

电子数字计算机原理

第 二 册

北京大学电子仪器厂

目 录

第七章 运算器

第一节	加法器	1
第二节	运算器各部件之间的代码传送	19
第三节	浮点规舍加(减)法	27
第四节	浮点规舍乘法	32
第五节	浮点规舍除法	37
第六节	逻辑运算及其它指令	42
第七节	关于运算速度	48
习 题		53

第八章 控制器

第一节	指令和指令系统	58
第二节	地址的形成	71
第三节	变址和读写控制	81
第四节	控制方式	87
第五节	分配器和启停线路	100
第六节	典型指令的执行过程	103
第七节	操作时间表	106
习 题		121

第九章 并行控制和快速运算

第一节	概述	127
第二节	存储器多体并行交叉存取	131
第三节	先行指令栈	139
第四节	先行读数栈。页面控制方法	149
第五节	并行工作的运算器。多累加寄存器结构	158
第六节	并行多位乘法部件	167
第七节	快速除法算法	180
第八节	多道程序与中断系统	187
附 录		197

第七章 运算器

运算器相当于一个算盘，它在控制器控制下完成算术运算(如定点、浮点“+”、“-”、“×”、“÷”等)和逻辑运算(如逻辑乘“∧”、逻辑加“∨”、按位加“A₊”、大于比较“>”、小于比较“<”以及移位等)指令所给出的运算。

为了完成这些运算，运算器需要有寄存器来存放参与运算的数(如被加数、加数、被乘数、乘数等)和运算结果(如和、乘积、商等)。有的寄存器还要有移位功能。

在一般计算机中，“+”、“-”、“×”、“÷”等算术运算都是化为定点加法完成的。因此，在运算器中还必须有加法器。在一些快速机器中，为了加快乘、除法的速度，有时还专门设有执行乘、除法的专用部件，如乘法器。

为了传送信息，在运算器中有时还专门设有传送信息代码的代码线。

在第六章，我们已经介绍了算术运算的算法。在本章，则主要是讨论这些算术运算和逻辑运算在运算器中是怎样实现的。为此，我们首先介绍运算器的加法器，其次介绍运算器其它部件的功能及其相互间的关系，并给出运算器的框图，再次，通过几条典型指令，介绍指令在运算器中的执行过程，最后，粗略地讨论一下关于运算器运算速度的一些问题。

第一节 加法器

在一般计算机中，“+”、“-”、“×”、“÷”等算术运算都是化为定点加法在加法器中完成的。这样，加法器自然地成了影响运算速度的一个重要因素。因此，加法器的设计对于运算器来说是十分重要的。

在第六章，我们已经详细地讨论了定点并行加法(模4加法)。比如，设

$$[x]_n = 00.1010101, \quad [y]_n = 00.0011101,$$

那么，当我们手算的时候，是用下面的算式来完成的：

$$\begin{array}{r}
 \begin{array}{r}
 \xrightarrow{a_4} \\
 [x]_n = 00.1010101 \\
 \xrightarrow{y_4} \\
 [y]_n = 00.0011101 \\
 000.011101 \leftarrow \text{进位信号} \\
 \phantom{\text{进位信号}} \\
 \phantom{\text{进位信号}} \\
 \phantom{\text{进位信号}} \\
 \hline
 [x+y]_n = 00.1110010 \\
 \phantom{\text{进位信号}}
 \end{array}
 \end{array}$$

现在来分析算式中的任何一位，如小数点后的第4位，那么它的和数 H_4 是由 a_4 、 y_4 及低位传来的进位 J_3 所决定的。并且它产生进位 J_4 影响着 H_5 的形成。因此，加法器的每一

位都是一个有三个输入端(x_i, y_i, J_{i+1})和两个输出端(H_i, J_i)的逻辑网络。这个逻辑网络就是第三章所介绍的全加器(参阅图 3.23)。将 $n+2$ 个全加器依次连接起来就构成 $n+2$ 位并行加法器(图 7.1)

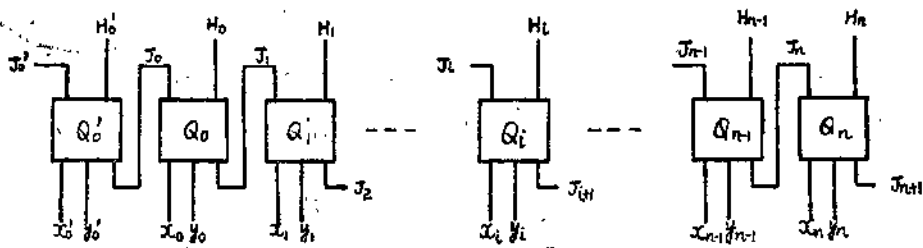


图7.1 加法器

在第三章已经给出了全加器的逻辑表达式,即:

$$\begin{cases} H_i = \bar{x}_i \bar{y}_i J_{i+1} + \bar{x}_i y_i \bar{J}_{i+1} + x_i \bar{y}_i \bar{J}_{i+1} + x_i y_i J_{i+1} \\ J_i = \bar{x}_i y_i J_{i+1} + x_i \bar{y}_i J_{i+1} + x_i y_i \bar{J}_{i+1} + x_i y_i J_{i+1} \end{cases}$$

从上述公式可知,“和数” H_i 的形成除 x_i, y_i 以外还依赖于由低位传来的进位 J_{i+1} ,而进位 J_{i+1} 又依赖于更低位传来的进位 J_{i+2} ,从而对进位而言形成了一条串联的链,称之为进位链。串行进位链的存在严重地影响着加法器的工作速度,这是任何采用进位制编码的并行运算器都不可避免的问题。为了提高运算器的工作速度,下面我们有必要详细地分析一下进位链的性质。

一、进位链

1. 本地进位与传送进位

“分析的方法就是辩证的方法。所谓分析,就是分析事物的矛盾。”我们分析进位链的性质,也就是分析进位链内部所存在的矛盾。

将进位公式稍加变换,可以改写成下面形式:

$$J_i = x_i y_i + (x_i \oplus y_i) J_{i+1}$$

式中“ $x_i \oplus y_i$ ”表示 x_i, y_i 之半和,即 $x_i \oplus y_i = \bar{x}_i y_i + x_i \bar{y}_i$ 。这个公式告诉我们:

- (1) 进位可以分解为“ $x_i y_i$ ”和“ $(x_i \oplus y_i) J_{i+1}$ ”这两部分。
- (2) “ $x_i y_i$ ”这部分与低位传来的进位 J_{i+1} 是无关的,它仅仅取决于本位参加运算的两个数码 x_i, y_i ,因此,我们称“ $x_i y_i$ ”为第 i 位产生的“本地进位”,记作 d_i 。
- (3) “ $(x_i \oplus y_i) \cdot J_{i+1}$ ”这部分则依赖于 J_{i+1} 。就是说,在第 i 位,如果 $x_i \oplus y_i = 1$,那么第 $i+1$ 位产生的进位 J_{i+1} 将被传送到更高位去,并且每一位的“ $x_i \oplus y_i$ ”都能起到这样的连通作用。所以,在第 i 位,如果 $x_i \oplus y_i = 1$,那么我们就称进位链在第 i 位被接通,或说进位能跳过第 i 位;如果 $x_i \oplus y_i = 0$,则第 i 位以后所产生的进位 J_{i+1}, J_{i+2}, \dots 都不能影响第 i 位以前的进位 J_{i-1}, J_{i-2}, \dots ,我们称进位链在第 i 位被打断。因此,我们称“ $(x_i \oplus y_i) J_{i+1}$ ”为第 i 位产生的传送进位,而“ $x_i \oplus y_i$ ”称为跳过条件(或传送条件),记作“ t_i ”。这样,传送进位可以记作“ $t_i \cdot J_{i+1}$ ”。

