第14卷 第1期 电子科学学刊 Vol. 14, No. 1
1992年1月 JOURNAL OF ELECTRONICS Jan., 1992

多路选择器树形结构逻辑网络设计

姜文彬
(淮北煤炭师范学院物理系,安徽淮北 235000)

摘要 本文首先讨论了逻辑函数的分解;然后给出利用多路选择器实现多变量组合函数时,获得最小或接近最小的树形结构网络的设计方法. 该方法适用于计算机自动综合.

关键词 逻辑设计;多路选择器;最小化

一、引 言

文献[1~3]已经表明,多路选择器(MUX)(亦称数据选择器)可以用作通用逻辑模块来实现任何组合函数. 当用多路选择器组成待实现函数的树形结构网络时,各级网络控制(地址)变量的配置将影响树形结构网络的繁简程度. 如果在树形结构网络中所用的多路选择器的数目最少,则称该网络为最小树形结构网络,此时各级网络的控制变量的配置称为最佳配置. 显然,待实现函数中所含变量越多,各级网络控制变量最佳配置的选取就越困难. 本文在讨论组合函数分解的基础上,给出利用多路选择器实现多变量组合函数时,各级网络控制变量的一种最佳或接近最佳配置的选取方法. 由此可以获得该函数利用多路选择器实现时的最小或接近最小的树形结构网络.

二、函数的分解

考虑任一个 n 个变量的组合函数 $f(X)$,其中 $X=\{x_1,x_2,\cdots,x_n\}$. 若将其变量集合分为两个子集合:

$$B_1=\{x_1,x_2,\cdots,x_{r-1},x_r\},\quad B_2=\{x_{r+1},x_{r+2},\cdots,x_{n-1},x_n\} \tag{1}$$

则由 Shannon 展开定理[4],可以证明

$$f(X)=f(B_1;B_2)=\sum_{i=0}^{p} f_i(B_2)I_i(B_1) \tag{2}$$

式中,$p=2^r-1$;$f_i(B_2)$ 为集合 B_2 中的变量的函数,称为原函数的子函数(剩余函数);$I_i(B_1)$ 为集合 B_1 的变量组成的某个最小项;i 为该最小项值为1时各变量的取值组成的二进制数对应的十进制数,$0\leqslant i\leqslant p$. 由(2)式可以看出,函数分解后得到的复合函数可视为 r 个变量 $x_1,x_2,\cdots,x_r$ 的函数,而各个子函数则可视为这个复合函数的真值表中相应的函数值. 将复合函数的真值表与原函数的真值表相对照,可得各个子函数的真值表,由此可以求得各个子函数. 例如,令 $B_1=\{$除 x_i 外的所有变量$\}$,$B_2=\{x_i\}$,将函数以此划分按(2)式分解后得到的子函数的可能类型(模式)最多只能有4种,即0,

1990.08.07收到,1991.06.10定稿.

