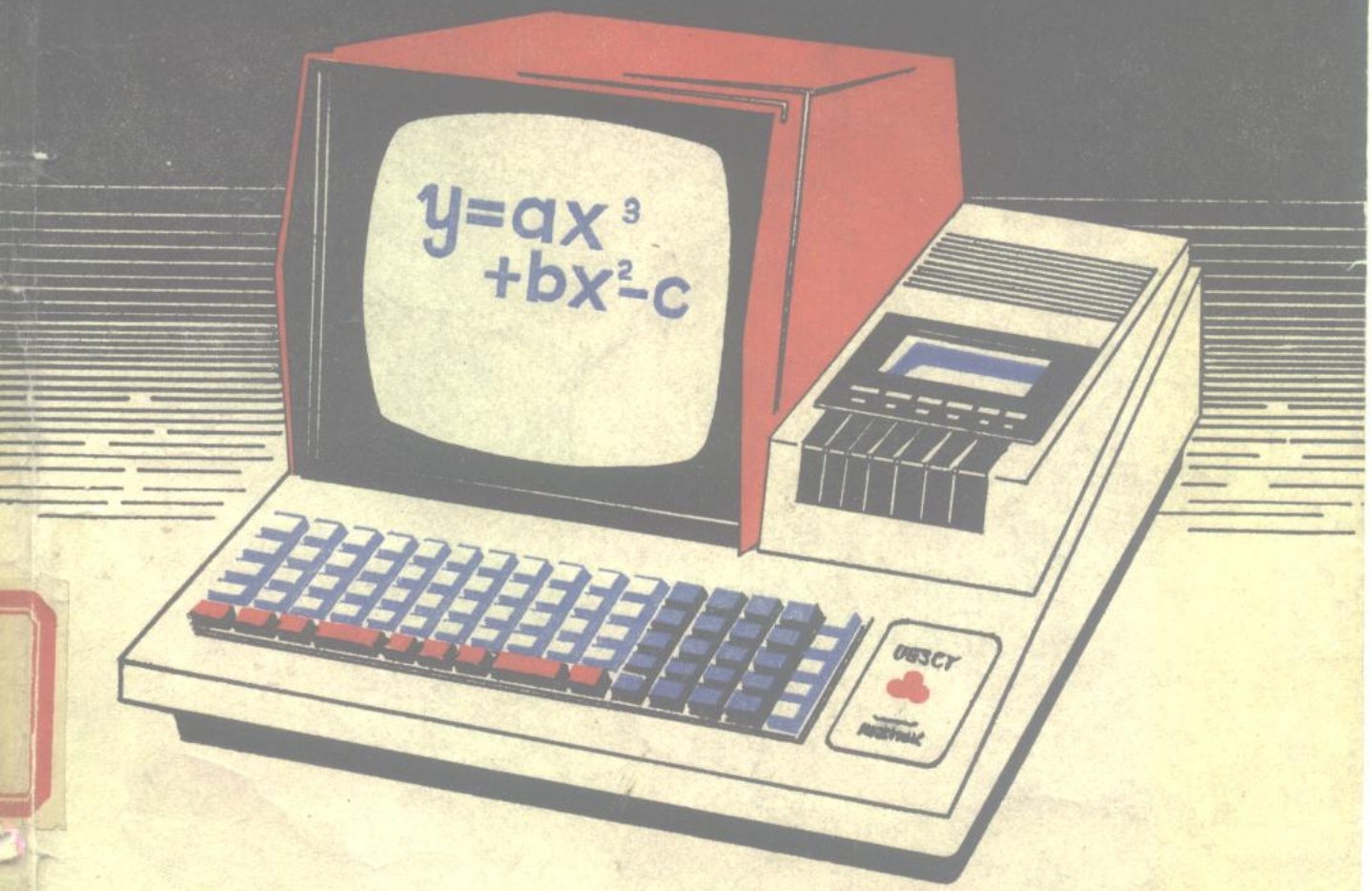
微型计算机通讯

第 一 集

WEIXING JISUANJI
TONG XUN



科学技术文献出版社重庆分社

微型计算机通讯（第一集）

中国科学技术情报研究所重庆分所	编　辑
科学技术文献出版社重庆分社	出　版
重庆市市中区胜利路91号	
四川省新华书店重庆发行所	发　行
重　庆　印　制　第　一　厂	印　刷

开本：787×1092毫米1/16	印张：7 字数：15万
1980年4月第一版	1980年4月第一次印刷
科技新书目：152—102	印数：9800

书号：15176·409　　定价：0.75元

目　　录

Motorola 68000微处理机的体系结构

E. Stritter, T. Gunter

微处理机工艺正在进入一个新的、向设计者提出更多的挑战的时代。尽管工艺上的限制尚未完全消除，但几乎达到了这样的程度：限制微处理机设计的因素不再是芯片内能包含多少功能，而是设计者能有多大的想象力和创造性[1]。因此，若干家公司都已引进了新的一代微处理机。本文旨在叙述其中之一——Motorola 68000 是如何应付这些特别情形的。

一、促成新的微处理机体系结构的原因

前几代微处理机均受到实际工艺的限制。Brook在一篇综述的文章[2]中讨论了早期的微处理机体系结构如何受到工艺的限制和受到所开拓的微处理机市场的影响。主要由于工艺水平的限制，无法在单块芯片上制作更多的东西，所以微处理机受到了寄存器数目、数据通路宽度和指令系统功能等方面的限制。微处理机还曾受到其它方面的限制，诸如地址空间太小[3]，地址计算不方便[4]等等。这不但是由于受工艺的限制，而且也可归因于对潜在的市场的主要看法[5]。尽管受上述种种原因所限，我们现在还是处在一个技术不断革新和激烈竞争的时期。

