值得注意的是，由于微程序控制的所有控制信号都来自微指令，而微指令又存于控制存储器中，因此欲完成上述这些微操作，必须先将微指令从控制存储器中读出，也即必须先给出这些微指令的地址。由图 10.4 可见，在取指微程序中，除第一条微指令外，其余微指令的地址均由上一条微指令的下地址字段直接给出，因此上述每一条微指令都需要增加一个将微指令下地址字段送至 CMAR 的微操作，记作 Ad(CMDR) →CMAR，而这一操作只能由下一个时钟周期 T 的上升沿将地址打入 CMAR 内。至于取指微程序的最后一条微指令，其后续微指令的地址是由微地址形成部件形成的，而且也只能由下一个 T 的上升沿将该地址打入 CMAR 中，即微地址形成部件→CMAR。为了反映该地址与操作码有关，故记作 OP(IR)→微地址形成部件→CMAR。

综上所述，考虑到需要形成后续微指令的地址，上述分析的取指操作共需 6 条微指令完成，即

T_0　PC→MAR，1→R

T_1　Ad(CMDR)→CMAR

T_2　M(MAR)→MDR，(PC)+1→PC

T_3　Ad(CMDR)→CMAR，

T_4　MDR→IR，OP(IR)→微地址形成部件(编码器)

T_5　OP(IR)→微地址形成部件→CMAR

所有微指令均由 T 的上升沿打入 CMDR 中。

(2) 执行阶段的微操作及节拍安排

执行阶段的微操作由操作码性质而定，同时也需要考虑后续微指令地址的形成问题。

1) CLA 指令

与组合逻辑控制一样，该指令在执行阶段只有一个微操作 0→AC，只需一个时钟周期 T，故对应一条微指令。该微指令的下地址字段应直接给出取指微程序的入口地址，而且由下一个 T 的上升沿将地址打入 CMAR 内。这样，对应 CLA 指令执行阶段的微指令有两条：

T_0　0→AC

T_1　Ad(CMDR)→CMAR　取指微程序入口地址→CMAR

同理可得其余 4 条非访存指令对应的微操作。

2) COM 指令

T_0　$\overline{AC}$→AC

T_1　Ad(CMDR)→CMAR　取指微程序入口地址→CMAR

3) SHR 指令

T_0　L(AC)→R(AC)，AC_0→AC_0

T_1　Ad(CMDR)→CMAR　取指微程序入口地址→CMAR

4) CSL 指令

T_0 R(AC)→L(AC),AC_0→AC_n (即ρ^{-1}(AC))

T_1 Ad(CMDR)→CMAR 取指微程序入口地址→CMAR

5) STP 指令

T_0 0→G

T_1 Ad(CMDR)→CMAR 取指微程序入口地址→CMAR

这里由于安排了 Ad(CMDR)→CMAR,使再次启动机器时,可直接用已存入 CMAR 中的取指微程序的入口地址。

6) ADD 指令

T_0 Ad(IR)→MAR,1→R

T_1 Ad(CMDR)→CMAR

T_2 M(MAR)→MDR

T_3 Ad(CMDR)→CMAR

T_4 (AC)+(MDR)→AC

T_5 Ad(CMDR)→CMAR 取指微程序入口地址→CMAR

7) STA 指令

T_0 Ad(IR)→MAR,1→W

T_1 Ad(CMDR)→CMAR

T_2 AC→MDR

T_3 Ad(CMDR)→CMAR

T_4 MDR→M(MAR)

T_5 Ad(CMDR)→CMAR 取指微程序入口地址→CMAR

8) LDA 指令

T_0 Ad(IR)→MAR,1→R

T_1 Ad(CMDR)→CMAR

T_2 M(MAR)→MDR

T_3 Ad(CMDR)→CMAR

T_4 MDR→AC

T_5 Ad(CMDR)→CMAR 取指微程序入口地址→CMAR

9) JMP 指令

T_0 Ad(IR)→PC

T_1 Ad(CMDR)→CMAR 取指微程序入口地址→CMAR

10) BAN 指令

T_0 $A_0 \cdot Ad(IR)+\overline{A}_0 \cdot (PC)$→PC

T_1　Ad(CMDR)→CMAR　取指微程序入口地址→CMAR

上述全部微操作共 20 个,微指令共 38 条。在上述指令中,1)~5)为非访存指令;6)~8)为访存指令;9)和 10)则为转移类指令。

2. 确定微指令格式

微指令的格式包括微指令的编码方式、后续微指令的地址形成方式和微指令字长等 3 个方面。

(1) 微指令的编码方式

上述微操作数不多,可采用直接编码方式,由微指令控制字段的某一位直接控制一个微操作。

(2) 后续微指令地址的形成方式

根据上述分析,可采用由指令的操作码和微指令的下地址字段两种方式形成后续微指令的地址。

(3) 微指令字长

微指令由操作控制字段和下地址字段两部分组成。根据直接编码方式,20 个微操作对应 20 位操作控制字段;根据 38 条微指令,对应 6 位下地址字段。这样,微指令字长至少取 26 位。

