黄瑞钧,刘杰, 刘红海,等. 基于改进压缩器的新型 Wallace 树乘法器设计[J]. 智能计算机与应用,2025,15(9):101-106.
DOI:10. 20169/j. issn. 2095-2163. 250916

基于改进压缩器的新型 Wallace 树乘法器设计

黄瑞钧, 刘 杰, 刘红海, 唐学峰, 洪 军, 李靖宇
(湖州师范学院 信息工程学院, 浙江 湖州 313000)

摘 要: 为了降低延时和功耗,提高处理器整体性能,优化乘法器关键路径成为乘法器设计的一个重要方法。首先,通过精减压缩器逻辑表达式中的逻辑冗余项,减少电路复杂度,缩短电路关键路径,达到降低压缩器延时的目的。然后,利用 Booth 算法,结合所提的改进型压缩器,采用 Wallace 树结构对 32 位乘法器进行重新设计,把关键路径延时减少到 11 个异或门。最后,基于 SIMC180 nm 工艺对所设计的乘法器进行仿真验证,实验结果表明所提设计方案能有效提高计算速度,降低功耗,优于现有的乘法器方案。

关键词: 乘法器; 压缩器; Booth 编码; Wallace 树
中图分类号: TP332 **文献标志码**: A **文章编号**: 2095-2163(2025)09-0101-06

Design of new Wallace tree multiplier based on improved compressor

HUANG Ruijun, LIU Jie, LIU Honghai, TANG Xuefeng, HONG Jun, LI Jingyu
(School of Information Engineering, Huzhou University, Huzhou 313000, Zhejiang, China)

Abstract: In order to reduce delay and power consumption and improve the overall performance of the processor, optimizing the critical path of the multiplier becomes an important option in the design of the multiplier. Firstly, by reducing the logical redundant items in the logical expression of the compressor, the complexity of the circuit is reduced, the critical path of the circuit is shortened, and the delay of the compressor is reduced. Then, using Booth algorithm, combined with the proposed improved compressor, the Wallace tree structure is used to redesign the 32-bit multiplier. Simultaneously analyzing the designed multiplier, its critical path delay has been reduced to 11 XOR gates. Finally, based on the SIMC180 nm process, the multiplier is simulated and verified. The experimental results show that the proposed design can effectively improve the calculation speed and reduce power consumption, which is better than the existing multiplier scheme.

Key words: multiplier; compressor; Booth code; Wallace tree

0 引 言

乘法器是处理器(CPU)的重要组成部分,其运算速度会严重影响 CPU 的运算速度[1],因此为了提高 CPU 的整体性能,必须对乘法器结构进行优化。

考虑到乘法运算可分为 3 个步骤:部分积生成、部分积压缩和最终结果相加[2],其中部分积生成和部分积压缩对其性能影响最大。在部分积生成方面,有 2 种主要方法。一种是由与门阵列生成部分积;另一种是由补码移位与取反操作生成部分积。研究可知,后者中最常采用的方案是使用 Booth 算法。该算法可以减少部分积生成。

在部分积压缩方面,主要有 2 种实现方法。一种为阵列乘法器,核心思想是对部分积进行移位,实现最低位对齐,并将其累加。另一种是树形乘法器,主要有 Dadda 树、Wallace 树和平衡延时树等,其中 Wallace 树结构最具优势。前一种方法的结构简单,但是关键路径长,延迟高,运算速度慢。而后一种方法的关键路径延时低,运算速度快,但结构略微复杂。因而在当前乘法器设计中,后一种方法中的 Wallace 树结构成为研究和设计的首选。例如,文献[3]采用 Wallace 树结构,舍弃简单与或门逻辑结构,使用特殊设计的选择电路,提高了部分积计算速度,但电路设计较为复杂。文献[4]对 Wallace 树结

基金项目: 湖州市公益重点项目(2019GZ10); 浙江省重点实验室项目(2020E10017)。
作者简介: 黄瑞钧(1998—),男,硕士研究生,主要研究方向:计算机体系结构,IC 测试,电路设计。
通信作者: 刘 杰(1970—),男,教授,主要研究方向:计算机体系结构,IC 测试,电路设计。Email:liujie@ zjhu. edu. cn。
收稿日期: 2024-01-03

