煤田物探数字技术讲座

第二讲　电子数字计算机介绍

煤田物探数字技术讲座编写组

一、概　　说

（一）电子计算机的解题过程

用电子计算机进行计算的过程包括以下两个环节：

首先要编制计算过程的“程序”，因为任何一个算题过程都需要按照一定的顺序，一步一步地进行。用计算机进行计算时，这个计算顺序是以一连串的“指令”的形式写出来。

举例来说，计算（a＋b）·c的过程可以用两条指令来完成：第一条指令是将a和b相加，第二条指令是把相加的结果和c相乘。

用指令形式写出的计算顺序叫做“程序”。程序编好以后，就将它送入计算机，然后计算机依次执行这些指令，完成整个解题过程。

由此可见，要使用电子计算机来进行计算，必须具备两个条件：一方面要有用各种元件、器材构成的计算机实体，这叫做“硬件”；另一方面还要有一套保证计算机正常工作的程序，这叫做“软件”，这两个方面是互相配合，不可分割的，在这一讲里面就分别介绍一下这两方面的基本常识。

（二）电子计算机的组成

一般的通用电子计算机包括以下几个主要组成部分：

运算器：是用来进行加减乘除等运算的装置。

存贮器：是用来存贮计算程序、原始数据、中间数据和计算结果等信息的装置。

控制器：是用来使计算按照事先编好的程序进行，使计算机各部分协调配合的装置。

输入装置：是把程序和原始数据送入计算机的装置。

输出装置：是输出计算结果的装置。

存贮器又分为内存贮器和外存贮器两部分。运算器、控制器和内存贮器合称为“主机”；外存贮器和输入、输出装置合称为“外部设备”。

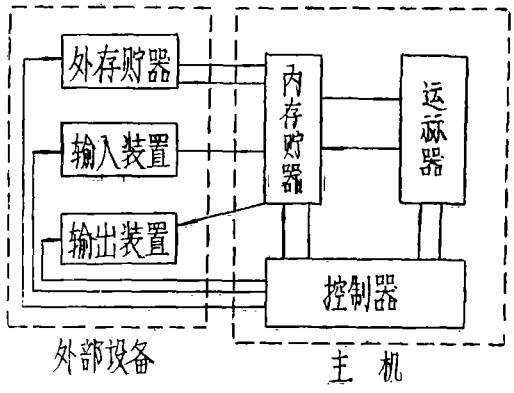


图1. 电子计算机框图

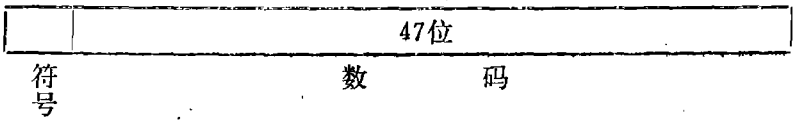
（三）电子计算机中的信息

计算机中的信息包括两种，即数码和指令。

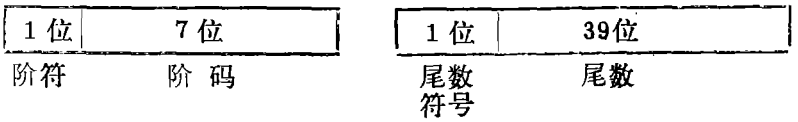
数码包括计算过程中的原始数据、中间数据和计算结果。在一台计算机中所能容纳的二进制数的位数是固定的，称为“字长”。数码可以用定点或浮点形式来表示（见本讲座第一讲：《煤田地质与勘探》1977年第2期）。

举例来说，假定一台计算机的字数是48位，则定点表示和浮点表示的数可以采用下面的格式。

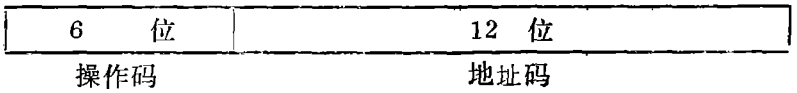
定点数：



浮点数：



指令通常包括操作码和地址码两部分，例如下面的格式：



操作码是表明这条指令所执行的是什么运算。

所谓地址是指内存贮器中存贮数据的位置的编号。地址码就是指明这条指令中用来进行运算的数据是保存在内存贮器的哪个地址中。

此外，为了方便程序设计，指令在执行过程中，其地址码往往要采用不同的方式进行修改。因此，指令中常常还要包括如何修改地址的信息。

二、运 算 器

运算器的作用是对二进制数进行各种算术运算和逻辑运算。在计算机中，各种运算都归结为加法运算。因此运算器的主要组成部分包括一套加法器和几套移位寄存器。计算可以用定点数，也可以用浮点数来进行。下面简单地说明一下运算器进行加、减、乘、除运算的基本原理。

（一）加法：加法是按位进行的，一位二进制数的加法规则如下：

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 10

（即本位的和是“0”同时向上一位进“1”）

图2. 半加器框图

实现上面这个加法规则的逻辑电路可以用图2的框图来代表，它的逻辑功能见表1。这样的电路称为“半加器”，它还不可能完成全部加法的功能，因为除了本位的加数和被加数相加以外，还要加上从下一位来的进位数，因此，完成全部加法功能的一位加法器，其逻辑功能应该是如表2所示。这种加法器称为全加器，其框图如图3所示。

