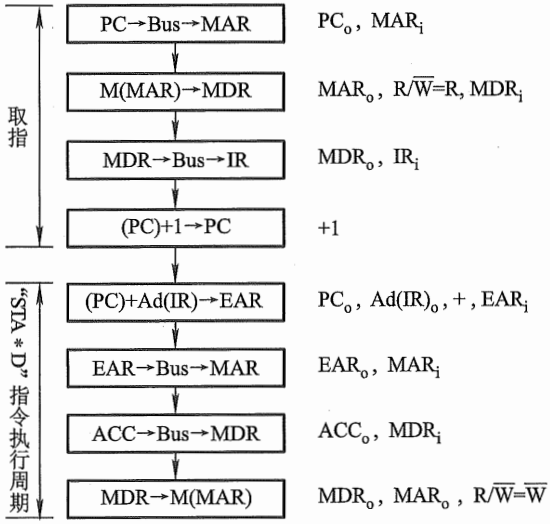


图 9.7　"STA ＊D"指令周期的信息流程及相应的控制信号

9.2.3　多级时序系统

1. 机器周期

机器周期可看作所有指令执行过程中的一个基准时间，机器周期取决于指令的功能及器件的速度。确定机器周期时，通常要分析机器指令的执行步骤及每一步骤所需的时间。例如，取数、存数指令能反映存储器的速度及其与 CPU 的配合情况；加法指令能反映 ALU 的速度；条件转移指令因为要根据上一条指令的执行结果，经测试后才能决定是否转移，所需的时间较长。总之，通过对机器指令执行步骤的分析，会找到一个基准时间，在这个基准时间内，所有指令的操作都能结束。若以这个基准时间定为机器周期，显然不是最合理的。因为只有以完成复杂指令功能所需的时间（最长时间）作为基准，才能保证所有指令在此时间内完成全部操作，这对简单指令来说，显然是一种浪费。

进一步分析发现，机器内的各种操作大致可归属为对 CPU 内部的操作和对主存的操作两大类，由于 CPU 内部的操作速度较快，CPU 访存的操作时间较长，因此通常以访问一次存储器的时间定为基准时间较为合理，这个基准时间就是机器周期。又由于不论执行什么指令，都需要访问存储器取出指令，因此在存储字长等于指令字长的前提下，取指周期也可看作机器周期。

2. 时钟周期（节拍、状态）

在一个机器周期里可完成若干个微操作，每个微操作都需要一定的时间，可用时钟信号来控制产生每一个微操作命令（如图 9.3 中的 C_i）。时钟就好比计算机的心脏，只要接通电源，计算

机内就会产生时钟信号。时钟信号可由机器主振电路(如晶体振荡器)发出的脉冲信号经整形(或倍频、分频)后产生,时钟信号的频率即为CPU主频。用时钟信号控制节拍发生器,就可产生节拍。每个节拍的宽度正好对应一个时钟周期。在每个节拍内机器可完成一个或几个需同时执行的操作,它是控制计算机操作的最小时间单位。图9.8反映了机器周期、时钟周期和节拍的关系,图中一个机器周期内有4个节拍 T_0、T_1、T_2、T_3。

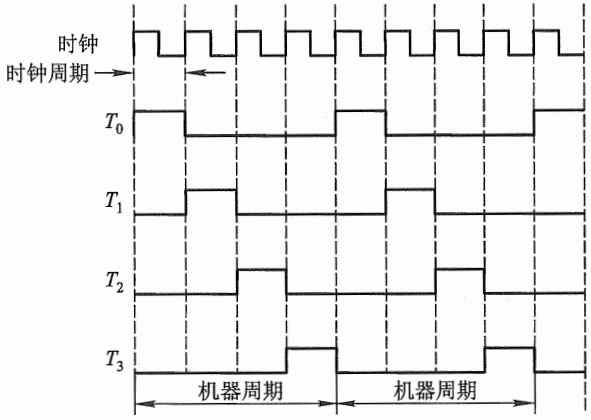
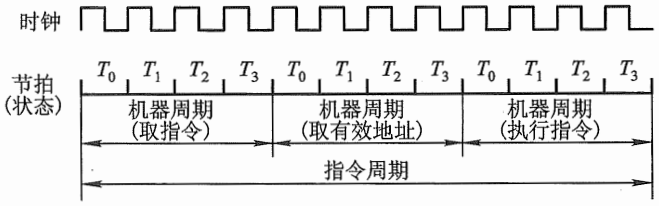


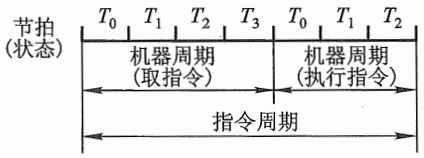
图9.8　机器周期、时钟周期和节拍的关系

3. 多级时序系统

图9.9反映了指令周期、机器周期、节拍(状态)和时钟周期的关系。可见,一个指令周期包含若干个机器周期,一个机器周期又包含若干个时钟周期(节拍),每个指令周期内的机器周期数可以不等,每个机器周期内的节拍数也可以不等。其中,图9.9(a)为定长的机器周期,每个机



