计算机组成原理

第三章 中央处理器

2025-8-22

信息与软件工程学院
School of Information and Software Engineering

主要内容

3.3 运算方法

- 01. 定点加减运算
- 02. 定点乘法运算
- 03. 定点除法运算

1、补码加减运算的基本关系式

补码加减：操作数用补码表示，连同符号位一起运算，

结果也用补码表示。

补码加减所依据的基本关系是：

$$(X + Y)_{补} = X_{补} + Y_{补} \qquad (1)$$

$$(X - Y)_{补} = X_{补} + (-Y)_{补} \qquad (2)$$

式（1）表明：当操作码为“加”时，可直接将补码表示的两个操作数（$X_{补}$、$Y_{补}$）相加，不必考虑它们的数符是正或负，所得结果即为补码表示的和。

一、定点加减运算

$$(X+Y)_{补} = X_{补} + Y_{补} \qquad (1)$$

$$(X-Y)_{补} = X_{补} + (-Y)_{补} \qquad (2)$$

式（2）表明：当操作码为“减”时，可转换为与减数的负数相加，从而化“减”为“加”。$(-Y)_{补}$是$Y_{补}$的机器负数。由于$Y_{补}$本身可正可负，$(-Y)_{补}$也可能为负或为正。由$Y_{补}$求$(-Y)_{补}$，称为对$Y_{补}$求补或变补，即将$Y_{补}$连同符号位一起变反，并在末位加1（不论$Y_{补}$本身为正或负）。如果减数Y为正，则变补后可转换为加一个负数；如果减数Y为负，则变补后可转换为加一个正数。