所以,用 d_i, t_i 代入进位公式,则可以改写成

$$J_i = d_i + t_i \cdot J_{i+1}$$

进位可以分解为本地进位和传送进位,是进位的基本性质,是我们所以能加快进位传送速度的客观依据。

为了简化设计,一般用 $t_i = x_i + y_i$ 来代替 $x_i \oplus y_i$, 因为进位公式不简化为

$$J_i = x_i y_i + (x_i + y_i) J_{i+1}$$

2. 串行进位链

经过上述分析,我们可以把加法器的进位链部分画成图 7.2

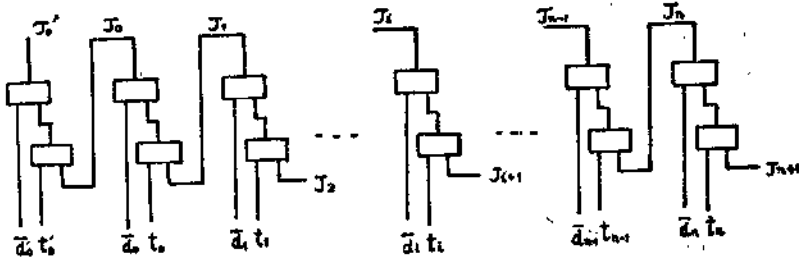


图 7.2

从图 7.2 可见,每一位所产生的进位,是串联地联接在一起的。假设经过一级“与非”门需要延迟时间 \bar{t}_y , 那么由 J_{i+1} 到 J_i 需要 $2\bar{t}_y$ 时间,对于 $n+2$ 级加法器来说,需要 $2(n+1)\bar{t}_y$ 才能形成 J_0 到 J_n 中所有的进位(因为一般说 J_0 是没有用的)。假设 $n=39$ 则 $t_{\Sigma} = 80\bar{t}_y$ (图 7.3)。

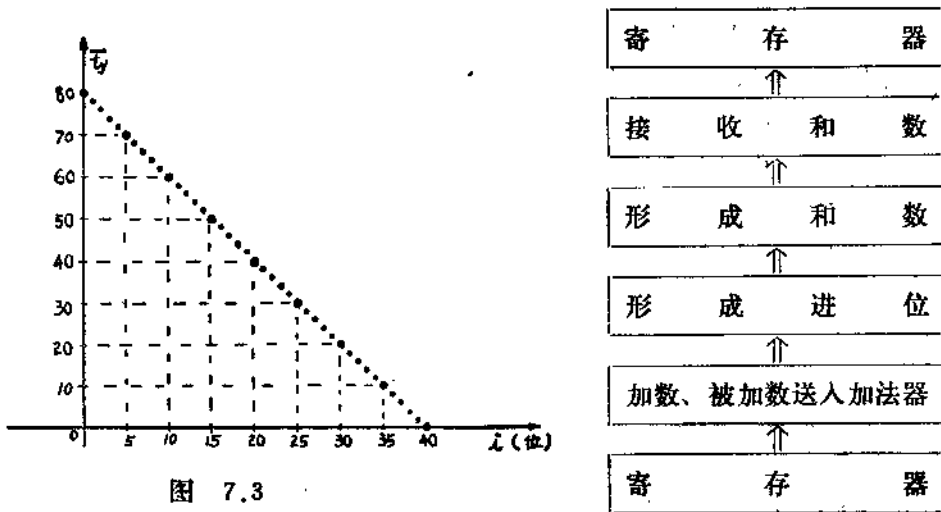


图 7.3

把进位时间和用于其它操作的时间相比较,就会看出这个时间太长了。一次定点加法一般说需要经过下述步骤:

- (1) 加数、被加数送入加法器: 从寄存器将加数、被加数送入加法器,直到在加法器中形成 d_i, t_i ,这一部分时间一般需要 $5\bar{t}_y$ 左右;
- (2) 形成进位: 从 d_i, t_i 到产生进位 J_i , 如上所说需要 $2(n+1)\bar{t}_y$, 若 $n=39$, 则需要 $80\bar{t}_y$;
- (3) 形成和数: 在加法器中形成和数,一般需要 $2\sim 3\bar{t}_y$;

(4) 接收和数：在加法器中产生和数后经控制线路送入寄存器直到寄存器稳定。一般说需 $10\bar{t}_y$ 左右。

因此，除进位时间外，其余时间一般说不超过 $20\bar{t}_y$ 。如果进位时间占 $80\bar{t}_y$ ，那么它将占定点加法时间 80% 以上，即：

$$\frac{\text{进位时间}}{\text{定点加法总时间}} > 80\%$$

所以，为了提高加法速度，首先必须减少进位时间。

加快进位时间的方法概括地可分为两种：一种是采用高速元件和高速线路，另一种是用逻辑方法。下面我们主要讨论一种常用的逻辑方法：分组跳跃进位（或称成组进位）法。

二、单重分组跳跃进位

1. 分组跳跃进位

为了寻求加快进位的逻辑方法，我们进一步分析各个进位之间的关系。我们选取进位链中的一段（如第 32~35 位），以分析进位之间的联系。

$$\begin{cases} J_{35} = d_{35} + t_{35}J_{36} \\ J_{34} = d_{34} + t_{34}J_{35} \\ J_{33} = d_{33} + t_{33}J_{34} \\ J_{32} = d_{32} + t_{32}J_{33} \end{cases}$$

由这几个公式可知，进位公式之间存在着递推关系。任何一位所产生的进位 J_i ，通过 J_{i-1}, J_{i-2}, \dots 而影响它前面进位的形成。对于这种递推公式，在第三章(94 页)曾介绍可以用展开法来压缩它的级数。上述公式可以展开成下列一组公式：

$$\begin{cases} J_{35} = d_{35} + t_{35}J_{36} \\ J_{34} = d_{34} + t_{34}d_{35} + t_{34}t_{35}J_{36} \\ J_{33} = d_{33} + t_{33}d_{34} + t_{33}t_{34}d_{35} + t_{33}t_{34}t_{35}J_{36} \\ J_{32} = d_{32} + t_{32}d_{33} + t_{32}t_{33}d_{34} + t_{32}t_{33}t_{34}d_{35} + t_{32}t_{33}t_{34}t_{35}J_{36} \end{cases}$$

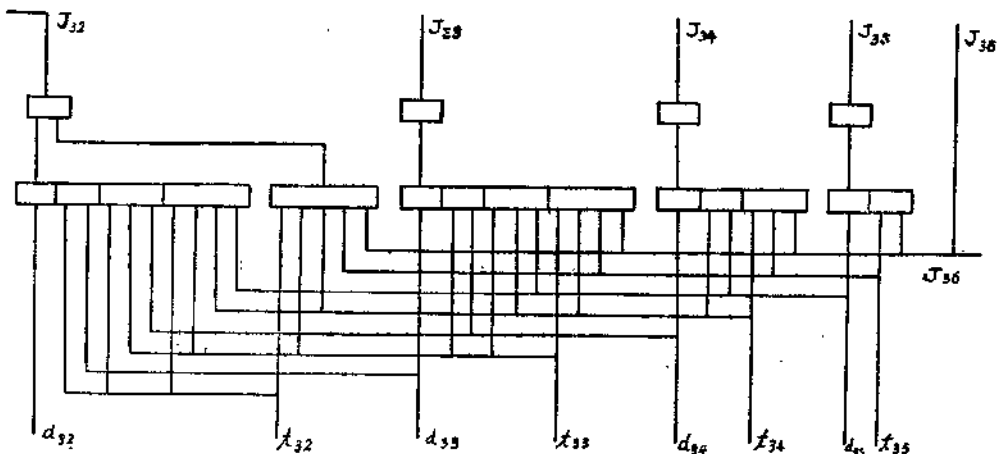


图 7.4

由这一组公式, 我们更清楚地看到下列关系:

- (1) 如果把这 4 位看做是一个小组, 那么, 进入这小组的进位只有 J_{36} , 而由这小组送到高位的进位只需送出 J_{32} ;
- (2) J_{36} 通过跳过条件 t_{35}, t_{34}, t_{33} , 而影响它前面几位进位的形成;
- (3) 在小组内部, 每一位所产生的本地进位 d_i , 通过 t_{i-1}, t_{i-2}, \dots , 而影响它前面几位进位的形成。

按照这一组展开式, 便可画出这个小组的进位线路(图 7.4)。由图 7.4 可知, 在 J_{36} 和 d_i, t_i 输入后, 只需 $2.4t_y$, 就可以形成本小组的所有进位($J_{35}, J_{34}, J_{33}, J_{32}$)。

根据这个原理, 可以将 40 位的加法器分成 10 组。这样, 从 d_i, t_i, J_{40} 都已产生时算起 ($t=0$), 经过 $2.4t_y$, 可以产生 $J_{36}, J_{37}, J_{38}, J_{39}$; 经 $4.4t_y$ 产生 J_{32} , 经 $4.8t_y$ 产生 J_{33}, J_{34}, J_{35} ……依此类推。经 $18.4t_y$, 就可产生 J_4 , 经 $18.8t_y$ 就可产生 J_5, J_6, J_7 ; 经 $20.4t_y$, 就可以产生 J_0 , 经 $20.8t_y$, 就可以产生第一组的每位进位: J_1, J_2, J_3 。也即只需 $20.8t_y$, 就产生了所有的进位(图 7.5)。

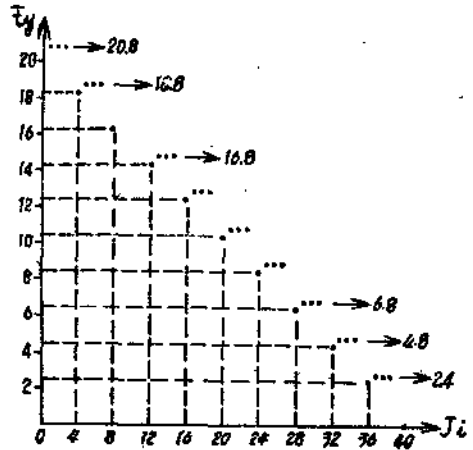


图 7.5

上述方法我们称为分组跳跃进位的方法。对于小组内来说, 每位的进位是同时(并行)产生的, 而小组之间则仍然是串行联接的。和后面所介绍的多重分组跳跃进位相区别, 我们称之为单重分组跳跃进位。

2. 关于分组方法的讨论

上面介绍的分组方法是各小组包含的位相同, 每组都是 4 位。但是, 分组的方法可以是多样的:

(1) 并不一定每组只包括 4 位, 也可以是 3 位一组、5 位一组等。至于怎样分组才合适, 这取决于机器的字长、元件的性能、对进位链速度的要求、允许使用于进位链部分的设备量等多种因素。一般说, 每小组包括的位数多些, 使用的元件就要多些, 进位时间就要短些; 反之, 每小组包括的位数少些, 使用的元件就会少些, 但进位时间就要拖长。例如, 如果每小组包含 5 位(如上例中增加第 31 位), 则公式中就要增加

$$J_{31} = d_{31} + t_{31}d_{32} + t_{31}t_{32}d_{33} + t_{31}t_{32}t_{33}d_{34} + t_{31}t_{32}t_{33}t_{34}d_{35} + t_{31}t_{32}t_{33}t_{34}t_{35}J_{36}$$

由于多这一位, 这一小组就要增加 7 个“与非”门。并且, 由于出现“ $t_{31}t_{32}t_{33}t_{34}t_{35}J_{36}$ ”这个乘积项, 就要求有 6 个输入头的“与非”门, 否则, 就要事先将 $t_{31}, t_{32}, t_{33}, t_{34}, t_{35}$ 中的几个合并在一起, 当然这又需要增加设备。

(2) 由于组间采取串联的方法, 所以并非所有小组组内都需要采取并行的方法, 有些小组只需将进入上一组的那一位(如上例中的 J_{32}) 展开就可以了, 其它位仍可采用串联的方法, 这样就可以节约设备。仍以第 9 小组为例, 公式可展开为:

$$\begin{cases} J_{35} = d_{35} + t_{35}J_{36} \\ J_{34} = d_{34} + t_{34}J_{35} \\ J_{33} = d_{33} + t_{33}J_{34} \\ J_{32} = d_{32} + t_{32}d_{33} + t_{32}t_{33}d_{34} + t_{32}t_{33}t_{34}d_{35} + t_{32}t_{33}t_{34}t_{35}J_{36} \end{cases}$$

图 7.6 就是上式的逻辑线路。

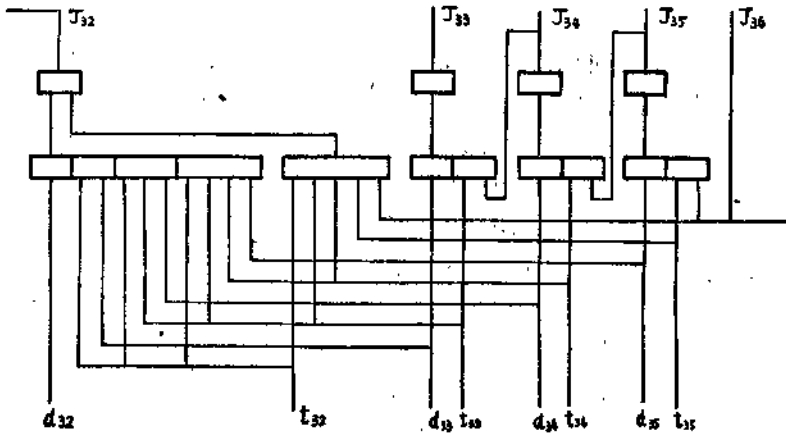


图 7.6

由进位与时间关系(图 7.7)可知, 如果除第 1、2、3 三个小组两小组用并行进位方法以外其它各组都采用图 7.6 的进位方法, 那么仍可在 $20.8\bar{t}$ 时间内产生所有的进位。不难看出, 如果全采用图 7.6 的进位方法, 只需再延长 $4.8\bar{t}$, 就能产生所有的进位。

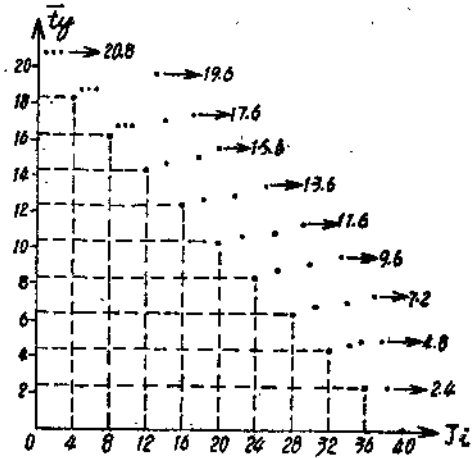


图 7.7

(3) 每小组所包含的位数也不一定是相等的, 也就是既可以按等长分组(每组位数相同), 又可以按不等长分组(每组位数不同)。这一点我们就不详细讨论了。

三、多重分组跳跃进位

1. 小组编成大组的方法

用多重分组跳跃进位的方法虽然已将进位时间压缩到原来的 $1/3-1/4$, 但对于速度较

快的机器来说,仍然是不够的,为了进一步加快进位的传送速度,对于已经分成了小组的进位链,我们再做进一步的分析。

首先来分析由第9小组产生的进位 J_{32} :

$$J_{32} = d_{32} + t_{32}d_{33} + t_{32}t_{33}d_{34} + t_{32}t_{33}t_{34}d_{35} + t_{32}t_{33}t_{34}t_{35}J_{36}$$

