基于 OKFDDs 的 Reed-Muller 逻辑混合极性转换算法

汪鹏君* 李 辉
(宁波大学电路与系统研究所 宁波 315211)

摘 要：混合极性转换是 RM (Reed-Muller)电路逻辑综合过程的一个重要环节，能够实现从 Boolean 逻辑最小项表达式到 RM 逻辑 MPRM (Mixed-Polarity Reed-Muller)表达式的转换。该文通过对 OKFDDs (Ordered Kronecker Functional Decision Diagrams)展开规律的研究，建立 MPRM 表达式与 OKFDDs 数据结构的对应关系。在此基础上，根据最小项系数与 MPRM 系数的下标包含关系，结合多输出函数描述方式，提出一种直接从最小项表达式展开到 MPRM 表达式的新型混合极性转换算法。最后通过对多个 Benchmark 测试的实验结果表明其转换效率相比其它混合极性转换算法有明显提高。

关键词：电路设计；电路逻辑综合；OKFDDs；MPRM 表达式；极性转换
中图分类号：TP391.72 **文献标识码**： A **文章编号**：1009-5896(2011)-04-0932-06
DOI: 10.3724/SP.J.1146.2010.00776

An Algorithm of Reed-muller Logic Mixed-polarity Conversions Based on OKFDDs

Wang Peng-jun Li Hui
(*Institute of Circuits and Systems*, *Ningbo University*, *Ningbo* 315211, *China*)

Abstract: Mixed-Polarity conversion is one of important phases in logic synthesis of Reed-Muller (RM) circuits, which implements the conversions from Boolean logic Minterm expressions to RM logic Mixed-Polarity Reed-Muller (MPRM) expressions. In this paper, based on the research of decomposed rules of Ordered Kronecker Functional Decision Diagrams (OKFDDs) the relations between MPRM expressions and OKFDDs database are established. On this basis, according to the subscripts included relations between Minterm coefficients and MPRM coefficients and combining the description of multi-output logic functions, a novel Mixed-Polarity conversion algorithm directly from Minterm expressions to MPRM expressions is proposed. Finally, through several Benchmark tests, the results show the efficiency of the methods, which is significantly improved compared to other conversion algorithms.

Key words: Circuit design; Circuit logic synthesis; Ordered Kronecker Functional Decision Diagrams (OKFDDs); Mixed-Polarity Reed-Muller (MPRM) expressions; Polarity conversion

1 引言

逻辑函数可用基于 AND/OR/NOT 运算的 Boolean 逻辑表示，也可用基于 AND/XOR 运算的 RM(Reed-Muller)逻辑表示。目前，逻辑综合技术主要针对 Boolean 逻辑展开。但大量研究表明：用 RM 逻辑实现的电路相比传统 Boolean 逻辑不仅有更好的可测试性[1]，而且部分功能电路(如算术逻辑部件、通信系统、错误校验等)在面积、功耗和速度等方面具有显著优势[2−5]。因此，开发快速有效的 RM 电路逻辑综合与优化算法，对完善电路优化设计 CAD 软件完备性有其重要现实意义。

MPRM(Mixed-Polarity Reed-Muller)表达式和 FPRM(Fixed-Polarity Reed-Muller)表达式是两种重要的 RM 逻辑表达式。FPRM 表达式中变量只能以原变量或反变量出现，而 MPRM 表达式中变量的出现形式不受此限制，因此其极性搜索空间更大，其优化结果也更佳[6]。RM 逻辑表达式可通过极性转换算法从 Boolean 逻辑最小项表达式转换得到。文献[7]提出了一种基于并行列表技术的固定极性转换算法，但此转换算法只能实现任意变量最小项表达式与 FPRM 表达式的函数转换；文献[8]提出了一种基于三角形数据结构的混合极性转换算法，但此转

2010-07-20 收到，2010-11-11 改回
国家自然科学基金(61076032, 60776022)，中国博士后科学基金(20090461355)，浙江省博士后科研项目，浙江省自然科学基金(Y1101078)和浙江省公益性技术应用研究计划项目(2010C31012)和浙江省大学生科技创新活动计划(新苗人才计划)资助课题
*通信作者：汪鹏君 wangpengjun@nbu.edu.cn