构中的 4-2 压缩器进行改进,在进位产生速度上取得了一定效果,但是在关键路径上并未进行优化,因而在运算速度上未见明显提升。文献[5]在 Wallace 树结构上又提出创新,利用压缩器将 9 个部分积压缩至 2 个部分积,但其整体的运算效率并不高,总共需要 5 级压缩。文献[6]提出了一种由 1 个半加器和 3 个全加器构成的 6 输入 3 输出的高阶压缩器,该压缩器的关键路径延时降低到 4 XOR,但电路结构较为复杂。文献[7]改进了由 2 个 3-2 压缩器组成的传统 4-2 压缩器,减少了 4-2 压缩器异或延时,但所设计的 Wallace 树压缩结构不对称,并且需要 4 级压缩。文献[8]中设计的 Wallace 树结构对称,但第一级压缩由 3 个异或延时的 4-2 压缩器和 4 个异或延时的 5-2 压缩器组成,在延时设计上并未达到效益最大化。

根据上述分析,本文对 4-2 压缩器和 5-2 压缩器进行了改进,并基于 Booth 算法提出一种新型 Wallace 树乘法结构。相比于当前文献给出的乘法器,本文所提乘法器具有明显性能优势。

1 压缩器的改进

1.1 新型压缩器设计

文献[7]提到了一种传统 4-2 压缩器,其结构如图 1 所示。该压缩器有 5 个输入信号 X_1、X_2、X_3、X_4 和 C_{in},以及 3 个输出信号 Sum、Carry 和 C_{out},其中 $X_i(i \in (1 \sim 4))$ 是部分积输入信号,C_{in} 是上一级部分积压缩的进位输入信号,Sum 是本位求和信号,Carry 是本级纵向进位输出,C_{out} 是本级横向进位输出。该压缩器由 2 个全加器组成,先对信号 X_4、X_3 和 X_2 进行压缩,产生的第一级求和信号再与信号 X_1、C_{in} 进行全加运算,由此生成第二级求和信号 Sum 和 Carry。该压缩器的逻辑表达式如下[9]:

$$\mathrm{Sum} = X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus C_{in} \tag{1}$$

$$\mathrm{Carry} = (X_4 \oplus X_3 \oplus X_2)X_1 + (X_4 \oplus X_3 \oplus X_2)C_{in} + X_1 C_{in} \tag{2}$$

$$C_{out} = X_4 X_3 + X_4 X_2 + X_3 X_2 \tag{3}$$

由式(1)~式(3)可知,在该压缩器的整个关键路径上,数据经过每一个全加器会占用 2 个 XOR 延时,总共消耗 4 个 XOR 延时。

对式(2)和式(3)进行转换,转换为[10]:

$$\mathrm{Carry} = X_4(\overline{X_1 \oplus X_2 \oplus X_3 \oplus X_4}) + C_{in}(X_1 \oplus X_2 \oplus X_3 \oplus X_4) \tag{4}$$

$$C_{out} = (X_4 \oplus X_3)X_2 + (\overline{X_4 \oplus X_3})X_4 \tag{5}$$

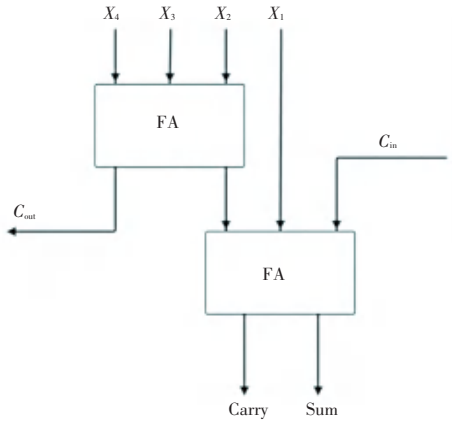

图 1 传统 4-2 压缩器

Fig. 1 Traditional 4-2 compressor

由式(1)、式(4)和式(5)重新绘制压缩器原理图,见图 2。由图 2 分析可知,该压缩器关键路径占有 3 个 XOR 门延时,相比于图 1 的传统 4-2 压缩器,减少了 1 个 XOR 门延时,并且还降低了硬件开销。这就是改进后的 4-2 压缩器。