一、定点加减运算

2、补码加减运算规则

① 参与运算的操作数用补码表示，符号位作为数的一部分直接参与运算，所得运算结果即为补码表示形式；

② 若操作码为加，则两数直接相加；

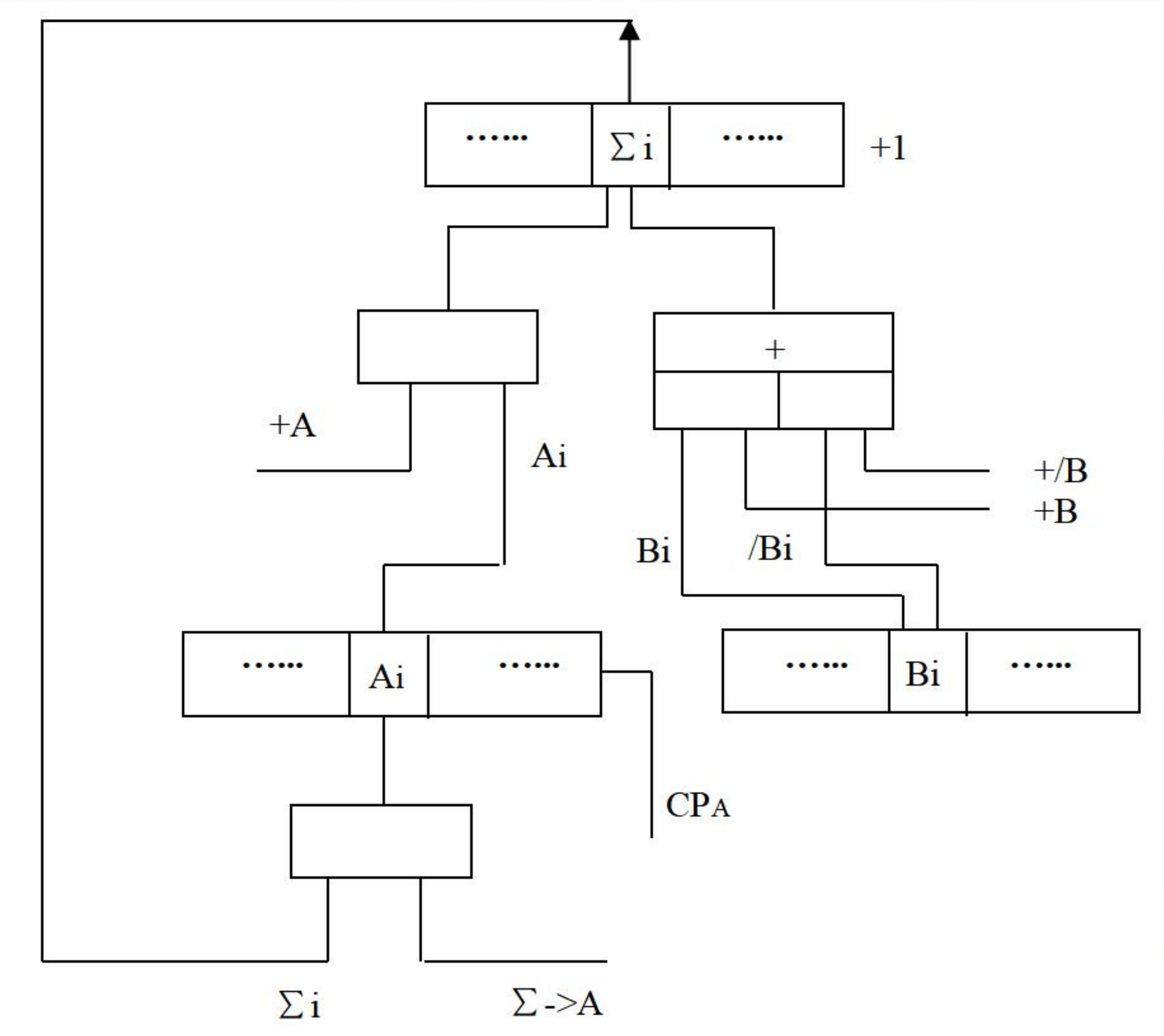
③ 若操作码为减，则将减数变补后再与被减数相加。

3、补码加减运算的逻辑实现

补码加减法所需控制命令

功能	所需控制命令		
加	+A +B	A+B	$\Sigma \to A$ CPA
减	+A $+\overline{B}$	$A+\overline{B}+1$	$\Sigma \to A$ CPA

一、定点加减运算



补码加减法运算器粗框

二、定点乘法运算

手算乘法如何实现?

例：0.1101×0.1011 = ?

```
      0.1101
  ×   0.1011
  ----------
        1101
       1101
      0000
     1101
  ----------
  0.10001111
```

由手算到机器实现，要解决三个问题：

①符号位如何处理?

②多项部分积相加，如何解决进位传递问题?

③乘数权值每高一位，新部分积需左移一位，才能保持两次部分积之间的位权对应关系，这导致加法器位数增加。为了保持加法器位数不变，能否改变移位方法?

二、定点乘法运算

对于符号位的处理方法：

可采用原码乘法或补码乘法。

对于后两种问题的处理方法：

一种乘法器是将n位乘法转换为n次累加与移位循环，因而可用常规加法器实现。

另一类乘法器结构，称为阵列乘法器。

本节主要讨论如何通过累加、移位实现分步乘法运算，即常规加法器。

二、定点乘法运算

1、原码一位乘法

（1）定义：取两个操作数的绝对值相乘，每步处理一位乘法，符号位单独处理。

（2）运算规则

①寄存器分配与初始值：A,B,C三个寄存器

A存放部分积累加和，初始值为0(双符号位00表示)；

B存放被乘数X（绝对值），此时，符号位为双符号位00（在乘的过程中，B中的值一直保持不变）；

二、定点乘法运算

C存放乘数Y（绝对值），将符号位去掉；C寄存器的初始值是乘数Y的尾数（有效位数），以后每乘一次，将已处理的低位乘数右移舍去，同时将A寄存器的末位移入C寄存器的高位。

②符号位：A,B均设置双符号位

③基本操作：

在原码一位乘中，每步只处理一位乘数，即位于C寄存器末位的乘数，也称之为判断位Cn；

二、定点乘法运算

若Cn=1，则部分积为B，执行A+B操作，然后将累加和右移一位，用“→”表示。（Cn位去掉）；执行部分积累加和+B操作，然后将新部分积累加和右移一位；

若Cn=0，则部分积为0，执行A+0操作，然后右移，或直接让A右移一位。（Cn位去掉）

右移时，A的末位移入C的高位，A的第二符号位移入尾数最高位，第一符号位移入第二符号位，而第一符号位本身则补0。

④操作步骤：

n次累加与n次移位（最后一次累加后要移位）

⑤处理符号位

（3）举例：

X = 0.1101，Y = -0.1011，求 XY =?

