

怎样用算法语言编程序

(709 计算机 TQ-16 计算机)

顾鼎铭 罗文化 陈恕行

上海科学技术出版社

内 容 简 介

本书通俗、详尽地介绍了在 709 电子数字计算机和 TQ-16 电子数字计算机上用算法语言编写计算程序的方法，并讲述了在这两台机器上算题的操作技术。另外，考虑到算题人员实际使用的需要，书中还选编了部分常用数值方法的标准程序。

本书可供科技人员及在电子数字计算机上算题的人员阅读、参考，也可作为算法语言短训班的普及教材。

怎样用算法语言编程序

(709 计算机 TQ-16 计算机)

顾鼎铭 罗文化 陈恕行

上海科学技术出版社出版

(上海瑞金二路 450 号)

新华书店上海发行所发行 上海日历印刷厂印刷

开本 787×1092 1/32 印张 8.5 字数 186,000

1980 年 2 月第 1 版 1981 年 4 月第 2 次印刷

印数 25,001—57,000

书号: 13119·763 定价: 0.69 元

目 录

引 言

第一章 电子数字计算机概述	1
第一节 709 电子数字计算机和 TQ-16 电子数字计算机简介	1
第二节 电子数字计算机上数的表示形式	3
一 十进制, 二进制, 八进制	3
二 数制的转换	5
三 数的浮点形式	8
四 计算机单元中数的表示形式	9
五 数的二十进制形式	13
第三节 电子数字计算机解算生产实际问题的全过程	15
一 构造数学模型, 选择计算方法	15
二 计算过程的程序设计	16
三 程序的信息化及其在计算机上的执行	19
第二章 程序的编写方法	21
第一节 程序的初步认识	21
第二节 算法语言的基本成分	22
一 基本符号	22
二 标识符, 常量, 变量, 标准函数	24
三 表达式	29
第三节 基本语句	34
一 赋值语句	34
二 转向语句	36
三 条件语句	37
四 循环语句	41

五	停机语句	48
六	空语句	48
七	输入语句	49
八	输出语句	52
九	十翻二语句	64
第四节	说明和分程序	69
一	简单变量说明	70
二	数组说明	73
三	分程序	80
四	开关说明	86
第五节	过程	92
一	过程的例子	92
二	过程的一般形式	98
三	常用算法的标准过程	110
第六节	程序的例子	115
第三章	常用数值方法的标准程序	127
一	一元三点插值	127
二	二元三点插值	128
三	龙贝方法计算单重积分	130
四	高斯法计算二重积分、三重积分	131
五	严格选大元消去法解线性代数方程组	136
六	共轭斜量法解线性代数方程组	138
七	高斯-塞德尔迭代法解线性代数方程组	141
八	部分主元消去法解高阶线性代数方程组	143
九	高阶对称正定稀疏线性代数方程组的分块解法	146
十	用消去法求矩阵的逆阵和行列式值	151
十一	用雅可比方法求对称矩阵特征值和特征向量	153
十二	求实对称矩阵全部特征值和特征向量的 QL 算法	156
十三	求实对称矩阵部分(或全部)特征值、特征向量的吉文斯-豪塞豪德方法	159

十四	用幂法和穷举法求实矩阵的部分特征值和特征向量	165
十五	求实矩阵全部特征值、特征向量的 QR 算法	168
十六	解广义特征值问题的子空间迭代法	180
十七	求高次代数方程全部实根的牛顿迭代法	188
十八	求实函数零点的姜勃法	191
十九	用拟牛顿法解非线性方程组	194
二十	用吉尔方法解常微分方程组(定步长)	197
二十一	用龙格-库塔法解常微分方程组(变步长)	199
二十二	逐步回归分析程序	202
二十三	用改进的单纯形法解线性规划问题	206
二十四	快速傅里叶变换(FFT)程序	209
第四章	计算实习	212
第一节	穿孔	212
一	源程序穿孔	213
二	原始数据穿孔	222
第二节	上机计算	227
一	控制台简介	227
二	上机操作的基本技术	235
三	上机操作的基本步骤	237
四	几种常用的操作技术	243
附录一	源程序书写限制表	248
附录二	源程序语法错误表	249
附录三	非正常停机表	252
习题解答		254
参考资料		263