换算法只适合单输出最小项表达式与 MPRM 表达式的函数转换。

而 OKFDDs(Ordered Kronecker Functional Decision Diagrams)是一种能有效表示和操作逻辑函数的图形数据结构[9]，已成功地应用于基于 XOR 电路的逻辑综合与优化中。鉴于此，本文在研究 MPRM 表达式、OKFDDs 数据结构和 OKFDDs 展开规律的基础上，通过推导最小项系数与 MPRM 系数的下标包含关系，结合多输出逻辑函数描述方式，提出一种直接从最小项表达式到 MPRM 表达式的并行混合极性转换算法，最后通过实验验证该混合极性转换算法的有效性。

2　MPRM 表达式与 OKFDDs 数据结构

2.1　MPRM 表达式

n 个输入变量的逻辑函数 f: $B^n \to B$ 有 3^n 个不同的极性，对应 3^n 个不同的 MPRM 表达式。极性 P 下 MPRM 表达式可表示为

$$f^P(x_{n-1}, x_{n-2}, \cdots, x_0) = \oplus\sum_{i=0}^{2^n-1} b_i \pi_i \tag{1}$$

其中 $\oplus\Sigma$ 为异或运算；i 为下标，其二进制数表示形式为 $i_{n-1}\ i_{n-2}\ \cdots\ i_0$；$b_i$ 为 MPRM 系数；P 为极性，其三进制数表示形式为 $p_{n-1}\ p_{n-2}\ \cdots\ p_0$；$\pi_i$ 为与项，可用符号表示为 $\pi_i = \dot{x}_{n-1}\dot{x}_{n-2}\cdots\dot{x}_0$。

MPRM 表达式中变量既可以原变量或反变量两者之一出现，也可以原变量和反变量两者同时出现，其出现形式与极性 P 和下标 i 有关[6]。表 1 为 π_i 中 $\dot{x}_k$ 与极性 p_k 和下标 i_k 的关系，$0 \leqslant k \leqslant n-1$，其中 x_k 表示 $\dot{x}_k$ 的原变量，$\overline{x}_k$ 表示 $\dot{x}_k$ 的反变量：当 p_k=0 时，$\dot{x}_k$ 以原变量出现；当 p_k=1 时，$\dot{x}_k$ 以反变量出现；当 p_k=2 时，$\dot{x}_k$ 的原变量和反变量同时出现。

2.2　OKFDDs 数据结构

OKFDDs 数据结构是一种直接固定非循环图形[9]，可表示为 G=(V,E)，其中 V 为节点，E 为连线上字符。节点可分为非终端和终端节点两种类型，非终端节点 v 有两个子继承节点 low(v)和 high(v)，而终端节点 v 与“0”或“1”相连，$v \in V$，节点之间使用连线连接。

表 1　$\dot{x}_k$ 查找表

p_k	i_k	$\dot{x}_k$
0	0	1
0	1	x_k
1	0	1
1	1	$\overline{x}_k$
2	0	$\overline{x}_k$
2	1	x_k

n 个输入变量的逻辑函数 f: $B^n \to B$，OKFDDs 中每个输入变量有以下 3 种展开类型[10]，变量可任选取其中之一进行分解：

$$f = f_0 \oplus x_i f_2 \quad \text{positive Davio(pD)} \tag{2}$$

$$f = f_1 \oplus \overline{x}_i f_2 \quad \text{negative Davio(nD)} \tag{3}$$

$$f = \overline{x}_i f_0 \oplus x_i f_1 \quad \text{Shannon(S)} \tag{4}$$

其中 f_0 和 f_1 分别为 x_i=0 和 x_i=1 逻辑函数 f 的子函数，且 $f_2 = f_0 \oplus f_1$，$0 \leqslant i \leqslant n-1$。OKFDDs 的输入变量的展开顺序和展开类型都是固定的，n 个输入变量的展开顺序为 $x_{n-1} \to \cdots \to x_1 \to x_0$，$x_i$ 在第 n–1–i 级进行展开；x_i 的展开类型为 d_i，$d_i \in$ {pD,nD,S}，n 个输入变量的展开类型组成的列表为 DTL (Decomposition Type List)，用符号表示为 D=(d_{n-1}, d_{n-2},$\cdots$,d_0)。

根据上述 OKFDDs 展开规律可知，MPRM 表达式与 OKFDDs 数据结构有如下对应关系：DTL D 对应极性 P，且 d_i=pD(nD, S)与 p_i=0(1, 2)相对应；终端节点 v_j 对应系数 d_j，$0 \leqslant j \leqslant 2^n - 1$，$v_j$ 为第 n 级从左到右数第 j 个节点；字符串 e_j 对应与项 π_i，e_j 为第 n 级从左到右数第 j 个节点上溯至顶端完整连线上字符连续与运算得到。