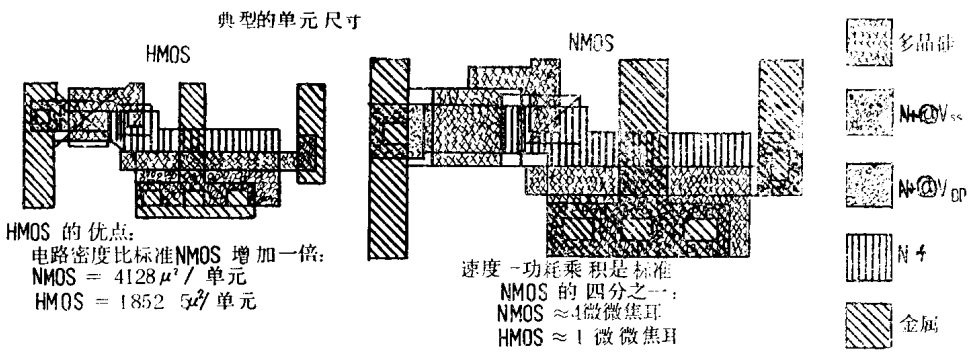


图 1　HMOS和NMOS工艺的比较。MC68000 采用的HMOS在电路密度和速度——功耗乘积等方面有显著的改进。

1. 工艺的进步

近几年来，基本的微处理机工艺-MOS工艺已取得了稳步的发展。最显著的进步是电路的密度提高了(图 1)，这可以直接看成是能集成在单片微处理机芯片上的功能量增加了。早先的微处理机每片容纳有5000～10000个晶体管，而现在的处理机芯片可容纳25000～70000个晶体管，这与许多最大的大型计算机的晶体管数相比只差一个数量级。工艺的进步不仅表现在电路密度的提高，而且在电路速度和功耗方面也取得了相应的改善。

工艺上取得的进步大部分是发展性的而

不是变革性的。主要的进步是电路密度增加了，这是由于加工工艺的不断改进，使得电路尺寸变小的结果。由于这一进步不是克服了基本的限制因素获得的，而只是靠对现有的加工工艺加以完善而取得的。因而，可以预期集成度还会不断提高。新的微处理机体系结构必须设计得可以利用将来的进步所提供的优点。

2. 市场的需求

直至几年以前人们还没有预见到在各种应用中对微处理机提出的要求给微处理机制造厂提供了新的机会。正象最初的微处理机设计者无法预计到他们的器件将会实现的许多用法一样，今天的设计者也只能想象出新的微处理机可能实现的用途中的几种。对设计者而言，其含意就是：如果希望新设计能适用于大量的可能出现的各种应用的话，这些新设计就必须灵活而通用。

3. 软件成本很高

微处理机的软件成本问题甚至比计算机中一般出现的软件成本问题更难解决。存储器的成本正在下降。处理机的功能正在增加，应用问题更为复杂，这些因素综合起来增加了微处理机程序的规模和复杂性。显然，十多万美元的软件成本与几百美元的硬件成本是不相称的。对于使用大量硬件的情况来说，这样悬殊的成本差异问题不大，因为软件成本可以由几千个硬件单元分摊；但是却常常使微处理机无法用于那些程序复杂而使用硬件数量不大的场合。为了有助于降低昂贵的软件成本，微处理机设计者必须求助于高级语言和进行程序设计的实践训练。

4. 设计成本很高

包含几万个晶体管的新器件的设计和投产成本很高。因而，计算机辅助设计是不可缺少的，但也是很贵的。设计者必须用各种办法来解决降低设计成本的问题。首先，采用规则的结构使得设计比较简捷，投产、测试和修正起来都比较容易，所以成本较之国外设计的来得低。其次，每个新的体系结构都必须尽可能考虑到以后的发展，即便于将来的扩展，使制造厂不再每隔几年就要搞一个新的体系结构。扩展和改进原来的八位微处理机体系结构的尝试经验表明，需要为将来的发展留有余地。设计者必须仔细地把将来可能限制其发展的因素排除在外。过去最常见的缺点是地址空间受到限制，且没有为将来的新指令提供备用的操作码。

5. 68000 的设计目标

莫托洛拉公司的68000微处理机的体系结构就设计得能够满足上述的那些要求（表1中归纳了MC68000的特性）。下面分别进行介绍。

表1 莫托洛拉公司 MC 68000 的特性

整数位数	8、16和32位
寻址能力	16777216字节
输入/输出	存储器映射
工　艺	HMOS
时钟周期	250 ns
存储器存取时间	500 ns
相对性能	6800的10～25倍
管脚数	64条
电　源	+5V

体系结构的系列

68000的设计形成了一种独特的计算机体系结构，根据这种结构能产生许多不同的形态或“实现方式”[6]。第一种形态就是MC68000，它实现的仅是整个68000体系中在目前的工艺条件下所能产生的一个子系统。

灵活性和实用性

68000的设计保证该处理机的程序很容易编制。尽可能在体系结构中剔除不合理的限制、人为的特殊情况或其它不适当的特性。

销路

68000微处理机的体系结构具有通用性，这反映了通用型微处理机在各种用途中不断增长的市场需求。

扩展性

68000的设计具有若干特点，例如：浮点运算和字符串操作。这些特点在第一种形态的68000微处理机上是实现不了的，而现在使它具有这些特点就能保证将来不会产生矛盾。此外，在体系结构中还保留了未用的空间，以适应将来工艺的发展可能产生的新的特点的需要。

能够使用高级语言

68000体系结构具有实现高级语言的特点，莫托洛拉公司已受到委托，提供满足高级语言的程序研制所需要的软件。

二、68000的内部结构

1. 资　源

68000的设计方案能提供一个有2^{32}字节（在最初的实现方案中为2^{24}字节）的地址空间。存储器可按字节寻址，利用位处理指令可寻址至某一位。可以以1、8、16或32位为单位对存储器进行存取。中央处理部件的资源包括十六个32位的寄存器、一个32位的指令（程序）计数器（在最初的实现方案中是24位）和一个16位的状态寄存器。

寄存器(图2)可分成两类。八个数据寄存器主要用于数据处理，这些数据可以成为所有各种操作中操作数的源或目的地址，而在寻址操作中只当作变址寄存器用。其余八个（地址）寄存器主要供寻址时用。堆栈指示器是地址寄存器中的一个。指令计数器和状态字寄存器都是独立的寄存器。

2. 寻　址

存储器是按8位字节、16位字或32位的“长字”进行逻辑寻址的。目前的实现方式要求字或长字数据是按字索取的。在“位”处理指令中也可单独对每一位分别寻址。

在68000的体系结构中采用了一种最佳的存储器管理方案，可以将地址空间分成长度不同的“段”，且为各个“段”规定存取权（访问权）。处理机可采用或不采用存储器管理。

地址计算（表2）方法由指令中的一个6位字段规定。指令的寻址规定与指令的操作规定无任何冲突之处；即在需要寻址的任何指令中可以使用任何寻址方式。

地址是一个32位的数据（目前的实现方式中是24位）。由于允许在几乎每一种寻址情况下规定、传送和计算16位的地址信号，所以这种体系结构可方便地构成小型系统(可寻址字节数小于2^{16})。例如，一条指令中所载的绝对地址可以用16位，也可以用32位；变址计算可以用一个寄存器的16位（符号位被扩展到24位）或32位作为输入。这个特点使得68000结构能满足地址空间很大的