第 一 章

电子数字计算机概述

电子数字计算机是由电子元件构成的一种现代化计算工具,它好象一个自动化的数字加工厂,用数字作为原料,经过加工、运算,得出数字形式的成品。由于电子数字计算机具有运算速度快、存贮容量大、计算精度高等特点,因此,可以利用它来完成大量繁复的计算工作,解决科学技术、工程设计以及其他各个领域中的计算问题。

我国自行设计制造的 709 机和 TQ-16 机是目前使用比较广泛的两台中型电子数字计算机。

第一节 709 电子数字计算机和 TQ-16 电子数字计算机简介

709 计算机是以集成电路为主要元件的通用电子数字计算机,平均每秒钟可以完成十二万次运算。机器配置有自动化程序的编译系统——用机器语言编写成的一个加工程序,它能够把用算法语言写成的源程序编排、翻译成用机器语言表示的目标程序。TQ-16 计算机基本上是对部分功能作了改进的 709 计算机,除了编译系统外,它还配置有简易的管理系统,具有如下三种功能:

1. 使用外部设备的管理;

2. 接受并处理算题人员对机器的干预;

3. 指示机器运行状态, 以便于算题人员对出错、故障的监督、维护和处理。

709 计算机、TQ-16 计算机都由内存、运控和外部设备三大部分组成:

内存 是指内存贮器, 它用来存贮程序、原始数据和计算结果。709 机和 TQ-16 机的内存贮器是由磁芯体构成, 它好比是一座大楼房, 由许多个称为单元的房间所组成, 单元的总数称为计算机的内存容量。磁芯体的单元是由许多个能够表示二进制数字的小元件串联而成的, 小元件的个数称为单元的字长。另外, 为了存取信息的需要, 磁芯体的单元还人为地依次编上一个号码, 称为单元的地址。

709 机和 TQ-16 机的内存容量是 32768, 单元地址依次编号为 0~32767 (八进制数编号时为 000000~077777)。每个单元的字长为 48 个二进位。

运控 是运算器和控制器的总称。机器的各种运算都在这里进行, 各种命令都由这里发出。这一部分中, 同算题人员直接有关的设备, 709 机是一个人工控制台, TQ-16 机是一个使用控制台 (包括控制台面板和控制台打字机)。利用这些设备可以使机器停止运算, 启动工作, 输入、输出信息以及改变单元内容或运算次序等。

外部设备 包括光电机、排印机、磁鼓和磁带机四种设备。

光电机用来输入程序和原始数据。排印机用来输出运算结果和程序。目前, 709 机和 TQ-16 机都配有两台光电机和两台排印机。

磁鼓和磁带统称为机器的外存贮器。每台磁鼓有 14336 个单元。与磁鼓相比, 磁带容量更大, 可达几十万个单元, 但

存取时间较慢。目前, 709 机未配磁带机, 只配有四台磁鼓, 算题人员可以使用三台, 磁鼓单元统一编号为 0~43007(八进制数编号时为 000000~123777)。TQ-16 机配有三台磁鼓、两台磁带机, 磁鼓编号为 0、1、2, 算题人员可以使用 1 号鼓、2 号鼓, 磁鼓单元分别编号为 0~14335(八进制编号时为 000000~033777)。磁带机编号为 0、1, 对算题人员一般只开放 0 号磁带机, 以组分段形式存取信息。

机器各部分的联系示于图 1。

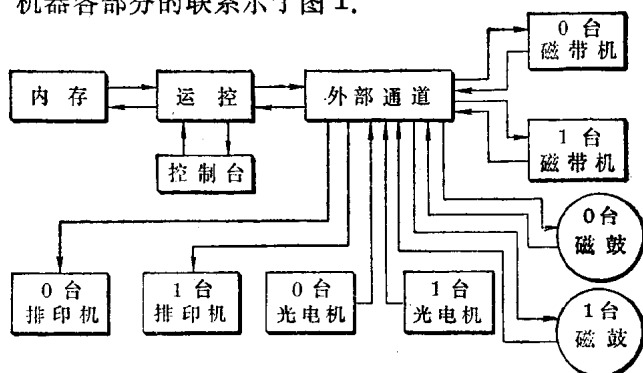


图 1 机器各部分联系图

第二节 电子数字计算机上数的表示形式

电子数字计算机加工、运算的对象是表示为浮点形式的二进制数。这种形式的数同我们日常接触的十进制数不一样, 因此, 要用计算机来解算生产实际问题, 就要了解数在计算机里的各种表示形式, 以及它们之间的转换方法。

