

引用格式: 仲亚, 叶瑶瑶. 基于新型压缩器的乘法器设计 [J]. 微电子学与计算机, 2019, 36(3): 28-31. [ZHONG Ya, YE Yao-yao. Design of multiplier based on new compressor [J]. Microelectronics & Computer, 2019, 36(3): 28-31.]

基于新型压缩器的乘法器设计

仲 亚, 叶瑶瑶

(上海交通大学 微纳电子学系, 上海 200240)

摘 要: 为了优化乘法器的延时以及功耗, 提高乘法器的性能, 针对乘法器的部分积压缩部分设计了新型的压缩器以及新型的压缩算法, 在此基础上设计实现了新型的 18 位乘法器. 新型压缩器针对常规压缩器横向进位占用延时过大的特点, 通过优化横向进位传递方式来缩短延时; 新型的压缩算法采用二叉树结构进行并行处理, 与常规的树结构相比, 其布线简单, 结构规则. 基于 simc 40 nm 工艺, 对 18 位乘法器进行综合. 经过仿真验证, 该乘法器关键路径延时 2.43 fs, 优于同类乘法器.

关键词: Booth 算法; 压缩器; 压缩算法

中图分类号: TN492

文献标识码: A

文章编号: 1000-7180(2019)03-0028-04

Design of multiplier based on new compressor

ZHONG Ya, YE Yao-yao

(Department of Micro/Nano Electronics, Shanghai Jiaotong University, Shanghai 200240, China)

Abstract: In order to optimize the delay and power consumption of the multiplier, a new compressor and a new compression algorithm are designed for the partial product compression part of the multiplier. Based on this, a new design of 18 bit multiplier is proposed and implemented. The new compressor reduces the delay by optimizing the lateral carry path for the characteristic that the conventional compressor has a large time lag for lateral carry. The new compression algorithm uses a binary tree structure for parallel processing, and compared with the conventional tree structure, its wiring is simple and its structure is regular.

Key words: Booth algorithm; compressor; compressed algorithm

1 引言

在硬件处理器的设计中, 算术逻辑单元 (ALU) 是不可或缺的组件. 乘法器是算术逻辑单元中基本的元件. 乘法器的性能制约着算术运算的整体性能. 乘法运算通常有三个步骤: 产生部分积、压缩部分积以及产生最终结果. 其中压缩部分积是整个乘法运算的关键, 产生大部分的延时和功耗. 文献[1]基本总结了在压缩部分积阶段所用到的常规的压缩器. 文献[2]则介绍了在这个阶段通常使用的 wallance 树结构压缩算法. 本文针对常规压缩器延时过高的特点提出了一种新型的压缩器 (包括新型 4-2 压缩

器和 5-2 压缩器), 并设计了一种新型的压缩算法: 二叉树结构压缩算法. 在此基础上本文提出了一种新型的 18 位乘法器结构: 在产生部分积阶段使用改进型 Booth 编码, 在压缩部分积阶段使用了新型的压缩器和压缩算法, 仿真结果表明其性能优化十分显著.

2 新型压缩器

2.1 传统压缩器

2.1.1 传统 3-2 压缩器

如图 1 所示, 传统的 3-2 压缩器有三个输入 (X_1, X_2, X_3) 和两个输出 (Sum, Carry), 无论采用

哪种结构实现 3-2 压缩器,都需要满足下列基本公式:

$$X_1 + X_2 + X_3 = \text{Sum} + 2 \times \text{Carry} \quad (1)$$

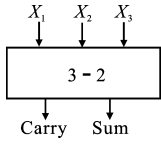


图1 3-2 压缩器

2.1.2 传统 4-2 压缩器

如图 2 所示,传统的 4-2 压缩器有四个输入 (X_1, X_2, X_3, X_4) 和两个输出 (Sum, Carry), 与 3-2 压缩器不同的是它将会产生一个中间进位 (C_{out}) 传递到下一列并且会接收上一列的进位 (C_{in}), 所以文献[4]也将 4-2 压缩器称为 5-3 计数器。同样, 无论使用何种结构来实现 4-2 压缩器, 它都需要满足下面的基本公式:

$$X_1 + X_2 + X_3 + X_4 + C_{in} = \text{Sum} + 2 \times (\text{Carry} + C_{out}) \quad (2)$$

