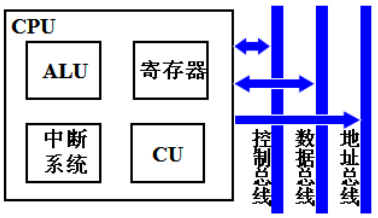
第 8 章　中央处理器(CPU)

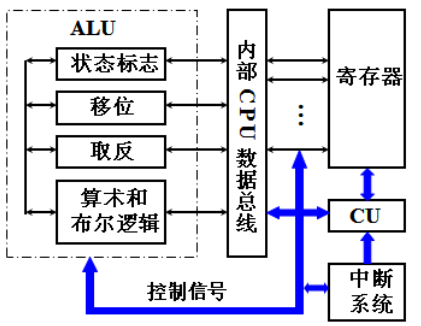
(一) CPU 的功能和基本结构

CPU 主要是由运算器和控制器组成，由于运算器(**实现算术运算和逻辑运算**)部分在第 5 章、第 6 章介绍过，所以本章主要介绍控制器的组成和工作原理。

1．CPU 的结构



CPU 与系统总线



CPU的内部结构

指令控制　PC　IR
操作控制
时间控制　CU　时序电路
数据加工　ALU　寄存器
处理中断　中断系统

CPU 的寄存器：

用户可见寄存器

(1) 通用寄存器　存放操作数
可作某种寻址方式所需的专用寄存器

(2) 数据寄存器　存放操作数（满足各种数据类型）
两个寄存器拼接存放双倍字长数据

(3) 地址寄存器　存放地址，其位数应满足最大的地址范围
用于特殊的寻址方式　段基值　栈指针

(4) 条件码寄存器　存放条件码，可作程序分支的依据
如正、负、零、溢出、进位等

控制和状态寄存器

(1) 控制寄存器
PC → MAR → M → MDR → IR
控制 CPU 操作
其中 MAR、MDR、IR　用户不可见
PC　用户可见

(2) 状态寄存器
状态寄存器　存放条件码
PSW 寄存器　存放程序状态字

2．控制器的功能

计算机对信息进行处理(或计算)是通过程序的执行而实现的，程序是完成某个确定算法的指令序列，要预先存放在存储器中。控制器的作用是控制程序的执行，它必须具有以下基本功能：

(1)．取指令

(2)．分析指令

(3)．执行指令

计算机不断重复顺序执行上述三种基本操作：取指、分析、执行；再取指、再分析、再执行，

如此循环，直到遇到停机指令或外来的干预为止。

(4). 控制程序和数据的输入与结果输出

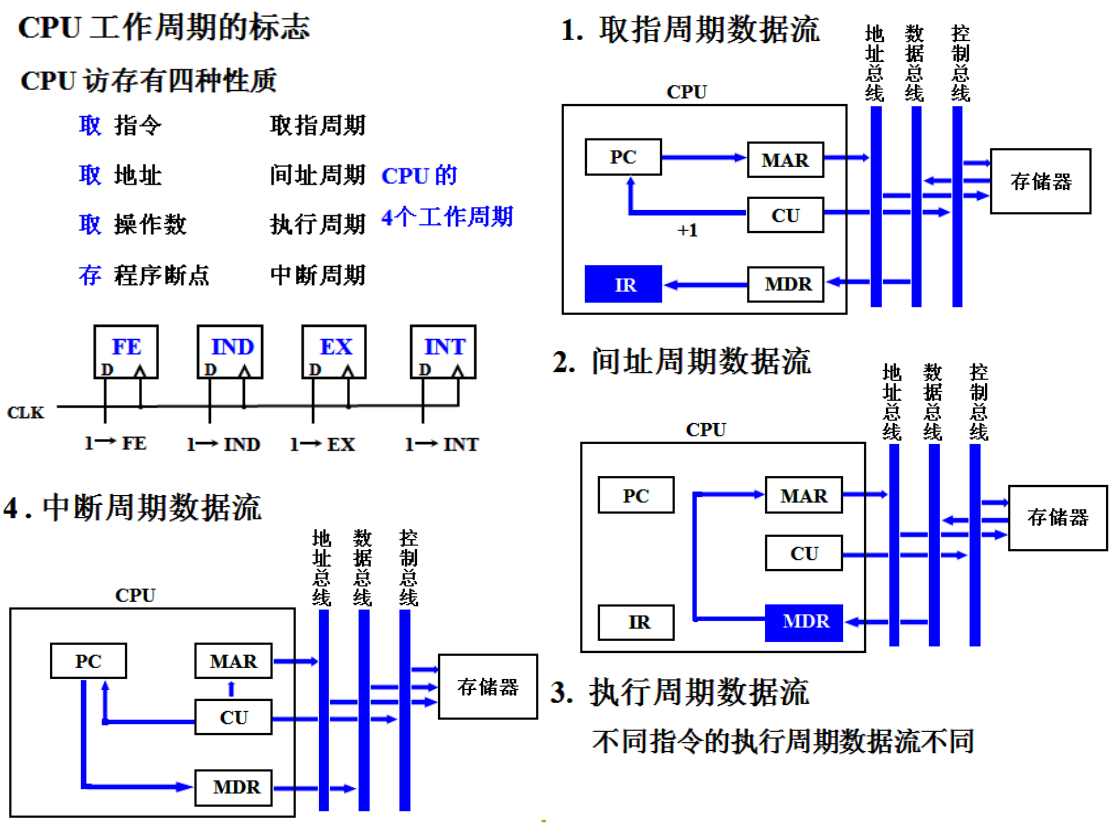
根据程序的安排或人的干预，在适当的时候向输入输出设备发出一些相应的命令来完成 I/O 功能，这实际上也是通过执行程序来完成的。