仔细分析发现,在 38 条微指令中有 19 条微指令是为了控制将后续微指令的地址打入 CMAR 的操作(其中 18 条是微指令下地址字段 Ad(CMDR)→CMAR,另一条是指令操作码 OP(IR)→微地址形成部件→CMAR),因此实际上是每两个时钟周期才能取出并执行一条微指令。如果能做到每一个时钟周期取出并执行一条微指令,将大大提高微程序控制的速度。

事实上如果将 CMDR 的下地址字段 Ad(CMDR)直接接到控制存储器的地址线上,并由下一个时钟周期的上升沿将该地址单元的内容(微指令)读到 CMDR 中,便能做到在一个时钟周期内读出并执行一条微指令。这就好比将 Ad(CMDR)当作 CMAR 使用。同理,也可将指令寄存器的操作码字段 OP(IR)经微地址形成部件形成的后续微指令的地址,直接送到控制存储器的地址线上。这两路地址可通过一个多路选择器,根据需要任选一路,如图 10.16 所示。

综上所述,在省去了 19 条微指令的同时也省去了两个微操作(微指令下地址字段 Ad(CMDR)→CMAR 和指令操作码 OP(IR) →微地址形成部件→CMAR)。这样,10 条机器指令共对应 20−2=18 个微操作和 38−19=19 条微指令。为了便于扩充,操作控制字段取 24 位,下地址字段取 6 位,其微指令格式如图 10.17 所示。

其中,第 0 位表示控制　PC→MAR 微操作

第 1 位表示控制　1→R 微操作

第 2 位表示控制　M(MAR)→MDR

第 3 位表示控制　(PC)+1→PC

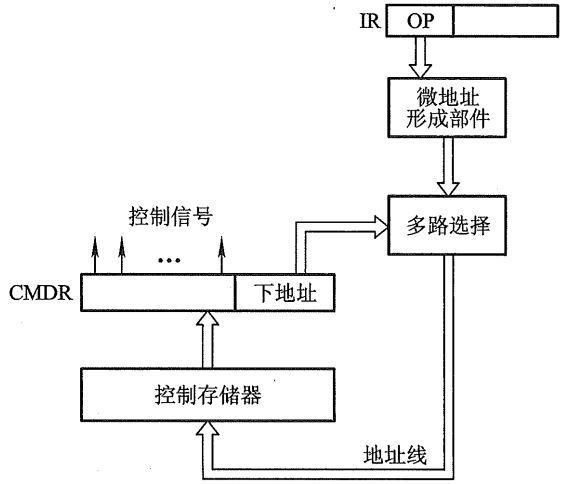


图 10.16 省去了 CMAR 的控制存储器

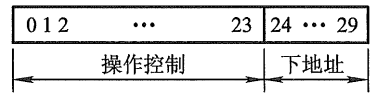


图 10.17 对应 10 条机器指令的微指令格式

第 4 位表示控制 MDR→IR

第 5 位表示控制 0→AC

第 6 位表示控制 $\overline{AC}\rightarrow AC$

第 7 位表示控制 $L(AC)\rightarrow R(AC), AC_0\rightarrow AC_0$

第 8 位表示控制 $R(AC)\rightarrow L(AC), AC_0\rightarrow AC_n$

第 9 位表示控制 0→G

第 10 位表示控制 Ad(IR)→MAR

第 11 位表示控制 (MDR)+(AC)→AC

第 12 位表示控制 1→W

第 13 位表示控制 AC→MDR

第 14 位表示控制 MDR→M(MAR)

第 15 位表示控制 MDR→AC

第 16 位表示控制 Ad(IR)→PC

第 17 位表示控制 $A_0 \cdot Ad(IR)+\overline{A}_0 \cdot (PC)\rightarrow PC$

3. 编写微指令码点

表 10.3 列出了对应 10 条机器指令的微指令码点。表中空格中“0”省略。

表 10.3　对应 10 条机器指令的微指令码点

微程序名称	微指令地址（八进制）	微指令（二进制代码）																											
		操作控制字段																								顺序控制字段					
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
取指	0 0	1	1																												1
	0 1			1	1																									1	
	0 2					1																				×	×	×	×	×	×
CLA	0 3						1																								
COM	0 4							1																							
SHR	0 5								1																						
CSL	0 6									1																					
STP	0 7										1																				
ADD	1 0		1									1																1			1
	1 1			1																								1		1	
	1 2												1																		
STA	1 3											1		1														1	1		
	1 4														1													1	1		1
	1 5															1															
LDA	1 6		1									1																1	1	1	1
	1 7			1																							1				
	2 0																1														
JMP	2 1																	1													
BAN	2 2																		1												