表 1

x_i	0	1
	0	0

$x_i\langle 0\rangle$

x_i	0	1
	1	1

$x_i\langle 1\rangle$

x_i	0	1
	0	1

$x_i\langle x_i\rangle$

x_i	0	1
	1	0

$x_i\langle \bar{x}_i\rangle$

表 2

x_i \ x_j	0	1
0	0	0
1	0	0

$x_ix_j\langle 0\rangle$

x_i \ x_j	0	1
0	1	1
1	1	1

$x_ix_j\langle 1\rangle$

x_i \ x_j	0	1
0	0	0
1	1	1

$x_ix_j\langle x_i\rangle$

x_i \ x_j	0	1
0	1	1
1	0	0

$x_ix_j\langle \bar{x}_i\rangle$

x_i \ x_j	0	1
0	0	1
1	0	1

$x_ix_j\langle x_j\rangle$

x_i \ x_j	0	1
0	1	0
1	1	0

$x_ix_j\langle \bar{x}_j\rangle$

表 3

x_ix_j \ x_k		0	1
0	0	0	0
0	1	0	0
1	0	0	0
1	1	0	0

$x_ix_jx_k\langle 0\rangle$

x_ix_j \ x_k		0	1
0	0	1	1
0	1	1	1
1	0	1	1
1	1	1	1

$x_ix_jx_k\langle 1\rangle$

x_ix_j \ x_k		0	1
0	0	0	0
0	1	0	0
1	0	1	1
1	1	1	1

$x_ix_jx_k\langle x_i\rangle$

x_ix_j \ x_k		0	1
0	0	1	1
0	1	1	1
1	0	0	0
1	1	0	0

$x_ix_jx_k\langle \bar{x}_i\rangle$

x_ix_j \ x_k		0	1
0	0	0	0
0	1	1	1
1	0	0	0
1	1	1	1

$x_ix_jx_k\langle x_j\rangle$

x_ix_j \ x_k		0	1
0	0	1	1
0	1	0	0
1	0	1	1
1	1	0	0

$x_ix_jx_k\langle \bar{x}_j\rangle$

x_ix_j \ x_k		0	1
0	0	0	1
0	1	0	1
1	0	0	1
1	1	0	1

$x_ix_jx_k\langle x_k\rangle$

x_ix_j \ x_k		0	1
0	0	1	0
0	1	1	0
1	0	1	0
1	1	1	0

$x_ix_jx_k\langle \bar{x}_k\rangle$

1，x_i 和 $\bar{x}_i$，它们均为平凡函数，如表 1 所示。将表 1 中的各函数称为关于变量 x_i 所定义的一维子函数，以符号“$x_i\langle\cdot\rangle$”表示。同理，可得关于 x_i 和 x_j 所定义的二维子函

数以及关于 x_i, x_j 和 x_k 所定义的三维子函数，等等。表 2 和表 3 分别示出了二维和三维的平凡子函数。

令符号“$\bullet\bullet\langle\bullet,\bullet\rangle$”表示由 x_ix_j（或 x_ix_k，或 x_jx_k）定义的两个子函数；符号“$A \Rightarrow C$”表示 C 是 A 的必要条件。分析表 2 和表 3 可得关于子集合 $\{x_i, x_j, x_k\}$ 所定义的平凡子函数与其子集合 $\{x_i, x_j\}$, $\{x_i, x_k\}$ 和 $\{x_j, x_k\}$ 分别定义的平凡子函数之间的关系为

$$\left.\begin{array}{l}
x_ix_jx_k\langle 0\rangle \Rightarrow x_ix_j\langle 0,0\rangle \text{ 且 } x_ix_k\langle 0,0\rangle \text{ 且 } x_jx_k\langle 0,0\rangle \\
x_ix_jx_k\langle 1\rangle \Rightarrow x_ix_j\langle 1,1\rangle \text{ 且 } x_ix_k\langle 1,1\rangle \text{ 且 } x_jx_k\langle 1,1\rangle \\
x_ix_jx_k\langle x_i\rangle \Rightarrow x_ix_j\langle x_i,x_i\rangle \text{ 且 } x_ix_k\langle x_i,x_i\rangle \text{ 且 } x_jx_k\langle 0,1\rangle \\
x_ix_jx_k\langle \bar{x}_i\rangle \Rightarrow x_ix_j\langle \bar{x}_i,\bar{x}_i\rangle \text{ 且 } x_ix_k\langle \bar{x}_i,\bar{x}_i\rangle \text{ 且 } x_jx_k\langle 1,0\rangle \\
x_ix_jx_k\langle x_j\rangle \Rightarrow x_ix_j\langle x_j,x_j\rangle \text{ 且 } x_ix_k\langle 0,1\rangle \text{ 且 } x_jx_k\langle x_j,x_j\rangle \\
x_ix_jx_k\langle \bar{x}_j\rangle \Rightarrow x_ix_j\langle \bar{x}_j,\bar{x}_j\rangle \text{ 且 } x_ix_k\langle 1,0\rangle \text{ 且 } x_jx_k\langle \bar{x}_j,\bar{x}_j\rangle \\
x_ix_jx_k\langle x_k\rangle \Rightarrow x_ix_j\langle 0,1\rangle \text{ 且 } x_ix_k\langle x_k,x_k\rangle \text{ 且 } x_jx_k\langle x_k,x_k\rangle \\
x_ix_jx_k\langle \bar{x}_k\rangle \Rightarrow x_ix_j\langle 1,0\rangle \text{ 且 } x_ix_k\langle \bar{x}_k\bar{x}_k\rangle \text{ 且 } x_jx_k\langle \bar{x}_k,\bar{x}_k\rangle
\end{array}\right\} \quad (3)$$