(5). 对异常情况和某些请求的处理

当机器出现某些异常情况，诸如算术运算的溢出和数据传送的奇偶错等;或者某些外来请求，诸如磁盘上的成批数据需送存储器或程序员从键盘送入命令等，此时由这些部件或设备发出：

① “中断请求”信号。

② DMA 请求信号。



3. 控制器的组成

根据对控制器功能分析，得出控制器的基本组成如下：

(1)程序计数器(PC)

即指令地址寄存器。在某些计算机中用来存放当前正在执行的指令地址；而在另一些计算机中则用来存放即将要执行的下一条指令地址；而在有指令预取功能的计算机中，一般还需要增加一个程序计数器用来存放下一条要取出的指令地址。

有两种途径来形成指令地址，其一是顺序执行的情况，通过程序计数器加“1”形成下一条指令地址(如存储器按字节编址，而指令长度为 4 个字节，则加“4”)。其二是遇到需要改变顺序执行程序的情况，一般由转移类指令形成转移地址送往程序计数器，作为下一条指令的地址。

（2)指令寄存器(IR)

用以存放当前正在执行的指令，以便在指令执行过程中，控制完成一条指令的全部功能。

（3)指令译码器或操作码译码器

对指令寄存器中的操作码进行分析解释，产生相应的控制信号。

在执行指令过程中，需要形成有一定时序关系的操作控制信号序列，为此还需要下述组成部分。

（4)脉冲源及启停线路

脉冲源产生一定频率的脉冲信号作为整个机器的时钟脉冲，是机器周期和工作脉冲的基准信号，在机器刚加电时，还应产生一个总清信号(reset)。启停线路保证可靠地送出或封锁时钟脉冲，控制时序信号的发生或停止，从而启动机器工作或使之停机。

（5)时序控制信号形成部件

当机器启动后，在 CLK 时钟作用下，根据当前正在执行的指令的需要，产生相应的时序控制信号，并根据被控功能部件的反馈信号调整时序控制信号。例如，当执行加法指令时，若产生运算溢出的异常情况，一般不再执行将结果送入目的寄存器(或存储单元)的操作，而发出中断请求信号，转入中断处理;又如执行条件转移指令时，根据不同的条件产生不同的控制信号，从而进入适当的程序分支。

(二) 指令执行过程

1. 指令执行的时序

计算机工作的过程是取指令，分析指令，执行指令三个基本动作的重复。考虑到所有的器件中(寄存器，存储器)存储器的速度最慢，因此，取最慢的器件工作时间(周期)作为整个工作的最长同步标准。

计算机的工作时序是按照存储器的工作周期划分的。每个存储器工作周期又称为机器周期。因此，每个机器周期至少完成一个基本操作。一般最长的操作是访问存储器(读/写)，这个时间也用于访问外设接口(寄存器)。如果，某个操作，比如利用运算器执行一次运算，如果不访问存储器，即使占用的时间很短，但是，也必须为其划分一个机器周期。因此，机器周期是计算时序划分的最大单位。