设寄存器 A = 00.0000，

B =|X|= 00.1101，

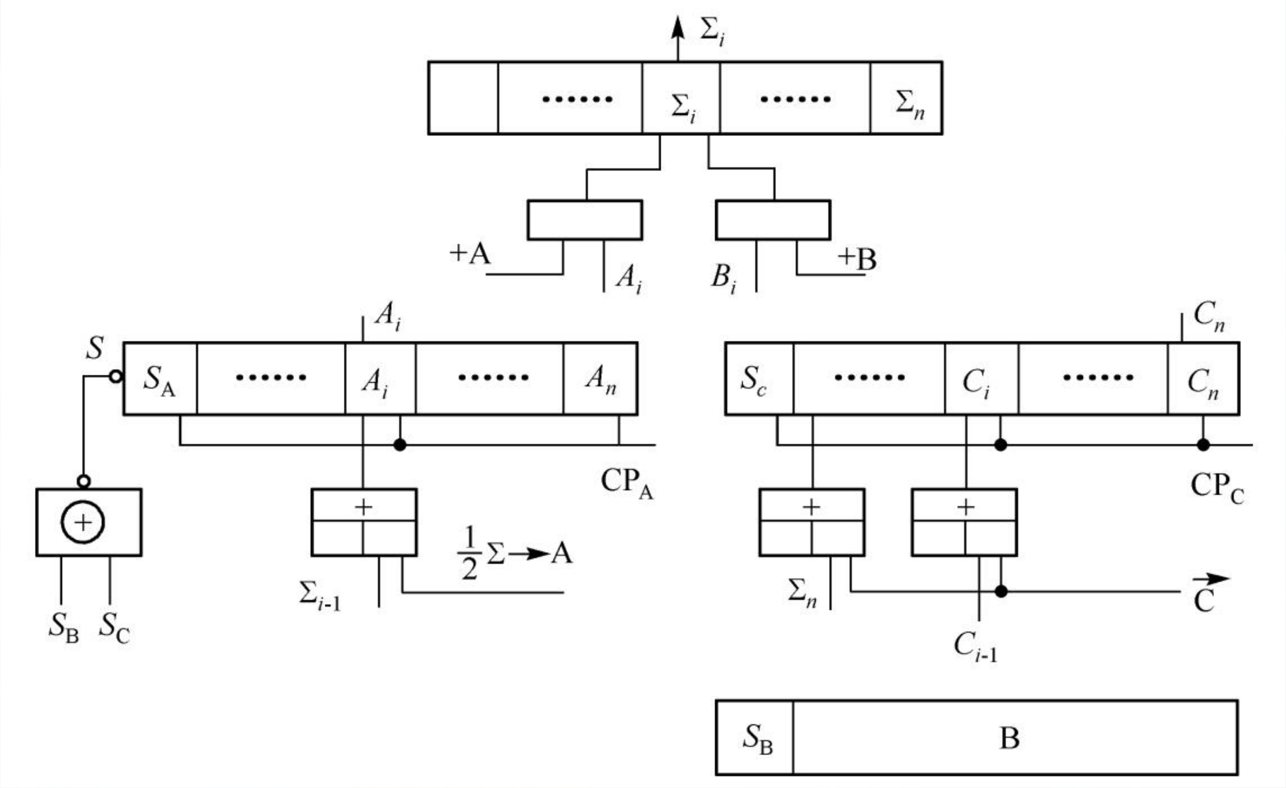
C =|Y|= .1011。

步数	条件	操作	A	C C_n(判断位)
			00.0000	.1011
第一步	C_n=1	+B	+ 00.1101	
			00.1101	A的末位移入C的高位
		整体右移一位，首位补0 →	00.0110	1.101
第二步	C_n=1	+B	+ 00.1101	A的末位移入C的高位
			01.0011	
		整体右移一位，首位补0 →	00.1001	11.10
第三步	C_n=0	+0	+ 00.0000	
			00.1001	A的末位移入C的高位
		整体右移一位，首位补0 →	00.0100	111.1
第四步	C_n=1	+B	+ 00.1101	
			01.0001	A的末位移入C的高位
		整体右移一位，首位补0 →	00.1000	1111

加符号位，则乘积为1.10001111

二、定点乘法运算

(4) 硬件逻辑框图



(5) 微命令设置

C_n = 1，即 A+B：+A、+B； A+B； $\Sigma/2 \rightarrow A$、$\vec{C}$，CP_A、CP_C

C_n = 0，即 A+0：+A；A； $\Sigma/2 \rightarrow A$、$\vec{C}$、CP_A、CP_C

二、定点乘法运算

2、补码一位乘法

（1）定义：操作数与结果均以补码表示，连同符号位一起，按相应算法运算。

（2）运算方法及关系式：比较法

$$[XY]_{补} = [A_n]_{补} + (Y_{n+1} - Y_n)[X]_{补}$$

注意：Y_{n+1}为低位，Y_n为高位

（3）运算规则

① 寄存器分配、初始值及符号位：

A, B, C三个寄存器

二、定点乘法运算

A存放部分积累加和，初始值为0(双符号位00表示)；

B存放被乘数$X_{补}$，（双符号位00、或11表示）；

C存放乘数$Y_{补}$，单符号位（符号位参与运算），Y的末位添0，称为附加位Y_{n+1}。

② 基本操作：

用C寄存器最末两位（含增加的C_{n+1}）作判断位，即(Y_n, Y_{n+1})为判断位。