以上关系给我们提供了由函数的某个子集合的各个子集合分别定义的平凡子函数来寻找这个子集合定义的平凡子函数的方法。由于(3)式中的条件为必要条件，故知当箭号右边的条件不满足时，就不能得到左边的子函数。

同理，可得子集合 $\{x_i, x_j, x_k, x_l\}$ 所定义的平凡子函数与其子集合 $\{x_i, x_j\}$, $\{x_i, x_k\}$, $\{x_i, x_l\}$, $\{x_j, x_k\}$, $x_j, x_l\}$ 和 $\{x_k, x_l\}$ 分别定义的平凡子函数之间的关系（必要条件）如表 4 所示。表 4 中括号里的数字为该类型子函数的个数。表 4 提供了由这些子集合的平凡函数寻找由它们组成的集合的平凡函数的方法。

表 4

$x_ix_jx_kx_l$	x_ix_j	x_ix_k	x_ix_l	x_jx_k	x_jx_l	x_kx_l
0	0(4)	0(4)	0(4)	0(4)	0(4)	0(4)
1	1(4)	1(4)	1(4)	1(4)	1(4)	1(4)
x_i	x_i(4)	x_i(4)	x_i(4)	0(2),1(2)	0(2),1(2)	0(2),1(2)
$\bar{x}_i$	$\bar{x}_i$(4)	$\bar{x}_i$(4)	$\bar{x}_i$(4)	1(2),0(2)	1(2),0(2)	1(2),0(2)
x_j	x_j(4)	0(2),1(2)	0(2),1(2)	x_j(4)	x_j(4)	0(2),1(2)
$\bar{x}_j$	$\bar{x}_j$(4)	1(2),0(2)	1(2),0(2)	$\bar{x}_j$(4)	$\bar{x}_j$(4)	1(2),0(2)
x_k	0(2),1(2)	x_k(4)	0(2),1(2)	x_k(4)	0(2),1(2)	x_k(4)
$\bar{x}_k$	1(2),0(2)	$\bar{x}_k$(4)	1(2),0(2)	$\bar{x}_k$(4)	1(2),0(2)	$\bar{x}_k$(4)
x_l	0(2),1(2)	0(2),1(2)	x_l(4)	0(2),1(2)	x_l(4)	x_l(4)
$\bar{x}_l$	1(2),0(2)	1(2),0(2)	$\bar{x}_l$(4)	1(2),0(2)	$\bar{x}_l$(4)	$\bar{x}_l$(4)

三、二级网络设计

由前节中函数分解公式(2)式可知，任一个 n 个变量的组合函数，都可用多路选择器

组成的二级树形结构网络实现.其中第一级网络的控制变量为集合 B_1 的所有变量,第二级网络的控制变量为集合 B_2 的所有变量. 在前节组合函数分解的讨论的基础上，给出求二级树形结构网络中各级网络控制变量的最佳(或接近最佳)配置的算法如下:

(1) (a) 给出 n, r. (b) 确定所给的函数的真值表（二维形式). (c) 令 $x_i = x_1$, $x_j = x_2$.

(2) 以 $B_1 = \{$除 x_i 和 x_j 外的所有变量$\}$, $B_2 = \{x_i, x_j\}$, 将函数按(2)式分解.求出函数分解后产生的二维平凡子函数的类型和每种类型所含子函数的个数，并计算二维平凡子函数的总数 N.

(3) 令 $x_i = x_{i'}, x_j = x_{j'}, i'j'$ 依次取 13,14,…,1n,23,24,…,2n,…,$(n-1)n$,重复(2),共做 $[n(n-1)/2-1]$ 次，$(n>2)$.