现在我们为计算机的执行时间进行最基本的划分：由于计算机不断地重复执行每个指令，所以，我们将执行的时间划分为一条一条指令执行所占用的时间，如下：

执行指令 1	执行指令 2	执行指令 3	执行指令 4	执行指令 5

我们将每指令占用的时间称为指令周期。由于每条指令的功能不一样，因此执行的时间也不同，指令周期长短不一样。

而每条指令的执行，又可以是取指令，分析指令，执行指令。由于取指令必须访问存储器，所以占用一个机器周期。分析指令是由指令译码电路完成的，所占用的时间极短，无需分配一个完整的机器周期。一般是在取指周期后期(结束之前的很短时间内)就可以完成。指令的执行较为复杂：可能不访问存储器；可能访问一次存储器；可能访问两次存储器等。因此，可能是一个机器周期到几个机器周期。

因此，每条指令的执行过程如下：

取指周期	执行周期 1	执行周期 2	执行周期 3	执行周期 4

第一个机器周期总是取指周期，而指令的地址总是从 PC 中获得，当发出读取存储器命令后，指令总是从数据总线 DB 送回，CPU 接受到指令之后，将指令放在指令寄存器 IR 之中。指令在 IR 中一直保留到取下一条指令为止。

第二个机器周期开始，根据指令有所不同：

执行一次 ALU 运算：分配一个机器周期。

执行访问一次存储器：分配一个机器周期。

所以，根据指令执行的不同情况，将会得到不同指令执行所占用的机器周期。

根据每个机器周期完成的任务不同，我们将每个机器周期按照任务命名。如同用取指周期命名第一个机器周期一样。

2．指令执行过程举例

假设指令格式如下：

操作码	rs，rd	rs1	imm(Disp)

rs，rd，rsl 为通用寄存器地址;imm(或 disp)为立即数(或位移量)。

加法指令功能：将寄存器(rs)中的一个数与存储器中的一个数(其地址为(rsl)＋disp)相加，结果放在寄存器 rd 中，rs 与 rd 为同一寄存器。

加法指令完成以下操作：

①取指周期

从存储器取指令，送入指令寄存器，并进行操作码译码(分析指令)。

程序计数器加 1，为下一条指令作好准备。

控制器发出的控制信号：PC→AB，W/R=0，M/IO=1;DB→IR;PC＋1。

②计算地址周期

计算数据地址，将计算得到的有效地址送地址寄存器 AR。

控制器发出的控制信号：rsl→GR，(rsl)→ALU，disp→ALU(将 rsl 的内容与 disp 送 ALU);“+”(加法命令送 ALU);ALU→AR(有效地址送地址寄存器)。

③取数周期

到存储器取数。

控制器发出的控制信号：AR→AB，W/R=0，M/IO=1;DB→DR(将地址寄存器内容送地址总线，同时发访存读命令，存储器读出数据送数据总线后，打入数据寄存器)。