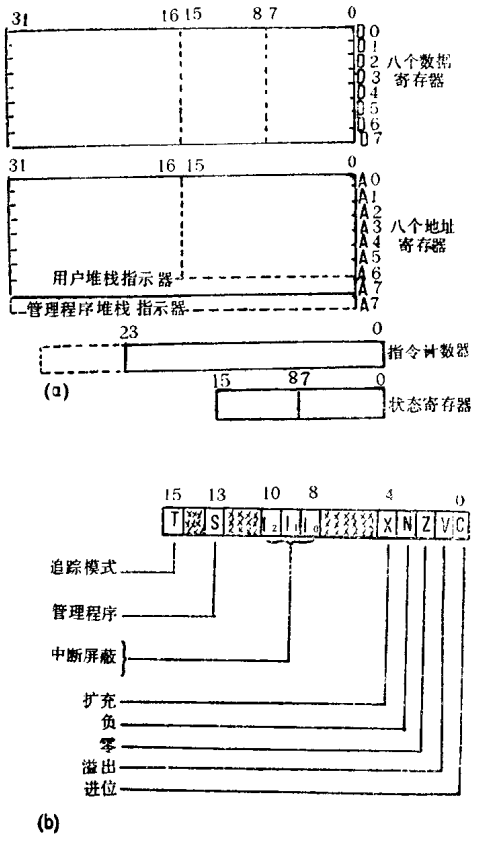


图2　(a) MC68000的程序编制模型
(b) 状态寄存器的内部结构

要求，而又不会降低仅需很小的地址空间的程序的效率。地址位数（16位或32位）是每次使用时单独规定的，所以程序中大地址和小地址可以任意混合。

表 2　　MC 68000 的 寻 址 方 式

寄存器直接寻址：	
数据寄存器直接寻址	EA=Dn
地址寄存器直接寻址	EA=An
状态寄存器直接寻址	EA=SR
寄存器间接寻址：	
寄存器间接寻址	EA=(An)
寄存器间接、后增量寻址	EA=(An)；　An←An+N
寄存器间接、先减量寻址	An←An−N；　EA=(An)
基址相对寻址	EA=(An)+d16
变址寻址	EA=(An)+(Xn)+d8
指令计数器相对寻址：	
带偏移地址相对寻址	EA=(PC)+d16
相对变址寻址	EA=(PC)+(Xn)+d8
短 PC 相对转移寻址	EA=(PC)+d8
长 PC 相对转移寻址	EA=(PC)+d16
绝对寻址：	
短绝对寻址	EA=（下一个指令字）
长绝对寻址	EA=（下面两个指令字）
立即数据寻址：	
立即寻址	数据=下一个或两个指令字
快速立即寻址	数据=指令的一个子字段（4位）

定义：

EA=有效地址　　An=地址寄存器

Dn=数据寄存器　　Xn=用作变址寄存器的地址或数据寄存器

SR=状态寄存器　　PC=指令计数器

d8=8位的位移量　　d16=16位的位移量

N=操作数为字节、字、长字时分别为1、2、4。　　(　　)=……的内容

←=取代

可采用的寻址方式有下述多种：

寄存器直接寻址　操作数存放在数据或地址寄存器中。

地址寄存器间接寻址　操作数的地址存放在规定的地址寄存器中。

地址寄存器间接、后增量寻址　操作数的地址存放在规定的地址寄存器中。在存取操作数以后，该寄存器中的地址递增1、2或4，可视操作数长度而定。

地址寄存器间接、先减量寻址　操作数的地址存放在规定的地址寄存器中。在存取操作数之前，地址寄存器的内容先减去一个相当于操作数长度的数（1、2或4）。

基址相对寻址　操作数地址是规定的地址寄存器的内容加上指令中的16位带符号的位移量。

指令计数器相对寻址　操作数的地址是指令计数器现行值加上指令中的16位带符号的位移量。

变址寻址　操作数地址是规定的地址寄存器的内容加上另一个规定的（数据或地址）寄存器的内容，再加上指令中一个8位的带符号位移量。

指令计数器变址寻址　操作数地址是指令计数器的现行值加上规定的数据或地址寄存器的内容，再加上指令中一个8位的带符号位移量。

绝对寻址　操作数地址就在后面的指令字中。

立即寻址　操作数就在指令字中。

位寻址　提供了一整套位处理指令（SET, CLEAR, CHANGE和TEST）。这些指令用上述各种寻址方式之一找到存储器中的某一个字。再根据某一位在该字中的位号找出所要进行处理的那一位。位号在指令中规定或先在一个数据寄存器中计算好。利用这一机构对“位”进行寻址就很简单，不要求使用逻辑指令和屏蔽字。对寄存器而言，可分别对所有32位的每一位进行寻址。

在所有的情形下，程序规定的地址都能复盖整个地址空间。没有任何的段长度方面的限制，也没有独立的段号需要处理。

地址和整数一样是一类具有全套操作方式的数据。在地址寄存器中可执行一整套地址处理操作（MOVE, COMPARE, INCREMENT, DECREMENT, ADD TO, SUBTRACT FROM）。此外，LOAD EFFCTIVE ADDRESS（装入有效地址）指令能完成任意的地址计算，并将其结果送入一个规定的地址寄存器，这使得程序员只用一条指令就可用处理机的任何寻址方式预先计算好地址。

因为有八个地址寄存器，所以对地址值进行暂存时就很少需要进行存储器存取操作，在程序的各个不同部分中也很少需要重新对地址进行计算。这些特点减少了花在处理地址上的程序时间，而所有地址处理在现有的微处理机中都要通过这个公共的“狭道”。这些特点还能使地址位数有一定的独立性[7]。尽管实际规定的可能是一个大地址（32位），但是指令中的地址字段却常常只有6位。

3. 数据处理

68000能满足许多数据类型的需要，为每种数据类型提供了一整套操作（表3和图3）。一般地说，寻址方式与数据类型无关。在处理整数、逻辑和地址数据时，也可以规定操作数的位数。源操作数既可在寄存器中也可以在可寻址的存储单元中，其结果既可存入寄存器中也可存入规定的存储单元中。这类“寄存器—存储器”操作减少了保存结果所需的寄存器存储操作的次数。绝大多数操作都可规定为完成存储器至寄存器、寄存器至寄存器、寄存器至存储器、立即数至寄存器或立即数至存储器等类型的操作。传送指令则更灵活，是一个完整的双地址指令。除了规定上述各种类型的操作外，还可规定存储器至存储器的传送操作。