(4) (a) 若 $n-r=2$, 则转(5). (b) 若 $n-r=3$, 则转(6). (c) 若 $n-r=4$,则转(8).

(5) (a) 在函数的 $n(n-1)/2$ 次分解所得到的N值中，取与最大的N值相对应的划分 $\{B_1;B_2\}$. (b) 以此划分将函数按(2)式分解，求出分解后产生的非平凡子函数(若有时)和复合函数,均以真值表形式给出,转(12).

(6) (a) 在函数的 $n(n-1)/2$ 次分解的划分中,寻找满足(3)式的 3 个集合并以它们为子集合组成集合 $\{x_i, x_j, x_k\}$. 若找不到，令 $N_1=0$, 转(10). (b) 以 $B_1 = \{$除 x_i, x_j 和 x_k 外的所有变量$\}$, $B_2 = \{x_i, x_j, x_k\}$, 将函数按 (2) 式分解. 求出函数分解后产生的三维平凡子函数的类型和每种类型所含子函数的个数，并计算三维平凡子函数的总数 N_1.

(7) 在 $N > 2N_1$ 的值中，寻找满足(3)式的 3 个集合，并以它们为子集合组成新集合$\{x_i, x_j, x_k\}$. 若找到,则转(6) (b); 否则,转(10).

(8) (a) 在函数的 $n(n-1)/2$ 次分解的划分中，寻找满足表 4 所示必要条件的 6 个集合,并以它们为子集合组成集合 $\{x_i, x_j, x_k, x_l\}$. 若找不到,令 $N_1=0$,转(10). (b) 以 $B_1 = \{$除 x_i, x_j, x_k 和 x_l 外的所有变量$\}$, $B_2 = \{x_i, x_j, x_k, x_l\}$, 将函数按(2)式分解. 求出函数分解后产生的四维平凡子函数的类型和每种类型所含子函数的个数，并计算四维平凡子函数的总数 N_1.

(9) 在 $N > 4N_1$ 的值中,寻找满足表 4 所示必要条件的 6 个集合,并以它们为子集合组成新集合 $\{x_i, x_j, x_k, x_l\}$. 若找到,则转(8) (b).

(10) 若 $N_1=0$, 对于所有的 N_1; 则转(12).

(11) (a) 在求得 N_1 值中,取与最大的 N_1 值相对应的划分 $\{B_1;B_2\}$. (b) 以此划分将函数按(2)式分解,求出分解后产生的非平凡子函数(若有时)和复合函数，均以真值表形式给出.

(12) 停.

例 对于六变量函数

$f(x_1, x_2, x_3, x_4, x_5, x_6) = \sum(0,1,2,3,9,11,12,14,16,17,18,19,25,27,28,30,32,33,40,43,48,49,52,53,54,55,57,58,60,61,62,63)$

应用上述算法,求得最佳配置为 $B_1 = \{x_1, x_3, x_4\}$, $B_2 = \{x_2, x_5, x_6\}$. 以此划分将函数按

(2)式分解如表 5 所示。得到的非平凡子函数和复合函数分别如表 6 和表 7 所示。

表 5

$x_1\ x_2\ x_3$ \ $x_4\ x_5\ x_6$	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
0 0 0	1	1	1	1	0	0	0	0
0 0 1	1	1	1	1	0	0	0	0
0 1 0	0	1	0	1	1	0	1	0
0 1 1	0	1	0	1	1	0	1	0
1 0 0	1	1	0	0	0	0	0	0
1 0 1	1	1	0	0	1	1	1	1
1 1 0	1	0	0	1	0	0	0	0
1 1 1	0	1	1	0	1	1	1	1