例 1　任一 3 个输入变量的逻辑函数 f(x_2, x_1, x_0)，D=(S, pD, nD)。根据 OKFDDs 展开规律此逻辑函数按 $x_2 \to x_1 \to x_0$ 顺序逐个展开，则其 OKFDDs 数据结构如图 1 所示。由对应关系可得 MPRM 表达式极性 P=19；b_0=f_{001}, b_1=f_{002},$\cdots$,b_7=f_{122}；π_0=$\overline{x}_2$, $\cdots$, π_7=$x_2x_1\overline{x}_0$。

3　基于 OKFDDs 的 Reed-Muller 混合极性转换

3.1　多输出逻辑函数的混合极性转换

n 个输入变量和 m 个输出变量的逻辑函数{f_k: $B^n \to B$|$0 \leqslant k \leqslant m-1$}用最小项表达式表示为

$$f_k(x_{n-1}, x_{n-2}, \cdots, x_0) = \sum_{i=0}^{2^n-1} a_{i,k} m_i \tag{5}$$

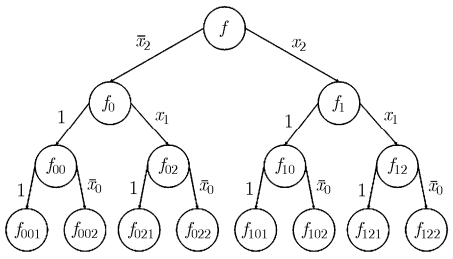

图 1 f (x_2,x_1,x_0)的 OKFDDs

其中 f_k 表示第 k 个输出变量的函数表达形式，$k=0,1,\cdots,m-1$；$a_{i,k}$ 为最小项系数，表示 m_i 是否在 f_k 中出现；最小项 m_i 用符号表示为 $\dot{x}_{n-1}\ \dot{x}_{n-2}\ \cdots\ \dot{x}_0$；下标 i 用二进制数表示形式为 $i_{n-1}\ i_{n-2}\ \cdots\ i_0$。其中当 i_c=0 时，$\dot{x}_c=\overline{x}_c$；当 i_c=1 时，$\dot{x}_c=x_c$，$0\leqslant c\leqslant n-1$。同理，此逻辑函数用 MPRM 表达式表示成

$$f_k^P(x_{n-1},x_{n-2},\cdots,x_0)=\oplus\sum_{i=0}^{2^n-1}b_{i,k}\pi_i \tag{6}$$

其中 P 表示对应表达式的极性；$b_{i,k}$ 为 MPRM 系数，$b_{i,k}$ 表示 π_i 是否在 f_k 中出现。

在单输出逻辑函数中仅用 m_i 或 π_i 就能表示最小项表达式或 MPRM 表达式，而在多输出逻辑函数中由于同一个 m_i 或 π_i 可能出现在不同输出的逻辑函数中。因此，为了表示多输出逻辑函数，定义一组信息对(t_i,o_i)，其中 t_i 为表达式与项，o_i 为与项出现情况，o_i 根据 t_i 在 f_k 中所出现的情况而定。在表示多输出最小项表示式时，$t_i=m_i$，$o_i=a_{i,m-1}\ a_{i,m-2}\ \cdots\ a_{i,0}$；在表示多输出 MPRM 表达式时，$t_i=\pi_i$，$o_i=b_{i,m-1}\ b_{i,m-2}\ \cdots\ b_{i,0}$。

令多输出逻辑函数$\{f_k:B^n\to B\ |0\leqslant k\leqslant m-1\}$中所有值为 1 的系数的下标的集合为 On-Set[11]，用数学符号表示为

$$\text{On-Set}=\{i\ |0\leqslant i\leqslant 2^n-1,\ o_i\neq 0\} \tag{7}$$

最小项表达式的 On-Set 为$\{i|0\leqslant i\leqslant 2^n-1,\ a_{i,m-1}\ a_{i,m-2}\ \cdots\ a_{i,0}\neq 0\}$，用 $\text{On-Set}_{\text{Minterms}}$ 表示，MPRM 表达式表示的 On-Set 为$\{i|0\leqslant i\leqslant 2^n-1, b_{i,m-1}\ b_{i,m-2}\ \cdots\ b_{i,0}\neq 0\}$，用 $\text{On-Set}_{\text{MPRM}}$ 表示。RM 逻辑混合极性转换即可简单地表述为：已知 $\text{On-Set}_{\text{Minterms}}$ 的条件下求解极性 P 下 $\text{On-Set}_{\text{MPRM}}$ 的过程。