半加器逻辑功能　　表1

输入		输出	
加数	被加数	和	进位
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

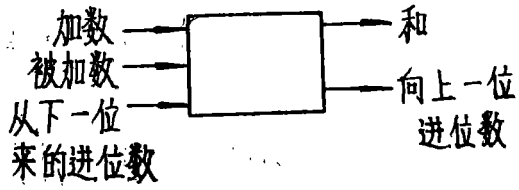


图3. 全加器框图

这样的全加器可以用前一讲所说的“门电路”来做成，也可以用触发器来做成。图4是一个用触发器做成的多位全加器，每一个触发器代表一位。

全加器的逻辑功能　　表2

输入			输出	
加数	被加数	从下一位来的进位数	和	向上一位进位数
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
1	0	0	1	0
0	1	1	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

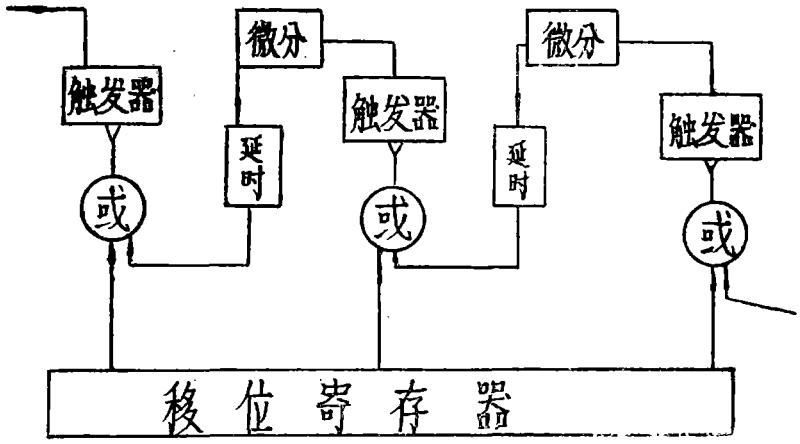


图4. 多位全加器框图

先把被加数保存在各个触发器中，然后把加数（事先保存在移位寄存器中）通过或门加在触发器的计数输入端，这个触发器就起到一个半加器的作用。如果某一位的加数是“0”，则触发器中的被加数保持不变；如果加数是“1”，而被加数是“0”，则相加后，触发器由“0”变为“1”状态；如果被加数也是“1”，则触发器由“1”变为“0”状态，此时，触发器的电位变化经过微分电路产生一个脉冲，也就是进位的信号。这个进位信号经过一定的延迟后，再送到上一位去相加。这样就完成了全加器的功能。

（二）减法：在计算机中，减法是化为加法来做的。因为从a减去b，就等于a加上-b。因此，在计算机中计算a－b时，只要把b的代码变为补码，然后再和a相加就行了。

（三）乘法：计算机进行乘法的原理，和笔算是一样的。例如，$9 \times 13 = 117$这个乘法用二进制形式来计算，可以写成下面的式子：

```
    1 0 0 1
    1 1 0 1
  ---------
    1 0 0 1   ……第一次部分乘积1001 × 1 = 1001
  0 0 0 0     ……第二次部分乘积1001 × 0 = 0000
1 0 0 1       ……第三次部分乘积1001 × 1 = 1001
1 0 0 1       ……第四次部分乘积1001 × 1 = 1001
-----------
1 1 1 0 1 0 1
```

由这个计算过程可以看出来：二进制乘法是这样进行的，先用乘数的最低一位来乘被乘数，得到第一次部分乘积。这很简单，如果乘数最低位是“1”，就把被乘数写在下面，如果是“0”，就在下面写“全0”。然后再用乘数的第二位来乘，其规则相同，只不过写部分乘积时向左移一位，如此乘下去，最后把各个部分乘积相加就得到最后结果。

可见“乘法”运算在计算机中可以由“加法”和“移位”这两种更简单的运算来实现。但是，和上面的算式有两点不同，第一是每乘一位就加一次，不是等到最后总加；第二，不是把每次的部分乘积向左移位，而是每乘一次以后，把相加后的部分乘积向右移一位，这两种做法的效果是相同的。

图5是一个简化的乘法框图的例子。被乘数的各位通过一组“与门”送入全加器，这一组“与门”由乘数的最后一位来控制，如果乘数末位是“1”，则“与门”打开，被乘数进

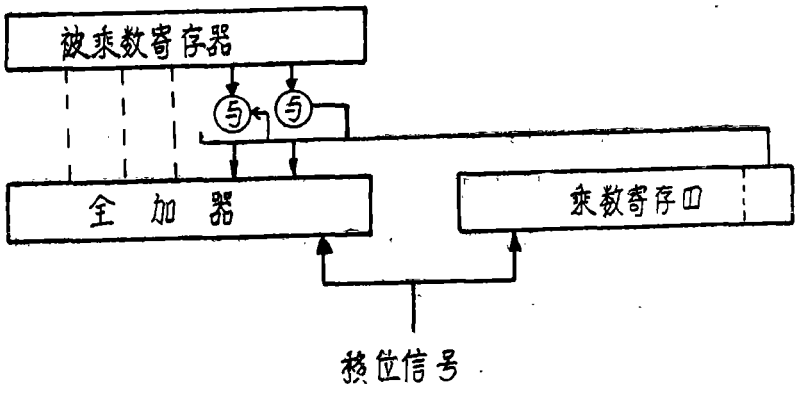


图5. 乘法运算的简化过程示意图

入全加器，如果末位是“0”，则被乘数不送入全加器，然后把全加器和乘数寄存器中的数码同时向右移一位，此时，“与门”就由乘数的次一位来控制了，如此下去，全部移完后，全加器中所保存的数就是乘积。