④执行周期

进行加法运算，结果送寄存器，并根据运算结果置状态位 N，Z，V，C。

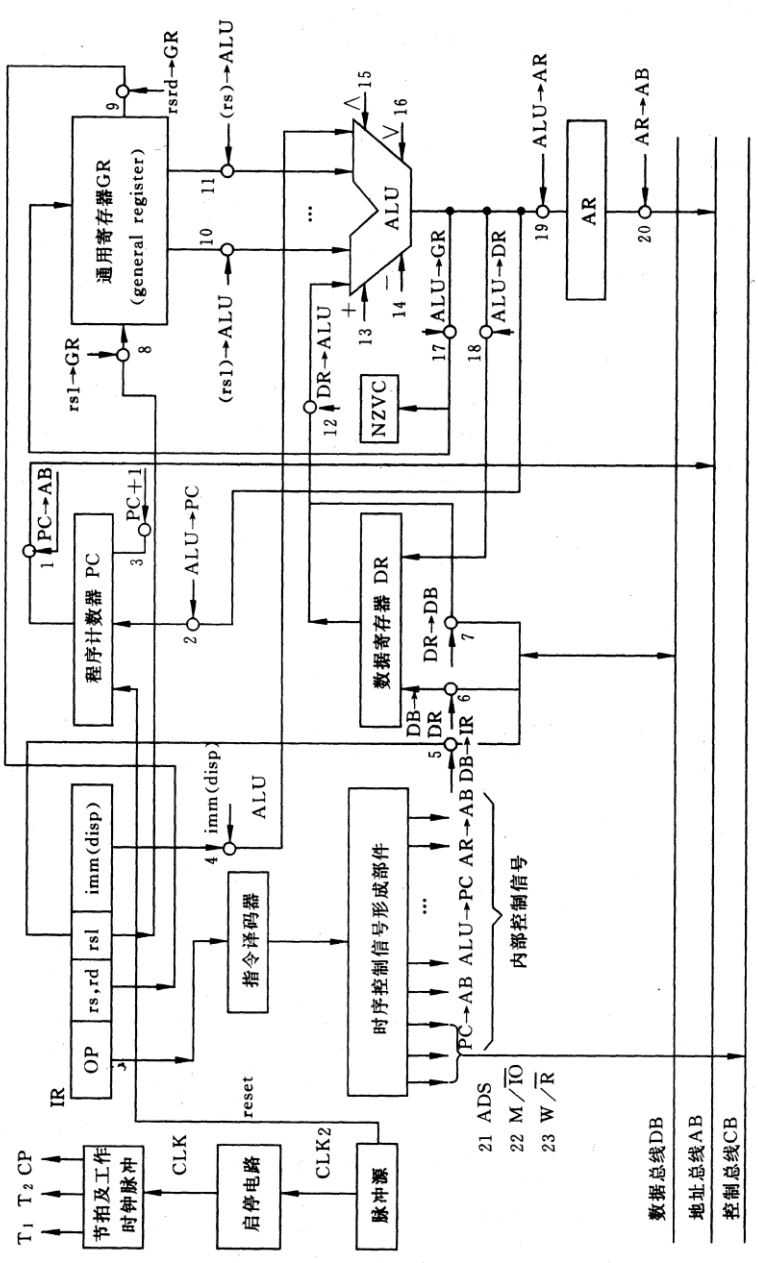
控制器送出的控制信号：rs，rd→GR，　(rs)→ALU，DR→ALU(两个源操作数送 ALU);

ALU→rd(运算结果送寄存器 rd)

(三) 数据通路的功能和基本结构

CPU 的数据通路是连接 CPU 内部各个部件以及和 CPU 外部个部件之间的数据和控制信号的连接关系图。

数据通路的基本结构：



(四) 控制器的功能和工作原理

1. 硬布线控制器

控制器控制信号的产生是采用逻辑电路，也称组合逻辑电路控制方式。“时序控制信号形成部件”是由硬逻辑布线完成的。实际设计中，需要几十~几百条指令，确定每条指令所需的机器周期，将情况相同的指令归并在一起，列出表达式，画出逻辑图。

(1)时序与节拍

每一步由一个机器周期来完成，假设采用 4 个机器周期，总之，需要 4 个不同的信号输出，代表 4 个不同的周期。

(2)操作码译码器

指令的操作码部分指出本指令将执行什么指令，如加法，减法等。对于不同的指令，采用不同的代码表示。

(3)操作控制信号的产生

以加法指令为例，加法指令的完成是由 4 个机器周期 cy1，cy2，cy3，cy4 组成，分别是取指，计算地址，取数，计算 4 个机器周期。

将所有的机器周期的操作控制信号的逻辑表达式全部写出来，就会得到各个操作控制信号的所有表达式，再将这些表达式安每个操作控制信号组合起来，就得到某个操作控制信号的表达式。

取指周期需要产生的操作控制信号如下：

- PC→AB=cy1　　；将 PC 送地址总线
- ADS=cy1·T1　　；存储器地质有效
- M/IO=cy1　　；存储器操作
- W/R=cy1　　；读操作
- DB→IR=cy1　　；将读出的结果送 IR
- PC+1=cy1　　；将程序计数器加 1

计算地址周期 cy2 需要完成有效地址((rs1)+Disp)的计算。产生的操作控制信号如下：

- rs1→GR=加法指令·cy2　　；送通用寄存器地址
- (rs1)→ALU=加法指令·cy2　　；通用寄存器送 ALU
- Disp→ALU=加法指令·cy2　　；偏移量送 ALU
- “+”=加法指令·cy2　　；ALU 执行加法操作
- ALU→AR=加法指令·cy2　　；运算结果送地址总线