3.2 混合极性转换算法

对任意多输出逻辑函数$\{f_k:B^n\to B|0\leqslant k\leqslant m-1\}$，若其按 D=(S,S,$\cdots$,S)进行 OKFDDs 展开，则对应的极性 3^n-1 下 MPRM 表达式为

$$\begin{aligned}f_k(x_{n-1},x_{n-2},\cdots,x_0)&=b_{0,k}\overline{x}_{n-1}\overline{x}_{n-2}\cdots\overline{x}_0\\&\oplus b_{1,k}\overline{x}_{n-1}\overline{x}_{n-2}\cdots x_0\oplus\cdots\oplus b_{2^n-1,k}x_{n-1}x_{n-2}\cdots x_0\end{aligned} \tag{8}$$

分析式(5)，式(8)和此 OKFDDs 数据结构的终端节点可得

$$d_{i,k}=f_{t_{n-1}t_{n-2}\cdots t_0}=a_{i,k} \tag{9}$$

其中 $t_c=i_c$，i_c 为 i 的第 c 位二进制数，$0\leqslant c\leqslant n-1$；若其按任意的 $D=d_{n-1},d_{n-2},\cdots,d_0$ 进行 OKFDDs 展开，其中 d_c=pD(nD,S)，$0\leqslant c\leqslant n-1$，假设 t_{n-1}，$t_{n-2},\cdots,t_0$ 存在 t_c=2，根据式(4)可得

$$f_{t_{n-1}\cdots t_{c+1}2t_{c-1}\cdots t_0}=f_{t_{n-1}\cdots t_{c+1}0t_{c-1}\cdots t_0}\oplus f_{t_{n-1}\cdots t_{c+1}1t_{c-1}\cdots t_0} \tag{10}$$

根据式(10)逐个将下标 2 展开为 0 和 1，结合式(9)即可得到

$$\begin{aligned}&b_{i,m-1}b_{i,m-2}\cdots b_{i,0}\\&=\oplus\sum_{t_c=2}a_{t_{n-1}t_{n-2}\cdots t_0,m-1}a_{t_{n-1}t_{n-2}\cdots t_0,m-2}\cdots a_{t_{n-1}t_{n-2}\cdots t_0,0}\end{aligned} \tag{11}$$

其中 $\oplus\sum_{t_c=2}$ 为将 $t_{n-1}t_{n-2}\cdots t_0$ 中 t_c=2 的下标逐个展开后求异或和。如例 1 中 $t_2t_1t_0$=122 可展开成 100，101,110,111，则 $b_{111,0}=a_{100,0}\oplus a_{101,0}\oplus a_{110,0}\oplus a_{111,0}$。

若已知 $\text{On-Set}_{\text{Minterms}}$，可逐个对 OKFDDs 数据结构进行 Shannon 展开，并结合式(11)求解极性 P 下 MPRM 系数，此过程即为串行混合极性转换，其伪代码如表 2 所示。该转换算法的外循环次数为 2^n，内部期望展开 $2^{\lfloor n/3\rfloor}-1$ 次，其中$\lfloor\ \rfloor$表示下取整，且需存储 2^n 个 OKFDDs 终端节点，则其算法复杂度和内存耗费分别为 $O(2^{\lfloor 4n/3\rfloor})$和 $O(2^n)$。

表 2 串行混合极性转换算法伪代码

```
/*
*    input:   On-Set_Minterms, P, n
*    output:  On-Set_MPRM
*/
Serial_Conversion(On-Set_Minterms, P, n)
{
   vertex-array[2^n] = get_OKFDDs_vertex (n, P); /*obtain the
   OKFDDs data structure */
   for(i = 0; i < 2^n; i ++)
   {
      set_temp=expend_s(vertex-array[i]);/*recursiveusing formula
      (10) */
      o_i = XOR_sum(On-Set_Minterms, set_temp); /*and using formula
      (11) */
      if(o_i != 0)
         insert_element(On-Set_MPRM, (i, o_i));
      clear_set(set_temp); /* clear a set */
   }
   return On-Set_MPRM;
}
```