4.除法：正如乘法运算可以由加法和移位来实现，除法也可以由减法和移位来实现。

例如，30 ÷ 6 = 5 这个除法，用二进制可以写成下面的形式：

```
          1 0 1
      ---------
1 1 0 ) 1 1 1 1 0
        1 1 0        第一位商是1，把除数从被除数中减去
        -------
          0 1 1      ←右移一位再除
          0 0 0      第二位商是0，不用减去除数，
          -----
            1 1 0    ←右移一位再除
            1 1 0    第三位商是1，把除数减去，
            -------
                0
```

由上式可见，当某一位商是1时，就把除数从被除数中减去，如是商是0，就不用减去除数了。因此，在每次除的时候，需要先判断一下商是1还是0，然后决定要不要减去除数。

在计算机中进行除法运算时，有时采用另一种方法，就是每一位商不管是“0”，还是“1”，总是先把除数从被除数中减去，然后判断一下所得的差是正的还是负的。如果正的，则该位商记“1”，然后移位再减；如果差是负的，就说明不够除，该位商记“0”，同时把减掉的除数再加上去，恢复原来的数，然后移位再减，如下式所示：

```
    1 1 1 1 0   被除数
-   1 1 0 0 0   减去除数
  -------------
    0 0 1 1 0   余数为正，商记“1”
-   0 1 1 0 0   除数右移一位，再减
  -------------
-   0 1 1 0 0   余数为负，商记“0”
+   0 1 1 0 0   把除数加上，恢复原来的余数
  -------------
    0 0 1 1 0
-   0 0 1 1 0   除数右移一位，再减
    ---------
    0 0 0 0 0   余数为0，商记“1”
```

三、内 存 贮 器

（一）磁心存贮的基本原理

因为计算机中所要存贮的是二进制的数码，因此用来存贮数码的元件必须具有两种不同的物理状态。目前计算机中所使用的内存贮器，是以磁心存贮器为主要形式，它的基本存贮元件是磁心。

磁心是用具有矩形磁滞回线的材料做成的磁环，如图6所示。如果在磁环中穿过一根导线，并通以电流，磁心内部就产生磁通，磁通随电流而变化的曲线如图7所示。由图中可见，磁心中的磁通有两种状态。如果在导线中通过电流 $+I_m$，则磁心中的磁通是 $+\Phi_m$，电流停止以后，磁心中保留的剩磁为 $+\Phi_c$；如果通过电流 $-I_m$，则磁通是 $-\Phi_m$，电流消失以后，磁通中的剩磁为 $-\Phi_c$。如果令 $+\Phi_c$ 代表“1”，则 $-\Phi_c$ 就代表“0”。

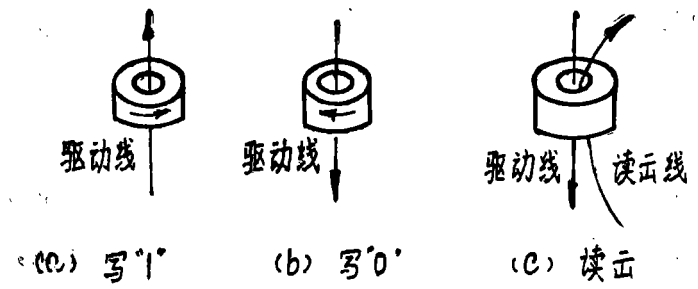


图6. 磁 心

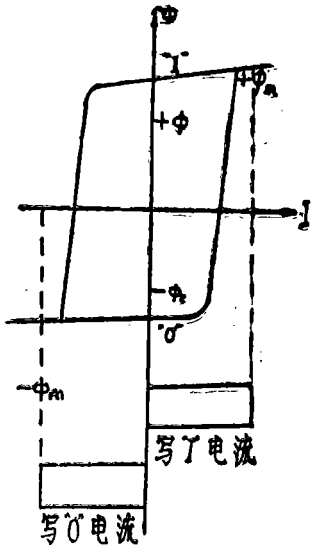


图7. 磁心的磁滞回线

把数码写入磁心中去的手续叫做“写入”，“写入”的方法如图6a，6b所示。在磁心中穿过一根导线，称为“驱动线”。6a表示驱动线中通过幅度为 $+I_m$ 的脉冲电流以后，磁心中的剩磁为 $+\Phi_c$，也就是在磁心中写入“1”。6b表示驱动线通过振幅为 $-I_m$ 的脉冲电流，磁心中的剩磁为 $-\Phi_c$，即写入“0”。

把写入磁心中的代码取出来的手续叫“读出”。为了把代码从磁心中读出，在磁心中穿有另一根导线，叫做“读出线”。在读出的时候，先向驱动线中通过写“0”脉冲电流。如果磁心原来是处于“1”状态，那么，磁通就要从 $+\Phi_c$ 变为 $-\Phi_c$，磁通的突然变化就在读出线中感应一个脉冲电压，这个电压经过放大整形后，就是读出来的“1”代码。如果磁心