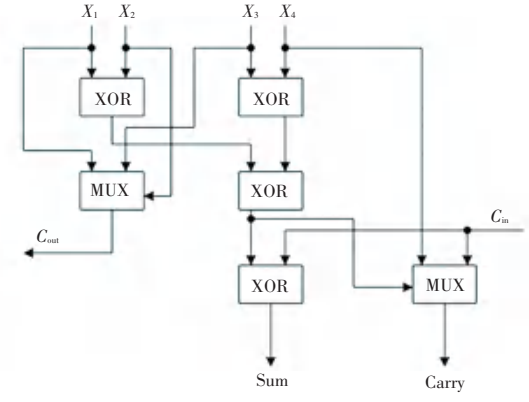

图 2 改进的 4-2 压缩器

Fig. 2 Improved 4-2 compressor

传统 5-2 压缩器的结构如图 3 所示[8]。其中,$X_1 \sim X_5$ 表示部分积输入信号,C_{in1} 和 C_{in2} 表示上一级部分积压缩的进位输入信号,C_{out1} 和 C_{out2} 表示本级部分积压缩产生的横向进位输出,求和信号 Sum 与进位 Carry 表示部分积压缩的纵向输出。图 3 的逻辑表达式分别如下[11]:

$$\mathrm{Sum} = X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus X_5 \oplus C_{in1} \oplus C_{in2} \tag{6}$$

$$C_{out1} = (X_1 \oplus X_2) \cdot X_3 + (\overline{X_1 \oplus X_2}) \cdot X_4 \tag{7}$$

$$C_{out2} = (X_3 \oplus X_4) \cdot C_{in1} + (\overline{X_3 \oplus X_4}) \cdot X_4 \tag{8}$$

$$\mathrm{Carry} = (X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus X_5 \oplus C_{in1}) \cdot X_5 + (\overline{X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus X_5 \oplus C_{in1}}) \cdot C_{in2} \tag{9}$$

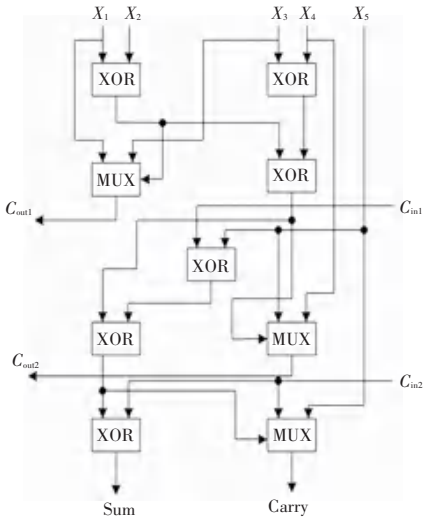

图 3　传统的 5-2 压缩器结构图[8]

Fig. 3　Structural diagram of a traditional 5-2 compressor[8]

由图 3 看到，由于 C_{out2} 必须等待 C_{in1} 的输入，因此低位进位 C_{in1} 在逻辑上直接影响横向输出 C_{out2}，这就会产生额外的延时；纵向输出 Carry 需要 X_1 和 X_2 经过 4 级 XOR 与 1 级 MUX 运算产生，求和信号 Sum 需要 X_1 和 X_2 经过 5 级异或门运算产生。Sum 处于压缩器最长路径末端，如果以一个 XOR 延时为一个单位，那么图 3 中计算出 Sum 的关键路径将占用 5 个 XOR 门延时。根据以上分析，本文对式(6)~式(9)进行改进，提出一种新型的 5-2 压缩器，对应结构如图 4 所示。改进后的逻辑表达式见如下[12]：

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

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

$$C_{out2} = (X_3 \oplus X_4) \cdot (X_1 \oplus X_2) + (\overline{X_4 \oplus X_5}) \cdot X_4 \quad (12)$$

$$\mathrm{Carry} = (X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus X_5 \oplus C_{in1}) \cdot C_{in2} + (\overline{X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus X_5 \oplus C_{in1}}) \cdot (X_4 \oplus X_5) \quad (13)$$