令最小项系数和 MPRM 系数分别为 a_i 和 b_j，下标 i 和 j 的二进制数表示形式分别为 $i_{n-1}i_{n-2}\cdots i_0$ 和 $j_{n-1}j_{n-2}\cdots j_0$，$0\leqslant I,\ j\leqslant 2^n-1$。根据 OKFDDs 的展开规律，结合式(10)和式(11)，可推导出如下 3 种最小项系数与 MPRM 系数的下标包含关系：

(1)当 p_c=0 时，根据式(2)可知终端顶点下标中 t_c=0 与 t_c=2 对应着 j_c=0 与 j_c=1，根据式(4)可知 t_c=2 展开成 0 和 1，则 i_c=0 与 j_c=0 和 j_c=1 有包含关系，i_c=1 与 j_c=1 有包含关系；

(2)当 p_c=1 时，根据式(3)可知终端顶点下标中 t_c=1 与 t_c=2 对应着 j_c=0 与 j_c=1，根据式(4)可知 t_c=2

展开成 0 和 1，则 i_c=0 与 j_c=1 有包含关系，i_c=1 与 j_c=0 和 j_c=1 有包含关系；

(3)当 p_c=2 时，根据式(4)可知终端顶点下标中 t_c=0 与 t_c=1 对应着 j_c=0 与 j_c=1，则 i_c=0 与 j_c=0 有包含关系，i_c=1 与 j_c=1 有包含关系。

上述 i_c 与 j_c 的包含关系可用图 2 表示，其中方格内“1”表示包含关系成立，而“0”表示包含关系不成立。若包含关系成立，i_c 可展开成“1”格对应的 j_c。如例 1 中 001∈ On-Set$_{\text{Minterms}}$，则在极性 P=19 下 001 可展开成 000, 001, 010, 011。

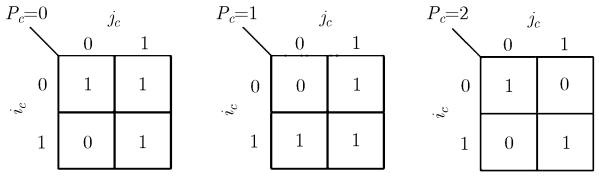

图 2 最小项表达式与 MPRM 逻辑表达式的系数下标包含关系

针对$\{f_k: B^n \to B | 0 \leqslant k \leqslant m-1\}$，定义一个长度为 2^n 数组 index-array，数组索引号 j 与 MPRM 系数 b_j 下标对应。index-array[j]用二进制数形式表示为 $v_{m-1,j}v_{m-2,j}\cdots v_{0,j}$，$v_{k,j}$ 的取值与 $b_{j,k}$ 包含最小项的个数有关。根据式(11)可知，若 $b_{j,k}$ 包含最小项的个数为奇数，则 $v_{k,j}$=1，否则 $v_{k,j}$=0。有且仅当 $v_{m-1,j}v_{m-2,j}\cdots v_{0,j}$=0 时，$j$∉ On-Set$_{\text{MPRM}}$，否则 j∈ On-Set$_{\text{MPRM}}$。

根据上述系数下标包含关系，即可避开 OKFDDs 终端节点的计算，直接从 On-Set$_{\text{Minterms}}$ 求解 On-Set$_{\text{MPRM}}$，此过程为并行混合极性转换，其伪代码如表 3 所示。其算法的外循环次数为 l，内部期望展开 $2^{\lfloor n/3 \rfloor}-1$ 次，需要耗费 l 个内存单元存储 On-Set$_{\text{Minterms}}$，其中 l 为最小项个数。则其算法复杂度和内存耗费分别为 $O(l\cdot 2^{\lfloor n/3 \rfloor})$和 $O(l)$。因此，该转换算法可提高极性转换效率，并减少算法内存消耗。

例 2　某一多输出逻辑函数$\{f_k: B^3 \to B | 0 \leqslant k \leqslant 1\}$，已知 On-Set$_{\text{Minterms}}$={(000,10),(001,11), (110, 01)}。根据极性 P=19 下的包含关系可得：(000,10)展开成(001,10)，(011,10)；(001,11)展开成(000,11)，(001,11)，(010,11)，(011,11)；(110,01)展开成(111, 01)。经过 3 次异或运算后：index-array=(11,01,11, 01,00,00,01,01)；最后将 index-array 中所有不为 0 的索引号和数值插入到 On-Set$_{\text{MPRM}}$ 中，则 On-Set$_{\text{MPRM}}$={(000,11),(001,01),(010,11),(011,01),(111, 01)}。