在确定微指令格式及其字长的过程中，还可将一些微操作命令合用一位代码来控制，这样可大大压缩微指令的操作控制字段，缩短微指令字长。

例 10.7　某机有 5 条微指令，每条微指令发出的控制信号如表 10.4 所示。采用直接控制方式设计微指令的控制字段，要求其位数最少，而且保持微指令本身的并行性。

表 10.4 例 10.7 表格

微指令	激活的控制信号									
	a	b	c	d	e	f	g	h	i	j
I_1	√		√		√		√		√	
I_2	√	√		√		√		√		√
I_3	√			√	√	√				
I_4	√									
I_5	√			√						√

解:由表 10.4 可见,控制信号 c、g、i 仅在微指令 I_1 同时出现,可合并用 1 位控制字段表示。控制信号 b、h 仅在微指令 I_2 中同时出现,也可合并用 1 位控制字段表示。这样 10 个控制信号 a~j可压缩到 7 个,其格式如图 10.18 所示。

a	bh	cgi	d	e	f	j
1	2	3	4	5	6	7

图 10.18 例 10.7 压缩后的微指令控制字段

思考题与习题

10.1 假设响应中断时,要求将程序断点存在堆栈内,并且采用软件方法寻找中断服务程序的入口地址,试写出中断隐指令的微操作及节拍安排。

10.2 写出完成下列指令的微操作及节拍安排(包括取指操作)。

(1) 指令"ADD R_1, X"完成将 R_1 寄存器的内容和主存 X 单元的内容相加结果存于 R_1 的操作。

(2) 指令"ISZ X"完成将主存 X 单元的内容增 1,并根据其结果若为 0,则跳过下一条指令执行。

10.3 按序写出下列程序所需的全部微操作命令及节拍安排。

指令地址	指 令
300	LDA 306
301	ADD 307
302	BAN 304
303	STA 305
304	STP

10.4　在单总线结构的计算机中，用该总线连接了指令寄存器 IR、程序计数器 PC、存储器地址寄存器 MAR、存储器数据寄存器 MDR、通用寄存器 $R_0 \sim R_7$ 的输入和输出端。ALU 的两个输入端分别与总线和寄存器 Y 的输出端相连，ALU 的输出端与寄存器 Z 的输入端相连。Y 的输入端与总线连接，Z 的输出端与总线连接。该机有下列指令：

ADD　R_1, R_2, R_3　　；$(R_2)+(R_3) \rightarrow R_1$

JMP　$*K$　　；$(PC)+(K-1) \rightarrow PC$

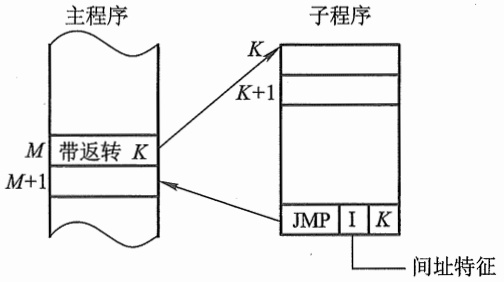
LOAD　R_1, mem　　；$(mem) \rightarrow R_1$

STORE　mem, R_2　　；$R_2 \rightarrow mem$

写出控制器执行上述指令的微操作及节拍安排。

10.5　假设 CPU 在中断周期用堆栈保存程序断点，而且进栈时指针减 1（具体操作是先修改栈指针后存数），出栈时指针加 1。分别写出组合逻辑控制和微程序控制在完成中断返回指令时，取指阶段和执行阶段所需的全部微操作命令及节拍安排。