若Y_nY_{n+1}为00或11，执行A+0，右移，实际上可直接让A右移一位；

二、定点乘法运算

若Y_nY_{n+1}为01，执行$A+X_补$，右移；

若Y_nY_{n+1}为10，执行$A+[-X_补]$，右移。

在右移时，A寄存器中的第二符号位值移入尾数的最高数位（有效位的最高位），第一符号位值移入第二符号位，第一符号位本身不变，而A寄存器末位移入C寄存器。

③ 操作步数：为有效位位数的n+1

（4）举例：X = -0.1101，Y = -0.1011，求$[XY]_补$ = ？

设 A = 00.0000，B = $X_补$ = 11.0011，

-B = $-X_补$= 00.1101，C = $Y_补$ = 1.0101。

二、定点乘法运算

步数	条件C_nC_{n+1}	操作	A	C C_{n+1}
			00.0000	1.0101 0
第一步	10	+B	+ 00.1101 00.1101	
		→	00.0110	11.010 1
第二步	01	+B	+ 11.0011 11.1001	
		→	11.1100	111.01 0
第三步	10	−B	+ 00.1101 00.1001	
		→	00.0100	1111.0 1
第四步	01	+B	+ 11.0011 11.0111	
$[XY]_{补}$=0.1000111		→	11.1011	11111. 0
第五步	10	−B	+ 00.1101 00.1000	1111

二、定点乘法运算

（5）微命令设置

Y_nY_{n+1}为00或11：

+A，A，$\Sigma \rightarrow$A，$\Sigma/2 \rightarrow$A、$\vec{C}$，CP_A、CP_C

Y_nY_{n+1}为01：

+A，+B，A+B，$\Sigma \rightarrow$A，$\Sigma/2 \rightarrow$A、$\vec{C}$，CP_A、CP_C

Y_nY_{n+1}为10：

+A、$+\overline{B}$，A+B，+1，$\Sigma \rightarrow$A，$\Sigma/2 \rightarrow$A、$\vec{C}$，CP_A 、CP_C