(a)　定长的机器周期



(b)　不定长的机器周期

图9.9　指令周期、机器周期、节拍和时钟周期的关系

器周期包含 4 个节拍(4 个 T);图 9.9(b)为不定长的机器周期,每个机器周期包含的节拍数可以为 4 个,也可以为 3 个,后者适合于操作比较简单的指令,它可跳过某些时钟周期(如 T_3),从而缩短指令周期。

机器周期、节拍(状态)组成了多级时序系统。

一般来说,CPU 的主频越快,机器的运行速度也越快。在机器周期所含时钟周期数相同的前提下,两机平均指令执行速度之比等于两机主频之比。例如,CPU 的主频为 8 MHz,其平均指令执行速度为 0.8 MIPS。若想得到平均指令执行速度为 0.4 MIPS 的机器,则只需要用主频为(8 MHz×0.4 MIPS)/0.8 MIPS=4 MHz 的 CPU 即可。

实际上机器的速度不仅与主频有关,还与机器周期中所含的时钟周期数以及指令周期中所含的机器周期数有关。同样主频的机器,由于机器周期所含时钟周期数不同,运行速度也不同。机器周期所含时钟周期数少的机器,速度更快。

例 9.3　设某计算机的 CPU 主频为 8 MHz,每个机器周期平均含 2 个时钟周期,每条指令的指令周期平均有 2.5 个机器周期,试问该机的平均指令执行速度为多少 MIPS? 若 CPU 主频不变,但每个机器周期平均含 4 个时钟周期,每条指令的指令周期平均有 5 个机器周期,则该机的平均指令执行速度又是多少 MIPS? 由此可得出什么结论?

解:由于主频为 8 MHz,所以时钟周期为 1/8=0.125 μs,机器周期为 0.125×2=0.25 μs,指令周期为 0.25×2.5=0.625 μs。

① 平均指令执行速度为 1/0.625=1.6 MIPS。

② 若 CPU 主频不变,机器周期含 4 个时钟周期,每条指令平均含 5 个机器周期,则指令周期为 0.125×4×5=2.5 μs,故平均指令执行速度为 1/2.5=0.4 MIPS。

③ 可见机器的运行速度并不完全取决于主频。

此外,机器的运行速度还和其他很多因素有关,如主存的运行速度、机器是否配有 Cache、总线的数据传输率、硬盘的运行速度以及机器是否采用流水技术等。机器速度还可以用 MIPS(执行百万条指令数每秒)和 CPI(执行一条指令所需的时钟周期数)来衡量。

9.2.4　控制方式

控制单元控制一条指令执行的过程实质上是依次执行一个确定的微操作序列的过程。由于不同指令所对应的微操作数及其复杂程度不同,因此每条指令和每个微操作所需的执行时间也不同。通常将如何形成控制不同微操作序列所采用的时序控制方式称为 CU 的控制方式。常见的控制方式有同步控制、异步控制、联合控制和人工控制四种。

1. 同步控制方式

同步控制方式是指,任何一条指令或指令中任何一个微操作的执行都是事先确定的,并且都是受统一基准时标的时序信号所控制的方式。

图 9.9(a)就是一种典型的同步控制方式,每个机器周期都包含 4 个节拍。如果机器内的存

储器存取周期不统一，那么只有把最长的存取周期作为机器周期，才能采用同步控制，否则取指令和取数时间不同，无法用统一的基准。又如有些不访存的指令，执行周期的微操作较少，无须4个节拍。因此，为了提高CPU的效率，在同步控制中又有三种方案。

(1) 采用定长的机器周期

这种方案的特点是：不论指令所对应的微操作序列有多长，也不管微操作的简繁，一律以最长的微操作序列和最繁的微操作作为标准，采取完全统一的、具有相同时间间隔和相同数目的节拍作为机器周期来运行各种不同的指令，如图9.9(a)所示。显然，这种方案对于微操作序列较短的指令来说，会造成时间上的浪费。

(2) 采用不定长的机器周期

采用这种方案时，每个机器周期内的节拍数可以不等，如图9.9(b)所示。这种控制方式可解决微操作执行时间不统一的问题。通常把大多数微操作安排在一个较短的机器周期内完成，而对某些复杂的微操作，采用延长机器周期或增加节拍的办法来解决，如图9.10所示。

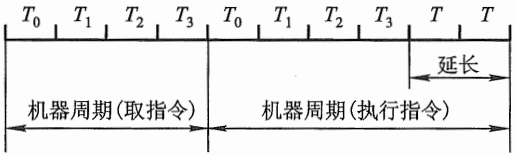


图9.10　延长机器周期示意

(3) 采用中央控制和局部控制相结合的方法

这种方案将机器的大部分指令安排在统一的、较短的机器周期内完成，称为中央控制，而将少数操作复杂的指令中的某些操作(如乘除法和浮点运算等)采用局部控制方式来完成，图9.11所示为中央控制和局部控制的时序关系。