在组成 J_{32} 的5个乘积项中,只有最后一项“ $t_{32}t_{33}t_{34}t_{35}J_{36}$ ”才依赖于下一小组(即第10组)所产生的进位 J_{36} 。因此,我们仍可仿照分析每一位进位的方法,把每小组产生的进位也分成两部分:“ $d_{32} + t_{32}d_{33} + t_{32}t_{33}d_{34} + t_{32}t_{33}t_{34}d_{35}$ ”这部分不依赖于下一小组送来的进位 J_{36} ,称为第9小组的本地进位,记作 D_9 ;“ $t_{32}t_{33}t_{34}t_{35}J_{36}$ ”这部分依赖于下一小组送来的进位,称为第9小组的传送进位,而“ $t_{32}t_{33}t_{34}t_{35}$ ”是进位 J_{36} 跳过第9小组的条件,记作 T_9 。这样第9小组产生的传送进位可记作“ T_9J_{36} ”, J_{32} 的表达式可以写为

$$J_{32} = D_9 + T_9J_{36}$$

仿此,可以将各小组产生的小组进位都写成这样的形式:

$$J_{36} = D_{10} + T_{10}J_{40}$$

$$J_{32} = D_9 + T_9J_{36}$$

$$J_{28} = D_8 + T_8J_{32}$$

.....

$$J_4 = D_2 + T_2J_8$$

$$J_0 = D_1 + T_1J_4$$

这样就又出现一组递归表达式。按照组内加快进位的方法,再将其中几个小组编成一个大组(如将第6—9组编成一大组),并再把这些表达式展开:

$$\begin{cases} J_{32} = D_9 + T_9J_{36} \\ J_{28} = D_8 + T_8D_9 + T_8T_9J_{36} \\ J_{24} = D_7 + T_7D_8 + T_7T_8D_9 + T_7T_8T_9J_{36} \\ J_{20} = D_6 + T_6D_7 + T_6T_7D_8 + T_6T_7T_8D_9 + T_6T_7T_8T_9J_{36} \end{cases}$$

对比小组组内进位公式可知,这个大组的进位线路(图7.8)和小组组内进位线路(图7.4)是完全一样的

2. D_i, T_i 的产生

为了在大组内形成进位,需要每个小组事先产生小组本地进位 D_i 和小组跳过条件 T_i 。

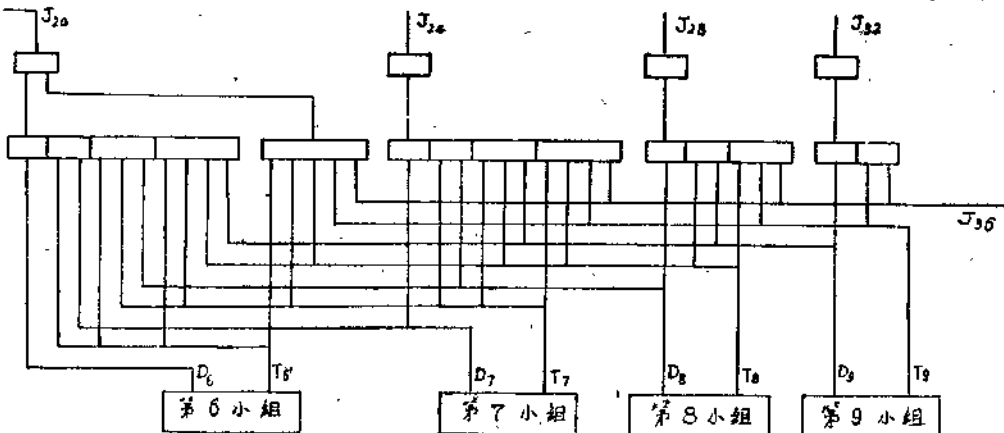


图 7.8

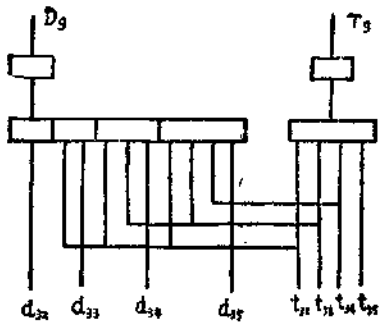


图 7.9

因此,图 7.4 的组内进位线路应做适当修改。

仍以第 9 小组为例,这时不再需要产生 J_{32} , 而要将产生 J_{32} 的线路改成产生小组本地进位 D_9 和跳过条件 T_9 。

由上面分析,我们知道

$$\begin{cases} D_9 = d_{32} + t_{32}d_{33} + t_{32}t_{33}d_{34} + t_{32}t_{33}t_{34}d_{35} \\ T_9 = t_{32}t_{33}t_{34}t_{35} \end{cases}$$

所以稍加修改就得到产生 D_9 和 T_9 的线路(图 7.9)。

3. 双重分组的进位链结构

上面我们已将第 6—9 小组编成一个大组,如果我们再将第 2—5 小组编成一个大组,那么我们就得到一个 41 位加法器的快速进位链。

图 7.10 是这个进位链的框图。

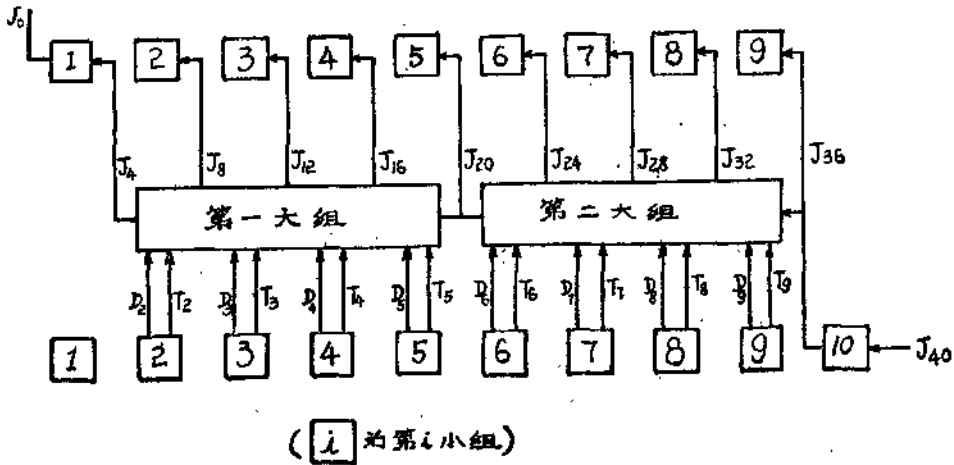


图 7.10

在图 7.10 中,第 2—9 小组都需产生 D_i 和 T_i , 因此要用能产生 D_i 和 T_i 的小组进位线路(即将图 7.4 按图 7.9 加以修改)。而第 1 和第 10 两小组需产生 J_0 和 J_{36} , 所以要用图 7.4 那样的小组进位线路。

这样的进位链,假设从 d_i, t_i, J_{40} 已经产生时算起($t=0$),那么,经过 $2.4\bar{t}_y$ 产生的有:

第 10 组的 $J_{36}, J_{37}, J_{39}, J_{39}$ 及所有的 D_i 和 T_i ;

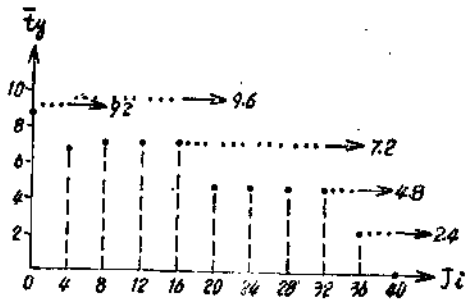


图 7.11

经过 $4.8\bar{t}_y$ 产生的有:

$J_{20}, J_{24}, J_{28}, J_{32}$ 及第 9 组的 J_{33}, J_{34}, J_{35} ;

经过 $6.8\bar{t}_y$ 产生 J_4 ;

经过 $7.2\bar{t}_y$ 产生的有:

J_8, J_{12}, J_{16} , 及第 5 组至第 8 组中 (J_{17} 至 J_{31}) 所有的进位;

经过 $8.8\bar{t}_y$ 产生 J_0 ;

经过 $9.2\bar{t}_y$ 产生的有 J_1, J_2, J_3 ;

经过 $9.6\bar{t}_y$ 产生的有:

第 2 组至第 4 组中除 J_8, J_{12} 以外)所有的进位(图 7.11)。

上述分组方法仅仅是一个例子。和划分小组类似,大组的划分方法也是多种多样的。此外,也不一定只是双重分组,还可以是三重的。每大组内包含的小组个数也不一定是相等的。至于怎样划分大组才合适,这也取决于机器的字长、元件的性能、对进位链速度的要求、允许使用的设备量等多种因素。

四、加法器接收数及产生和数的方法

对于加法器来说,除了进位链以外,还有两个重要部分,即:(1)加数和被加数送入加法器并产生 d_i 和 t_i ; (2)产生和数。这两者是相互联系的,在产生 d_i, t_i 时要充分考虑便于形成和数;在产生和数时要充分利用产生 d_i 和 t_i 的线路。

1. 加法器接收数的线路

一般说,数总是在寄存器中存放的。对于不同操作,参与运算的两个数往往是不相同的。如加法器的尾数部分,有时要做 $L_s + C_s$, 有时要做 $L_s - C_s$, 而乘法(两位一乘)时还要做 $L_s + 2C_s$ 等。机器的设计要求不同,送入加法器的加数和被加数也不相同。假设只考虑在加法器尾数部分中做 $L_s + C_s, L_s - C_s, L_s + 2C_s$ 三种运算,那么加法器的 x 端只接收 L_s , 而 y 端则有时要接收 C_s , 有时要接收 \bar{C}_s , 有时要接收 $2C_s$ 。为了控制在不同的情况下送入不同的数,在控制器要产生 $L_s \rightarrow Q_s$ (L 寄存器的尾数部分送入加法器的尾数部分), $C_s \rightarrow Q_s, \bar{C}_s \rightarrow Q_s, 2C_s \rightarrow Q_s$ 等控制电位。图 7.12 是全加器 Q_{20} 接收数的线路。

当进行 $L_s + C_s$ 时, $L_s \rightarrow Q_s, C_s \rightarrow Q_s$ 两控制电位为高电位, 而 $\bar{C}_s \rightarrow Q_s, 2C_s \rightarrow Q_s$ 都是低电位, 这样 $x_{20} = L_{20}, y_{20} = C_{20}$ 。当进行 $L_s - C_s$ 时, $L_s \rightarrow Q_s, C_s \rightarrow Q_s$ 是高电位, 而 $\bar{C}_s \rightarrow Q_s, 2C_s \rightarrow Q_s$ 是低电位, 这样 $x_{20} = L_{20}, y_{20} = \bar{C}_{20}$ 。当进行 $L_s + 2C_s$ 时, $L_s \rightarrow Q_s$ 和 $2C_s \rightarrow Q_s$ 是高电位, 而 $\bar{C}_s \rightarrow Q_s, C_s \rightarrow Q_s$ 是低电位, 这样 $x_{20} = L_{20}$ 而 $y_{20} = C_{21}$ 。在加法器尾数部分不工作时, $L_s \rightarrow Q_s, C_s \rightarrow Q_s, \bar{C}_s \rightarrow Q_s, 2C_s \rightarrow Q_s$ 都是低电位, 这时 $x_{20} = 0, y_{20} = 0$ 。实际上, 在一些较复杂的机器中, 由于指令较多, 接收门还要复杂些。

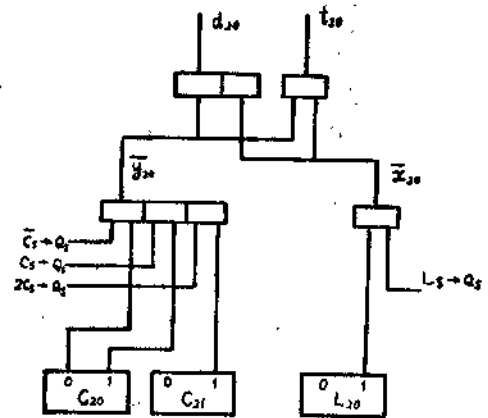


图 7.12

用“与非”门、“与或非”门做为接收门来接收数, 进入加法器的数一般是 \bar{x}_i 和 \bar{y}_i , 这是我们在形成 d_i 和 t_i 时必须考虑的。因此, 一般用图 7.12 的线路来形成 d_i 和 t_i 。

2. 和数的产生

在第三章中我们已经介绍了几种产生和数的方法。在实际线路中, 由于加速进位的需要, 所以一般都先产生 d_i 和 t_i 。此外, 由于控制线路的需要(如判断整个和数是否为全“0”等), 有时先产生 \bar{H}_i 比先产生和数 H_i 反而方便些。

因此, 在实际机器中, 往往采用下列方法产生和数。

(1) 两次半加法。设 h_i 为 x_i, y_i 之半和, 则

$$h_i = \bar{x}_i y_i + x_i \bar{y}_i = (\bar{x}_i + \bar{y}_i)(x_i + y_i) = \bar{x}_i y_i (x_i + y_i) = \bar{d}_i t_i$$

所以, 利用 d_i, t_i , 很容易产生 h_i 。

第二次半和 $H_i = h_i \bar{J}_{i+1} + \bar{h}_i J_{i+1}$ 仍可利用一个半加器得出。

图 7.13 是两次半加法的 4 种求和线路。

在(a)图中, 进位线路提供的是 J_{i+1} , 要求产生和数 H_i 。这时, d_i 经 YF_1 产生 $\bar{d}_i, \bar{d}_i, t_i$ 经 YF_2 而产生 \bar{h}_i, \bar{h}_i 和 J_{i+1} 经半加器而产生 H_i :

$$H_i = \bar{h}_i \bar{J}_{i+1} + h_i J_{i+1} = (h_i + \bar{J}_{i+1})(\bar{h}_i + J_{i+1}) = \bar{h}_i \bar{J}_{i+1} + h_i J_{i+1}$$

在图(b)中, 进位线路提供的是 \bar{J}_{i+1} , 要求产生和数 H_i , 这时, 还可以省去图(a)的 YF_2 而直接用 $\bar{d}_i \cdot t_i$ 代替 h_i 。

在图(c)中, 进位线路提供的是 J_{i+1} , 要求产生 \bar{H}_i 。在图(d)中, 进位线路提供的是 \bar{J}_{i+1} , 要求产生 \bar{H}_i 。其原理都和图(a)相同, 不再详细叙述。