三、定点除法运算

手工变为机器实现时，需要解决三个问题：

①如何判断够减

方法：
- 先判后减（硬件方式）
- 先减后判（软件方式）
  - 恢复余数法
  - 不恢复余数法（加减交替除法）

②如何处理符号位

③如何提高除法运算速度

除法器分类：

方法：
- 常规除法器
- 迭代除法器
- 阵列除法器

三、定点除法运算

1、原码不恢复余数法

（1）定义：取两个操作数的绝对值相除，符号位单独处理。

（2）运算关系式：根据余数r_i符号判断是否够减：

r_i为正表示够减，上商$Q_i = 1$；

r_i为负表示不够减，上商$Q_i = 0$；

通式： $$r_{i+1} = 2r_i + (1 - 2Q_i)Y$$

若第i步够减，Q_i=1，则第i+1步应做$2r_i$−Y；

若第i步不够减，Q_i=0，则第i+1步应做$2r_i$+Y。

(3)运算规则

①寄存器分配与符号位:

A,B,C三个寄存器;

A初始值存放被除数(绝对值),以后存放各次余数,

A取双符号位,从第一符号位判断是否够减,从而决定商值;

B寄存器存放除数的绝对值,取双符号位;

C存放商,取单符号位;商由末位置入,在每次置入新商时,原商同时左移一位。

三、定点除法运算

②基本操作与上商：

a. 第一步操作必为$2r_0-Y$；

b. 以后各步根据如下条件进行：

r_i为正表示够减，即$Q_i=1$，则第i+1步应为$2r_i-Y$，

r_i为负表示不够减，即$Q_i=0$，则第i+1步应为$2r_i+Y$；

c. 最后一步：若第n步（最后一步）余数为负，则需增加一步恢复余数，这增加的一步不移位，操作为r_n+Y。

三、定点除法运算

③操作步数：

要求得n位商（不含符号位），则需做n步（次）“左移---加减”循环。

④符号：同号相除为正，异号反之。

（4）举例： $X \div Y = -0.10110 \div 0.11111 = ?$

设 $A = |X| = 00.10110$，$B = |Y| = 00.11111$，

则 $-B = 11.00001$，$C = |Q| = 0.00000$。

三、定点除法运算

步数	条件 C_nC_{n+1}	操作	A		C	C_n	
			00.10110	r_0	0.00000		整体左移一位，末位补0
第一步		←	01.01100	$2r_0$			
		−B	+ 11.00001				舍去；整体左移一位，末位商1
	$S_A=0$		100.01101	r_1	0.00001	Q_1	r_1为正；整体左移一位，末位补0
第二步	$C_n=1$	←	00.11010	$2r_1$			
		−B	+ 11.00001				整体左移一位，末位商0
	$S_A=1$		11.11011	r_2	0.00010	Q_2	r_2为负；整体左移一位，末位补0
第三步	$C_n=0$	←	11.10110	$2r_2$			
		+B	+ 00.11111				舍去；整体左移一位，末位商1
	$S_A=0$		100.10101	r_3	0.00101	Q_3	r_3为正；整体左移一位，末位补0
第四步	$C_n=1$	←	01.01010	$2r_3$			
		−B	+ 11.00001				舍去；整体左移一位，末位商1
	$S_A=0$		100.01011	r_4	0.01011	Q_4	r_3为正
第五步	$C_n=1$	←	00.10110	$2r_4$			
		−B	+ 11.00001				整体左移一位，末位商0
	$S_A=1$		11.10111	r_5'	0.10110	Q_5	r'_5为负
第六步	$C_n=0$	+B	+ 00.11111				舍去
		恢复余数	100.10110	r_5			

商=-0.101110

余数=0.10110×2^{-5}

(5) 微命令设置

$r_i=0$，即$Q_i=1$，则$2r_i-Y$：$+\overleftarrow{A}$、$+\overline{B}$; $\underline{2A\rightarrow\Sigma、+\overline{B}、+1}$;