表 6

x_2 \ $x_5\ x_6$	0 0	0 1	1 0	1 1
0	1	0	0	1
1	0	1	1	0

$a(x_2, x_5, x_6)$

表 7

x_1 \ $x_3\ x_4$	0 0	0 1	1 0	1 1
0	1	0	x_6	$\bar{x}_6$
1	$\bar{x}_5$	x_2	a	x_2

$f(x_1, x_3, x_4; x_2, x_5, x_6, a)$

由表 6 和表 7 作出函数以八选一选择器实现的二级树形结构网络如图 1 所示。

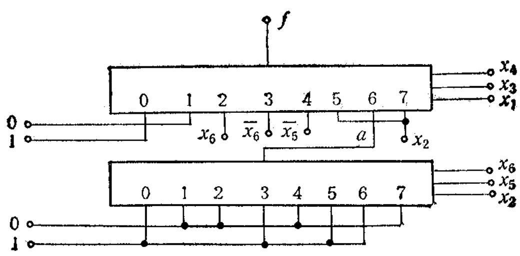

图 1　八选一选择器树形结构网络

四、多级网络设计

仍以上述六变量函数为例来说明。考虑上述函数用四选一选择器实现时，最小树形结构网络的设计。显然，此时所设计的网络应为三级网络。设计方法一般是先求出函数用二选一选择器实现的最小树形结构网络，然后把它转换为用四选一选择器实现的最小树形结构网络。对于多级网络设计，原则上可以重复使用上述算法，使每次迭代过程所得

到的复合函数中出现最多的平凡子函数．但是这种每级网络控制变量次序固定的设计方法有时只能得到接近最小的树形结构网络．在上述算法的基础上，考虑到各级网络控制变量次序可以是混合的，则有时可以得到最小的树形结构网络．

先考虑函数用二选一选择器实现．此时多级网络设计从第一级开始．将表 5 中的函数值用变量替换法[3]替换，以使得形成的各个平凡子函数的维数达到尽可能高，如表 8 所示．可以看出，若平凡子函数的维数是 $(n-k), k=1,2,\cdots,(n-1)$，则多级网络的第 k 级网络上以这个平凡子函数为其子函数的 $(n-k+1)$ 维非平凡子函数(函数 f 可视为 $k=1$ 时的特例)的首级(即第 1 级）是级可实现的[5]，即这个非平凡子函数的首级

表 8

$x_1\ x_3\ x_2$ \ $x_4\ x_5\ x_6$		000	001	010	011	100	101	110	111	
0 0 0		$\bar{x}_4$	$\bar{x}_4$	$\bar{x}_4$	$\bar{x}_4$	$\bar{x}_4$	$\bar{x}_4$	$\bar{x}_4$	$\bar{x}_4$	← ①
0 0 1		$\bar{x}_4$	$\bar{x}_4$	$\bar{x}_4$	$\bar{x}_4$	$\bar{x}_4$	$\bar{x}_4$	$\bar{x}_4$	$\bar{x}_4$	
0 1 0		x_6	x_6	x_6	x_6 ← ③	$\bar{x}_6$	$\bar{x}_6$	$\bar{x}_6$	$\bar{x}_6$	← ④
0 1 1		x_6	x_6	x_6	x_6	$\bar{x}_6$	$\bar{x}_6$	$\bar{x}_6$	$\bar{x}_6$	
1 0 0		$\bar{x}_5$	$\bar{x}_5$	$\bar{x}_5$	$\bar{x}_5$ ← ⑤	x_2	x_2	x_2	x_2	← ②
1 0 1		$\bar{x}_5$	$\bar{x}_5$	$\bar{x}_5$	$\bar{x}_5$	x_2	x_2	x_2	x_2	
1 1 0	⑥ →	$\bar{x}_6$	$\bar{x}_6$	x_6	x_6 ← ⑦	x_2	x_2	x_2	x_2	
1 1 1	⑧ →	x_6	x_6	$\bar{x}_6$	$\bar{x}_6$ ← ⑨	x_2	x_2	x_2	x_2	

网络用单个二选一选择器实现，其控制变量为该非平凡子函数中的某个变量 x_i，并且有一个数据输入连接那个 $(n-k)$ 维的平凡子函数，而另一个数据输入连接一个由该非平凡子函数以 $B_1=\{x_i\}, B_2=\{$除 x_i 外这个非平凡子函数的所有变量$\}$为划分，按(2)式分解所产生的另一个子函数，它可以是非平凡子函数，或者是平凡子函数．