68000采用的数据类型以及对这些类型的数据能进行的操作如下：

整数　其操作是加、减、乘、除、取负、比较和算术移位。整数可以是1、2或4个字节。移位是多位一起移位，既可左移又可右移，移位次数在指令中规定或预先在一个数据寄存器中计算好，如果需要的话，还可指示出溢出状态。

多精度整数　为便于实现多精度整数算术运算而提供的基本操作是：带扩充加、带扩充减、带扩充取负、无符号乘和无符号除。除了乘、除以外，操作数可以是1、2或4个字节，而乘、除法可能对二字节操作数进行运算。

逻辑　其操作包括“与”、“或”、“异”、取反、比较、移位和循环移位（可进行多位置的移位和循环移位，可左移亦可右移，可带可不带扩充）。可对1、2或4个字节进行逻辑操作。

布尔运算　其中包括“与”、“或”、“异”、取反、隐含和按条件码进行置位等。“按条件码进行置位”用于检索条件转移指令所执行的任何条件测试的逻辑

值。布尔数据是单字节量。

位 其操作是置位、清除、改变和测试。各“位”可独立进行寻址。

十进制数 十进制数的操作是加、减、取负和比较。十进制（BCD）指令对存储器中的操作数进行操作（存储器至存储器），每次两个数字（一个字节）。十进制指令与循环指令相结合能实现可变字长的存储器至存储器的十进制操作。

字符 字符指令有传送和比较两种，对存储器中的操作数进行操作（存储器至存储器）。

地址 地址操作包括增量（1、2或4）、减量（1、2或4）、加整数、减整数、比较和传送有效地址。

实数 安排有浮点加、减、乘和除等操作，但是在早期的器件中尚无法实现。

字符串 安排有字符串传送、字符串检索和转换等操作，但在早期的器件中尚无法实现。

表3 68000的指令系统

助记符	说明	记助符	说明
ABCD	带扩充的十进制加法	MOVE	传送
ADD	加法	MULS	带符号的乘法
ADDX	带扩充的加法	MULU	不带符号的乘法
AND	逻辑与	NBCD	带扩充的十进制数取负
ASL	算术左移	NEG	2的补数
ASR	算术右移	NEGX	带扩充的2的补数
BCC	条件转移	NOP	无操作
BCHG	位测试和改变其值	NOT	1的补数
BCLR	位测试和清除	OR	逻辑或
BRA	无条件转移	PEA	推入有效地址
BSET	位测试和置位	RESET	外部设备复位
BSR	转子程序	ROTL	不带扩充的循环左移
BTST	位测试	ROTR	不带扩充的循环右移
CHK	检查寄存器是否超出范围	ROTXL	带扩充的循环左移
CLR	清除操作数	ROTXR	带扩充的循环右移
CMP	算术比较	RTR	返回和恢复
DCNT	减量和不为零则转移	RTS	子程序返主
DIVS	带符号的除法	SBCD	从扩充部分中减去十进制数
DIVU	不带符号的除法	SCC	有条件地置位
EOR	异	STM	存储多个寄存器
EXG	寄存器交换	STOP	停机
EXT	带符号的扩充	SUB	减法
JMP	跳转	SUBX	带扩充的减法
JSR	跳转至子程序	SWAP	数据寄存器两半部分交换
LDM	存入多个寄存器	TAS	测试和设置操作数
LDQ	快速送入寄存器	TRAP	陷阱
LEA	送入有效地址	TRAPV	溢出陷阱
LINK	通过堆栈进行连接	TST	测试
LSL	逻辑左移	UNLK	解除堆栈连接
LSR	逻辑右移		

4. 程序控制

程序控制指令包括条件转移（指令计数器相对寻址）、跳转、跳转至子程序、子程序返主和中断返回，全都是传统的指令。用陷阱指

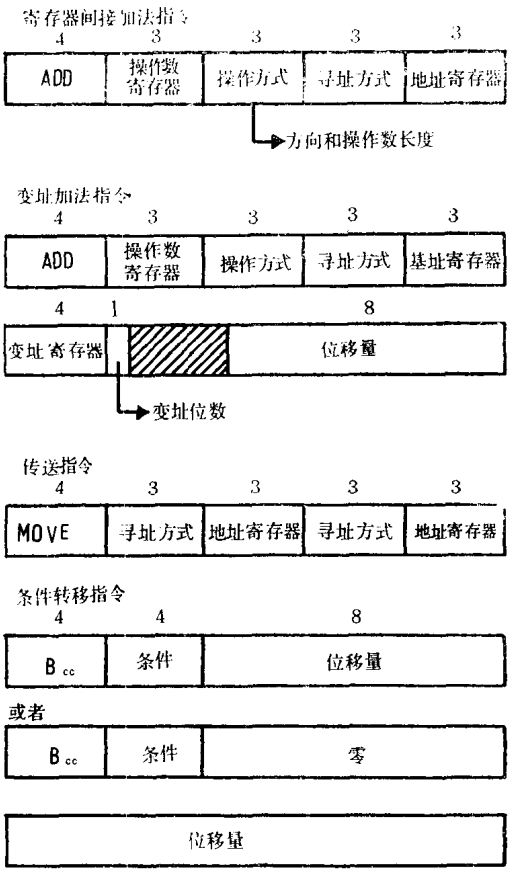


图3 典型的MC68000指令格式

令可规定十六个独立的操作系统调用操作。下面将会讨论条件陷阱、循环和子程序控制。“停机”指令使处理机停机；“复位”指令能重新预置系统；“传送”指令能对处理机状态字进行操作。

5. 特许状态

MC 68000处理机能在用户状态和管理程序状态下工作。在管理程序状态下，整个指令系统都可使用。对外界给出相应的内部状态指示。例如，当处理机处于管理程序状态时，则可禁止地址变换。在用户状态下不允许使用某些指令（例如：STOP、RESET以及修改状态字的那些指令），否则会造成对管理程序状态的陷阱（将现行指令字和状态字推入堆栈，从预定的陷阱矢量中取出新的内容打入指令计数器和状态寄存器）。非法指令、不可实现的指令、中断和陷阱（作为系统调用操作）都会使处理机引起陷阱并切换至管理程序状态。为了保证返回管理程序状态时能正常工作，无论是否处于用户状态，都有两个堆栈指示寄存器——一个在用户状态时工作，另一个在管理程序状态下工作。通过专用指令可以将用户堆栈指示器的内容加到管理程序状态下的程序中去。