传统结构会使用两个 3-2 压缩器(全加器)级联来实现 4-2 压缩器, 具体的示意图如图 3 所示。

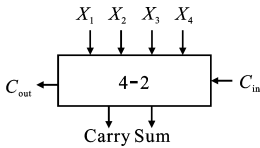


图2 4-2 压缩器

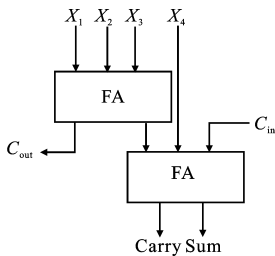


图3 使用全加器实现 4-2 压缩器

2.1.3 传统 5-2 压缩器

如图 4 所示, 5-2 压缩器有 5 个输入 (X_1, X_2, X_3, X_4, X_5) 和两个输出 (Sum, Carry), 同 4-2 压缩器类似, 5-2 压缩器产生中间进位 (C_{out1}, C_{out2}) 并且会接收低位的进位 (C_{in1}, C_{in2})。同样, 传统结构的 5-2 压缩器也会使用 3 个全加器来实现, 如图 5 所示。同样存在下面的基本公式^[3]:

$$X_1 + X_2 + X_3 + X_4 + X_5 + C_{in1} + C_{in2} = \text{Sum} + 2 \times (\text{Carry} + C_{out1} + C_{out2}) \quad (3)$$

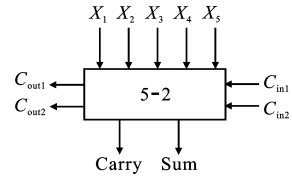


图4 5-2 压缩器

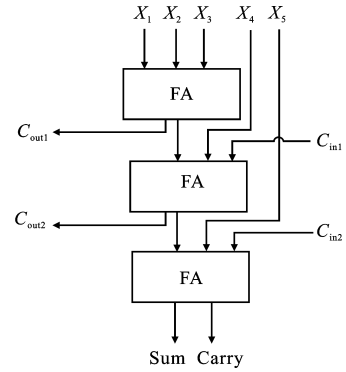


图5 使用 3 个全加器实现 5-2 压缩器

2.2 改进型压缩器

多位乘法器会生成多个部分积, 而且部分积的位数会非常高, 此时高性能的压缩器对乘法器的整体性能影响会非常大。传统的压缩器逻辑冗余高, 而且部分信号之间的逻辑依附关系会导致无法并行运行^[4]。

2.2.1 改进型 4-2 压缩器

传统的 4-2 压缩器(结构如图 3)使用两个全加器来实现, 其逻辑表达式如下所示:

$$\text{Sum} = X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus C_{in} \quad (4)$$

$$C_{out} = (X_1 \oplus X_2) \cdot X_3 + (X_1 \oplus X_2) \cdot X_4 \quad (5)$$

$$\text{Carry} = (X_1 \oplus X_2 \oplus X_3 \oplus X_4) \cdot C_{in} + (X_1 \oplus X_2 \oplus X_3 \oplus X_4) \cdot X_4 \quad (6)$$

由图 2 可知 C_{out} 会处于乘法器关键路径, 会产生过量的延时和功耗, 对这部分进行优化收益会比较大。通过观察传统 4-2 压缩器的逻辑表达式(公式(4)~(6))可以发现 C_{out} 有明显逻辑冗余, 经过优化转换, 本文提出新的 4-2 压缩器逻辑表达式:

$$\text{Sum} = X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus C_{in} \quad (7)$$

$$C_{out} = (X_1 \oplus X_2) \cdot X_3 + X_1 \cdot X_2 \quad (8)$$

$$\text{Carry} = (X_1 \oplus X_2 \oplus X_3 \oplus X_4) \cdot C_{in} + (X_1 \oplus X_2 \oplus X_3 \oplus X_4) \cdot X_4 \quad (9)$$

2.2.2 改进型 5-2 压缩器

同样, 由图 5 所示 5-2 压缩器的结构可以得出传统型 5-2 压缩器的逻辑表达式:

$$\text{Sum} = X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus X_5 \oplus C_{in1} \oplus C_{in2} \quad (10)$$

$$C_{out1} = (X_1 + X_2) \cdot X_3 + X_1 \cdot X_2 \quad (11)$$

$$C_{out2} = (X_4 \oplus X_5) \cdot C_{in2} + (X_4 \oplus X_5) \cdot X_4 \quad (12)$$

$$\begin{aligned} \text{Carry} = & (X_1 \oplus X_2 \oplus X_3) \oplus (X_4 \oplus X_5 \oplus C_{in1}) \cdot \\ & C_{in2} + (((X_1 \oplus X_2 \oplus X_3) \oplus \\ & (X_4 \oplus X_5 \oplus C_{in1}))) \cdot (X_1 \oplus X_2 \oplus X_3) \end{aligned} \quad (13)$$