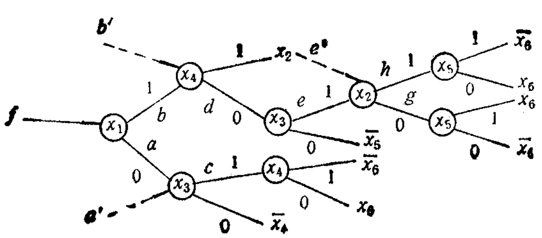

图 2 函数 f 的二叉判定图（B 图)

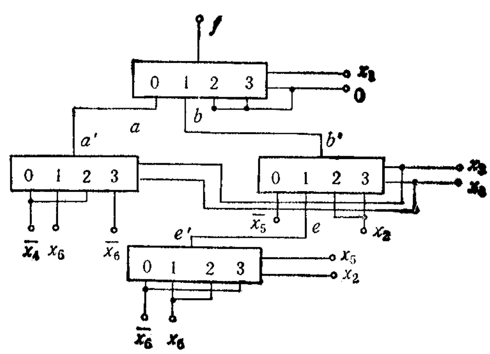

图 3 四选一选择器树形结构网络

由表 8 可直观地得出函数 f 的二叉判定图(简称 B 图)[6]，如图 2 所示。可以看出，此时 B 图是最小(最简)的。图中每个结点对应一个二选一选择器，其中结点变量为选择器的控制变量，出枝变量或出口变量为选择器的输入变量，入枝变量或入口变量为选择器的输出。于是，由 B 图可直接得到函数用二选一选择器实现的树形结构网络。

为了得到函数用四选一选择器实现的最小树形结构网络，在函数的 B 图中嵌入子函数 a', b', e'，如图 2 中的虚线所示。由 B 图可知，子函数 a', b', e' 分别具有与 a, b, e 相同的子功能。由 B 图列出 a'，b'，e' 的真值表如表 9 所示。于是可作出相应的用四选一选择器组成的最小树形结构网络如图 3 所示。

表 9

x_4 \ x_3	0	1
0	$\bar{x}_4$	x_6
1	$\bar{x}_4$	$\bar{x}_6$

a'

x_4 \ x_3	0	1
0	$\bar{x}_5$	e
1	x_2	x_2

b'

x_2 \ x_5	0	1
0	$\bar{x}_6$	x_6
1	x_6	$\bar{x}_6$

e'

五、结　　论

本文首先讨论了组合函数的分解方法，这种分解方法可以利用真值表的分解较方便地实现。然后给出了利用多路选择器实现多变量组合函数时，各级网络控制变量的最佳(或接近最佳)配置的选取方法，从而设计出待实现函数的最小(或接近最小)的树形结构网络。利用本文设计方法得到的最小树形结构网络不一定是唯一的，有时可能有两种或多种最小树形结构网络，此时任取一种即可。本文给出的设计方法易于实现使用计算机自动综合。当函数中所含变量较少时，利用手工综合也较方便。

作者感谢吴训威教授的指导和帮助。

参　考　文　献

[1] R. P. Voith, *IEEE Trans. on C*, C-26(1977)5, 117—424.
[2] C. C. Cheung, R. W. Ehrich, *IEEE Trans. on C*, C-24(1975)11, 1110—1113.
[3] K. Y. Fang, A. S. Wojcik, *IEEE Trans. on C*, C-37(1988)10, 1293—1301.
[4] S. C. Lee, Modern Switching Theory and Digital Design, Prentice-Hall, Inc., Englewood Cliffs, N. J., (1978).
[5] R. K. Goral, A. Pal, *IEE Proc. -E*, 137(1990)2. 164—170.
[6] S. B. Akers, *IEEE Trans. on C*, C-27(1978)6, 509—516.

DESIGN OF TREE-TYPE LOGIC NETWORKS USING MULTIPLEXERS

Jiang Wenbin

(Huaibei Coal Techers College, Huaibei, Anhui 235000)

Abstract A decomposition approach of logic functiony is discussed. The design method, by which the minimization or near minimization of tree-type logic networks can be obtained, is presented when multiple variables combinational logic function is implemented by using multiplexers. Using the method, the automatic synthesis of the logic function can easily be accomplished on conputers.

Key words Logic design; Multiplexer; Minimization