MC 68000用户/系统状态的这种差别，使得用一个很小的操作系统的核心部分，可为若干个独立的任务或用户提供受到可靠保护的虚拟地址空间。

对非法和不可实现的指令进行陷阱使得操作系统能为用户状态的任务提供一个功能更强的虚拟机。例如，操作系统可用软件来实现用户状态的任务中任何目前尚不能实现的指令（例如浮点或字符串处理指令）。

6. 高级语言的软件后援

最近，Allison提出了一种方法[8]，即将微处理机体系结构设计得能符合高级语言的要求。这种方法（已为MC 68000的设计员所采用）是“检查…执行这类语言所要求的运行时间表示法”，“提供适当的指令…以满足所要求的运行时间表示法”和“勿需很多的联机计算”就能对该运行时间表示法实现变换。

MC 68000的设计考虑到高级语言的软件后援。它以清楚而协调的指令系统、硬件实现常用的功能（乘、除和地址计算）以及为适应处理高级语言程序的运行时间的要求而设计的一套专用指令，从而使其在编译时间和执行时间上都能符合高级语言的要求。借助这些专用指令所能实现的高级语言中的语句有：数组存取、有限精度的算术运算、循环、布尔表达式求值和过程的调用。下面

分别加以说明：

数组存取 边界检查（BOUND CHECK）指令将预先计算好的数组标引（存在一个数据寄存器中）与零和由指令进行寻址的一个限定值进行比较。若标引超出了该数组的边界，则发生陷阱中断。这样，用一条指令就代替了常用的一组指令（至少四条）。

有限精度算术运算 若前一次操作产生了溢出，则溢出陷阱指令会造成一次陷阱中断。这可用有效的溢出测试功能协助对算术运算的结果进行适当的检查。

循环 FOR循环语句的一种约束形式可用一条指令予以实现，该指令使一个计数值减去一个量，若结果不为零则转回至循环处。

布尔表达式的求值 有条件置位指令可在条件转移指令所用的那些条件下为一个布尔变量赋于“1”或“0”值。由于这些指令不会产生额外的条件转移，因此有助于布尔表示式的求值，特别是在可能受边界效应影响而不希望出现“短路”求值的情况（正象用Pascal语言那样）下尤其如此。

过程的调用 MC68000用堆栈（以地址寄存器之一——称为堆栈指示器的地址寄存器进行指示）建立被调用过程的嵌套环境。用三条指令（每个参数还得附加一条指令字）就可实现一次高级语言程序的过程调用（图4）。整个调用机构只用到堆栈，而且是完全可以重入的（图5）。下面将对这些指令进行详细介绍。

将参数值或地址压入堆栈 MOVE指令可将一个值压入堆栈；PEA（压入有效地址）指令（见前面有关“送入有效地址”指令的说明），则可将任意的地址计算的结果压入堆栈，供访问时调用。

建立新的局部环境 LINK指令完成下述所有的操作：保存堆栈上帧指示器（一个

SAMPLE 程序：
  程序实例；
  变参1，变参 2；整型；
  过程PROC（X：整型；变量Y：整型）；
    变量A，B；整型；
    BEGIN
    〈过程体〉
    END；
    BEGIN
    PROC（形参1，形参2）
    END
程序体；
  MOVE 参数1至堆栈指示器@“压入第一个参数”
  PEA 参数2“压入第二个参数的地址”
  JSR PROC“调用过程”
  ADD #6至堆栈指示器“从堆栈弹出参数”
过程体；
  LINK FP,4“连接和分配三个局部变量”
  MOVEM〈寄存器表〉至堆栈指示器@“还原寄存器内容”
  UNLK FP “还原堆栈内容”
  RETURN “返回到调用过程”

图4 用Pascal语言写的程序实例及其对应的MC68000代码

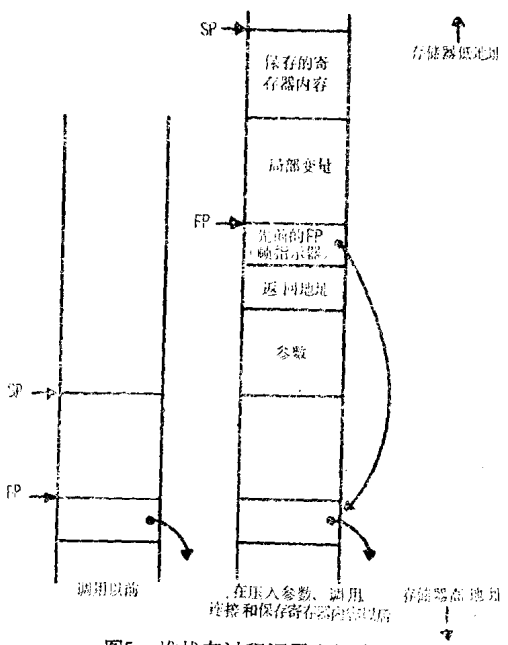


图5 堆栈在过程调用中的作用

调用过程 “跳转至子程序”指令将返回地址压入堆栈，并跳转至过程的入口。

任意的地址寄存器）原来的内容，使帧指示器指向新的栈顶，从堆栈指示器中减去该过程所要求的局部存储用的字节数。这就为调用过程建立了局部存储区，并为局部变量和参数的变址寻址提供了一个帧指示器（地址寄存器）。

将任意一组寄存器的内容保存在堆栈　“多个寄存器传送”(MOVE MULTIPLE REGISTER)指令可用一条指令将任意一组寄存器的内容保存在堆栈（或存储器中任何地方）。内容需要保存的寄存器是通过在指令的位字段中设置相应位来指示的。