一 十进制, 二进制, 八进制

我们日常生活中, 用得最多的数是表示为十进制形式的

数，叫做十进制数。这种数的每一位数字都是 0、1、2、3、4、5、6、7、8、9 十个数字中的一个，而且，数中不同位置上的数字具有不同的含义。

例如，十进制数 2.375 表示为

$$2.375_{(十)} = 2 \times 10^0 + 3 \times 10^{-1} + 7 \times 10^{-2} + 5 \times 10^{-3},$$

其中，自左至右的十进位数字 2、3、7、5 分别是 10^0 、 10^{-1} 、 10^{-2} 、 10^{-3} 数位上的数字。每一位上的数字满了十，就向高一位上进一，所以，十进制就是“逢十进一”的计数制。

除了十进制数之外，我们还经常碰到许多非十进制数。譬如，计算时间的时候，60 秒钟作为 1 分钟，60 分钟作为 1 小时，这就是六十进制。在使用电子数字计算机时，我们将更多地碰到二进制数和八进制数。

二进制数就是数的每一位数字都由 0、1 两个数字中的一个所构成，而且，数中不同位置上的数字具有不同的含义。

例如，二进制数 10.011 表示为

$$10.011_{(二)} = 1 \times 2^1 + 0 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3} \\ = 2.375_{(十)},$$

其中，自左至右的二进位数字 1、0、0、1、1 分别是 2^1 、 2^0 、 2^{-1} 、 2^{-2} 、 2^{-3} 数位上的数字。每一位上的数字满了二，就向高一位上进一，所以，二进制就是“逢二进一”的计数制。

从本例看出，二进制中的 10.011 就等于十进制中的 2.375。

由于二进制数的每一位数字只有 0 或 1，形式很简单，运算也方便，所以，大多数计算机都采用二进制作为数的表示形式。当然，这种二进制形式的数使用时也有不便之处，譬如，它的数位很长，书写起来很累赘，而且，它同习惯使用的十进制数关系也不明显等。为了弥补这些缺点，我们引进数的八

进制形式。因为，用八进制形式表示一个数时，它所占的位数同用十进制表示时相差不大，而且，数的八进制形式同二进制形式有简明的换算关系。

八进制数就是数的每一位数字都是 0、1、2、3、4、5、6、7 八个数字中的一个，而且，数中不同位置上的数字具有不同的含义。

例如，八进制数 2.3 表示为

$$2.3_{(8)} = 2 \times 8^0 + 3 \times 8^{-1} = 2.375_{(10)},$$

其中，2 是 8^0 数位上的数字，3 是 8^{-1} 数位上的数字。每一位上的数字满了八，就向高一位上进一，所以，八进制就是“逢八进一”的计数制。

从本例看出，八进制表示中的 2.3，等于十进制表示中的 2.375，也等于二进制表示中的 10.011。

二 数制的转换

(一) 数的十进制形式同八进制形式的互化

任何一个数 x ，在十进制里的表示形式与八进制里的表示形式是不一样的，但它们代表的数值大小是相同的，所以这两种数的表示形式之间有一定的转换关系。

先讲数的十进制形式转换成八进制形式的方法。

1. 整数的化法——“除八取余”法则。

【例】把十进制整数 $75_{(10)}$ 转换成八进制形式的整数。

假设转换后的八进制整数是 $k_n k_{n-1} \cdots k_0_{(8)}$ ，则应有

$$75_{(10)} = k_n \times 8^n + k_{n-1} \times 8^{n-1} + \cdots + k_0 \times 8^0.$$

我们从这个式子中把 $k_n, k_{n-1}, \cdots, k_0$ 确定出来，为此，等式两端除以 8，得

$$9 + 3 \times 8^{-1} = k_n \times 8^{n-1} + k_{n-1} \times 8^{n-2} + \cdots + k_0 \times 8^{-1}.$$

比较等式两端得

$$k_0 = 3,$$

$$9 = k_n \times 8^{n-1} + k_{n-1} \times 8^{n-2} + \cdots + k_1 \times 8^0.$$

将第二式两端除以 8, 得

$$1 + 1 \times 8^{-1} = k_n \times 8^{n-2} + k_{n-1} \times 8^{n-3} + \cdots + k_1 \times 8^{-1}.$$

再比较等式两端得

$$k_1 = 1,$$

$$1 = k_n \times 8^{n-2} + k_{n-1} \times 8^{n-3} + \cdots + k_2 \times 8^0.$$

又从第二式比较两端得

$$k_2 = 1.$$

从而

$$75_{(+) } = (k_2 k_1 k_0)_{(A)} = 113_{(A)}.$$

2. 小数的化法——“乘八取整”法则.