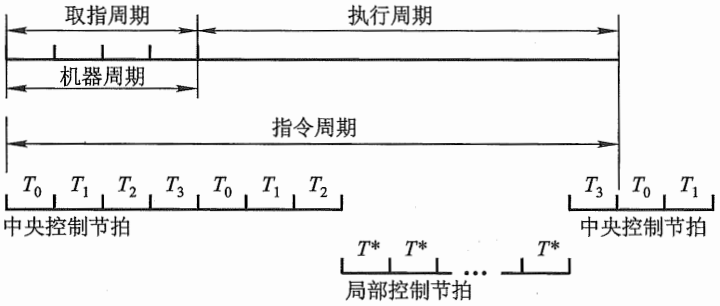


图9.11　中央控制和局部控制的时序关系

在设计局部控制线路时需要注意两点：其一，使局部控制的每一个节拍 T^* 的宽度与中央控制的节拍宽度相同；其二，将局部控制节拍作为中央控制中机器节拍的延续，插入中央控制的执行周期内，使机器以同样的节奏工作，保证局部控制和中央控制的同步。T^* 的多少可根据情况

而定，对于乘法，当操作数位数固定后，T^* 的个数也就确定了。而对于浮点运算的对阶操作，由于移位次数不是一个固定值，因此 T^* 的个数不能事先确定。

以乘法指令为例，第一个机器周期采用中央控制的节拍控制取指令操作，接着仍用中央控制的 T_0、T_1、T_2 节拍去完成将操作数从存储器中取出并送至寄存器的操作，然后转局部控制，用局部控制节拍 T^* 完成重复加和移位的操作。

2. 异步控制方式

异步控制方式不存在基准时标信号，没有固定的周期节拍和严格的时钟同步，执行每条指令和每个操作需要多少时间就占用多少时间。这种方式微操作的时序由专门的应答线路控制，即当 CU 发出执行某一微操作的控制信号后，等待执行部件完成该操作后发回“回答”（或“结束”）信号，再开始新的微操作，使 CPU 没有空闲状态，但因需要采用各种应答电路，故其结构比同步控制方式复杂。

3. 联合控制方式

同步控制和异步控制相结合就是联合控制方式。这种方式对各种不同指令的微操作实行大部分统一、小部分区别对待的办法。例如，对每条指令都有的取指令操作，采用同步方式控制；对那些时间难以确定的微操作，如 I/O 操作，则采用异步控制，以执行部件送回的“回答”信号作为本次微操作的结束。

4. 人工控制方式

人工控制是为了调机和软件开发的需要，在机器面板或内部设置一些开关或按键，来达到人工控制的目的。

（1）Reset（复位）键

按下 Reset 键，使计算机处于初始状态。当机器出现死锁状态或无法继续运行时，可按此键。若在机器运行时按此键，将会破坏机器内某些状态而引起错误，因此要慎用。有些微型计算机未设此键，当机器死锁时，可采用停电后再加电的办法重新启动计算机。

（2）连续或单条执行转换开关

由于调机的需要，有时需要观察执行完一条指令后的机器状态，有时又需要观察连续运行程序后的结果，设置连续或单条执行转换开关，能为用户提供这两种选择。

（3）符合停机开关

有些计算机还配有符合停机开关，这组开关指示存储器的位置，当程序运行到与开关指示的地址相符时，机器便停止运行，称为符合停机。

9.2.5　多级时序系统实例分析

为了加深对本章内容的理解，下面以 Intel 8085 为例，通过对一条 I/O 写操作指令运行过程的分析，使读者进一步认识多级时序系统与控制单元发出的控制信号的关系。

1. Intel 8085 的组成

图 9.12 是 Intel 8085 的组成框图，其内部有 3 个 16 位寄存器，即 SP、PC 和增减地址锁存器 IDAL，11 个 8 位寄存器，即 B、C、D、E、H、L、IR、AC、暂存器 TR 以及地址缓冲寄存器 ABR 和地址数据缓冲寄存器 ADBR，以及一个 5 位的状态标志寄存器 FR。ALU 能实现 8 位算术运算和逻辑运算。控制单元的具体组成将在第 10 章讲述，图中的定时和控制(CU)能对外发出各种控制信号。8085 内还有中断控制和 I/O 控制，内部数据总线为 8 位。图中未标出 8085 片内的控制信号。