10.6　已知带返转指令的含义如下图所示，写出机器在完成带返转指令时，取指阶段和执行阶段所需的全部微操作及节拍安排。



10.7　画出组合逻辑控制单元的组成框图，根据指令处理过程，结合有关部件说明其工作原理。

10.8　画出微程序控制单元的组成框图，根据指令处理过程，结合有关部件说明其工作原理。

10.9　试比较组合逻辑设计和微程序设计的设计步骤和硬件组成，说明哪一种控制速度更快，为什么？

10.10　微指令的操作控制有几种编码方式？各有何特点？哪一种控制速度最快？

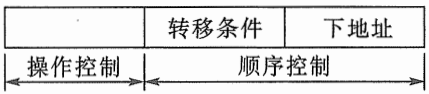
10.11　什么是垂直型微指令？什么是水平型微指令？各有何特点？

10.12　能否说水平型微指令就是直接编码的微指令，为什么？

10.13　微指令的地址有几种形成方式？各有何特点？

10.14　微指令操作控制字段采用直接编码或显式编码时，其微指令字长如何确定？

10.15　设控制存储器的容量为 512×48 位，微程序可在整个控存空间实现转移，而控制微程序转移的条件共有 4 个（采用直接控制），微指令格式如下：



试问微指令中的 3 个字段分别为多少位？

10.16　试比较静态微程序设计和动态微程序设计。

10.17 解释机器指令、微指令、微程序、毫微指令和毫微程序以及它们之间的对应关系。

10.18 毫微程序设计的特点是什么？与微程序设计相比，其硬件组成有何不同？

10.19 假设机器的主要部件有程序计数器 PC，指令寄存器 IR，通用寄存器 R_0、R_1、R_2、R_3，暂存器 C、D，ALU，移位器，存储器地址寄存器 MAR，存储器数据寄存器 MDR 及存储矩阵 M。

(1) 要求采用单总线结构画出包含上述部件的硬件框图，并注明数据流动方向。

(2) 画出“ADD (R_1)，(R_2)”指令在取指阶段和执行阶段的信息流程图。R_1 寄存器存放源操作数地址，R_2 寄存器存放目的操作数的地址。

(3) 写出对应该流程图所需的全部微操作命令。

10.20 假设机器的主要部件同上题，外加一个控制门 G。

(1) 要求采用双总线结构(每组总线的数据流动方向是单向的)画出包含上述部件的硬件框图，并注明数据流动方向。

(2) 画出“SUB R_1, R_3”指令完成 (R_1)−(R_3)→R_1 操作的指令周期信息流程图(假设指令地址已放在 PC 中)，并列出相应的微操作控制信号序列。

10.21 下表给出 8 条微指令 I_1～I_8 及所包含的微命令控制信号，设计微指令操作控制字段格式，要求所使用的控制位最少，而且保持微指令本身内在的并行性。

微指令	所含的微命令
I_1	a b c d e
I_2	a d f g
I_3	b h
I_4	c
I_5	c e g i
I_6	a h j
I_7	c d h
I_8	a b h

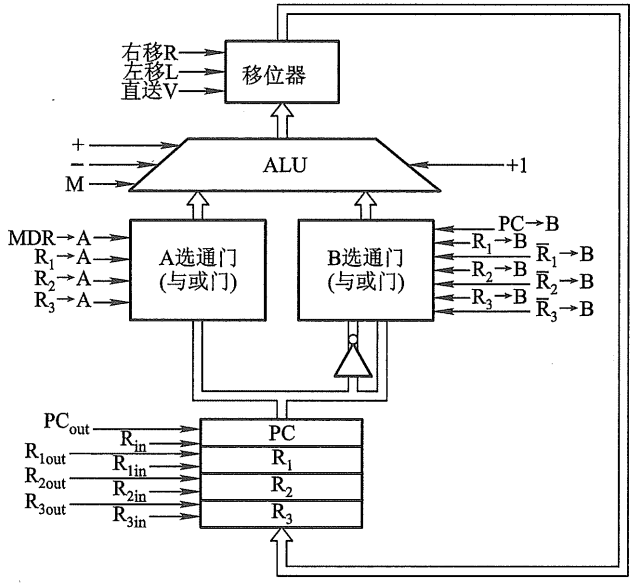
10.22 设有一运算器通路如下图所示，假设操作数 a 和 b(均为补码)分别放在通用寄存器 R_2 和 R_3 中，ALU 有+、−、M(传送)三种操作功能，移位器可实现左移、右移和直送功能。

(1) 指出相容性微操作和相斥性微操作。

(2) 采用字段直接编码方式设计适合于此运算器的微指令格式。

(3) 画出计算 $2(a+b)\rightarrow R_3$ 的微程序流程图，试问执行周期需用几条微指令？

(4) 按设计的微指令格式，写出满足(3) 要求的微代码。



附录 10A　PC 整机介绍

PC 整机主要由主板、处理器、主存、芯片组，以及通过各个端口和总线插槽接入的外部设备等组成。

10A.1　主板

主板或系统板是装在 PC 主机箱中的一块印制电路板，其上安装了组成微型计算机的主要电路系统，并带有扩展插槽和多种接插件，用于插装各种接口卡和有关部件(如键盘、鼠标等)。主板性能的好坏直接影响整机的性能。