原来处于“0”状态，则在通过写“0”电流时，磁通几乎没有什么变化（只有从$-\Phi_s$到$-\Phi_m$的很小的变化），读出线上就没有脉冲输出。一个磁心经过一次读出以后，其中的“1”就变为“0”，也就是存贮的信息被破坏了。因此，在读出“1”以后，要把读出信号再送回到驱动线，在磁心中写入“1”，使它恢复原来的信息。

（二）怎样按地址存取代码

一个磁心只能存一位二进制数。如果要存多位数，例如存一个40位的数码，就需要用40个磁心构成一组。在磁心存贮器中用大量的磁心组成一个磁心体，其中每一组磁心都有一个编号，称为“地址”。每一个地址可以存一个数据。

现在的问题是存取数码时，如何能自动地选择所需要的地址呢？为了达到这个目的，在磁心体中把磁心排列成方形的矩阵，如图8。每一个磁心中穿过两根驱动线，称为X驱动线和Y驱动线。在图8中共有16个磁心，有4根X驱动线和4根Y驱动线。X驱动线和Y驱动线都分别编号。这两个编号合起来就是磁心的地址。例如，编号为“10”的X驱动线和编号为“01”的Y驱动线所穿过的那个磁心，其地址就是“1001”。

在读写的时候，按照地址在相应的X和Y驱动线中通以脉冲电流，其幅度各等于$\frac{1}{2}I_m$。假定原来磁心都处于“0”状态，现在X驱动线中通以$+\frac{1}{2}I_m$脉冲电流，则凡是该线所穿过的磁心都受到这个电流的作用。但是因为电流的幅度只有$\frac{1}{2}I_m$，不足以使磁心翻转到“1”状态。同样地，在编号为“01”的Y驱动线中通过$+\frac{1}{2}I_m$脉冲电流，这条线所穿过的磁心也不会翻转到“1”状态。只有在这两条线同时穿过的那个磁心，也就是地址为“1001”的磁心，其中的励磁电流为$\frac{1}{2}I_m+\frac{1}{2}I_m=I_m$。因此，只有这个磁心翻转到“1”状态，也就是在这个地址中写进了“1”。读出的原理也是一样，即在交叉的两根驱动线中都通过以$\frac{1}{2}I_m$的写“0”电流。读出线只有一根，穿过这个矩阵中的全部磁心。

驱动线是怎样选择的呢？这是通过译码器来进行的。例如，需要在“1001”这个地址中存取代码，就把地址码的前两位“10”送入X地址译码器，经过译码后，在译码器输出端的“10”线上，就送出驱动电流；后两位代码“01”送入Y地址译码器，在译码器输出端的“01”线上送出驱动电流。这样就可以在地址为“1001”的这个磁心中存取代码。

（三）磁心存贮器的构造

像图8这样一个磁心矩阵通常装在一块板上，用来存贮一位数。对于多位数，每一位需要一块磁心板。这些磁心板组成一个磁心体，如图9。图中每块磁心板上纵横各有64个磁心，

共有4096个磁心，也就是可以存贮4096个数。X和Y驱动线也各有64条。假定所要存贮的数是40位，则一共要有40块磁心板。每一块磁心板各有一条读出线。由磁心中读出的数码送入数码寄存器，然后送往运算器。地址码是由控制器送来的，它存在地址寄存器中，然后分别由X和Y两个地址译码器译码后，选择所需要的地址。

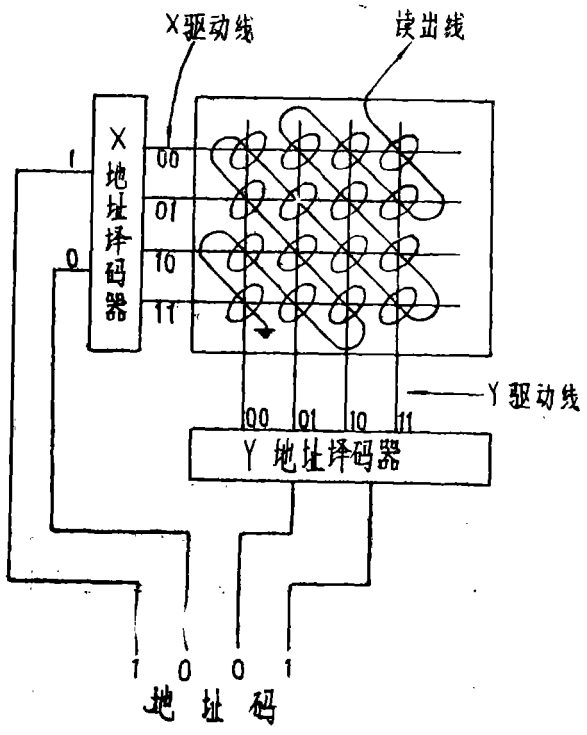


图8.