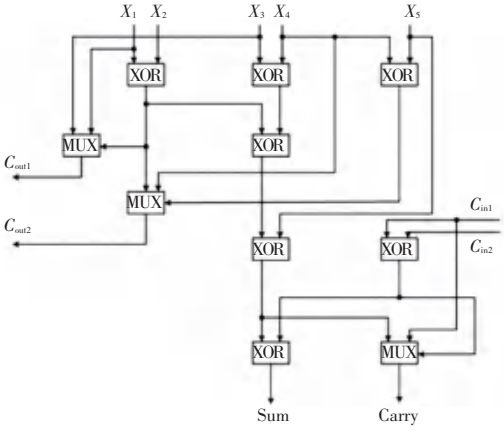

图 4　改进的 5-2 压缩器

Fig. 4　Improved 5-2 compressor

由图 4 分析可知，该 5-2 压缩器关键路径占有 4 个 XOR 门延时，相比于图 3 的 5-2 压缩器，减少了 1 个 XOR 门延时，并且同样也减少了硬件开销。这就是改进后的 5-2 压缩器。

1.2　新型压缩器仿真分析

基于 SMIC180 nm 工艺，采用 Synopsys 的 Design Compile 工具对本文所提新型压缩器进行综合验证分析。表 1 列出本文和文献[7]所提 4-2 压缩器的性能比较，表 2 列出本文和文献[8]所提 5-2 压缩器的性能比较。由表 1 和表 2 结果可以看出，在关键路径延时和功耗方面，本文所提的压缩器比文献[7]和文献[8]所提压缩器都有明显的提升。

表 1　4-2 压缩器性能对比

Table 1　Comparison of 4-2 compressor performance

参数	功耗/mW	关键路径延时/ns
文献[7]中 4-2 压缩器	0.18	0.71
本文所提 4-2 压缩器	0.17	0.65

表 2　5-2 压缩器性能对比

Table 2　Comparison of 5-2 compressor performance

参数	功耗/mW	关键路径延时/ns
文献[8]中 5-2 压缩器	0.28	0.72
本文所提 5-2 压缩器	0.24	0.66

2　乘法器设计

本文所提乘法器是 32 位数补码乘法，主要包括：部分积生成，部分积压缩和超前进位加法器。所提乘法器结构如图 5 所示。

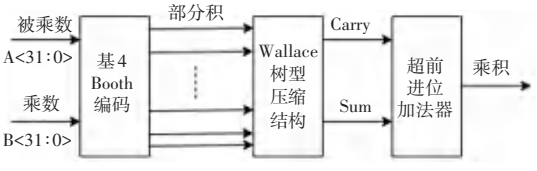

图 5　乘法器结构图

Fig. 5　Structure diagram of the multiplier

为了得到有利于进行树形压缩的部分积阵列，本文运用基 4 Booth 算法，将 2 个 32 位二进制数据送入部分积产生模块，形成一个由 22 个部分积组成的阵列，此后由 Wallace 树结构再对部分积阵列进行压缩。

2.1　由基 4 Booth 算法生成部分积

为解决有符号的乘法运算中的符号修正难题，Booth 于上世纪中叶提出了一种乘数编码方法，称之为 Booth 算法[13]。当前研究较广泛的是基 4 Booth 算法和基 8 Booth 算法，前者的编码公式具体如下：

$$P = A \times \sum_{t=0}^{n/2-1} ((-2B_{[2t+1]} + B_{[2t]} + B_{[2t-1]}) \times 2^{2t}) \quad (14)$$

其中,A 表示被乘数;B 表示乘数;P 表示乘法运算结果。

由式(14)可知,基4 Booth 算法在计算 n 比特数相乘时(n 为偶数),会产生 $n/2$ 个部分积。而基 8 Booth 算法在计算 n 比特数相乘时,会生成 $n/3$ 个部分积[14]。尽管基 8 产生的部分积比基 4 产生的部分积更少,但是对被乘数 A 的操作中产生了 $+3A$ 和 $-3A$,导致部分积生成电路更加复杂。综上所述,本文选择基 4 Booth 算法来生成部分积。