[例] 把十进制小数 $0.825_{(+)}$ 转换成八进制形式的小数.

假设转换后的八进制小数是 $0.k_1 k_2 \cdots k_n \cdots_{(A)}$, 则应有

$$0.825_{(+) } = k_1 \times 8^{-1} + k_2 \times 8^{-2} + \cdots + k_n \times 8^{-n} + \cdots.$$

我们从这个式子中把 $k_1, k_2, \cdots, k_n, \cdots$ 确定出来, 为此, 等式两端乘 8, 得

$$6 + 0.6 = k_1 + k_2 \times 8^{-1} + \cdots + k_n \times 8^{-n+1} + \cdots.$$

比较等式两端得

$$k_1 = 6,$$

$$0.6 = k_2 \times 8^{-1} + k_3 \times 8^{-2} + \cdots + k_n \times 8^{-n+1} + \cdots.$$

第二式两端乘 8, 得

$$4 + 0.8 = k_2 + k_3 \times 8^{-1} + \cdots + k_n \times 8^{-n+2} + \cdots.$$

比较等式两端得

$$k_2 = 4,$$

$$0.8 = k_3 \times 8^{-1} + k_4 \times 8^{-2} + \cdots + k_n \times 8^{-n+2} + \cdots.$$

第二式两端乘 8, 得

$$6 + 0.4 = k_3 + k_4 \times 8^{-1} + \cdots + k_n \times 8^{-n+3} + \cdots.$$

比较等式两端得

$$k_3 = 6,$$

$$0.4 = k_4 \times 8^{-1} + k_5 \times 8^{-2} + \cdots + k_n \times 8^{-n+3} + \cdots$$

.....

如此一直下去, 可以求出 $k_1, k_2, k_3, \cdots$. 从而,

$$0.825_{(+)} = (0.k_1k_2k_3\cdots)_{(\lambda)} = 0.6463146314\cdots_{(\lambda)}.$$

3. 一般十进制数的化法.

先用上面两种方法, 把一般十进制数的整数和小数分别转换成相应的八进制形式, 再综合两部分的结果, 就得到一般十进制数的八进制形式.

[例] 把十进制数 $75.825_{(+)}$ 转换成八进制形式.

先把十进制整数 $75_{(+)}$ 按“除八取余”法则转换成八进制形式的整数 $113_{(\lambda)}$:

$$\begin{array}{r} 8 \overline{) 75} \\ \underline{8 9} 3 \\ 1 1 \end{array} \quad \begin{array}{c} \uparrow \\ 3 \\ 1 \end{array}$$

再把十进制小数 $0.825_{(+)}$ 按“乘八取整”法则转换成八进制形式的小数 $0.6463146314\cdots_{(\lambda)}$:

$$\begin{array}{r} 0.825 \\ \times 8 \\ \hline \boxed{6}.600 \\ \times 8 \\ \hline \boxed{4}.800 \\ \times 8 \\ \hline \boxed{6}.400 \\ \vdots \end{array} \quad \begin{array}{c} \downarrow \\ 6 \\ 4 \\ 6 \end{array}$$