(2) 利用向高位的进位产生和数。

在第三章曾介绍了利用向高位的进位产生和数的方法。

即

$$H_i = \bar{J}_i(x_i + y_i + J_{i+1}) + x_i y_i J_{i+1}$$

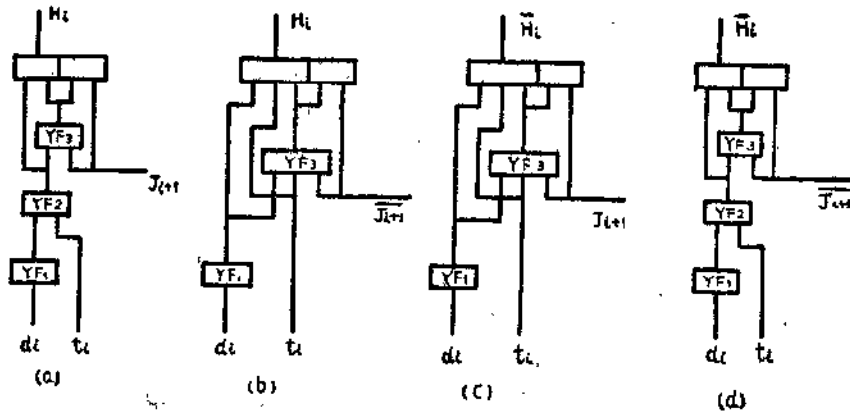


图 7.13

$$\bar{H}_i = J_i(\bar{x}_i + \bar{y}_i + \bar{J}_{i+1}) + \bar{x}_i \bar{y}_i \bar{J}_{i+1}$$

利用 $d_i = x_i y_i, t_i = x_i + y_i$, 上式可改写为:

$$\begin{aligned} \bar{H}_i &= \bar{J}_i(\bar{x}_i + \bar{y}_i + \bar{J}_{i+1}) + \bar{x}_i \bar{y}_i \bar{J}_{i+1} \\ &= \bar{J}_i t_i + \bar{J}_i \bar{J}_{i+1} + d_i \bar{J}_{i+1} \end{aligned} \quad (\text{图 7.14a})$$

$$\begin{aligned} H_i &= J_i(\bar{x}_i + \bar{y}_i + \bar{J}_{i+1}) + x_i y_i J_{i+1} \\ \text{或 } H_i &= J_i(\bar{x}_i \bar{y}_i + \bar{J}_{i+1}) + \bar{x}_i \bar{y}_i J_{i+1} \\ &= J_i \bar{d}_i + J_i \bar{J}_{i+1} + \bar{x}_i \bar{y}_i J_{i+1} \end{aligned} \quad (\text{图 7.14b})$$



图 7.14

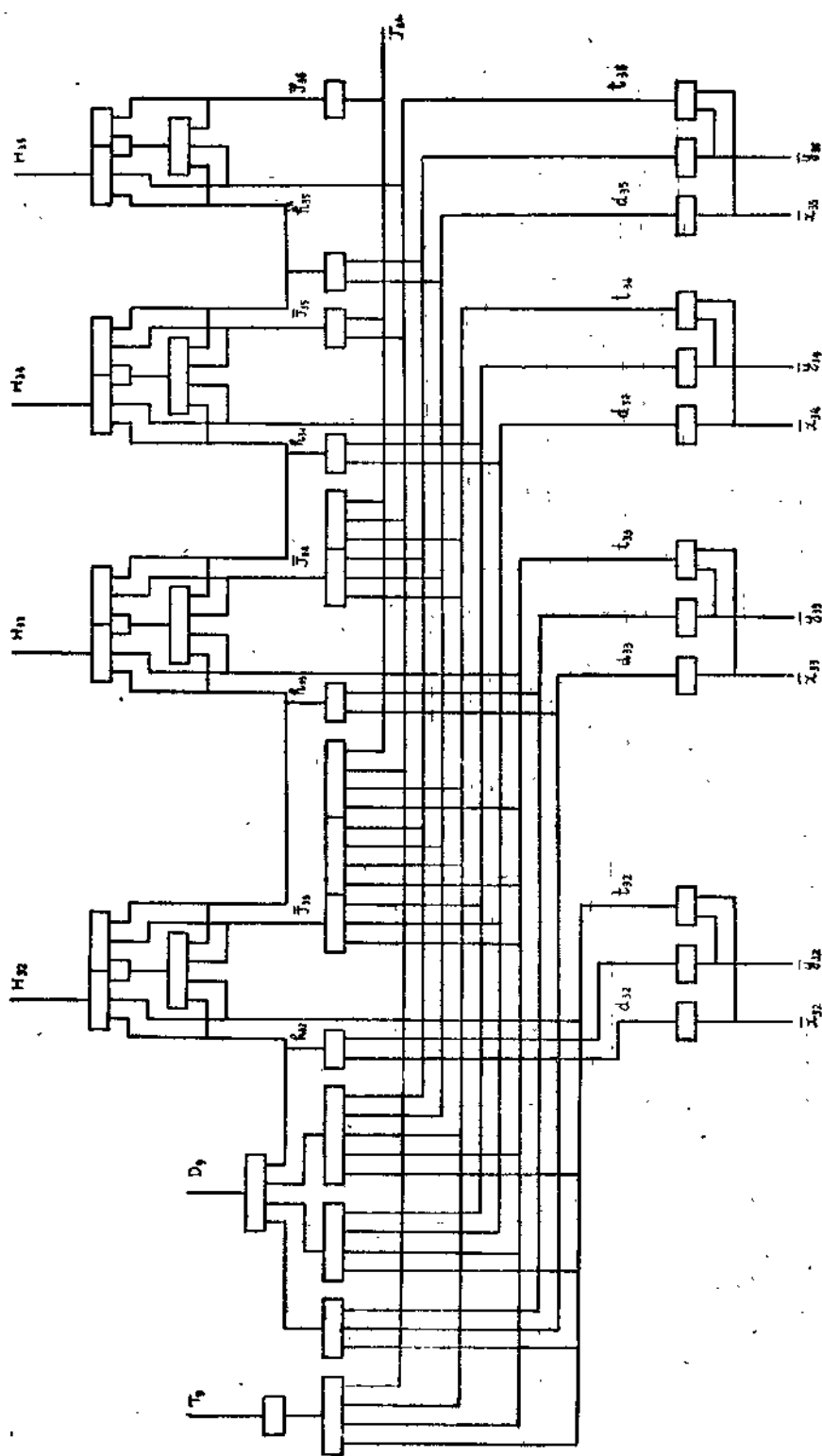
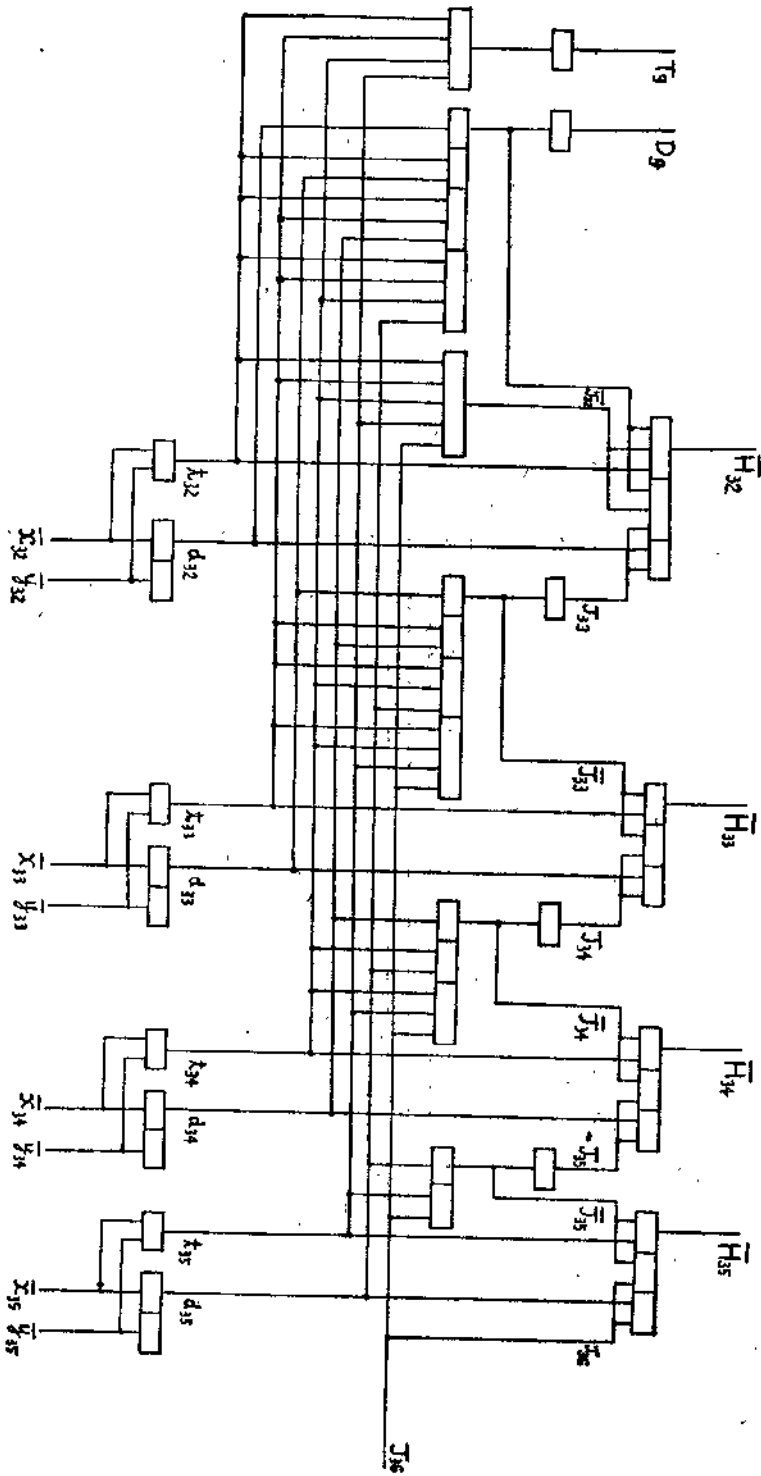