10A.1.1　主板的主要组成部件

① CPU 插座(Socket)或插槽(Slot)。

② 内存条插槽。

③ 连接硬盘、软盘驱动器和光盘驱动器等外部设备接口的插座。

④ 接插各种用途的接口卡所需的扩展插槽。

⑤ 连接鼠标、键盘、打印机和调制解调器的串并端口。

⑥ CPU 芯片、内存条、系统芯片组、BIOS(基本输入输出系统)芯片和 CMOS 芯片等。

⑦ 电源、电池、电阻和电容等。

⑧ 跳线和开关。

10A.1.2 CPU 芯片及插座(插槽)

主板的性能主要取决于 CPU,CPU 芯片内部总线的宽度和时钟频率(主频)是决定 CPU 性能的主要参数。内部总线越宽,在每个时钟节拍内可以传送的数据越多,有利于进行更大的数据量运算。主频越高,执行每条指令的时间越短,运算速度越快,整机性能也越好。

CPU 芯片可以采用插座或插槽固定在主板上。

10A.1.3 内存条插槽

内存条插槽的插孔数必须与内存条引脚数一致。72 线的单面内存条使用 72 线内存条插槽,其数据宽度为 32 位;168 线和 184 线的双面内存条使用 168 线和 184 线的内存条插槽,其数据宽度为 64 位。

10A.1.4 扩展插槽

主板上的扩展插槽(又称总线插槽)是主机通过系统总线与外部设备联系的通道,外设接口电路的接口卡可插在扩展槽内。常用的插槽有 ISA 扩展槽、EISA 扩展槽、VESA(VL-Bus)扩展槽和 PCI 扩展槽。

10A.1.5 配套芯片和器件

为使 CPU 正常工作,主板上还必须有配套的芯片和器件,主要有如下几种。

1. BIOS 芯片

BIOS(Basic Input Output System)芯片装有基本输入输出系统程序,完成冷启动、热启动、上电自检、基本输入输出驱动程序、系统硬件配置分析、引导 DOS 启动或引导 ROM BASIC 解释程序以及 BIOS 中断管理。该程序固化在 ROM 芯片上,早期采用 EPROM 作为 BIOS 芯片,现在大多数 PC 采用 Flash Memory 作为 BIOS 芯片。

2. 芯片组

主板上除了 CPU、内存条、BIOS 芯片外,还有众多支持芯片和接口芯片,这些芯片按其功能不同,采用 VLSI 技术将它们分别集成在几块芯片中,如总线的缓冲和控制芯片,CPU 的复位、Cache 控制、存储器控制、协处理器接口芯片,以及 CMOS 与 8042 振荡源芯片等。

3. CMOS RAM 芯片

CMOS RAM 芯片用于提供系统的日期、时间、保存系统的硬件配置参数和软硬盘规格、显示器接口的类型、键盘和其他硬件的设置。PC 系统上电启动到引导完成,必须从 CMOS RAM 中读取若干参数数据,设定为开机的初始状态。如果 CMOS RAM 芯片损坏或内容丢失,会造成软硬盘无法使用、鼠标工作失效等故障。

10A.1.6 主板结构的改进

1984 年,IBM 在推出 IBM PC/AT 时,以产品定义了内部结构的标准,称为 AT 结构标准,以后又发展成为 Baby/Mini-AT 结构标准。随着计算机技术的进一步发展,特别是“电子计算机、电器和电信”三电一体化和多媒体技术的发展,使集成到 PC 中的功能日益增多,对主板的尺寸、CPU 和内存条的位置以及软硬盘控制器及软硬盘机支架位置都有更高要求,先后又推出了 ATX 和 BTX 主板结构规范。后者的散热性能更好,安装与固定方式更科学,并采用了大量的新型总

线及接口,增加了 USB 接口的数量,使外设的接插更为简化。

在高性能的 PC 系统中,主板上还采用了一些提高系统性能的新技术,例如:

① 采用集成的数据捕获芯片,构成硬件监控系统,增加系统的监测能力,以提高系统的稳定性。

② 支持 Ultra DMA/33、DMA/66 芯片,使硬盘的数据传输率比原来提高了 1~2 倍。在多任务环境中执行数据传输,或在单任务环境中进行大批量数据传输时,采用 DMA 方式,允许数据直接从硬盘传输到主存,而不占用 CPU 资源,使 CPU 从频繁的数据传输中脱离出来去执行其他任务,从而提高整机系统的性能,这一点对多任务环境特别有用。

③ 除了采用 USB 接口外,Apple 公司和 TI(Texas Instrument)公司又开发了一种高速数据传输的串行接口 IEEE 1394,有了 USB 和 IEEE 1394,使 PC 的使用简化到同家电一样容易。