在采用基4 Booth 算法生成部分积的过程中,必须遵循一定的生成规则[15](见表 3):当乘数的连续 3 位数采用基 4 生成部分积时,输入位不同,部分积不同,式(14)中"+0"、"+1"、"-1"、"+2"、"-2"分别对应了部分积操作 PPn"+0"、"A"、"$-A$"、"$+2A$"、"$-2A$"。

表 3 基 4 Booth 编码

Table 3 Base 4 Booth encoding

B_{i+1}	B_i	B_{i-1}	$-2\times(B_{i+1})+B_i+B_{i-1}$	PPn
0	0	0	+0	0
0	0	1	+1	$1\times A$
0	1	0	+1	$1\times A$
0	1	1	+2	$2\times A$
1	0	0	-2	$-2\times A$
1	0	1	-1	$-1\times A$
1	1	0	-1	$-1\times A$
1	1	1	-0	0

2.2 Wallace 树结构设计

在采用基4 Booth 编码对部分积压缩后,一般再采用 Wallace 树结构对所剩部分积进行再压缩,直到压缩成 2 个数[16]。由基 4 Booth 编码产生的 22 个部分积,如果完全采用 3-2 压缩器[4]对部分积相加,需要 7 级压缩才能将 22 个部分积压缩成 2 个部分积,共占用 14 个 XOR 门延时;如果完全采用 4-2 压缩器[5]对部分积进行相加,需要 4 级压缩才能将 22 个部分积压缩到 2 个部分积,共占用 12 个 XOR 门延时;文献[7]采用的是 3-2 压缩器和 4-2 压缩器相混合的 Wallace 树结构,经过 4 级压缩,共占用了 12 个 XOR 门延时。上述 3 种结构都消耗 40 个 XOR 门和 20 个 MUX 硬件资源。

针对 Wallace 树结构关键路径延时长和消耗硬件资源多的问题[17],本文基于改进 4-2 压缩器和 5-2 压缩器提出了一种新的实现方案,见图 6。该结构采用 3 级压缩,即:第一级包含 2 个 5-2 压缩器和 2 个 6-3 压缩器,第二级包含 2 个 5-2 压缩器,第三级包含一个 4-2 压缩器。该结构首先将经由基 4 Booth 算法获得的 22 个部分积送入第一级压缩,压缩成 10 个部分积,占用了 4 个 XOR 门延时;接着经过第二级压缩后被压缩成 4 个部分积,占用 4 个 XOR 门延时;最后经过第三级压缩获得 2 组输出信号,占用了 3 个 XOR 门延时。这样,参见图 6,把 22 部分积压缩成 2 个部分积、仅占用 11 个 XOR 延时,而且压缩结构对称。