图 7.15

五、加法器设计实例

例 1 利用两次半加产生和数的加法器

图 7.15 是小组进位线路和求和线路, 共 41 位, 分成 10 小组, 每 4 位为一小组。本图标号是第 9 小组, 第 2—8 小组组内线路与第 9 组相同, 第 1 组和第 10 组不产生 $D_i A_i$ 而产生 J_0 和 J_{36} (参考图 7.4 的 J_{32}) 其组间进位线路如图 7.8 和图 7.10。为了节约元件, 求和线路采取图 7.13(b), 即用 $\bar{a}_i a_i$ 代替 h_{4i} , 而 \bar{a}_i 与进位中的一项共同使用一个“与非”门。



在本例中,假设字长为40位,那么从 $\bar{x}_i, \bar{y}_i, J_{40}$ 输入算起到形成 H_i 共需 $11.6\bar{t}_y$ (如果组间进位部分全改为“单与非”门则为 $10.8\bar{t}_y$)。按“单与非”门计算10个小组共需元件410个,组间部分共需36个,合计446个,平均每位为11.2个。

例2 利用向高位进位求和线路的加法器

本例见图7.16,本例字长、分组情况与上例均相同,不同的是,为便于产生控制电位而形成 \bar{H}_i ,产生和数时利用了向高位的进位。本例自 $\bar{x}_i, \bar{y}_i, J_{40}$ 输入到 \bar{H}_i 形成共需 $12.4\bar{t}_y$ (如果将 d_i 改为输出 x_i 和 y_i ,产生 D_i 的“与或非”门改为“单与非”门,组间进位部分也改为“单与非”门,则同样可降至 $10.8\bar{t}_y$)。

按“单与非”门计算,本例每小组共使用元件44个,组间进位部分和上例相同,共用36个,合计476个,平均每位为11.9个。

例3 快速加法器

下面介绍一种快速加法器。

所介绍的这种加法器字长为44位,每4位分成1小组,共分为11小组,其分组情况如图(7.17)

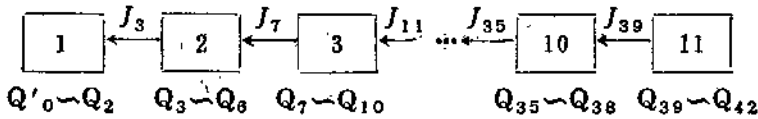


图 7.17

这种快速加法器和前面两例有几个重要的区别:

(1) 它具有快速进位线路,自 \bar{x}_i, \bar{y}_i 输入到产生 H_i 只需 $7.2\bar{t}_y$ 。所以能达到高速度,是和它的分组方法有关。它不再分为两个大组,而是把11个小组看成是一个大组。

根据前而的讨论,如果把11个小组看成是一个大组,那么 J_3 的表达式要多达10项,而且有的乘积项的因子多达10个:

$$J_3 = D_2 + T_2 D_3 + T_2 T_3 D_4 + \dots + T_2 T_3 T_4 T_5 T_6 T_7 T_8 T_9 T_{10} D_{11}$$

为了使用只有少量输入头的元件(假定“与非”门输入头不超过5个,“与或非”门不超过“5与或非”门)实现这个快速线路,就要设法减少项数并减少每个乘积项的因子个数。

首先,能不能减少项呢?我们先看头两项。

头两项 $D_2 + T_2 D_3$ 就能用下面方法合并为一项:

$$D_2 + T_2 D_3 = (D_2 + T_2)(D_2 + D_3) = \bar{D}_2 \bar{T}_2 \cdot \bar{D}_2 \bar{D}_3$$

在第2小组和第3小组中,由图7.9可知, $\bar{D}_2, \bar{T}_2, \bar{D}_3$ 都已经和组内进位同时产生了,而且比 D_2, T_2 要早 $1\bar{t}_y$ 的时间,因此 $\bar{D}_2 \bar{T}_2$ 和 $\bar{D}_2 \bar{D}_3$ 可以和 D_2, T_2, D_3, T_3 同时产生,这样我们可以用 $\bar{D}_2 \bar{T}_2 \cdot \bar{D}_2 \bar{D}_3$ 来代替 $D_2 + T_2 D_3$,从而减少了一项。

实际上, $\bar{D}_2 \bar{T}_2 \cdot \bar{D}_2 \bar{D}_3 = D_2 + T_2 D_3$ 可以看做是2、3两小组所组成的一个中组的本地进位,因此,为了书写方便,我们用 D_{2-3} 表示 $\bar{D}_2 \bar{T}_2 \cdot \bar{D}_2 \bar{D}_3$ 。

仿照这样的方法, J_3 的第3、4两项也可以合并,因为

$$T_2 T_3 D_4 + T_2 T_3 T_4 D_5 = T_2 T_3 (D_4 + T_4 D_5)$$

提出公因子 $T_2 \cdot T_3$ 后余下部分也可表为两因子的乘积,即 $\bar{D}_4 \bar{T}_4 \cdot \bar{D}_4 \bar{D}_5$ 。同理我们把它看做是4、5两小组所组成的中组的本地进位,记作 D_{4-5} 。依此类推,每两小组都产生一个它们的本

地进位:

$$\begin{aligned} D_{2-3} &= \overline{D_2} \overline{T_2} \cdot \overline{D_2} \overline{D_3} \\ D_{4-5} &= \overline{D_4} \overline{T_4} \cdot \overline{D_4} \overline{D_5} \\ D_{6-7} &= \overline{D_6} \overline{D_6} \cdot \overline{D_6} \overline{D_7} \\ D_{8-9} &= \overline{D_8} \overline{T_8} \cdot \overline{D_8} \overline{D_9} \\ D_{10-11} &= \overline{D_{10}} \overline{T_{10}} \cdot \overline{D_{10}} \overline{D_{11}} \end{aligned}$$