④ 为了解决 PCI 总线无法执行实时 3D 图形处理,Intel 公司又开发了 AGP(Accelerated Graphics Port)加速图形端口,可获得较高的 3D 图形显示性能,增加了可用的带宽,其最高数据传输率可达 266 MBps 和 533 MBps,而采用 AGP 2.0 版本,最大数据传输率可达 1 GBps。

10A.2　芯片组

10A.2.1　芯片组的功能

CPU 芯片是 PC 的核心器件,但 CPU 要完成信息处理功能还必须有一系列的“接口电路”和“支持电路”。例如,CPU 要向外设输入输出信息,必须有并行接口电路和串行接口电路;CPU 要与主存进行数据传送,必须有主存控制电路;CPU 要具有中断处理功能,必须有中断控制电路;CPU 要支持 DMA 功能,必须有 DMA 控制电路;CPU 要通过系统总线与其他部件联系,必须有总线控制电路等;此外要向 CPU 及系统中其他部件提供时钟信号,必须有“时钟发生器(电路)”。所有这些电路在早期的 PC 中都是由一些中、小规模集成电路和成千上万个电阻、电容组成的。这不仅占用了主板中的很多位置,而且给维修带来很大困难。

在 PC 286 以上的微型计算机系统中,为了简化硬件部分的设计,减少主板上芯片的数量,增加硬件的可靠性,大部分厂商采用芯片组(Chipset)技术来设计 PC 主板。随着超大规模集成电路的发展,采用 VLSI 技术,把主板上众多的接口芯片和支持芯片按不同功能分别集成到一块集成芯片中。这样,用少量几片 VLSI 芯片的组合,即“控制芯片组”(简称“芯片组”)来代替众多的接口芯片和支持芯片,可以简化主板的设计,降低系统的成本,提高系统的可靠性,有利于测试、维护和维修。

在 PC 中,整个系统的有效运行都由芯片组来控制和协调,芯片组决定了系统的如下特征。

① CPU 的类型(是 Pentium、Pentium Pro、Pentium MMX,还是 Pentium Ⅱ、Pentium Ⅲ 和 Pentium 4)和芯片的主频范围。

② 内存条的类型,是快速页面模式、扩展数据输出、突发式,还是同步 DRAM(SDRAM)、带 Cache 的 DRAM、双数据速率的 SDRAM(DDR SRDAM),是支持一种,还是几种等。

③ 提供 USB 接口以及 IEEE 1394 接口的数目。

④ 存储器总线的最大频率。

⑤ PCI 总线的类型，是 32 位还是 64 位，同存储器总线速度是同步还是异步，是否支持 PCI-Express。

⑥ 支持几个 CPU。

⑦ 对内置 PCI、EIDE 控制的支持。

⑧ 内置 PS/2 鼠标和键盘控制器、BIOS 以及实时时钟电路。

芯片组一旦选定，系统的上述特征就被确定，在使用过程中，芯片组是无法升级的。

10A.2.2 芯片组的组成

不同时代的 PC 有不同的芯片组，随着 ISA、EISA、PCI 总线的出现，所推出的芯片组与总线标准有关，如 82350 EISA 芯片组、PCI 芯片组等。

基于 Pentium(包括 Pentium Pro)处理器系列和 PCI 总线的芯片组，通常称为 PCI 芯片组，如 Intel 430 系列。

1. Intel 430 系列

430 系列芯片组适用于 Pentium 和 PCI 总线。图 10.19 是 Pentium 计算机主板结构示意图。图中的 PCI 芯片组包括主存与 Cache 控制器芯片，CPU 总线-PCI 总线桥(又称北桥)芯片，PCI 总线-ISA 总线桥(又称南桥)芯片。主存与 Cache 控制芯片用来管理 CPU 对主存和 Cache 的存取操作，并提供在 CPU、Cache、主存和 PCI 局部总线之间传送的总线控制。北桥芯片将 CPU 总线和 PCI 总线相连，实现 CPU、主存和 PCI 之间的通路。南桥芯片将 PCI 总线和 ISA 总线相连。通过两个“桥”芯片将 CPU 总线、PCI 总线和 ISA 总线连成整体。桥芯片在此起到了速度缓冲、电平转换和控制协议的转换作用。当 CPU 芯片需要升级时，只需改变 CPU 总线和北桥芯片，全部已有的外部设备可继续工作。

图 10.19 中的 CPU 总线是一个 64 位数据线和 32 位地址线的同步总线。总线时钟频率为 66.6 MHz(或 60 MHz)，CPU 内部时钟是此时钟频率的倍频。此总线可与 4~128 MB 的主存相连。需要扩充主存容量时，可以内存条的形式插入主板的内存条插槽内。CPU 总线还接有 L2 级 Cache。CPU 对主存和 Cache 的存取由主存与 Cache 控制器芯片完成。