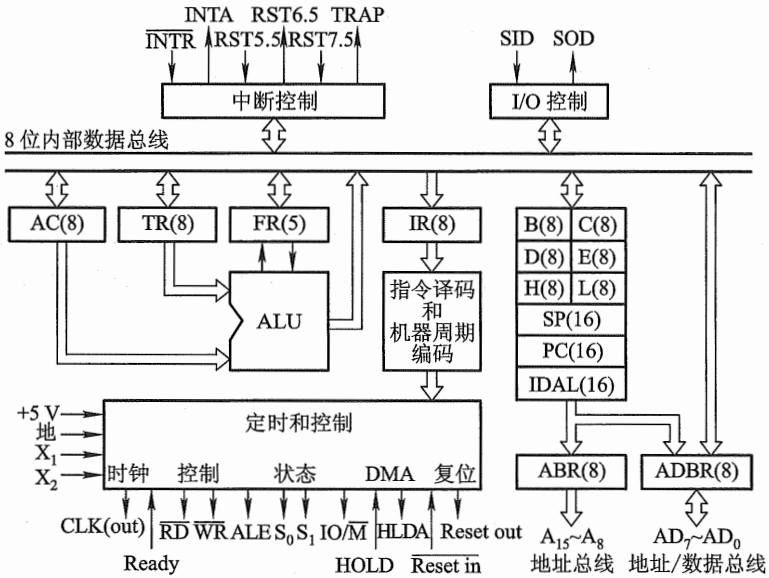


图 9.12 Intel 8085 的组成框图

2. Intel 8085 的外部信号

8085 芯片引脚图如图 9.13 所示，共 40 根引脚。外部信号分以下几类。

(1) 地址和数据信号

① A_{15}~A_8(出)：16 位地址的高 8 位。

② AD_7~AD_0(入/出)：16 位地址的低 8 位或 8 位数据，它们共用相同的引脚。

③ SID(入)：串行输入。

④ SOD(出)：串行输出。

(2) 定时和控制信号

① CLK(出)：系统时钟，每周期代表一个 *T* 状态。

② X_1、X_2(入)：来自外部晶体或其他设备，以驱动内部的时钟发生器。

③ ALE(出)：地址暂存使能信号，在机器周期的第一个时钟周期产生，使外围芯片保

存地址。

④ S_0、S_1(出):用于标识读/写操作是否发生。

⑤ IO/$\overline{M}$(出):使 I/O 接口或存储器读/写操作使能。

⑥ $\overline{RD}$(出):表示被选中的存储器或 I/O 接口将所读出的数据送至数据总线上。

⑦ $\overline{WR}$(出):表示数据总线上的数据将写入被选中的存储器或 I/O 接口中。

(3) 存储器和 I/O 的初始化信号

① HOLD(入):请求 CPU 放弃系统总线的控制和使用,总线将用于 DMA 操作。

② HLDA(出):总线响应信号,表示总线可被外部占用。

③ Ready(入):用于 CPU 与较慢的存储器或设备同步。当某一设备准备就绪后,向 CPU 发 Ready 信号,此时 CPU 可进行输入或输出。

(4) 与中断有关的信号

① TRAP(出):重新启动中断(RST7.5、RST6.5、RST5.5)。

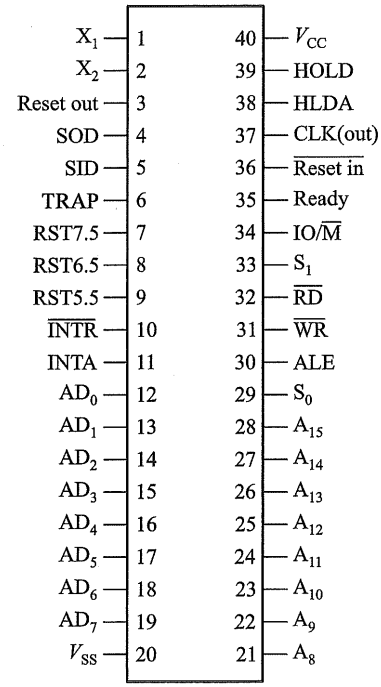


图 9.13　Intel 8085 外部引脚图

② $\overline{INTR}$(入):中断请求信号。

③ INTA(出):中断响应信号。

(5) CPU 初始化

① $\overline{Reset\ in}$(入):PC 清"0",假设 CPU 从 0 地址开始执行。

② Reset out(出):对 CPU 的置"0"做出响应,该信号能用于重置系统的剩余部分。

(6) 电源和地

① V_{CC}:+5 V 电源。

② V_{SS}:地。

3. 机器周期和节拍(状态)与控制信号的关系

8085 的一条指令可分成 1~5 个机器周期,每个机器周期内又包含 3~5 个节拍,每个节拍持续一个时钟周期。在每个节拍内,CPU 根据控制信号执行一个或一组同步的微操作。下面分析一条输出指令,其功能是将 AC 的内容写入所选择的设备中,执行该指令的时序图如图 9.14 所示。

由图可见,该指令的指令周期包含 3 个机器周期 M_1、M_2 和 M_3,每个机器周期内所包含的节拍数不同(M_1 含 4 拍,M_2 和 M_3 均含 3 拍)。该指令字长为 16 位,由于数据线只有 8 位,所以要