例如，“+”操作控制信号在加法指令的 cy2(计算有效地址)和 cy4(操作数相加)时需要；减法指令

的 cy2(计算有效地址)时需要;转移指令的 cy2(计算有效地址)时需要;。

所以，“+”操作控制信号的逻辑表达式如下：

“+”=加法指令 (cy2+cy4)+减法指令 cy2+转移指令·cy2+…

设机器有 7 位操作码(OP0～OP6)，假设加法指令的操作码为 0001100，形成的加法指令信号的逻辑表达式为：

加法指令= OP0OP1OP2OP3OP4OP5OP6

如，某机器 128 条指令，用 7 位操作码(OP0～OP6)，如果其中有 16 条算术逻辑运算指令，可以将这些指令的 3 位操作码都设计相同的编码，如 OP0OP1OP2= 001，而其他位 OP3～OP6 编码表示 16 个不同的指令。

设命令 A 是所有算术逻辑运算在 cy2 周期需要产生的，逻辑表达式：

A=加法指令 cy2+减法指令 cy2+逻辑加指令·cy2+…

=(加法指令 +减法指令+逻辑加指令+…)·cy2

= OP0 OP1 OP2 cy2

只需要一个与门，就可实现命令 A。

2. 微程序控制器

(1)微程序，微指令和微命令

在计算机中，一条指令的功能是通过按一定次序执行一系列基本操作完成的，这些基本操作称为微操作。例如，前面讲到的加法指令，分成四步(取指令，计算地址，取数，加法运算)完成，每一步实现若干个微操作。实现这些微操作的控制命令就是微命令。

微操作是指最基本的，不可再分的操作，如前面提到的：

PC→AB; W/R=0; DB→IR 等。

PC→AB 等就是微命令。

微指令：在微程序控制的计算机中，将由同时发出的控制信号所执行的一组微操作称为微指令，所以微指令就是把同时发出的控制信号的有关信息汇集起来而形成的。将一条指令分成若干条微指令，按次序执行这些微指令，就可以实现指令的功能。组成微指令的微操作又称微命令。

微程序：计算机的程序由指令序列构成，而计算机每条指令的功能均由微指令序列解释完成，这些微指令序列的集合就叫做微程序。

(2)微指令的编码方式;

① 直接控制法

在微指令的控制字段中，每一位代表一个微命令，在设计微指令时，是否发出某个微命令，只要将控制字段中相应位置成“1”或“0”，这样就可打开或关闭某个控制门，这就是直接控制法。

② 字段直接编译法

在计算机中的各个控制门，在任一微周期内，不可能同时被打开，而且大部分是关闭的(相应的控制位为“0”)。所谓微周期，指的是一条微指令所需的执行时间。如果有若干个(一组)微命令，在每次选择使用它们的微周期内，只有一个微命令起作用，那么这若干个微命令是互斥的。

选出互斥的微命令，并将这些微命令编成一组，成为微指令字的一个字段，用二进制编码来表示， 就是字段直接编译法。

③ 字段间接编译法

字段间接编译法是在字段直接编译法的基础上，进一步缩短微指令字长的一种编译法。如果在字段直接编译法中，还规定一个字段的某些微命令，要兼由另一字段中的某些微命令来解释，称为字段间接编译法。

(3)微地址的形式方式。

① 微程序入口地址的形成

<1>一级转移方式

当操作码的位数与位置固定时，可直接使操作码与入口地址的部分位对应。

<2>多级转移方式

先按照指令类型标志转移到某条微指令，以区分出是哪一大类，然后可以进一步按指令操作码转移，区分出是该指令中的哪一类具体操作。

② 微程序后继地址的形成

<1>以增量方式产生后继微地址。

在顺序执行微指令时，后继微地址由现行微地址加上一个增量(通常为 1)形成的;而在非顺序执行时则要产生一个转移微地址。

<2>增量与下址字段结合产生后继微地址

将微指令的下址字段分成两部分：转移控制字段 BCF 和转移地址字段 BAF，当微程序实现转移时，将 BAF 送 μPC，否则顺序执行下一条微指令(μPC＋1)。