$\underline{\Sigma\rightarrow A、\overleftarrow{C}、Q_i\rightarrow C_n、CP_A、CP_C}$

$r_i=1$，即$Q_i=0$，则$2r_i+Y$：$+\overleftarrow{A}$、$+B$; $\underline{2A\rightarrow\Sigma、+B}$;

$\underline{\Sigma\rightarrow A、\overleftarrow{C}、Q_i\rightarrow C_n、CP_A、CP_C}$

最后一步中，若余数为负，则需要恢复余数操作。

三、定点除法运算

2、补码不恢复余数除法

（1）定义：指被除数、除数，所求得的商，余数等都用补码表示，连同符号位一起运算。

（2）运算规则

① 寄存器分配与符号位：

A,B,C三个寄存器；

A初始值存放被除数（补码表示），以后存放各次余数，A取双符号位；

B寄存器存放除数（补码表示），取双符号位；

三、定点除法运算

C存放商，取单符号位，初始值为0。

② 假商符：在第一步操作之前，先根据r_0（即X）、Y符号比较确定假商符（与真商符相反），即：

r_0 、Y同号为1

r_0 、Y异号为0

③ 基本操作

a. 第一步操作，假商符为1(r_0 、Y同号)，$2X_{补}-Y_{补}$

假商符为0(r_0 、Y异号)，$2X_{补}+Y_{补}$

b. 其余操作根据如下规则进行：

三、定点除法运算

若$X_{i补}$、$Y_补$同号：

$r_{i补}$ $Y_补$同号（够减），上商1，下一步，$2r_{i补}-Y_补$

$r_{i补}$ $Y_补$异号（不够减），上商0，下一步，$2r_{i补}+Y_补$

若$X_{i补}$ ，$Y_补$异号：

$r_{i补}$ $Y_补$同号（不够减），上商1，下一步，$2r_{i补}-Y_补$

$r_{i补}$ $Y_补$异号（够减），上商0，下一步，$2r_{i补}+Y_补$

c. 最后一步要对假商校正。

（三）举例：$X \div Y = 0.1000 \div (-0.1010) = ?$

设$A = X_补 = 00.1000$，$B = Y_补 = 11.0110$，

$B = 00.1010$，$C = Q_补 = 0.0000$。

三、定点除法运算

步数	条件	操作	A		C　C_{n-1}	
	r_0、Y异号	求商符	00.1000	r_0	0.000	Q_0
第一步	C_{n-1}=0	←	01.0000	$2r_0$		
		+B	+ 11.0110			
	r_1、Y异号		00.0110	r_1	0.000	Q_1
第二步	C_{n-1}=0	←	00.1100	$2r_1$		
		+B	+ 11.0110			
	r_2、Y异号		00.0010	r_2	0.000	Q_2
第三步	C_{n-1}=0	←	00.0100	$2r_2$		
		+B	+ 11.0110			
	r_3、Y异号		11.1010	r_3	0.001	Q_3
第四步	C_{n-1}=0	←	11.0100	$2r_3$		
		−B	+ 00.1010			
			11.1110	r_4		

假商= 0.001，真商= 0.001 + 1.0001 = $1.0011_{补}$ = $0.1101_{真值}$；

余数=$2^{-4}r_4$ = $1.11111110_{补}$ = $-2^{-4}\times 0.0010_{真值}$。

(4) 微命令设置

$r_{i补}$、$Y_{补}$同号，即$2r_{i补}-Y_{补}$：

+2A、+A，+B、$+\overline{B}$、+1，$\Sigma\rightarrow$A，C，$Q_i\rightarrow C_n$，CP_A、CP_C

$r_{i补}$、$Y_{补}$异号，即$2r_{i补}+Y_{补}$：

+2A、+A，+B、$+\overline{B}$、+1，$\Sigma\rightarrow$A，C，$Q_i\rightarrow C_n$，CP_A、CP_C

谢谢观看

计算机组成原理

信息与软件工程学院
School of Information and Software Engineering