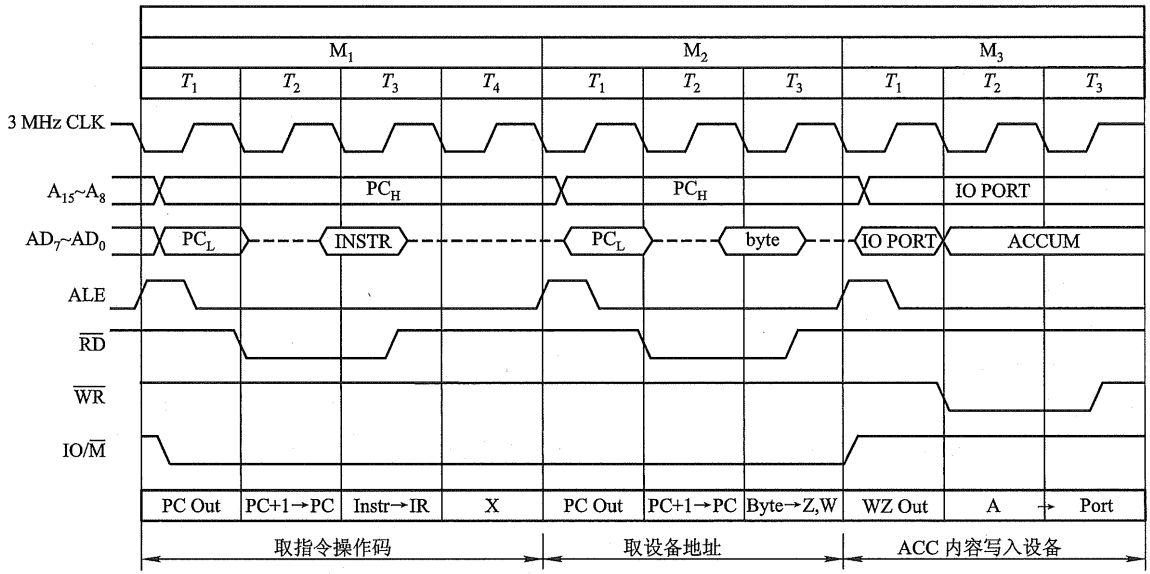


图 9.14 8085 输出指令时序图

分两次将指令取至 CPU 内。第一个机器周期取指令的操作码,第二个机器周期取被选设备的地址,第三个机器周期把 AC 的内容通过数据总线写入被选中的设备中。具体时序如下。

(1) 第一个机器周期 M_1:存储器读,取指令操作码

① T_1 状态,$IO/\overline{M}$ 低电平,表示存储器操作。CPU 将 PC 的高 8 位送至地址总线 $A_{15}\sim A_8$,PC 的低 8 位送至地址/数据总线 $AD_7\sim AD_0$,并由 ALE 的下降沿激活存储器保存地址。

② T_2 状态,$\overline{RD}$(低)有效,表示存储器读操作,存储器将指定地址的内容送至数据总线 $AD_7\sim AD_0$, CPU 等待数据线上的数据稳定。

③ T_3 状态,当数据线上的数据稳定后,CPU 接收数据,此数据为该指令的第一字节操作码。

④ T_4 状态,CPU 进入译码阶段,在 T_4 最后时刻 ALE(高)失效。

在 T_2 或 T_3 状态可安排(PC)+1→PC 操作,图中未标出此控制信号。

(2) 第二个机器周期 M_2:存储器读,取被选设备的地址

① T_1 状态,同 M_1 的 T_1 状态操作。

② T_2 状态,同 M_1 的 T_2 状态操作。

③ T_3 状态,当数据线上的数据稳定后,CPU 接收数据,此数据为被选设备的地址。

同样可以在 T_2 或 T_3 时刻完成(PC)+1→PC 操作。这个机器周期内设有指令译码,因此 T_4 省略。在 T_3 最后时刻 ALE(高)失效。

(3) 第三个机器周期 M_3:I/O 写

① T_1 状态,$IO/\overline{M}$ 高电平,表示 I/O 操作,CPU 将 I/O 口地址送至 $A_{15}\sim A_8$ 和 $AD_7\sim AD_0$,并

由 ALE 下降沿激活 I/O 保存地址。

② T_2 状态，$\overline{WR}$（低）有效，表示 I/O 写操作，AC 的内容通过 $AD_7 \sim AD_0$ 数据总线送至被选中的设备中。