四、控 制 器

（一）控制器的功能

控制器是全机的控制中心，它控制计算机各个部分使之协调动作，按照一定的顺序来完成整个解题过程。

计算机的解题过程是按照预先编制好的程序进行的。程序中的每一步称为一条指令。这些指令预先存放在内存贮器中，每一条指令存放于一定的地址。计算机解题的过程就是逐条地执行这些指令。在执行指令的过程中控制器要完成以下的任务：

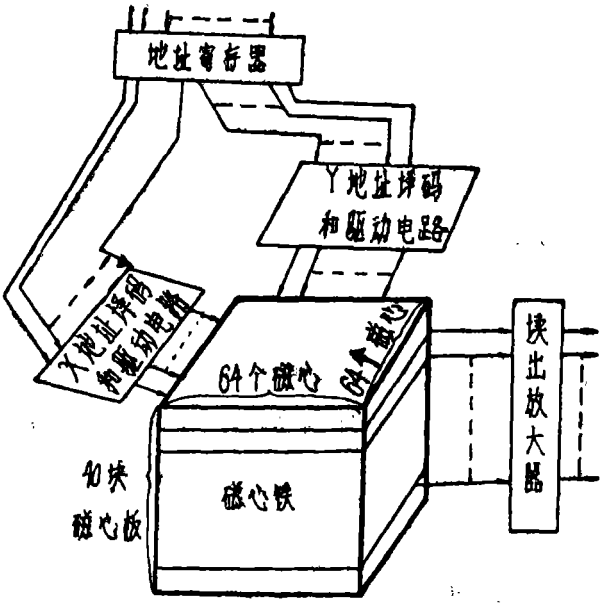


图9.

1.按照“指令地址”的前后次序从内存贮器中逐条取出指令，并寄存在控制器中。

2.根据指令中的操作码，确定机器应进行什么操作，然后发出相应的信号，去控制机器各个部分的动作。

3.与此同时，把指令中的地址

码送入内存贮器，按照规定的地址取出操作数。

4.执行指令所规定的操作。

（二）控制器的组成

为了完成上述的任务，控制器中需要有一套相应的部件，其中最主要的有以下几部分：

1.指令计数器：用来寄存指令地址，它的内容指出了当前要执行的指令应从存贮器的哪个地址去取。通常，计算程序中的指令是按顺序执行的。例如，当执行完“0001”条指令后，就执行“0002”条指令。此后执行“0003”、“0004”…条指令，这就要求指令计数器在每执行完一条指令后就自动加“1”，为执行下一条指令作好准备。

2.指令寄存器：按照指令计数器所给出的“指令地址”，从存贮器中取出的指令就寄存在指令寄存器中。

3.操作码译码器：指令寄存器中的指令操作码由操作码译码器进行译码。操作码译码器有很多输出线，每一种操作有一条输出线，操作码经译码后，在相应的输出线上就出现高电位。例如，进行加法时，加法的输出线上就出现高电位，这个高电位用来控制运算器中的各个电路按照加法运算的规则来动作。

4.节拍脉冲发生器：计算机中执行每一条指令的过程是由很多具体步骤所组成的。例如，做乘法时要进行多次的移位和相加。这些步骤是按照一定的时间节拍来进行，节拍脉冲发生器的作用就是产生一定长短的节拍脉冲，使计算机各个部件的动作按照节拍一步一步地进行。

5.控制台

控制台上装有控制面板。它由一些显示灯，开关按钮和有关控制线路组成。显示灯用以显示机器的某些内部状态（例如显示出在加法器中进行运算的数码）。开关和按钮用来控制计算机的工作方式。控制台的作用是供操作人员对计算机进行操作和检查用的。

（三）计算机执行指令的过程

控制器的工作过程和整个计算机的工作过程是分不开的。下面就以一个具体的例子来说明在算题过程中计算机各个部分是如何工作的。

我们以表4的程序中的前两条指令做例子。这两条指令是

	指令内容	
指令地址	操作码	地址码
0401	05	0001
0402	01	0101

这里，“0401”是指令在内存贮器中存放的地址，“05　0001”是指令的内容。其中，“05”（这里的指令是用八进制写的，因此，05就相当于二进制的000101）是“取数”操作码。第一条指令的内容就是把内存贮器中“0001”地址中的数（以a代表）取出送往运算器的加法器中。“01”是“加法”的操作码。第二条指令的内容就是把“0101”地址中的数（以b代表）取出送往加法器中，和上一次取出的数a相加。

执行一条指令的过程叫做一个周期，每一周期又分为若干节拍。为了加快计算速度，前后两个周期常常有一部分是重叠的。下面为简单起见，假定这两条指令分别在两个周期内执行。整个计算过程如下（图10）：

在第一周期开始时，指令计数器的内容为“0401”，即第一条指令的地址码，首先把这个地址送入内存贮器，经地址译码后，从“0401”这个地址中取出第一条指令，送往控制器的指令寄存器。同时指令计数器的内容自动地加1，变为“0402”，为下一条指令作好准