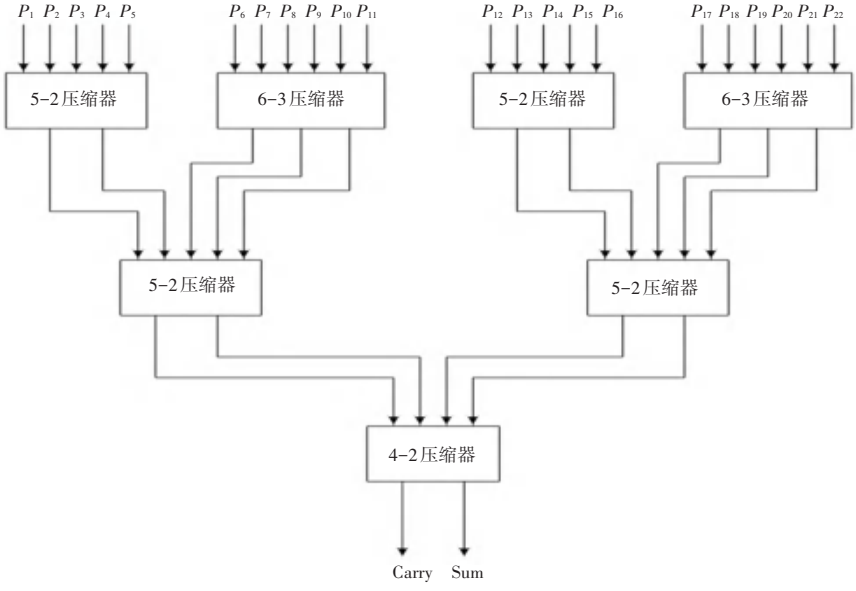

图 6 新型的 Wallace 树压缩结构

Fig. 6 New Wallace tree compression structure

文献[8]采用 4-2 压缩器和 5-2 压缩器相混合的 Wallace 树结构，此结构将 18 个部分积进行压缩，共占用了 12 个 XOR 门延时。本文采用改进的 4-2 压缩器替代传统 4-2 压缩器重新对文献[8]所提的 18 个部分积进行压缩，由于改进的 4-2 压缩器关键路径延时只有 3 个 XOR 门延时，所以改进后的 Wallace 树结构仅需要 10 个 XOR 门延时。相比于文献[8]，关键路径延时减少了 2 个 XOR 门延时。表 4 是上述采用不同的压缩器设计的压缩结构所压缩部分积个数、消耗的资源和关键路径延时的综合比较。

表 4　压缩树结构的比较

Table 4　Comparison of compressed tree structures

树形结构	压缩部分积个数	硬件消耗/(XOR、MUX)	关键延时/XOR
完全采用 3-2 压缩器[4]	22	40、20	14
完全采用 4-2 压缩器[5]	22	40、20	12
采用 3-2 与 4-2 压缩器混合[7]	22	40、20	12
本设计Ⅰ	22	34、17	11
采用 4-2 与 5-2 压缩器[8]	18	32、16	12
本设计Ⅱ	18	32、16	10

由表 4 可见，本文所提结构比完全使用 3-2 压缩器的结构减少了 3 个 XOR 门延时，比完全使用 4-2 压缩器的结构和文献[7]结构减少了 1 个 XOR 门延时，在整体评估上比这 3 种方法都少消耗 6 个 XOR 和 3 个 MUX 硬件资源。综上所述，本文所提结构与文献[7]中的 Wallace 树结构和完全使用 3-2 压缩器或 4-2 压缩器构成的 22 组部分积压缩结构相比，无论在关键路径延时、还是硬件资源消耗上表现都更加优异。

3　实验结果与分析

本文基于 SMIC180 nm 工艺，使用 Modelsim 和 Vivado 对所提乘法器进行实验。

通过编写测试激励文件来随机产生 32 位测试数据，并将其输入到本文所提的乘法器中，可得到一系列仿真结果[18]。图 7 是在一系列仿真结果中截取一对输入数据后的乘法结果仿真图。由图 7 可见，该乘法运算是正确的。

图 7　功能仿真图

Fig. 7　Functional simulation diagram

表 5 是所提乘法器与不同文献中乘法器的性能对比。由表 5 可见，在压缩 18 个部分积的结构上，改进后的乘法器比文献[8]所设计的乘法器运算速度提升 5.63%，面积缩小了 9.6%；另外，本文所提乘法器关键路径延时为 2.41 ns，功耗为 5.14 mW，压缩比为 22-2，比文献[7]所设计的乘法器功耗降低了 9.66%，速度提升 5.86%，面积缩小了 24.6%。因此，对压缩器的改进以及对 Wallace 树压缩结构的优化，能有效提高乘法器的运算速度，并且降低乘法器的实际功耗[19]。

表 5　乘法器性能比较

Table 5　Comparison of multiplier performance

乘法器	延时/ns	功耗/mW	压缩比	面积/mm^2
文献[8]	2.49	5.41	18-2	0.154 6
改进文献[8]	2.35	5.40	18-2	0.139 7
文献[7]	2.56	5.69	22-2	0.162 9
所提设计	2.41	5.14	22-2	0.122 7

4　结束语

本文通过对压缩器逻辑冗余的优化，删减非必

要的逻辑依附关系[20],对 4-2 压缩器和 5-2 压缩器进行了改进,减少了压缩器的延迟与功耗,并基于这种改进的压缩器构建了 Wallace 树压缩结构,增加压缩部分积位数,设计了改进型乘法器,有效提高了计算速度和压缩效率。

参考文献

[1] 李飞雄,蒋林. 一种结构新颖的流水线 Booth 乘法器设计[J]. 电子科技,2013,26(8):46-48.