可见，控制单元的每一个控制信号都是在指定机器周期内的指定 T 时刻发出的，反映了多级时序系统与控制信号间的关系。

思考题与习题

9.1 设 CPU 内有这些部件：PC、IR、MAR、MDR、AC、CU。

（1）写出取指周期的全部微操作。

（2）写出减法指令“SUB X”、取数指令“LDA X”、存数指令“STA X”（X 均为主存地址）在执行阶段所需的全部微操作。

（3）当上述指令为间接寻址时，写出执行这些指令所需的全部微操作。

（4）写出无条件转移指令“JMP Y”和结果溢出则转指令“BAO Y”在执行阶段所需的全部微操作。

9.2 控制单元的功能是什么？其输入受什么控制？

9.3 什么是指令周期、机器周期和时钟周期？三者有何关系？

9.4 能不能说 CPU 的主频越快，计算机的运行速度就越快？为什么？

9.5 设机器 A 的 CPU 主频为 8 MHz，机器周期含 4 个时钟周期，且该机的平均指令执行速度是 0.4 MIPS，试求该机的平均指令周期和机器周期，每个指令周期中含几个机器周期。如果机器 B 的 CPU 主频为 12 MHz，且机器周期也含 4 个时钟周期，试问 B 机的平均指令执行速度为多少 MIPS？

9.6 设某计算机的 CPU 主频为 8 MHz，每个机器周期平均含 2 个时钟周期，每条指令平均有 4 个机器周期，试问该计算机的平均指令执行速度为多少 MIPS。若 CPU 主频不变，但每个机器周期平均含 4 个时钟周期，每条指令平均有 4 个机器周期，则该机的平均指令执行速度又是多少 MIPS？由此可得出什么结论？

9.7 某 CPU 的主频为 10 MHz，若已知每个机器周期平均包含 4 个时钟周期，该机的平均指令执行速度为 1 MIPS，试求该机的平均指令周期及每个指令周期含几个机器周期。若改用时钟周期为 0.4 μs 的 CPU 芯片，则计算机的平均指令执行速度为多少 MIPS？若要得到平均每秒 80 万次的指令执行速度，则应采用主频为多少的 CPU 芯片？

9.8 某计算机的主频为 6 MHz，各类指令的平均执行时间和使用频度如下表所示，试计算该机的速度（单位用 MIPS 表示）。若上述 CPU 芯片升级为 10 MHz，则该机的运行速度又为多少？

指令类别	存取	加、减、比较、转移	乘除	其他
平均指令执行时间	0.6 μs	0.8 μs	10 μs	1.4 μs
使用频度	35%	45%	5%	15%

9.9 试比较同步控制、异步控制和联合控制的区别。

9.10 什么是典型的同步控制？为了提高 CPU 的效率，在同步控制方式中又有哪些方式？以 8085 的输出指令为例，说明它属于哪种控制方式？

9.11 设 CPU 内部结构如图 9.4 所示，此外还设有 B、C、D、E、H、L 6 个寄存器，它们各自的输入和输出端都与内部总线相通，并分别受控制信号控制（如 B_i 为寄存器 B 的输入控制；B_o 为寄存器 B 的输出控制）。要求从取指令开始，写出完成下列指令所需的全部微操作和控制信号。

（1）ADD B,C　　((B) + (C)→B)

（2）SUB A,H　　((AC) - (H) →AC)

9.12 CPU 结构同上题，写出完成下列指令所需的全部微操作和控制信号（包括取指令）。

（1）寄存器间接寻址的无条件转移指令“JMP @ B”。

（2）间接寻址的存数指令“STA @ X”。

9.13 设 CPU 内部结构如图 9.4 所示，此外还设有 $R_1 \sim R_4$ 4 个寄存器，它们各自的输入和输出端都与内部总线相通，并分别受控制信号控制（如 R_{2i} 为寄存器 R_2 的输入控制；R_{2o} 为寄存器 R_2 的输出控制）。要求从取指令开始，写出完成下列指令所需的全部微操作和控制信号。

（1）ADD R_2, @R_4　　；　$((R_2)+((R_4))\rightarrow R_2$，寄存器间接寻址）

（2）SUB R_1, @mem　　；　$((R_1)-((\text{mem}))\rightarrow R_1$，存储器间接寻址）

9.14 设单总线计算机结构如图 9.5 所示，其中 M 为主存，XR 为变址寄存器，EAR 为有效地址寄存器，LATCH 为锁存器。假设指令地址已存于 PC 中，画出“LDA * D”和“SUB X,D”指令周期信息流程图，并列出相应的控制信号序列。

说明：

（1）“LDA * D”指令字中 * 表示相对寻址，D 为相对位移量。

（2）“SUB X,D”指令字中 X 为变址寄存器 XR，D 为形式地址。

（3）寄存器的输入和输出均受控制信号控制，例如，PC_i 表示 PC 的输入控制信号，MDR_o 表示 MDR 的输出控制信号。

（4）凡是需要经过总线实现寄存器之间的传送，需在流程图中注明，如 PC→Bus→MAR，相应的控制信号为 PC_o 和 MAR_i。

第 10 章　控制单元的设计

本章以 10 条机器指令为例，介绍控制单元的两种设计方法，旨在使读者初步掌握设计控制单元的思路，为今后设计计算机打下初步基础。

10.1　组合逻辑设计

10.1.1　组合逻辑控制单元框图

图 9.2 示出了控制单元的外特性，其中指令的操作码是决定控制单元发出不同控制信号的关键。为了简化控制单元的逻辑，将存放在 IR 的 n 位操作码经过一个译码电路产生 2^n 个输出，这样，每对应一种操作码便有一个输出送至 CU。当然，若指令的操作码长度可变，指令译码线路将更复杂。