这样, J_3 的表达式就可以压缩为 5 项:

$$\begin{aligned} J_3 &= D_{2-3} + T_2 T_3 D_{4-5} + T_2 T_3 T_4 T_5 D_{6-7} + T_2 T_3 T_4 T_5 T_6 T_7 D_{8-9} + \\ &+ T_2 T_3 T_4 T_5 T_6 T_7 T_8 T_9 D_{10-11} \end{aligned}$$

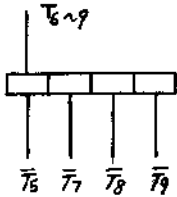


图 7.18

但是, 它的乘积项最多的仍包含有 10 个因子 (D_{10-11} 应看做是两个因子)。

为了减少每个乘积项所包含的因子, 我们采取了先产生 T_i 的乘积项的办法, 从图 7.9 可知, \overline{T}_i 比 T_i 要早一级产生, 因此, 我们用一个“4 与或非”门就可以在产生 D_i 的同时产生一个 T_i 的乘积项 (如 $T_2 T_3 T_4 T_5$ 或 $T_6 T_7 T_8 T_9$)。

图 7.18 是用“4 与或非”门产生的 $T_6 T_7 T_8 T_9$, 实际上它可以看做是进位跳过第 6~9 组的跳过条件, 因此, 我们用 T_{6-9} 来表示 $T_6 T_7 T_8 T_9$ 。仿此, 可以产生 T_{2-5} 和 T_{6-7} 等。

从而, J_3 的表达式可写成下面形式:

$$\begin{aligned} J_3 &= D_{2-3} + T_2 T_3 D_{4-5} + T_{2-5} D_{6-7} + T_{2-5} T_{6-7} D_{8-9} + \\ &+ T_{2-5} T_{6-9} D_{10-11} \end{aligned}$$

依此类推, $J_7, J_{11}, J_{15}, \dots$ 的表达式分别为:

$$\begin{aligned} J_7 &= D_3 + T_2 D_{4-5} + T_{3-5} D_{6-7} + T_{3-5} T_{6-7} D_{8-9} + \\ &+ T_{3-5} T_{6-9} D_{10-11} \\ J_{11} &= D_{4-5} + T_{4-5} D_{6-7} + T_{4-5} T_{6-7} D_{8-9} + T_{4-5} T_{6-9} D_{10-11} \\ J_{15} &= D_6 + T_5 D_{6-7} + T_5 T_{6-7} D_{8-9} + T_5 T_{6-9} D_{10-11} \\ J_{19} &= D_{6-7} + T_{6-7} D_{8-9} + T_{6-9} D_{10-11} \\ J_{23} &= D_7 + T_7 D_{8-9} + T_7 T_{8-9} D_{10-11} \\ J_{27} &= D_{8-9} + T_{8-9} D_{10-11} \\ J_{31} &= D_9 + T_9 D_{10-11} \\ J_{35} &= D_{10-11} \\ J_{39} &= D_{11} \end{aligned}$$

图 7.20(a)(b) 就是组间进位线路, 在图 20(b) 中, 为了使 D_{10-11} 不超过 8 个负载 (在本书快速线路中, 我们总是假定负载不超过 8 个), 所以产生 J_{35} 时用 $D_{10} + T_{10} D_{11}$ 来代替 D_{10-11} 。

(2) 它具有快速产生和数的线路。

在前两例中, 当组间进位产生以后, 总是要先回到小组与组内产生的进位汇合后再形成和数, 这样就至少要占用 $2.4t_p$ 的时间, 在本例则省去了这部分时间。

图 7.19 是本例第 9 小组的组内进位线路和产生和数的线路。以 H_{31} 为例, H_{31} 由 x_{31}, y_{31}, J_{32} 经两次半加而形成, 即

$$H_{31} = \overline{h_{31}} J_{32} + h_{31} J_{32} \quad \text{其中 } h_{31} = \overline{x_{31}} y_{31} + x_{31} \overline{y_{31}}$$

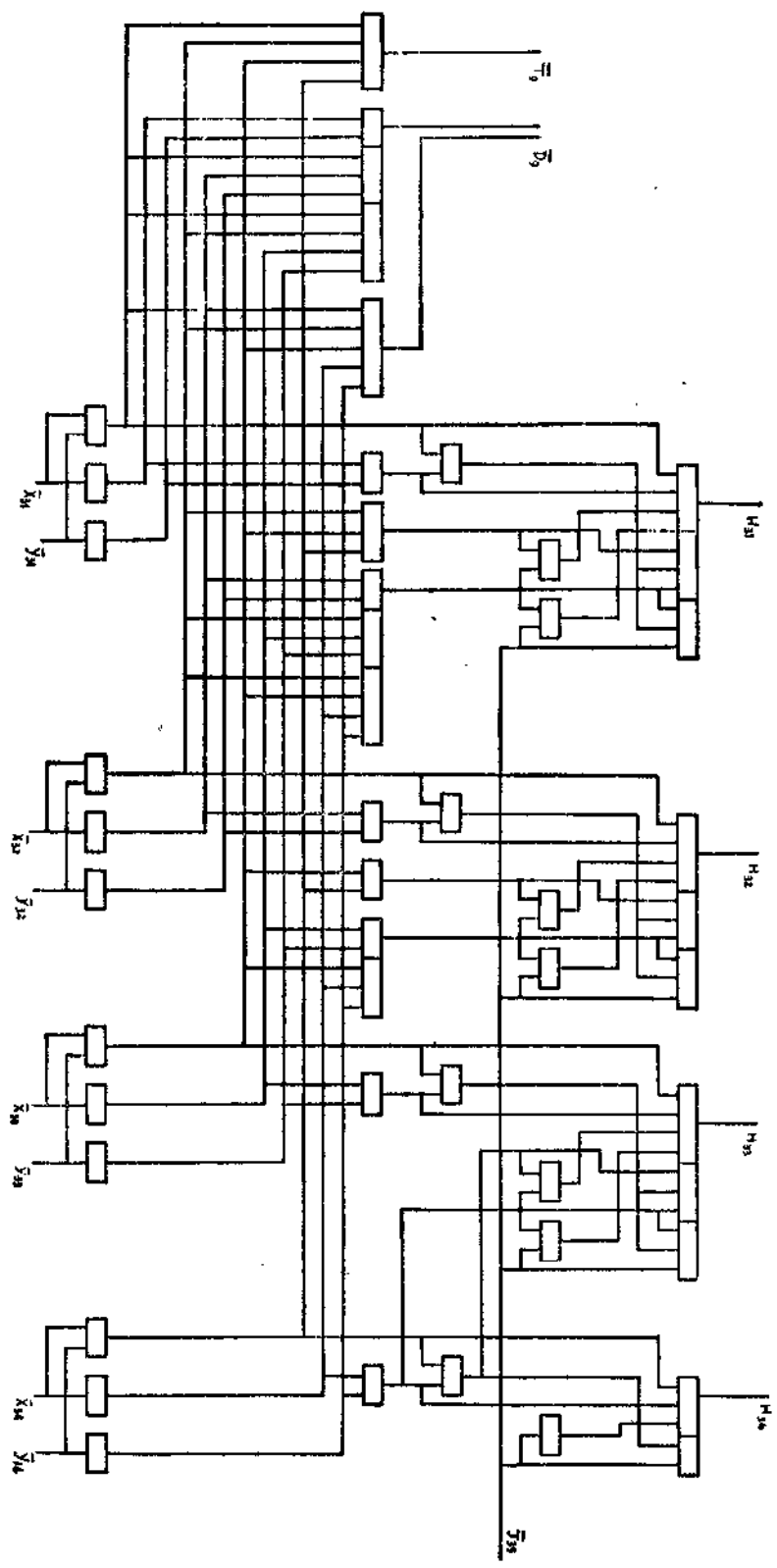


图 7.19