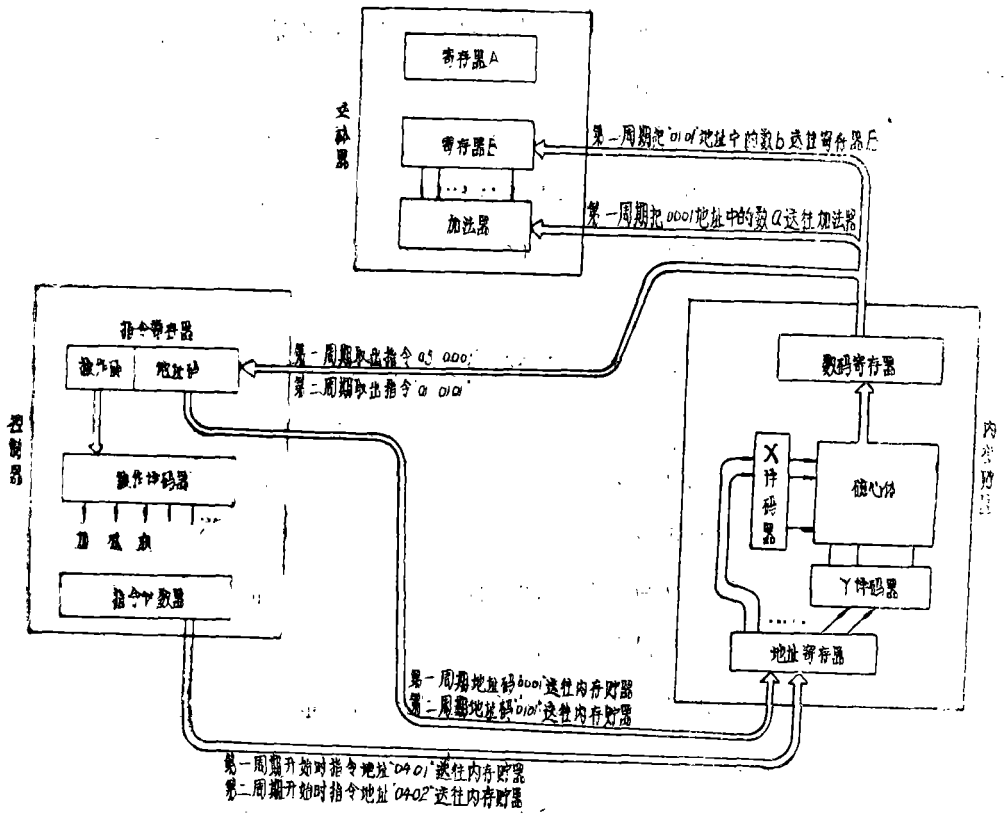


图10

备。指令寄存器中的操作码“05”由操作译码器译码后，发出相应的“取数”操作信号。地址码“0001”则送往存贮器，经地址译码后，从“0001”这个地址中取出数a送往运算器的加法器。

第二周期开始时，将指令计数器的内容“0402”送往内存贮器，从“0402”地址中取出第二条指令“01 0101”，送入指令寄存器。操作码“01”经操作译码器译码后，产生“加法”信号；地址码“0101”送往内存贮器，从“0101”地址中取出数b送往运算器中的移位寄存器B。然后在“操作信号的控制下，将数b送往加法器和a相加，得到和数a + b。

五、外部设备

（一）外存贮器

对计算机的存贮器有两方面要求：一方面要求它所能存贮的信息容量大，另一方面要求存取信息的速度快。但是一种形式的存贮器很难兼顾这两方面的要求。为此，把存贮器分为内存贮器和外存贮器两部分。内存贮器全部由电路组成（主要采用磁心存贮器，近年来开始采用了由半导体元件组成的存贮器）。其存取速度快。存贮容量较小。外存贮器的特点是存贮容量大，但这一般都包含有机械运动，因此存取速度慢。在运算过程中，内存贮器直接和运算器、控制器互相配合。计算过程中的数据可以随时存入内存贮器或从内存贮器中取出。而外存贮器只是在必要时成批地把信息送往内存贮器或成批地从内存贮器接收信息。

常用的外存贮器有磁鼓、磁带和磁盘等形式，它们的原理和磁心存贮器相同，都是利用磁性材料的涂层作为存贮介质，用磁性材料的两种剩磁状态来表示“1”和“0”。

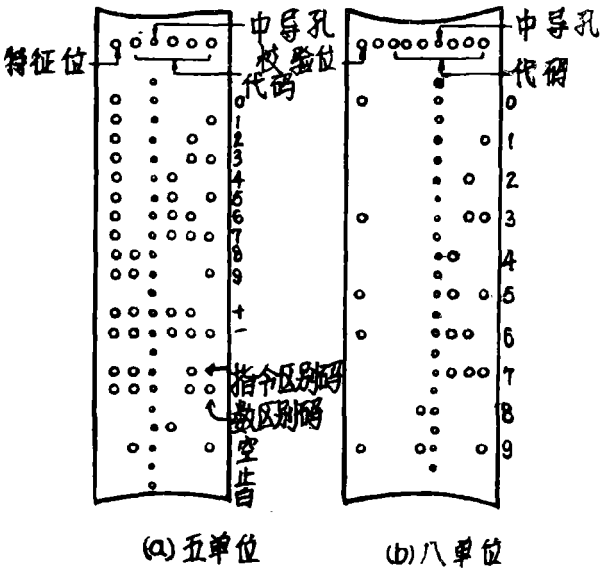


图11. 穿孔纸带

（二）输入装置

输入装置的作用是把算题的程序和原始数据变为电信号送入计算机

中。最常用的输入装置是光电输入机。

利用光电输入机进行输入，首先要把信息记录在穿孔纸带上。常用的穿孔纸带形式有五单位和八单位两种，见图11。除纸带上的小孔外，五单位纸带每排有五个信息孔，八单位纸带每排有8个信息孔。纸带上的每一个孔代表一位二进制数码。有孔表示“1”，无孔表示“0”。