下面四条指令形成的一组指令构成相反的过程，供过程返回用：

被保存的寄存器内容重新打入原寄存器　这里还是用多个寄存器传送指令。

重新建立先前的环境　“解除堆栈连接”(UNLINK)指令能解除 LINK(连接)指令所做的工作。

从过程中返回　返回(RETURN)指令从堆栈中弹出返回地址，并返回至调用过程去。

从堆栈中弹出参数　对堆栈指示器运用“加立即数”(ADD IMMEDIATE)指令，就可将任何数目的数值弹出堆栈。

三、MC68000的系统结构

计算机的体系结构是通过确定中断结构、存储器分段、总线接口和输入/输出结构等环节来规定处理机与周围环境之间的交互作用的。MC 68000的系统结构设计得尽可能灵活。例如，就象其它的莫托洛拉微处理机一样，输入/输出(I/O)设备的寄存器可象存储单元一样寻址(存储器“映射”I/O)。存储器“映射”I/O可为程序员进行设备控制和设备数据寄存器操作提供整个指令系统所具有的灵活性和能力。由于I/O不需要其它的指令，所以处理机就较为简单，指令系统也容易记住。存储器的重要区域用一个存储器保护装置来保护，也同样用它来保护I/O空间。

设计 MC 68000的总线结构时亦力求简单、高速和灵活性。地址线和数据线是分开的，不必进行多路转换。因此不需要任何信号分离设备去进行多路分配，确保在速度要求很高的系统中能有最好的性能。总线是异步的；总线上的传送过程由相应的信息交换信号进行控制，因此不必对时序或系统同步性作任何假设。采用信息交换信号就使得能在同一条处理机总线上连接若干个响应时间相差很大的设备和存储器。处理机可以等待任意长的时间，直至被访问的设备或存储器发出信号表示传送已实现了为止。

芯片上安排了简单的总线请求/允许规约，使得处理机和直接存储器存取设备能协调地共享系统总线，而不必增加任何逻辑电路。而且，芯片上还有一个“总线出错”管脚，能在进行非法的或错误的存储器存取操作时终止指令的执行，并进行一次陷阱中断。这为存储器保护提供了方便。

MC 68000的中断结构与绝大多数小型计算机的中断结构一样。中断有八个优先级，并能实现矢量中断，使软件对中断处理子程序的位置及其执行过程能进行完全的控制。处理机当前的优先级存在其状态字中，能够禁止那些与当前的优先级同级或级别更低的中断。但是可以发生较高级的中断，因此中断处理可以嵌套。当有一个允许发生的中断时，处理机发出一个响应信号。中断的设备就以一个矢量数作为回答。处理机用这个矢量数作为对存储器低地址处一张中断矢量表进行检索的索引，找出适当的中断处理程序的入口，这样的矢量有256个(图 6)。同一优先级上的各个不同的设备可通过不同的矢量数加以区别，所以不需要进行定时查询。软件陷阱和处理机中的异常状态也可通过矢量表来转接，在这种情形下，矢量数则由处理机安排。矢量表存放在主存储器中，所以必要时可由操作系统进行处理。处理机能实现一组不执行的矢量（每级优先级一

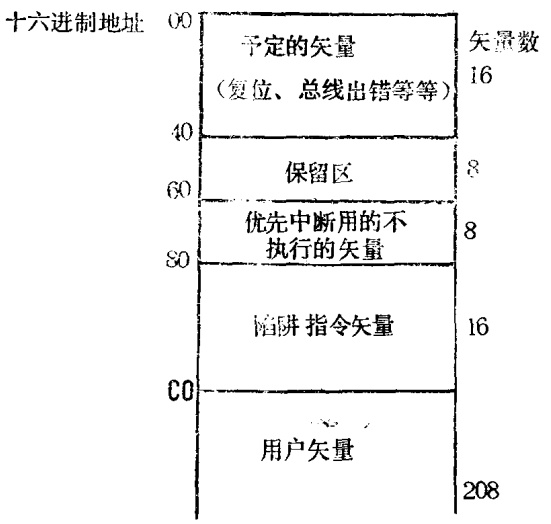


图 6　陷阱和中断矢量的分配

个），这就使得现有的外围设备(即使未装备以矢量数作出响应的机构）也都能使用。

可把处理机直接接于存储器来构成68000系统。这时，程序产生的地址就是实际地址。这对于许多用途而言已是足够了。更复杂的应用（特别是那些多任务，甚至多用户的情况)则需要更复杂的存储器管理手段。可以另外用一片单片器件来实现存储器分段、地址变换和存储器保护等功能。

四、MC68000的设计及其实施

单片 MC 68000 微处理机（图7）部分地实现了68000体系结构，实现了完整的体系结构中尽目前工艺水平所能达到的最大的子系统。由于工艺的局限性而限制了管脚数和电路密度。目前的封装技术将使每个组件的管脚数限制在64个以下，因此使得地址范围限制在24位。同样，到存储器的数据通路也只有16位宽。这不是系统结构上的限制，但是却因此而使得存取一个32位的数据需要进行两次存储器存取操作。

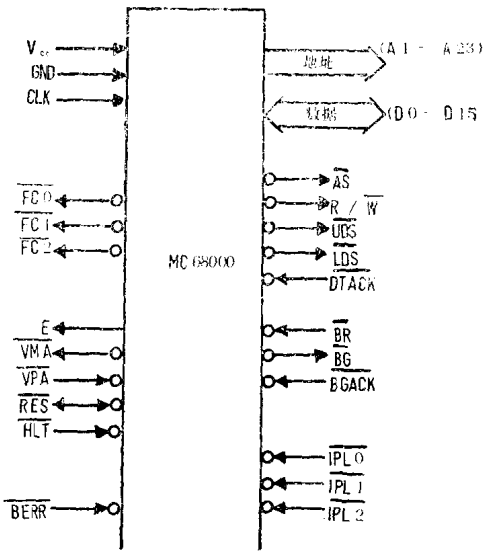
电路密度限制了可实现的指令条数。通常，操作码组合中有八分之一由于受电路密度限制而实现不了。这些空白指令中有一些在68000体系结构中已作了安排，例如，供浮点和字符串操作用。还有一部分暂且不作规定，供将来系统结构扩展时用。所有不可实现的指令都会引起陷阱，因此实现软件仿真是可能的。

将来，这种系统结构的实现方式可在性能和功能方面向上发展或向下缩小。由于工艺的进步，将很快会实现整个体系结构。当电路密度进一步提高时，新的微处理机将更快和更小(因而也更便宜)，功耗也更小。电路密度增加后还可将存储器制作在同一芯片上，并采用复杂的高速技术。

现在 MOS 大规模集成电路的工艺水平几乎达到了在每平方密耳的电路面积上可制作一个晶体管的程度，且逻辑门可设计得使速度功耗乘积达 1×10^{-12} 焦耳。68000的设计选择了一种先进的高密度n沟道硅栅MOS工艺。这种工艺能满足 3 微米器件尺寸的要求，并可向设计员提供多个 MOS 晶体管阈值电压。这种工艺使电路设计员能采用最小尺寸的器件研制高性能的逻辑门和低功耗的内部缓冲电路。