4　实验及其结果分析

本文所提出的算法均用 C 语言实现，通过 GNU-gcc 编译器编译，在 Linux 操作系统运行，程序运行的硬件环境为 Pentium D CPU(2.8 GHz)1 G RAM。测试电路采用 MCNC 和 ISCAS Benchmark，若所用 Benchmark 不是 PLA 格式文件，则使用电路逻辑综合与优化工具 SIS 转化成 PLA 格式。

表 3 并行混合极性转换算法伪代码

```
/*
*     input:   On-Set_Minterms, P, n
*     output:  On-Set_MPRM
*/
Parallel_Conversion(On-Set_Minterms, P)
{
    for(i = 0; i < size_of(On-Set_Minterms); i ++)
    {
        (t_i, o_i) = get_set_element(On-Set_Minterms, i);
                    /*obtain the i^th element of set */
        set_temp = expend_i(t_i, P);
                    /* expend a minterms coefficient */
        for(j = 0; j <size_of(set_temp); j ++)
        {
            index-array[j] ^ = o_i;   /* tag inclusion index */
         }
         clear_set(set_temp);
    }
    for(i = 0; i < 2^n-1; i ++)
    {
        if(index-array[i]!=0) /*determinethe MPRM coefficient*/
            insert_element(On-Set_MPRM,(i, index-array[i]));
    }
    return On-Set_MPRM;
}
```

首先，设计两组实验分别测试逻辑函数的输入变量和最小项个数对串行转换和并行转换算法的影响。第 1 组随机产生最小项数目为 50 的输入变量为 10~19 的 10 个最小项表达式，第 2 组随机产生最小项数目为 500 的输入变量为 10~19 的 10 个最小项表达式，并针对随机生成 100 个极性分别使用这两种算法对以上两组最小项表达式进行测试。试验结果如表 4 所示，其中“输入”为输入变量个数，“测试 1”与“测试 2”分别为 50 个与 500 个最小项的测试结果，“串行转换”与“并行转换”分别为串行转换和并行转换算法的平均 CPU 运行时间。由表 4 可知：随输入变量个数增加，串行转换算法运行时间呈指数倍增加，而并行转换算法运行时间呈线性增加；随最小项个数增加，串行转换算法运行时间增加不明显，但并行转换算法运行时间增加明显；总体上并行转换相比串行转换算法效率有较大幅度

表 4 对随机产生最小项表达式测试的试验结果

输入	测试 1		测试 2	
	串行转换 (s)	并行转换 (s)	串行转换 (s)	并行转换 (s)
10	0.0022	0.0001	0.0025	0.0012
11	0.0060	0.0002	0.0053	0.0017
12	0.0135	0.0002	0.0139	0.0020
13	0.0336	0.0004	0.0340	0.0024
14	0.0873	0.0005	0.0872	0.0039
15	0.2306	0.0010	0.2273	0.0044
16	0.5618	0.0018	0.5634	0.0067
17	1.4869	0.0033	1.4824	0.0109
18	4.0284	0.0062	4.0047	0.0175
19	10.7216	0.0105	10.7117	0.0266

提高。

然后，使用国际通用的 MCNC 和 ISCAS Benchmark 测试转换算法的有效性。针对 13 个中大规模 Benchmark，分别使用这两种算法对随机产生 100 个极性进行测试。实验结果如表 5 所示，其中“名称”为 Benchmark 的名称，“输入”、“输出”和“最小项”分别为 Benchmark 的输入变量、输出变量和最小项个数，“串行转换”与“并行转换”分别表示串行转换和并行转换算法的平均 CPU 运行时间。由表 5 可知：相比串行转换，对所有 13 个 Benchmark 并行转换算法的转换效率有不同程度提高，且平均提高 36.4%；当 l 与 2^n 相仿时，效率提高不多，如“apex4”，“t”等；当 $l<<2^n$ 时，效率提高明显，如“table3”，“table5”等，l 和 n 分别为最小项和输入变量个数。