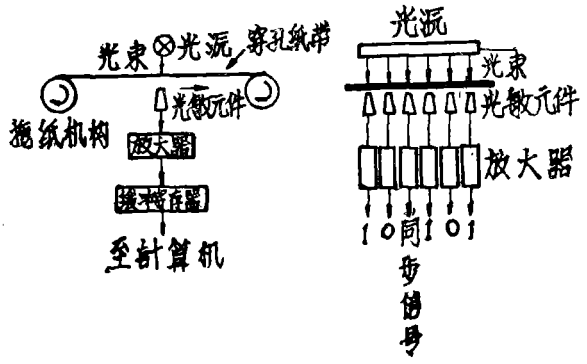


图12　光电输入机原理图

光电输入机是由光源、光敏元件、放大器及纸带拖动机构组成，见图12。当纸带由拖纸机构送进时，光源发出的窄光束连续地照在纸带上。光束照到孔时，光就通过孔照射在光敏元件上，使光敏元件导电，经放大器产生一个脉冲信号，使寄存器的触发器置于“1”状态，相当于把“有孔”所代表的代码“1”输入到寄存器。当光束照到无孔的纸带上，光被挡住，光敏元件不导电，放大器没有电脉冲输出，寄存器的触发器不能触发而置于“0”状态，相当于把“无孔”所表示的代码“0”输入到寄存器。这样，当纸带在小孔的同步下不断前进时，便把纸带上的信息孔逐个地送入缓冲寄存器，并由缓冲寄存器转送到计算机的存贮器中。

（三）输出装置

输出装置是把计算的结果以人所能识别的形式，如数字、字母或符号、图形等表示出来。输出装置的形式很多，下面简单介绍一下行式打印机。

打印机包括机械部分和控制电路。机械部分如图8所示，它的机械部分包括一个圆柱形的字轮，圆柱体的表面刻有凸出的数字、字母和符号。例如，80行64字符的宽行打印机，字轮的圆周方向有64个字符，字轮的轴线方向有80位相同的字符。这80位字符都有一相应的字锤。字锤由电磁铁直接驱动。打印是通过字轮旋转，字锤打击的方式实现的。通过色带把要打印的字符印于纸上。另有一个编码盘和字轮一同旋转，当某一个字符转到字锤下面时，编码盘就发出和该字符相应的代码。如果我们需要某一字锤打出该字符，就把该字符的代码送入打印控制电路的数码寄存器中。例如需要第二位字锤打出“3”这个字符，就把“3”的代码“0011”送到第二位的数码寄存器中。当字轮上的“3”转到字锤下面时，编码盘也发出“0011”这个代码，和数码寄存器中的代码完全符合。这时就产生一个控制信号，控制

第二位电磁铁打出字符“3”来。字轮每转一圈，就把一行字中的各个不同的符号都打出来了。

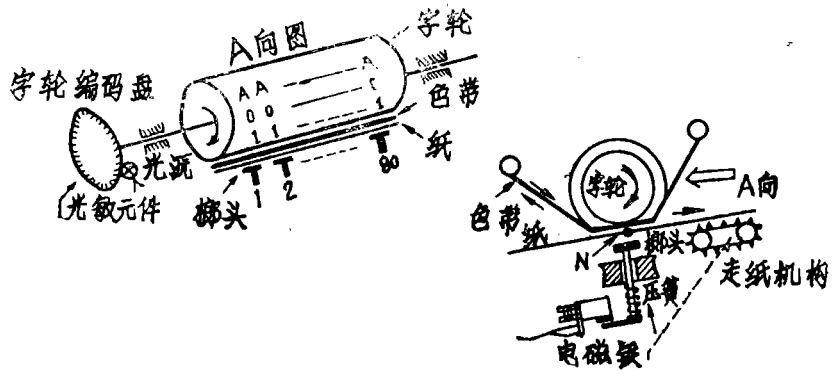


图13. 打印机械部分
右图为左图的侧视图

六、程 序 设 计

要在电子计算机上解题，首先要把需要解决的问题写成数学形式，也就是把这个问题变为数学问题。其次要给这个数学问题选择一个计算方法，确定计算步骤。然后把这个计算步骤用程序的形式写出来。程序是由若干条指令组成的，每条指令完成一个操作。对任何一台计算机来说，它能够进行哪些操作，是在设计计算机的时候规定好了的。一台计算机所能实现的全部操作称为该计算机的“指令系统”。因此，要利用某一台计算机来解题，就必须根据这台计算机的指令系统来编制程序。

下面结合实例简单介绍一下程序设计的基本概念，一般编写程序都是用八进制或十六进制，下面用的是八进制。

我们假定有一台简单的计算机，它的指令系统如表3所示：

一个简单计算机的指令系统　　表3

操作名称	加	减	乘	除	取 数	存 数	比 较	无条件转移	条件转移	打 印
操 作 码	01	02	03	04	05	06	07	10	11	12

现在以计算下面的公式作为例子

$$x_i = (a_i + b_i) \cdot c_i \qquad i = 1, 2, \cdots 64$$

这里一共有64组数据，即从a_1、b_1、c_1到a_{64}、b_{64}、c_{64}。编制程序的第一步，是在内存

贮器中给这些数据分配一定的地址。现在规定把a_1……a_{64}存放在地址“0001”至“0100”中，把b_1、…b_{64}存放在地址“0101”至“0200”中；把c_1……c_{64}存放在地址“0201”至“0300”中。此外，计算结果X_1……X_{64}也要暂时存放在内存贮器中，现在规定将他们存放在地址“0301”至“0400”中。程序存放在从“0401”开始的地址中。以上的地址分配见表4。