PCI 总线用于连接高速的 I/O 设备，它是 32 位(或 64 位)的同步总线，地址线、数据线共用一组，分时复用。它采用集中式仲裁方式，有专用的 PCI 总线仲裁器。主板上还有 3 个 PCI 总线扩展插槽，可作为 PC 扩充外设使用。

ISA 总线可与低速 I/O 设备连接，主板上配有 3 ~ 4 个 ISA 总线扩展插槽，以便使用 16 位/8 位的接口(适配器)卡。ISA 总线控制逻辑还可以通过主板上的板级总线与实时钟/日历、ROM、键盘和鼠标控制器(8042 微处理器)等芯片相连接。

Pentium Ⅱ处理器推出后，为适应这一高频 CPU 芯片的需要，又推出了满足 AGP 加速图形端口要求的 Intel 440 芯片组。

2. Intel 440 BX 芯片组

Intel 440 BX 是专用于 Pentium Ⅱ的芯片组，它把系统总线的频率提高到 100 MHz，同时还支持 66 MHz 的系统总线，保护了以前的投资。它使 100 MHz 的 SDRAM 的使用成为可能。它提高

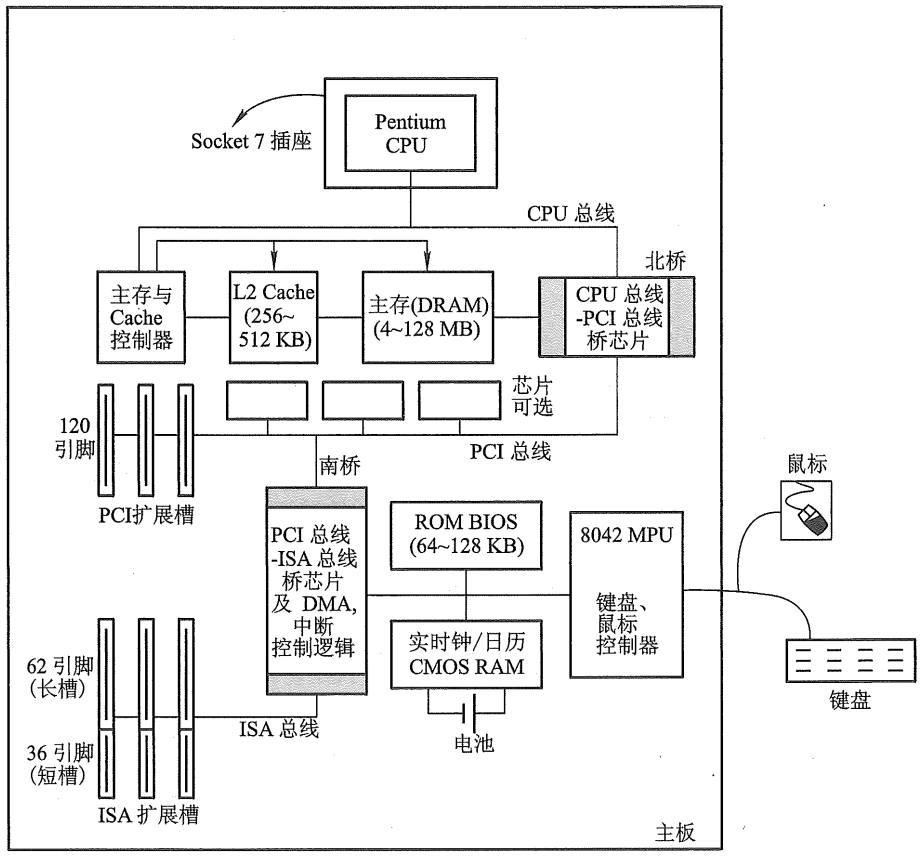


图 10.19　Pentium 主板结构示意图

了 Pentium Ⅱ CPU、AGP、100 MHz SDRAM 和 PCI 总线间的速度和带宽，使在 MPEG-2 视频解码及 DVD-ROM 等应用领域的性能得到提高。

图 10.20 是采用 440 BX 芯片组的主板结构示意图。

440 BX 芯片组由 82443 BX 主桥(Host Bridge)芯片和 82371 AB/EB 多功能 I/O 芯片组成。它基本保持了 430 芯片组由北桥芯片和南桥芯片组成的结构。北桥集成了存储器控制器和 PCI 控制器，南桥集成了 ISA 控制器和 IDE(Integrated Drive Electronics)控制器(硬盘驱动器接口)。