控制单元的时钟输入实际上是一个脉冲序列，其频率即为机器的主频，它使 CU 能按一定的节拍（T）发出各种控制信号。节拍的宽度应满足数据信息通过数据总线从源到目的所需的时间。以时钟为计数脉冲，通过一个计数器，又称节拍发生器，便可产生一个与时钟周期等宽的节拍序列。如果将指令译码和节拍发生器从 CU 中分离出来，便可得简化的控制单元框图，如图 10.1 所示。

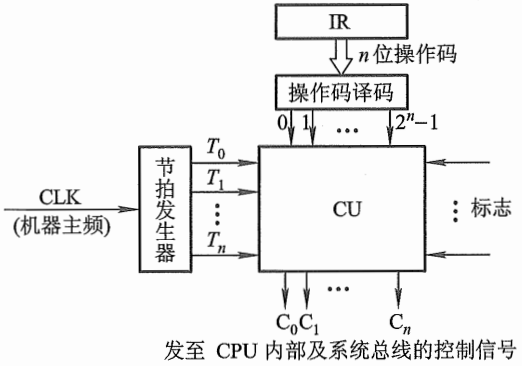


图 10.1　带译码和节拍输入的控制单元框图

10.1.2 微操作的节拍安排

假设机器采用同步控制,每个机器周期包含3个节拍,而且CPU内部结构如图9.3所示,其中MAR和MDR分别直接和地址总线和数据总线相连,并假设IR的地址码部分与MAR之间有通路。

安排微操作节拍时应注意以下3点。

① 有些微操作的次序是不容改变的,故安排微操作节拍时必须注意微操作的先后顺序。

② 凡是被控制对象不同的微操作,若能在一个节拍内执行,应尽可能安排在同一个节拍内,以节省时间。

③ 如果有些微操作所占的时间不长,应该将它们安排在一个节拍内完成,并且允许这些微操作有先后次序。

按上述3条原则,以9.1节所分析的10条指令为例,其微操作的节拍安排如下:

1. 取指周期微操作的节拍安排

- 根据原则②,T_0 节拍可安排两个微操作:PC→MAR,1→R。
- 根据原则②,T_1 节拍可安排 M(MAR)→MDR 和(PC)+1→PC 两个微操作。
- T_2 节拍可安排 MDR→IR,考虑到指令译码时间较短,根据原则③,可将指令译码OP(IR)→ID也安排在 T_2 节拍内。

实际上(PC)+1→PC操作也可安排在 T_2 节拍内,因一旦PC→MAR后,PC的内容就可修改。

2. 间址周期微操作的节拍安排

T_0 Ad(IR)→MAR,1→R

T_1 M(MAR)→MDR

T_2 MDR→Ad(IR)

3. 执行周期微操作的节拍安排

(1) 非访存指令

1) 清除累加器指令 CLA

该指令在执行周期只有一个微操作,按同步控制的原则,此操作可安排在 $T_0 \sim T_2$ 的任一节拍内,其余节拍空,例如:

T_0

T_1

T_2 0→AC

2) 累加器取反指令 COM

同理,累加器取反操作可安排在 $T_0 \sim T_2$ 的任一节拍中,即

T_0

T_1

T_2　$\overline{AC}\rightarrow AC$

3）算术右移一位指令 SHR

T_0

T_1

T_2　$L(AC)\rightarrow R(AC), AC_0\rightarrow AC_0$

4）循环左移一位指令 CSL

T_0

T_1

T_2　$R(AC)\rightarrow L(AC), AC_0\rightarrow AC_n$（即 $\rho^{-1}(AC)$）

5）停机指令 STP

T_0

T_1

T_2　$0\rightarrow G$

（2）访存指令

1）加法指令 ADD X

T_0　$Ad(IR)\rightarrow MAR, 1\rightarrow R$

T_1　$M(MAR)\rightarrow MDR$

T_2　$(AC)+(MDR)\rightarrow AC$（该操作实际包括$(AC)\rightarrow ALU, (MDR)\rightarrow ALU, ALU\rightarrow AC$）

2）存数指令 STA X

T_0　$Ad(IR)\rightarrow MAR, 1\rightarrow W$

T_1　$AC\rightarrow MDR$

T_2　$MDR\rightarrow M(MAR)$

3）取数指令 LDA X

T_0　$Ad(IR)\rightarrow MAR, 1\rightarrow R$

T_1　$M(MAR)\rightarrow MDR$

T_2　$MDR\rightarrow AC$

（3）转移类指令

1）无条件转移指令 JMP X

T_0

T_1

T_2　$Ad(IR)\rightarrow PC$

2）有条件转移(负则转)指令 BAN X

T_0

T_1

T_2　$A_0 \cdot Ad(IR) + \overline{A_0} \cdot (PC) \rightarrow PC$

4. 中断周期微操作的节拍安排

在执行周期的最后时刻，CPU 要向所有中断源发中断查询信号，若检测到某个中断源有请求，并且未被屏蔽又被排队选中，则在允许中断的条件下，CPU 进入中断周期，此时 CPU 由中断隐指令完成下列操作(假设程序断点存入主存 0 号地址单元内)：