下面开始编制程序，首先计算$X_1=(a_1+b_1)\cdot c_1$，其步骤是先从地址“0001”取出数据a_1，存放在加法器中，再从地址“0101”取出数据b_1，将它和a_1相加。相加的结果即a_1+b_1仍存放在加法器中。然后从地址“0201”取出数据c_1，和加法器中的和数a_1+b_1相乘。乘积就是X_1，将它存放到地址“0301”中，这一共需要4条指令，即

指令地址	指令内容	
0401	05 0001	取a_1
0402	01 0101	取b_1，和a_1相加，得到a_1+b_1
0403	03 0201	取c_1，和a_1+b_1相乘，得到X_1
0404	06 0301	存X_1

下面就应该接下去计算$X_2=(a_2+b_2)\cdot c_2$。但是，像这样编出来的程序，每计算一个数据，需要4条指令，算出全部数据就需要64×4即256条指令。不但要花费程序人员大量的时间，而且要占用内存贮器的大量地址。我们可以看出来，每计算一个数据，所用的4条指令，都是相似的，其操作码是重复的，其地址码是对每个数据加1，例如a_1是从0001取出的，而a_2则是从0002取出的，等等。因此，可以采用“修改指令”的方法，编成“循环程序”，利用这4条指令来计算全部数据。修改指令的方式很多。在这里为简单起见，规定在每条指令的后面增加一位代码。如果这一位是“1”，就表示在这条指令执行完了以后，需要把它的地址码加1，仍旧保存在内存贮器中，作为计算下一个数据用。在每次执行完上面4条指令以后，利用一条转移指令来从新从前面开始执行。这样编出来的程序如下：

→0401	05	0001	1	取a_i
0402	01	0101	1	取b_i，和a_i相加，得到a_i+b_i
0403	03	0201	1	取c_i，和a_i+b_i相乘，得到x_i
0404	06	0301	1	存x_i
—0405	10	0401	0	返回去执行0401

操作码“10”表示“无条件转移”。执行这条指令时，是把指令计数器中的地址改变为“0401”，取出的下一条指令就是“0401”了。

这样只要5条指令就可以进行全部计算。但是，这样编出的程序还不能用。因为计算机并不知道什么时候应该停止计算（在上面这个例子中，应该在循环64次，即算出X_{64}以后，就自动停止）。因此，必须把这一点在程序中指明。这样编出来的就是表4中的程序。

计算$X_i=(a_i+b_i)\cdot c_i$　　（i＝1，2，……64）的程序　　表4

地　址	内　容	意　义
0001	a_1	原始数据
0002	a_2	
⋮	⋮	
0100	a_{64}	
0101	b_1	
⋮	⋮	
0200	b_{64}	
0201	c_1	
⋮	⋮	
0300	c_{64}	
0801	x_1	计算结果
⋮	⋮	
0400	x_{64}	
→0401	05　0001　1	取a_i
0402	01　0101　1	取b_i，和a_i相加，得到a_i+b_i
0403	03　0201　1	取c_i，和a_i+b_i相乘，得到x_i
0404	06　0301　1	存x_i
0405	05　0413　0	取出“0413”的内容（循环次数）
0406	01　0414　0	“循环次数”加1
0407	06　0413　0	送回“0413”
0410	07　0415　0	将循环次数和“0415”中的数“64”比较
0411	11　0401　0	若循环次数<64，则执行“0401”； 若=64则执行下一条指令
0412	12　0000　0	打印计算结果
0413	0000	“循环次数”计数器
0414	0001	常数“1”
0415	0100	常数“64”

其中“0413”这个地址作为循环次数计数器用，每算完一个数据，就把这个地址中的内容加1，也就是把“0414”中的数码“0001”加到“0413”中的数码上。然后把相加的结果和“64”来比较（“64”这个数存在“0415”中）。如果相加的结果小于64，就继续算下去，如果等于64，计算就停止。这一点是利用“0411”这条指令来执行的。它的操作码“11”表示“条件转移”，就是要根据上一条指令的比较结果来决定向哪一条指令转移，如果比较结果小于64，则转向“0401”，如果等于64，就执行下一条指令“0412”，即把全部算出的数据$X_1 \cdots X_{64}$打印出来。

随着计算技术的发展，计算机的使用范围愈来愈广，所计算的问题也愈来愈复杂，编制程序就需要大量的人员。为了节省编制程序的人力和时间，用计算机来编制程序的自动化程序设计就发展起来了。所谓程序设计自动化，首先还是要由人来编程序，但不用写出详细的程序，而只要规定具体的计算步骤。这种计算步骤叫做“源程序”，它是用一种特殊的符号（称为“算法语言”）来编写的。“源程序”编好后，就输入计算机，在计算机中另有一套预先编好的“编译程序”，计算机利用“编译程序”，就能自动地把“源程序”编译成机器上真正能执行的程序了。