图 5 所示的传统结构使用 3 个全加器来实现压缩器,第一个全加器产生 C_{out1} ,第二个全加器产生 C_{out2} ,第三个全加器产生最终的 Sum 和 Carry. 第二个全加器同时接收 X_4 和 C_{in1} 以及第一个全加器产生的输出,同样第三个全加器同时接收 X_5 和 C_{in2} 以及第二个全加器的输出,可以发现 C_{out2} 在逻辑关系上与 C_{in1} 相关,于是输出横向进位 C_{out2} 只能等待 C_{in1} 的输入,这种逻辑上的依附关系会产生额外延时,影响性能。

要想实现高性能的 5-3 压缩器,必须使得 C_{out2} 独立于 C_{in1} ^[5]. 在本文的设计中,将 X_5 取代 C_{in1} 传递进第二个全加器,重新设计 5-2 压缩器的结构,这样就打断了横向进位的联系. 需要注意的是逻辑关系将发生深刻变化,其具体的逻辑表达式也将发生变化,但是输入输出变量必须要满足通式(3),这是 5-2 压缩器基本原理的要求. 最终新型的 5-2 压缩器逻辑表达式如下:

$$\text{Sum} = X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus X_5 \oplus C_{in1} \oplus C_{in2} \quad (14)$$

$$C_{out1} = (X_1 + X_2) \cdot X_3 + X_1 \cdot X_2 \quad (15)$$

$$C_{out2} = (X_4 \oplus X_5) \cdot (X_1 \oplus X_2) + (X_4 \oplus X_5) \cdot X_4 \quad (16)$$

$$\begin{aligned} \text{Carry} = & (C_{in1} \oplus C_{in2}) \cdot (X_1 \oplus X_2 \oplus X_3 \oplus X_4 \\ & \oplus X_5) + (C_{in1} \oplus C_{in2}) \cdot C_{in1} \end{aligned} \quad (17)$$

由逻辑表达式即可看出经过优化之后, C_{out2} 与 C_{in1} 无逻辑依附关系,且 Carry 信号的逻辑表达式也得到了优化。

3 新型的压缩算法

传统的 wallance 树形压缩算法逻辑级过多,且各级之间的布线非常复杂。

以 18 位乘法器为例,其部分积压缩采用 wallance 树结构的点图如图 6 所示. 需要注意的是,图 6 仅为 wallance 树结构乘法器处理步骤中的一步,

事实上 wallance 树实现的乘法器是运用压缩器反复的“覆盖”部分积树,直至得到最终的深度为 2 的树. 图 6 已经是改进之后的 wallance 结构,仍然可以发现其每一级的“覆盖”均需要等待前一级的输出,这样越大的乘法器延时就会越高;更为重要的是 wallance 树结构实现的乘法器非常的不规则,这会使得后面高质量的版图设计任务变得十分困难。

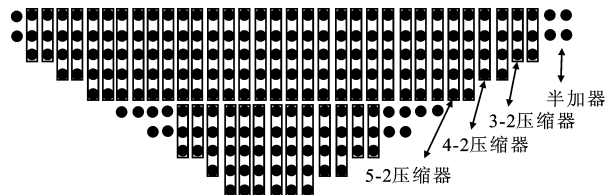


图 6 wallance 树结构

针对 wallance 乘法器的不足,本文提出新型的 二叉树压缩算法,其结构见图 7. 二叉树结构是每个结点最多有两个子树的树结构,由图 7 可见其结构非常规整。

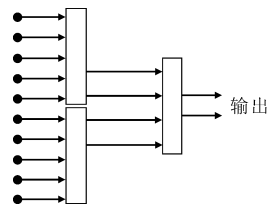


图 7 二叉树压缩算法(以 18 位乘法器为例)

图 7 仍以 18×18 乘法器为例,经过 Booth 编码总共会形成 10 个部分积(如图 8 所示). 在上文介绍的新型 4-2 压缩器和 5-2 压缩器的基础上,将整个处理过程分为两级:(1)第一级中运用 2 个 5-2 压缩器将所有的 10 个部分积转换成 4 个输出;(2)第二级中运用 4-2 压缩器将第一级的输出进一步处理,从而形成最终求和所需的两个操作数. 与 wallance 树结构相比,二叉树压缩算法逻辑级数较少,结构规则,有利于布局布线;更重要的是,第一级的两个 5-2 压缩器逻辑上相互独立,并行运行,可以很大程度上提高乘法器的整体速度。