表 5 对 MCNC 和 ISCAS Benchmark 测试的试验结果

名称	输入	输出	最小项	串行转换(s)	并行转换(s)
apex4	9	19	512	0.001	0.001
sao2	10	4	511	0.002	0.001
ex1010	10	10	812	0.003	0.003
misex3	14	14	12281	0.098	0.095
table3	14	14	3273	0.094	0.024
spla	16	46	42177	0.856	0.603
t481	16	1	42016	0.602	0.399
table5	17	15	28668	1.971	0.486
t	18	1	216084	5.223	4.365
pcle	19	9	326656	10.254	6.340
cm150a	21	1	1572864	81.432	57.707
cc	21	20	2097152	92.015	78.650
mux	21	1	524288	79.230	24.190

5 结束语

本文涉及到 RM 电路逻辑综合的一个重要方面——从 Boolean 逻辑表达式到 RM 逻辑表达式的混合极性转换，为 MPRM 表达式的快速获得提供了有效途径。在研究 MPRM 表达式和 OKFDDs 数据结构的基础上，将 MPRM 表达式对应的 OKFDDs 数据结构终端节点进行 Shannon 展开，然后根据最小项系数下标与 MPRM 系数下标之间的包含关系，并结合多输出函数描述方式，提出了一种直接从最小项表达式到 MPRM 表达式的并行混合极性转换算法，实现了 Boolean 逻辑表达式到 RM 逻辑表达式的快速函数转换。通过两组随机生成的测试电路和 13 个 MCNC 和 ISCAS Benchmark 测试表明：并行转换算法的复杂度随输入变量个数增加呈线性比例增加，但随最小项个数增加明显增加，当最小项较少时，相比串行转换算法，并行转换算法的转换效率提高明显，而当最小项较多时，其转换效率与串行转换的转换效率相仿。此转换算法将作为 RM 逻辑最佳混合极性搜索的基础，为进一步开发 RM 电路自动逻辑综合工具提供有效的极性转换算法。

参 考 文 献

[1] Rahaman H, Das D K, and Bhattacharya B B. Testable design of AND-EXOR logic networks with universal test sets[J]. *Computers and Electrical Engineering*, 2009, 35(5): 644–658.

[2] Pradhan S N and Chattopadhyay S. Two-level AND-XOR network synthesis with area-power trade-off[J]. *International Journal of Computer Science and Network Security*, 2008, 8(9): 368–375.

[3] Al Jassani B A, Urquhart N, and Almaini A E A. Minimization of incompletely specified mixed polarity Reed Muller functions using genetic algorithm[C]. International Conference on Signals, Circuits and Systems, Tunis, 2009: 1–6.

[4] Chaudhury S and Chattopadhyay S. Fixed polarity Reed-Muller network synthesis and its application in AND-OR /XOR-based circuit realization with area-power trade-off[J]. *IETE Journal of Research*, 2008, 54(5): 353–364.

[5] Cheng J, Chen X, and Faraj K M, *et al*.. Expansion of logical function in the or-coincidence system and the transform between it and maxterm expansion[J]. *Computers and Digital Techniques*, 2003, 150(6): 397–402.

[6] Al Jassani B A, Urquhart N, and Almaini A E A.

Manipulation and optimization techniques for Boolean logic[J]. *IET Computers and Digital Techniques*, 2010, 4(3): 227–239.

[7] Wang P and Chen X. Tabular techniques for or-coincidence logic[J]. *Journal of Electronics* (*China*), 2006, 23(2): 269–273.

[8] Dueck W, Maslov D, and Butler T, *et al*.. A method to find the best mixed-polarity Reed-Muller expression using transeunt triangle[C]. The 5th International Workshop on Applications of Reed-Muller Expansion in Circuit Design (RM), Starkville, 2001: 82–93.

[9] Becker B, Drechsler R, and Theobald M. On the expressive power of OKFDDs[J]. *Formal Methods in System Design*, 1997, 11(1): 5–17.

[10] Li H, Wang P, and Dai J. Area minimization of MPRM circuits[C]. 2009 IEEE 8th International Conference on ASIC, Changsha, 2009: 521–524.

[11] Zhang X Y, Wang L L, and Zhou X G. Efficient RM conversion algorithm for large multiple output functions[C]. The 9th International Conference on Solid-State and Integrated Circuits Technology, Beijing, 2008: 2300–2303.

汪鹏君：　男，1966 年生，博士，教授，博士生导师，研究方向为多值逻辑电路理论、低功耗集成电路设计以及数字集成电路优化等.

李　辉：　男，1985 年生，硕士生，研究方向为低功耗集成电路设计和数字集成电路优化.