T_0　0→MAR，1→W

T_1　PC→MDR

T_2　MDR→M(MAR)，向量地址→PC

此外，由图 8.30 可知，CPU 进入中断周期，由硬件置“0”允许中断触发器 EINT，即关中断。

例 10.1　设 CPU 中各部件及其相互连接关系如图 10.2 所示。图中 W 是写控制标志，R 是读控制标志，R_1 和 R_2 是暂存器。

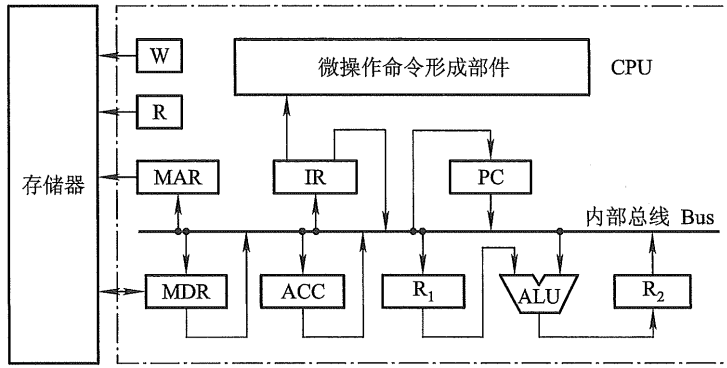


图 10.2　例 10.1 CPU 内部结构框图

(1) 假设要求在取指周期由 ALU 完成 (PC)+1→PC 的操作(即 ALU 可以对它的一个源操作数完成加 1 的运算)。要求以最少的节拍写出取指周期全部微操作命令及节拍安排。

(2) 写出指令“ADD # α”(#为立即寻址特征，隐含的操作数在 ACC 中)在执行阶段所需的微操作命令及节拍安排。

解：(1) 由于 (PC)+1→PC 需由 ALU 完成，因此 PC 的值可作为 ALU 的一个源操作数，靠控制 ALU 做+1 运算得到 (PC)+1，结果送至与 ALU 输出端相连的 R_2，然后再送至 PC。

此题的关键是要考虑总线冲突的问题，故取指周期的微操作命令及节拍安排如下：

T_0　PC→Bus→MAR，1→R　　；PC 通过总线送 MAR

T_1　M(MAR)→MDR，

(PC)→Bus→ALU_{+1}→R_2　　;PC 通过总线送 ALU 完成 (PC)+1→R_2

T_2　MDR→Bus→IR,　　;MDR 通过总线送 IR

OP(IR)→微操作命令形成部件

T_3　R_2→Bus→PC　　;R_2 通过总线送 PC

(2) 立即寻址的加法指令执行周期的微操作命令及节拍安排如下:

T_0　Ad(IR) →Bus→R_1　　;立即数→R_1

T_1　(ACC) + (R_1) →ALU→R_2　　;ACC 通过总线送 ALU

T_2　R_2→Bus→ACC　　;结果通过总线送 ACC

例 10.2　设 CPU 内部结构如图 10.2 所示,且 PC 有自动加 1 功能。此外还有 B、C、D、E、H、L 6 个寄存器(图中未画),它们各自的输入端和输出端都与内部总线 Bus 相连,并分别受控制信号控制。要求写出完成下列指令组合逻辑控制单元所发出的微操作命令及节拍安排。

(1) ADD B,C　　;(B)+(C) →B

(2) SUB E,@ H　　;(E)-((H)) →E　寄存器间接寻址

(3) STA @ mem　　;ACC→((mem))　存储器间接寻址

解:(1) 完成"ADD B,C"指令所需的微操作命令及节拍安排如下:

取指周期

T_0　PC→Bus→MAR,1→R

T_1　M(MAR) →MDR,(PC) + 1→PC

T_2　MDR→Bus→IR,OP(IR) →微操作命令形成部件

执行周期

T_0　C→Bus→R_1

T_1　(B) + (R_1) →ALU→R_2　　;B 通过总线送 ALU

T_2　R_2→Bus→B

(2) 完成"SUB E,@ H"指令所需的微操作命令及节拍安排如下:

取指周期

T_0　PC→Bus→MAR,1→R

T_1　M(MAR) →MDR,(PC) + 1→PC

T_2　MDR→Bus→IR,OP(IR) →微操作命令形成部件

间址周期

T_0　H→Bus→MAR,1→R

T_1　M(MAR) →MDR

执行周期

T_0　MDR→Bus→R_1

T_1　(E)-(R_1) →ALU→R_2　　;E 通过总线送 ALU