4 18 位乘法器

在上述理论的基础上,本文将探讨使用新型压缩器以及新型压缩算法实现新型的 18 位乘法器,并在仿真结果中比较各种实现方案的优劣。

4.1 实现方案

大小不同的部分积阵列对乘法器设计并不合

适,因此经常使用的是改进型 Booth 编码.改进型的 Booth 编码通过使用负的部分积来代替倍数大的部分积,进而可以减小硬件开销.乘数三位一组并相互重叠一位,按照表 1 编码来形成一个部分积,进入编码的输入位是两个当前位和一个来自相邻低位组的最高位,所得到的部分积的数目是乘数宽度的一半,编码方式如表 1 所示.

表 1 改进型 Booth 编码

乘数位	部分积
000	0
001	+被乘数
010	+被乘数
011	+2×被乘数
100	− 2×被乘数
101	− 被乘数
110	− 被乘数
111	0

按照表 1 可以获得 18 位乘法器的部分积,其点图如图 8,图中的每个点表示部分积的每一位.

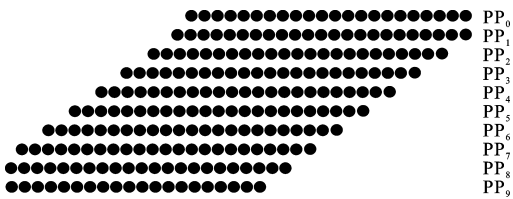


图 8 18 位乘法器部分积点图

在改进型 Booth 编码过程中,由于会出现负的部分积并且会参与运算,所以需要将负的部分积转换成补码参与运算.本文这里的处理是将部分积取反,额外的 1 将会加到下一行的最低位,来形成负的部分积的补码;此外,符号位的拓展也需要在此阶段进行考虑,这样就造成了图 8 中的部分积点图比常规的的部分积点图更加复杂.在获得部分积点图的基础上,使用本文提出的新型压缩器以及压缩算法便可求得乘法器最终的结果.

4.2 仿真结果

基于 SMIC 55 nm 工艺库,使用 vcs2016 进行功能上的仿真,使用 Synopsys 综合工具 Design Compiler 对本文的新型压缩器以及新型的 18 位乘法器进行综合.其中,表 2 列出了新型 4-2 压缩器与传统 4-2 压缩器性能上的比较;表 3 列出了新型 5-2 压缩器和传统 5-2 压缩器的比较;表 4 则展示了新型的 18 位乘法器与常规 wallance 树形乘法器的比较.仿真均以 18 位乘法器设计为例,以延时和功耗

为参考指标.

表 2 4-2 压缩器比较

压缩器	延时/ns	功耗/mw
传统 4-2 压缩器	0.70	0.18
新型 4-2 压缩器	0.64	0.17

表 3 5-2 压缩器比较

压缩器	延时/ns	功耗/mw
传统 5-2 压缩器	0.72	0.27
新型 5-2 压缩器	0.67	0.24

表 4 18 位乘法器比较

乘法器	延时/ns	功耗/mw
wallance 树形乘法器	2.8	5.2
新型乘法器	2.5	4.8

5 结束语

经过综合和实验,文中提出的新型 4-2 压缩器和新型 5-2 压缩器性能优化明显,延时以及功耗均有优化;而运用了新型二叉树压缩算法实现的新型乘法器则性能提升更为明显:关键路径的延时减少约 11%;总的功耗减少约 8%.值得注意的是,本文提出的新型压缩器和压缩算法仅仅以 18 位乘法器为参考进行性能评估,相信在大数乘法运算中,其性能的提升会更加明显,这也是下一步的工作目标.

参考文献:

[1] VEERAMACHANENI S, KRISHNA K M, AVI-NASH L, et al. Novel architectures for high-speed and low-power 3-2, 4-2 and 5-2 compressors[C]//20th International Conference on VLSI Design held jointly with 6th International Conference on Embedded Systems (VLSID' 07). Bangalore, 2007: 324-329.

[2] NAJAFI A, TIMARCHI S. High-speed energy-efficient 5:2 compressor[C]//2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). Opatija, 2014: 80-84.

[3] ABHILASH R, DUBEY S, CHINNAAIAH M C. Asic design of low power VLSI architecture for different multiplier algorithms using compressors[C]//2016 11th International Conference on Industrial and Information Systems (ICIIS). Roorkee, 2016: 387-392.