执行单元采用双总线结构，用来处理地址和数据(图 8)。这两条总线都是 16 位的，如果需要的话，每条线都可由微指令动态地重新组成为三个互相独立的部分。用三个独立的算术单元完成这些计算，而且还为执行长移位、优先级编码和位处理提供了专门的逻辑功能。这些单元都与两条内部总线相接，从寄存器同时接收两个操作数。每条总线都有原码和反码两种逻辑值，因此可用不同的电路设计实现较高速的操作。执行单元直接与外部总线逻辑电路和缓冲电路相接，而它的操作与总线的外部时序要求无关。

68000 的控制是通过微指令实现的。微程序控制器的实际结构在其它文章[9]中已作了详细讨论。通过两级控制结构实现微控制。第一级存放着具有短“垂直”格式和复杂的转移能力的微指令序列。含有次微指令地址、宽“水平”控制字的微指令则存在第二级中。由次微指令直接控制执行单元。由



$A_1 \sim A_{23}$	地址线	23位地址总线，与UDS 和 LDS 结合能寻址16777216字节。
$D_0 \sim D_{15}$	数据线	16 位数据总线，传送 8 位或16位的信息。
AS	地址选通	表示地址有效，并为不可分的操作提供总线锁定功能。
R/W	读/写	定义总线操作是读还是写，并控制外部总线缓冲器。
UDS, LDS	数据选通	按R/W和AS的状态识别出要进行操作的字节。
DTACK	数据传送响应	使总线周期与慢速设备或存储器同步。
BR	总线请求	请求总线的设备送至处理机的输入信号
BG	总线允许	处理机发出的允许总线仲裁的输出信号。
BGACK	总线允许的响应	由BG产生的批准信号，表示仲裁过程的选择有效。
IPL0 IPL1 IPL2	中断优先级	向处理机提供中断功能的优先级。
FC0 FC1 FC2	功能码	向外部设备提供有关当前总线周期和处理机状态的信息。
CLK	时钟	输入给处理机的TTL主时钟。
RES	复位	向处理机和外围设备提供复位（预置）信号。
HLT	停机	停止处理机工作，允许单步工作。
BERR	总线出错	若无响应或收到无效的响应，则提供一个总线周期结束信号。
E	选通	向6800系统提供的选通时钟。
VPA	有效外围地址	确定地址区域与6800兼容
VMA	有效存储器地址	向6800系列的设备指出总线上是一个有效地址。
V_{cc}	+5V（两条引脚）	
	地（两条引脚）	

图 7　68000 管腿符号及其定义。该微处理机封装在64条腿的管壳内，因此地址和数据总线都是独立的（不进行多路转换）

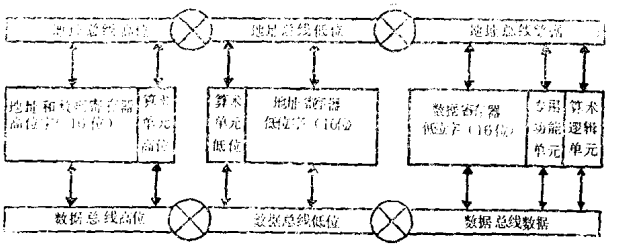


图 8　MC68000 执行单元的结构

于考虑到新的超大规模集成电路芯片的设计成本很高，所以采用了微指令。与组合逻辑电路相比，微指令的结构很有规律，显著减小了设计的复杂性。采用微指令也使某些工程上的决定（例如专用指令的细节）可推迟做出。换言之，在微机器的结构决定以后，可同时进行硬件（电路设计）和固件（微程序设计）的设计工作。

五、结　　论

Motorola 68000的体系结构是先进的工艺与设计者对微处理机用户和微处理机应用在结构方面的要求有了较深的理解相结合的产物。68000 是进入先前由高档小型计算机所占据的领域的第一步。它是一个32位的结构，能满足许多数据类型和数据范围的需要。68000 的优点是寻址机构灵活、指令系统简单而有效（可以很容易地实现复杂的操作）、具有多级矢量中断结构以及快速、异步、勿需多路转换的总线结构。68000 体系结构代表了一个为开拓高档微处理机市场而设计的微处理机系列。

参　考　文　献

1. J. R. Rattner, "Microprocessor Architecture——Where Do We Go From Here", COMPCON Spring 1977 Digest of Papers, pp223—224。
2. F. P. Brooks, "An Overview of Microcomputer Architecture and Software", Proc. EUROMICRO 1976, North Holland, pp. 1—6。
3. C. G. Bell and W. D. Strecker, "Computer Structures: What have We learned from the PDP-11?", Proc. 3rd Symposium on Computer Architecture, 1976, pp. 1—11。
4. L. A. Levanthal and W. C. Walsh, "Addressing Considerations in Microprocessor Design", COMPCON Spring 1977 Digest of Papers, pp. 225—229。
5. B. L. Peuto and L. J. Shustek, "Current Issues in the Architecture of Microprocessors", Computer, Vol. 10, №2, Feb. 1977, pp. 20—25。
6. S. A. Ward, "Toward the Renaissance Computer Architecture", MIDCON 1977 Preprints, pp. 1—6。
7. P. E. Stanley, "Address Size Independence in a 16—bit Minicomputer", Proc. 5th Symposium on Computer Architecture, 1978, pp. 152—157。
8. D. R. Allison, "A Design Philosophy for Microcomputer Architectures", Computer, Vol. 10, №2, Feb. 1977, pp. 35—41。
9. E. P. Stritter and H. L. Tredennick, "Microprogrammed Implementation of a Single Chip Microprocessor", Proc. 11th Annual Microprogramming Workshop, Nov. 1978, pp. 8—16。

[白英彩、刘寿和译自《Computer》，1979，12，№2，43—52。林云梯校]

16位单片微处理器—Z8000

嵇　正　利

题要：本文介绍了美国 Zilog 公司新近研制的16位单片微处理器。试图既作微处理器用，亦作小型机应用。Z8000在寄存器组成和指令形式等体系结构方面和小型机相近。芯片有40引线的和48引线的两种。前者的逻辑地址空间是64K字节，后者是 8 兆字节。8 兆字节的逻辑地址空间是由 7 位的段号和16位偏移量组成的23位地址线输出来实现的。另外还备有能提供再定位和存储保护功能的片子。