82443 BX 芯片采用了四端口加速技术，它把 CPU(支持单/双 Pentium 处理器)、AGP 端口、主存和 PCI 总线相互连接起来并控制四者的数据传输。它支持 1 GB 的扩展数据输出 EDO(Extended Data Output) DRAM 和 SDRAM 的主存，但不允许 EDO 和 SDRAM 混合使用，它也支持 Ultra DMA/33。

82371 EB(Pentium Ⅱ×4E)芯片是一个高度集成的多功能 I/O 芯片，它包括 PCI-ISA 桥接器，提供两个硬盘驱动器接口 IDE，支持 Ultra DMA/33 接口标准，具有 USB 控制器，支持两个 USB 端

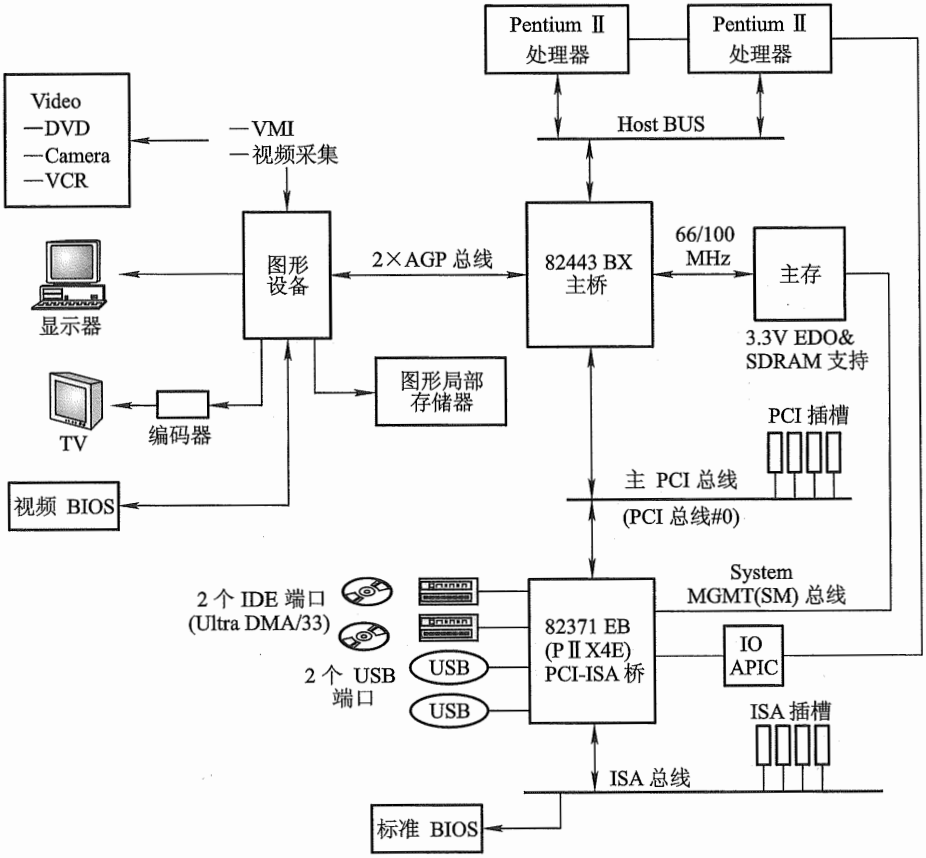


图 10.20 440 BX 主板结构示意图

口,还支持 I/O 高级可编程中断控制器(Advanced Programmable Interrupt Controller,APIC)。采用 DMA 方式的硬盘数据可通过 SM 总线直接传输到主存。

参 考 文 献

[1] STALLINGS W. Computer Organization and Architecture: Designing for Performance[M]. 7th ed. Upper Saddle River: Prentice Hall, 2005.

[2] PATTERSON D A, HENNESSY J L. Computer Organization and Design: The Hardware/Software Interface[M]. 3rd ed. Burlington: Morgan Kaufmann, 2004.

[3] BRYANT R E, O'HALLARON D R. Computer Systems: A Programmer's Perspective[M]. Upper Saddle River: Prentice Hall, 2002.

[4] HYDE R. Write Great Code: Volume 1: Understanding the Machine[M]. San Francisco: No Starch Press, 2004.

[5] 唐朔飞. 计算机组成原理[M]. 2版. 北京: 高等教育出版社, 2008.

[6] 唐朔飞. 计算机组成原理: 学习指导与习题解答[M]. 2版. 北京: 高等教育出版社, 2012.

[7] 孙德文. 微型计算机技术[M]. 修订版. 北京: 高等教育出版社, 2005.

[8] 张晨曦, 王志英, 张春元, 等. 计算机体系结构[M]. 2版. 北京: 高等教育出版社, 2005.

[9] 白中英. 计算机组成原理[M]. 3版. 北京: 科学出版社, 2002.