常好. 另外, EOMMIWO 的求解精度没有受到维度变化的影响, 而且求解的标准差都很低, 从这两方面看, 相对于基本的 IWO, EOMMIWO 的鲁棒性得到了很大提高. 图 3 是三个算法在低维下求解 8 个函数的平均收敛曲线图. 由于精英反向学习提高了 EOMMIWO 算法在整个进化过程中的勘探能力, 使得算法在求解精度上一直高于 IWO 和 DE; 阶段性变异在算法前期, 旨在提高个体多样性, 使 EOMMIWO 算法在进化前期就能快速向最优解收敛, EOMMIWO 算法的收敛速度明显高于 IWO 和 DE, 求解精度也远远高于他们.

5 结束语

本文针对 IWO 算法易早熟, 求解精度不高, 收敛速度慢的缺陷, 提出了基于精英反向学习的阶段性变异杂草算法 (EOMMIWO). EOMMIWO 对基本杂草算法进行了三个方面的改进: 第一, 提出了自适应标准差的策略, 将固定的初始标准差和终止标准差转化为根据问题的搜索区域而变化的值, 充分利用问题本身的条件在父个体周围产生子个体, 提高了搜索的精准度. 第二, 在杂草进化过程中让精英个体反向学习, 提高了算法的勘探能力和求解精度, 并且提出了浮动反向点的概念, 提升了反向学习的效果. 第三, 在算法进化的不同阶段, 对杂草群采用了不同变异策略, 不但增强了群体多样性和算法的全局寻优能力, 同时提高了算法的收敛速度. 通过 8 个 Benchmark 函数的测试表明, 该算法大大提高了 IWO 算法的求解精度和收敛速度.

参考文献:

[1] MEHRABIAN A R, LUCAS C. A novel numerical optimization algorithm inspired from weed colonization [J]. Ecological Informatics, 2006, 1(3):355-366.

[2] DORNELLES R, PAIM G, SILVEIRA B, et al. A power-efficient 4-2 adder compressor topology[C]// 2017 15th IEEE International New Circuits and Systems Conference (NEWCAS). Strasbourg, 2017: 281-284.

[5] PISHVAIE A, JABERIPUR G, JAHANIAN A. Re-designed CMOS (4; 2) compressor for fast binary multipliers[J]. Canadian Journal of Electrical and

[2] POUYA A R, SOLIMANPUR M, REZAEI M J. Solving multi-objective portfolio optimization problem using invasive weed optimization[J]. Swarm & Evolutionary Computation, 2016(28):42-57.

[3] 曹磊, 叶春明. 杂草-蚁群算法在应急管理中的应用[J]. 微电子学与计算机, 2016, 33(9):150-154.

[4] BARHAGHTALAB M H, MEIGOLI V, ZAVVAR M. Optimal fuzzy controller based on chaotic invasive weed optimization for damping power system oscillation[J]. Smart Science, 2017 (2) :1-16.

[5] BASAK A, MAITY D D. A differential invasive weed optimization algorithm for improved global numerical optimization[J]. Applied Mathematics and Computation, 2013, 219(12):6645-6668.

[6] TIZHOOSH H R. Opposition-based learning: a new scheme for machine intelligence[C]//Proceedings of IEEE Computational Intelligence for Modelling, Control and Automation Vienna, Austria, 2005:695-701.

[7] 李森, 任晓娜. 采用差分进化策略的入侵杂草改进算法及函数优化应用[J]. 计算机应用与软件, 2014, 31(6):285-259.

[8] 贺毅朝, 王熙照. 差分演化的收敛性分析与算法改进[J]. 软件学报, 2010, 21(5): 875-885.

[9] 陈欢, 周永权, 赵光伟. 基于混沌序列的多种群入侵杂草算法[J]. 计算机应用, 2012, 32(7):1958-1961.

作者简介:

吴聪聪 女, (1975-), 讲师. 研究方向为智能计算、信息检索、机器学习. E-mail: hebwucongcong@126.com.

贺毅朝 男, (1969-), 教授. 研究方向为进化算法理论与应用、算法设计与分析、计算复杂性理论与群测试理论.

赵建立 男, (1976-), 讲师. 研究方向为机器学习、智能计算.

李 宁 女, (1977-), 硕士, 副教授. 研究方向为智能计算、机器学习.

(上接第 31 页)

Computer Engineering, 2013, 36(3): 111-115.

作者简介:

仲 亚 男, (1994 -), 硕士研究生. 研究方向为超大规模集成电路设计、运算电路设计. E-mail: zhongya0614@qq.com.

叶瑶瑶 女, (1986-), 博士, 讲师. 研究方向为系统级芯片设计