最后, 把两部分结果合并起来, 即得

$$75.825_{(+)} = 113.6463146314 \dots_{(A)}.$$

如果要把一个八进制形式的数转换成十进制形式，则可以根据八进制数中各位数字的含义直接得出。

【例】把八进制数 $15.6_{(A)}$ 化为十进制形式，只要按公式直接计算即可：

$$15.6_{(A)} = 1 \times 8^1 + 5 \times 8^0 + 6 \times 8^{-1} = 13.75_{(+)}.$$

(二) 数的八进制形式同二进制形式的互化

由于 $8 = 2^3$ ，所以，八进制数的一位可以用二进制的三位来表示：

八进制	0	1	2	3	4	5	6	7
二进制	000	001	010	011	100	101	110	111

由此，数的八进制形式化为二进制形式时，可以采用“一位拉三位”的方法。

【例】八进制数 $113.6463_{(A)}$ 化为二进制形式时有

$$113.6463_{(A)} = 001001011.110100110011_{(2)}.$$

反过来，数的二进制形式化为八进制形式时，应以小数点作为基准，往左、往右都是用“三位并一位”，不足三位用“0”补。

【例】二进制数 $1110.011111_{(2)}$ 化为八进制形式时有

$$001110.011111_{(2)} = 16.37_{(A)}.$$

三 数的浮点形式

上面提到的数的各种表示形式，有一个共同的特点，就是它们的小数点位置都是固定的。例如， $75.825_{(+)}$ 中，小数点固定在数字 5 与 8 之间；再如， $110.011_{(2)}$ 中，小数点固定在左起第三、第四位数字中间； $75.32_{(A)}$ 中，小数点固定在左起第二、第三位数字中间等等。这种小数点具有固定位置的数

叫做定点数。

另外,对于任何一个数来说,除了可以表示为定点形式外,还可以表示成浮点的形式,即表示成小数点是浮动的形式。

[例] 十进制数 $75.825_{(+)}$ 可以表示为

$$\begin{aligned}75.825_{(+)} &= +10^{+2} \cdot 0.75825_{(+)} \\ &= +10^{+3} \cdot 0.075825_{(+)} = \dots\end{aligned}$$

[例] 二进制数 $110.011_{(=)}$ 可以表示为

$$\begin{aligned}110.011_{(=)} &= +2^{+11} \cdot 0.110011_{(=)} \\ &= +2^{+100} \cdot 0.0110011_{(=)} = \dots\end{aligned}$$

[例] 八进制数 $75.32_{(\wedge)}$ 可以表示为

$$\begin{aligned}75.32_{(\wedge)} &= +8^{+2} \cdot 0.7532_{(\wedge)} \\ &= +8^{+3} \cdot 0.07532_{(\wedge)} = \dots\end{aligned}$$

一般地,任何一个数 x 都可以表示为

$$x = \pm N^{\pm p} \cdot q,$$

这里, N ——数制的底数. 十进制数的底数 N 等于 10, 二进制数的底数 N 等于 2, 八进制数的底数 N 等于 8.

p ——称为“阶数”, 它是 N 进制表示的正整数.

q ——称为“尾数”, 它是 N 进制表示的正小数, 即

$$0 \leq q < 1.$$

若数的浮点形式中, 有 $\frac{1}{N} \leq q < 1$, 则称该浮点数为 N 进制浮点规格化数. 例如, 二进制浮点数 $+2^{+101} \cdot 0.101001$ 是浮点规格化的数, 而 $+2^{+110} \cdot 0.0101001$ 是二进制非规格化的浮点数.

四 计算机单元中数的表示形式

电子数字计算机要求把参与运算的数表示为二进制浮点

形式

$$x = \pm 2^{\pm p} \cdot q,$$

其中, 阶数 p 是二进制正整数, 尾数 q 是二进制正小数. 在 709 机和 TQ-16 机上限定

$$-1000000 \leq \pm p < 1000000,$$

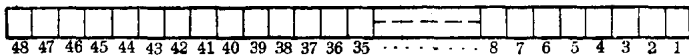
$$0 \leq q < 1.$$

且若 $q \geq \frac{1}{2}$, 则称该数为二进制浮点规格化数.

电子数字计算机为什么要把数表示为二进制浮点形式呢？这是因为：

第一,二进制数在计算机单元里容易表示,

如前所述, 计算机存放数据的单元是由一串互相联系着的小元件构成的, 每一个单元有 48 个小元件, 即



单元的每一个小元件都具有两种不同的稳定状态(如晶体管的饱和或截止)。如果规定一种状态代表1, 另一种状态代表0, 那么, 每一个小元件就可以表示一个二进位数字, 因而, 在确定正负号位和小数点位置后, 一个单元中小元件任何一种稳定状态的组合, 就代表着一个有限位长度的二进制数。譬如, 在一个具有48个小元件的单元中, 若把第48位定为符号位(用0代表正, 1代表负), 小数点位置固定在第47、48位之间, 这样, 这个单元就能把任何一个具有47个二进位数字的小于1的定点数表示出来。

第二,浮点形式可以扩大数的表示范围。如果单元中表示的是定点数,则数的表示范围就会受到限制,例如上面的例子就只能把二进制的小数表示出来。为了扩大数在单元中的

表示范围, 可以采用小数点不固定的浮点形式.