一种微处理器，如果①能直接适合现有的应用 8 位和16位微处理器的产品和②具有保证产品长寿命的可扩展的先进体系结构，它就能马上在各种产品中被采用。Zilog公司产的 Z8000不仅能满足第一个条件，而且可获得比现有微处理器大十倍的总处理能力。此外，为了满足第二个条件，Z8000 摆脱了传统的面向字节的微处理器设计，而采用较规则的小型机体系结构。

着眼于小型机应用领域的设计

Z8000除小型机体系结构的许多特点外，还兼有一些附加的特点。也就是说，在设计Z8000时，既考虑到了微型计算机应用的需要，又考虑到小型计算机应用的需要。

首先，作为体系结构上的特征可先举出的，是使用由一位至可变长字（字串）的 7 种数据类型，而且还有可供选择的 8 种寻址方式。81种操作码同各种数据类型和寻址方式结合，形成414条指令。可以说，这种指令系统和大部分小型计算机相比，功能较强且较丰富。况且，这种指令的规则性非常高。因此，90%以上的指令可把主要的五种寻址方式和字节、16位字、双倍长字等各种数据类型作任意组合使用。

在Z8000体系结构方面，其特点是芯片上有足够多的寄存器（16位寄存器共有24个）。因此，程序设计时的访存次数大为减少。这些寄存器中有16个是通用寄存器，除其中一个外，其余15个，都可作变址寄存器使用，不受任何限制。

以小型计算机的应用领域为目标这一点，还表现在Z8000具有存储器直接寻址能力为 8 兆字节的大容量上，只是 Z8000机上没有把这种存储器空间作为线性空间来使用。Z8000的存储器是由128个段构成，每段最大达 65536 个字节。地址空间的分段方法与程序员使用存储器的方法相似，即把各过程区和数据区（局部或全部的）分别置入各自固有的段内。为了更有效地利用整个存储器空间，研制了称为“存储管理器”的大规模集成电路。存储管理器和Z8000配合，完成大系统的动态存储再定位和存储器保护。

由于还必须满足现有微处理器的需要，生产了两种 Z8000芯片。一种是48引线的分段方案，它具有供 8 兆字节寻址用的23根地址线，另一种是40引线的，有16根地址线，可对相当于一个段的64K字节寻址。可以由40引线扩展到48引线。亦即分段的 48 引线 Z8000，通过利用送程序状态指令，可在128个段的任意段中，执行非分段产品的代码。

最后，Z8000和一般通用计算机系统一样，为了把操作系统与应用程序分开，备有两种工作模式（系统模式和正常模式）。而且，每种模式都具有各自的堆栈。因此，借特权指令执行的与全系统有关的功能是与一般程序分开的。

约5800个门(17，500个晶体管)的大规模集成电路

Z8000是用硅栅n沟MOS技术的芯片。负载是耗尽型，设计采用了按比例缩小的原则。在6.05×6.50毫米的硅片上集成约17,500个管子。集成度为每平方毫米148个门，超过了过去的微处理器（参见附录2 Z8000系谱图）。每片使用5伏电源，用的是4兆赫（250毫微秒）的单相时钟。中央处理器一个存储器周期至少包括3个时钟周期。因此，Z8000需要周期时间为750毫微秒和存取时间为430毫微秒的存储器。

各种控制信号端子

如图1所示，Z8000除地址/数据总线、时钟、电源端子外，还有6种控制总线，即总线定时控制、状态控制、CPU状态控制、中断控制、总线控制以及多微处理器控制总线。

总线定时控制输出线有三根，它们对片中地址/数据总线上的数据流实行调整。地址选通信号表明地址为有效。数据选通信号对中央处理器输入输出数据的读写定时。存储器请求线便于动态存储器连接，它送出刷新定时信号。

状态控制总线传输与中央处理器状态有关的信息。读/写线预先给出下个周期状态。正常/系统线指明中央处理器在现周期是处于两工作方式中的哪一个。字/字节线指明中央处理器是存取16位数据或是存取8位数据。四根状态控制线（ST_0—ST_3）组成一个四位字，指明若干种总线状态，如存储器请求、堆栈、取第一指令字、取第二指令字、中断响应和内部操作等。

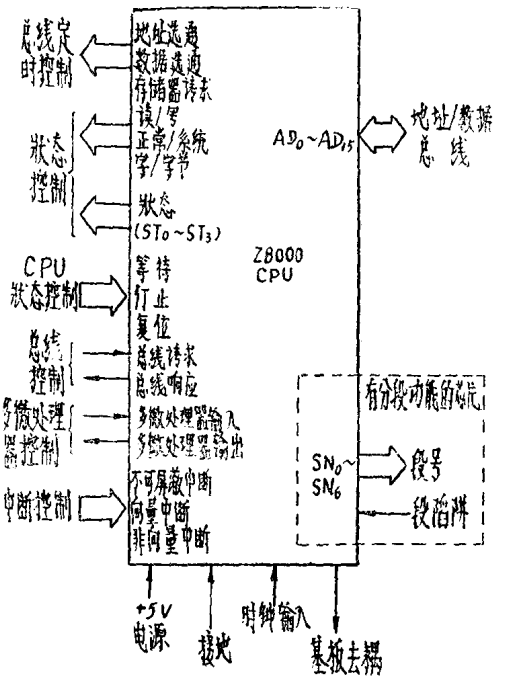


图1　两种芯片。用标准的40引线微处理器封装的Z8000是不分段的，具有16根地址线，可对64K字节存储器直接寻址。与小型计算机相似的48引线芯片附加有7根段地址线（图内方框所示），从而具有8兆字节的寻址能力。

中央处理器状态控制总线由三根输入线组成。复位信号（输入）使中央处理器处于初始状态。等待信号给CPU指明，数据传送还未准备好。停止信号是停止中央处理器内部操作（但动态存储器的刷新功能并未停止）。中央处理器检测到停止信号后，实际停止工作是在取出各指令的第一字之后。

采用一对线即可完成Z8000所有总线的控制。总线请求信号（输入）指明其他装置要求使用总线。也即总线请求输入信号为低电位，则中央处理器使地址/数据总线、总线定时控制总线和状态控制总线都是高阻状态。而且通过发出总线响应输出信号，表明中央处理器已放弃总线控制。这时，要求使用总线的装置收到该总线响应信号，就可使用上述的所有总线。

另一对线加上一系列指令，用于实现多