[2] MALATHI L, BHARATHI A, JAYANTHI A N. RDO-WT: Optimised Wallace tree multiplier based FIR filter for signal processing applications[J]. International Journal of Electronics, 2022,109(10):1759-1780.

[3] ALIOTO M. Ultra-low power VLSI circuit design demystified and explained: A tutorial[J]. IEEE Transactions on Circuits and Systems I: Regular Papers,2012,59(1):3-29.

[4] CHO K, LEE K, CHUNG J. Design of low-error fixed width modified booth multiplier[J]. IEEE Transactions on Very Large Scale Integration(VLSI) Systems,2004,12(5):522-531.

[5] 姚上上,沈立. 基于混合压缩结构的新型浮点乘法器设计[J]. 微电子学与计算机,2021,38(9):74-78.

[6] 吴美琪,赵宏亮,刘兴辉,等. 一种基于改进基 4 Booth 算法和 Wallace 树结构的乘法器设计[J]. 电子设计工程,2019,27(16):145-150.

[7] 盛勇侠,梁华国,肖远,等. 基于新型压缩树的近似 Booth 乘法器[J]. 微电子学,2022,52(3):425-430.

[8] ZHANG Yongqiang, CHEN Xiaoyue, HE Cong. Energy-efficient multipliers using imprecise compressors for image multiplication [J]. International Journal of Circuit Theory and Applications, 2022, 50(11): 3875-3890.

[9] AYOUB S ,NABIOLLAH S, MAHMOOD R, et al. An efficient counter-based Wallace-tree multiplier with a hybrid full adder core for image blending [J]. Frontiers of Information Technology Electronic Engineering,2022,23(6):950-965.

[10] LIU Hao, WANG Mingjiang, YAO Longxin, et al. A Piecewise linear mitchell algorithm - based approximate multiplier [J]. Electronics,2022,11(12):1913.

[11] ZHANG Neng, QIN Qiao, YUAN Hang, et al. NTTU:An area-efficient low-power NTT-uncoupled architecture for NTT-based multiplication[J]. IEEE Transactions on Computers, 2020, 69(4):520-533.

[12] NOBUTAKA K, KAZUYOSHI T. An RSFQ flexible-precision multiplier utilizing bit-level processing[J]. Journal of Physics: Conference Series,2021,1975(1): 012025.

[13] NANDHINI V, SAMBATH K. Implementation of normal urdhva tiryakbhayam multiplier in VLSI[J]. International Journal of Performability Engineering,2021,17(6):511-518.

[14] RENITA J, EDNA N, ASOKAN N. Implementation and performance analysis of elliptic curve cryptography using an efficient multiplier[J]. Journal of Semiconductor Technology and Science,2022,22(2):53-60.

[15] RAFIQ A, CHAUDHRY M S . Design of an Improved low-power and high-speed Booth multiplier[J]. Circuits, Systems, and Signal Processing,2021,40:5500-5532.

[16] PADMANABHAN B, RAUNAQ N, MASKELL D L. Approximate array multipliers[J]. Electronics,2021,10(5):630.

[17] 操天,刘伟强,朱玉莹. 面向可容错计算的近似 Booth 乘法器设计[J]. 微电子学与计算机,2018,35(7):67-71.

[18] AMPOLU M, SWARUP K S. A survey on applications of alternating direction method of multipliers in smart power grids [J]. Renewable and Sustainable Energy Reviews, 2021, 152: 111687.

[19] KONG Tianqi, LI Shuguo. Design and analysis of approximate 4-2 compressors for high-accuracy multipliers [J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2021,29(10):1771-1781.

[20] FANG Bao, LIANG Huaguo, XU Dawen, et al. Approximate multipliers based on a novel unbiased approximate 4-2 compressor [J]. Integration,2021,81:17-24.