在 709 机和 TQ-16 计算机单元中, 二进制数规定表示为如下的浮点形式:

$$x_{(2)} = \pm 2^{\pm p_1 p_2 p_3 p_4 p_5 p_6} \cdot 0. q_1 q_2 \cdots q_{39} q_{40},$$

其中, 阶数 $p_1 p_2 p_3 p_4 p_5 p_6$ 是二进制正整数, 尾数 $0. q_1 q_2 \cdots q_{39} q_{40}$ 是二进制正小数, 且

$$-1000000 \leq \pm p_1 p_2 p_3 p_4 p_5 p_6 < 1000000,$$

$$0 \leq 0. q_1 q_2 \cdots q_{39} q_{40} < 1.$$

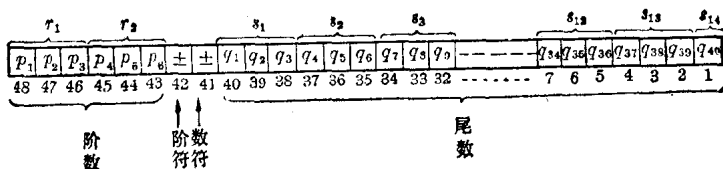
这个数如果用八进制浮点形式表示, 则为

$$x_{(8)} = \pm 2^{\pm r_1 r_2} \cdot 0. s_1 s_2 \cdots s_{13} s_{14},$$

其中, 阶数 $r_1 r_2$ 是八进制正整数, 尾数 $0. s_1 s_2 \cdots s_{13} s_{14}$ 是八进制正小数, 且

$$-100 \leq \pm r_1 r_2 < 100, 0 \leq 0. s_1 s_2 \cdots s_{13} s_{14} < 1, s_{14} \leq 1.$$

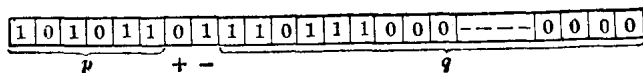
二进制浮点数在计算机单元中的形式是



其中, 阶符和数符规定用 0 表示正, 用 1 表示负, 另外还规定, 当阶符为负时, 阶数要用 $1000000 - |p_1 p_2 p_3 p_4 p_5 p_6|$ 来代替.

现在举几个例子.

[例] 二进制数 $x_{(2)} = -2^{+101011} \cdot 0.11011100 \cdots 0$ 在计算机单元中表示为



写成八进制形式, 即

$$x_{(A)} = -2^{+53} \cdot 0.67000000 \dots 0.$$

[例] 二进制数 $x_{(2)} = +2^{-101021} \cdot 0.11011100 \dots 0$ 在计算机单元中表示为



写成八进制形式, 即

$$x_{(A)} = +2^{-53} \cdot 0.67000 \dots 0.$$

由于计算机单元的字长是有限的, 因而, 它能表示的二进制浮点规格化数的数值范围是:

$$2^{-63} \cdot \frac{1}{2} \leq |x| \leq 2^{+63} \cdot (1 - 2^{-40}).$$

这一范围在十进制计数制中, 大约等于 $10^{\pm 19}$.

对于绝对值小于 $2^{-63} \cdot 1/2$ 的数, 计算机将表示为机器零: $2^{-64} \cdot 0$. 若在运算过程中, 出现大于机器所能表示的最大的数, 则计算机将产生溢出停机.

最后, 我们用一个例子写出把一个十进制数化成二进制浮点规格化形式并置入单元的步骤.

[例] 把 $-53.14_{(十)}$ 化成二进制浮点规格化形式后置入单元.

1. 把 $-53.14_{(十)}$ 化成八进制形式.

先用“除八取余”法则把 $53_{(十)}$ 化成八进制整数 $65_{(八)}$:

$$\begin{array}{r} 8 \overline{) 53} \\ \underline{48} \\ 5 \\ \underline{40} \\ 13 \end{array} \quad \begin{array}{|c|c|} \hline 6 & 5 \\ \hline \end{array} \quad \uparrow$$

再用“乘八取整”法则把 $0.14_{(十)}$ 化成八进制小数 $0.107 \dots_{(八)}$: