

文章编号:1004-9037(2013)03-0376-06

小数乘法器的低功耗设计与实现

袁 博 刘红侠

(西安电子科技大学宽禁带半导体材料与器件国家重点实验室,西安,710071)

摘要:提出一种针对小数乘法器的低功耗设计算法,其优化指标为综合后小数乘法器内部寄存中间运算结果的寄存器位宽,解决了目前低功耗设计中算法自身逻辑单元被引入系统从而降低系统优化效果的问题。该算法能够在不降低系统工作效率、不损失系统运算精度、不增加额外逻辑单元的条件下,大幅降低系统功耗和面积。在使用该算法对某一射频模块进行优化后,硬件测试结果显示该射频模块对某型号FPGA的逻辑占用率相比优化前降低17.9%,寄存器总数降低30.7%,存储单元占用率降低21.5%。该算法适用于对含有大量小数乘法运算的系统进行低功耗优化。

关键词:小数乘法器;低功耗设计;数据宽度;优化逻辑

中图分类号:TN702;TN402

文献标志码:A

Low-Power Design and Application for Decimal Multiplier

Yuan Bo, Liu Hongxia

(Key Lab of Ministry of Education for Wide Band-Gap Semiconductor Devices, Xidian University, Xi'an, 710071, China)

Abstract: A low-power design methodology is presented for decimal multiplier, the methodology optimization object is the width of the adders in synthesized multiplier. The methodology resolves the problem of optimization logic joining into optimized system existed in present low-power design. It can reduce system power and area significantly without additional logic, and the system working efficiency and calculation accuracy are remained. After optimizing a radio-frequency circuit using the proposed method, FPGA test result shows that logic utilization is reduced by 17.9%, total registers number is reduced by 30.7%, and total block memory bits utilization is reduced by 21.5%. The methodology perform well in the system optimization, including the optimization of large-scale multipliers.

Key words: decimal multiplication; low-power methodology design; data width; optimization logic

引 言

随着 SoC(片上系统)设计的高速发展,如何降低系统功耗和面积成为越来越多设计者们共同关心的课题^[1]。功耗主要有 3 种:即静态功耗、动态功耗和状态跳变时电源与地间的短路功耗。其中动态功耗,即电路节点电平翻转时对负载电容的充放电功耗,是电路的主要功耗源;静态功耗即当电

路处于静态即状态保持不变时的功耗,在电路功耗中所占比例很小。因此低功耗设计的一个重要思想就是通过减少电路中冗余的状态翻转,实现降低功耗的目的^[2]。在传统数字设计中,人们往往希望通过降低系统时钟频率、减少冗余信号翻转等方法来降低系统功耗。其中降低系统时钟频率会有效降低系统功耗,但系统性能和工作效率也会随之降低;而减少冗余信号翻转虽然不会影响系统性能,但需要在系统中增加额外控制电路,这会使得系统

基金项目:国家自然科学基金(60976068)资助项目;教育部科技创新工程重大项目培育资金(708083)资助项目;教育部博士点基金(200807010010)资助项目。

收稿日期:2011-09-13;**修订日期:**2012-10-20

引入额外的功耗和面积^[3]。常见的低功耗设计有:

(1)门控时钟 其主要采用触发器的设计方法,即当触发器状态出现冗余翻转时,通过关闭时序部件的时钟,使触发器保持静态,同时以这些时序部件输出为输入信号的组合逻辑也将处于静态,可以实现降低功耗的作用。该方法的缺点是需要时序部件的时钟输入端加入控制逻辑,使其能够在部件处于冗余状态时关闭输入时钟,即在降低部件功耗的同时引入控制逻辑带来的额外功耗,影响部件低功耗优化效果。

(2)操作数隔离 其优化对象是系统中的算术、逻辑运算模块,主要方法是在系统处于冗余状态下,即不进行算术、逻辑运算的时候,使模块的所有输入保持“0”值,禁止操作数进入系统产生冗余信号翻转,该方法使系统输出结果保持静止。而当系统需要进行运算时,将模块的所有输入还原使其正常工作。该方法的缺点是需要运算模块的输入端增加控制逻辑,使其能够在处于冗余状态所有输入信号赋“0”,但同时引入了控制逻辑带来的额外功耗,影响模块低功耗优化效果。

(3)存储器分块访问 主要方法是将系统中存储器按照其内部各子模块所需容量进行分块,然后用高位地址线进行片选译码。假设某系统分配到一块128 KB的RAM,其内部两个子模块各自需要一块64 KB的RAM,这时可以选用两块64 KB的RAM和17位的地址线。其中低16位地址线直接提供给两个RAM,最高位地址线接到下面RAM的片选端CS。通过这种方法,不管从CPU出来什么样的地址,则每次只会选中一个64 KB的RAM。如果采用单块128 KB的RAM,则每次都要选中一块128 KB的RAM,众所周知,一块64 KB RAM的功耗要远小于一块128 KB RAM的功耗。该方法的缺点是需要将地址总线的位宽扩大,同时加入片选逻辑,使系统在降低功耗同时引入了总线扩宽和片选逻辑带来的额外功耗,影响系统的低功耗优化效果。

随着超大规模集成电路设计技术的进步,高性能信号处理芯片已经成为通信、电子、空间技术等领域必不可少的组成部分,因此如数字滤波器及数字信号处理器等含有大量小数乘法运算的模块也频繁地被应用于各种芯片和电路中^[3]。对于小数乘法运算而言,为了保持较高的运算精度,要求寄存中间运算结果的寄存器保留较宽的位宽,但系统功耗和面积也会随之增大;如果试图减小寄存器位宽,乘法运算的精度损失则不可避免^[4,5]。

本文的目的在于针对上述已有技术的不足,提出一种针对小数乘法运算的低功耗设计算法,该算法的优点在于:(1)采用了一种全新的设计和实现方案,使得综合后只有优化结果参与生成系统门级电路,而优化算法自身的逻辑单元不会引入到系统中。对于含有大规模乘法运算的系统,避免其内部各乘法器引入优化逻辑自身的功耗和面积累加入被优化系统,提升优化效果。(2)由于在运算过程中只需搜索乘法器系数中“1”的排列和位置并加以计算^[5],因此具有运行速度快,占用资源少,简单易行的优点。(3)该设计能够在不降低系统工作效率、不损失运算精度、不增加额外逻辑单元的条件下,大幅降低系统功耗和面积。

1 数据宽度优化

数字设计中,数据宽度 n 所能表达出的最大数被归一化为“1”,小数则被表示为所占该“归一化1”的比例,因此 n 位的小数 B 可以被整数化处理为 X ^[6]

$$X = B(2^n - 1) \quad (1)$$

例如8位的小数 $\frac{31}{255}$ 可以整数化处理为 $31/255$ $(2^8 - 1) = 31(0111111_2)$,这里 $255(11111111_2)$ 被归一化为“1”。这样小数乘法运算就可以暂时转化为整数乘法运算,只是运算结果需要转化回小数才能得到最终的小数乘法运算结果。

整数乘法运算 $A \times B$ 的计算过程如式(2)所示,其中被乘数和乘数分别为 A 和 B , B 以二进制表示为 $b_3b_2b_1b_0$

$$A(b_32^3 + b_22^2 + b_12^1 + b_02^0) = b_32^3A + b_22^2A + b_12^1A + b_02^0A \quad (2)$$

提出系数 b_0 后,式(2)可转化为

$$b_0(A + 2b_1/b_0A + 2^2b_2/b_0A + 2^3b_3/b_0A) = b_0(A + b_1/b_02(A + 2(A + b_3/b_22(A)))) \quad (3)$$

由式(3)可得,如果 b_0 为“0”,则 b_i2^iA 项也为“0”。因此对于每一项,其分母必为“1”,否则整个项均为“0”。由此可得多项式

$$b_0(A + b_12(A + b_22(A + b_32(A)))) \quad (4)$$

其等价多项式

$$b_3(A + b_22^{-1}(A + b_12^{-1}(A + b_0(A)))) \quad (5)$$

由式(5)可以看出本级加法结果的末位不参与下级加法运算,依然作为下级加法结果的末位存在。如果能够缺省本级加法结果的最末 i 位(i 等于下级加法运算中另一个加数的左移位),则可

以减少下级加法结果的位宽,而缺省掉的也只是“最小贡献位”^[7,8]。在小数乘法运算中,为了保持数据宽度,这些“最小贡献位”会在最终运算结果中被缺省掉,但不能在中间各级加法结果中缺省,否则会出现较大的误差累积从而影响运算精度和准确度。若能预先计算出各级加法结果中的“最小贡献位”进而缺省,从而降低各级加法结果位宽,而且保证最终运算结果的一致性,不损失任何运算精度和准确度^[9]。

以 8 位小数乘法 $153 \times 31/255$ 为例,依式(1)整数化处理为 153×31 ,其乘法竖式如图 1 所示,结果小数化处理并保留 8 位数据宽度为 10010.100_b 。对该乘法依式(5)进行优化,缺省各级加法结果的“最小贡献位”后的乘法竖式如图 2 所示,结果小数化处理并保留 8 位数据宽度同样为 10010.100_b 。该乘法运算转化为 4 个加法器,优化前其内部加法结果位宽分别为 9,11,12,13,优化后其内部加法结果位宽分别为 8,9,9,9。因此在乘法运算 $153 \times 31/255$ 内部共节省寄存器位宽 $(9+11+12+13)-(8+9+9+9)=10$ 位。对含有大量小数乘法运算的系统,优化效果将十分明显。

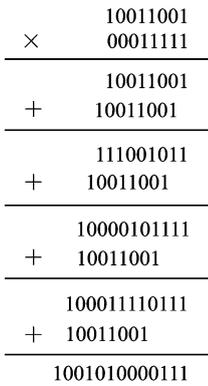


图 1 153×31 的乘法竖式

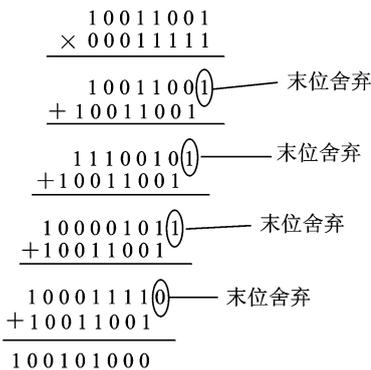


图 2 缺省优化过程

2 优化算法实现及仿真

图 3 是 n 位小数乘法 $A \times B$ 的优化算法实现框图,首先对小数系数 B 整数化处理为 X ,对整数 X 从最低位向高位搜索,逢“1”则记录其位置,以变量 `position_of_1` 表示;逢“0”则继续向高位搜索。将以上过程实现为一个 VHDL 函数 `find_multi_factor_f(x)`,并放入库中与系统分离,函数的输入 x 为乘法器的小数系数,输出为表示系数中各“1”位置的变量 `position_of_1`,记为 `find_multi_factor_f(x) = position_of_1 = (n_1, n_2, \dots, n_k)`。因此乘法器被转化为 $k-1$ 个加法器,第一个加法器是由被乘数 A 左移 n_1 位加 A 左移 n_2 位构成,并缺省加法结果的最末 n_2 位存入寄存器作为第一个中间结果 A_1 ;第 i 个加法器是由 A 左移 n_i 位加前一个中间结果 A_{i-1} 构成,并缺省加法结果的最末 n_i 位存入寄存器作为第 i 个中间结果 A_i ,以此类推直到构建出 $k-1$ 个加法器。最后需要对加法运算结果进行小数化处理,由于算法开始先对 n 位小数乘法器 $A \times B$ 中小数系数 B 进行了整数化处理,相当于将 B 左移 n 位,因此最后一级加法器的结果需要右移 n 位还原回小数后得到最终乘法结果。然而在优化过程中最大缺省位为 n_k ,即在优化过程中已经右移了 n_k 位,所以最后一级加法器的结果只需右移 $n-n_k$ 位并由 `z_o` 输出。

设计优化乘法器模块 `mtplr_multiplication`,其类属参数为 `multi_find_g` 输入端口为 `x_i` 与 n ,输出端口为 `z_o`。其中小数乘法器系数由模块类属参数 `multi_find_g` 输入,被乘数由 `x_i` 输入,数据宽度由 n 输入,最终乘法运算结果由 `z_o` 输出,图 4 所示为该优化乘法器模块的实体。

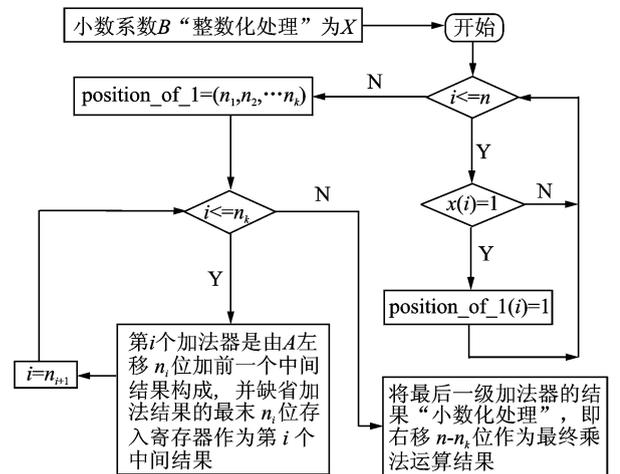


图 3 优化算法实现框图

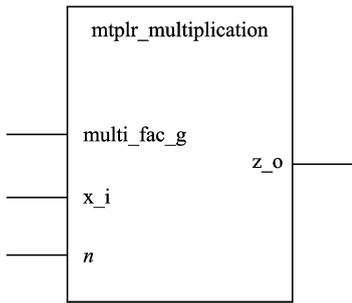


图 4 优化乘法器模块 mtplr_multiplication 的实体

在模块内部,调用函数 find_multi_factor_f(x),并使优化乘法器模块的类属参数 multi_find_g 作为该函数的输入,即 find_multi_factor_f(multi_find_g),输出结果以常数阵列 shift_bits_c 表示:constantshift_bits_c:nature_array:=find_multi_factor_f(x).position_of_1 其与被乘数构建移位加法运算并且进行优化。

在综合初期,常数 shift_bits_c 会根据乘法器系数计算得出。综合后,乘法器仅仅根据这些常数便可转化为对应的移位加结构并进行优化,而库中的运算逻辑自身不会引入乘法器,而且优化算法是缺省对下级加法无进位贡献的本级加法结果的末位,从而减少存放各级加法结果的寄存器位宽。因此在联合优化乘法器模块 mtplr_multiplication 内部,没有引入任何额外的运算和逻辑单元。

将优化乘法器模块 mtplr_multiplication 置于库中与系统设计分离。在系统设计中实例化该模块并替换掉原有各定系数乘法器,替换时只需将各实例化模块的类属参数设定为所对应乘法器系数即可完成该系统的优化。系统设计的参数和特性一旦确定,其内部各乘法器系数也将确定。而乘法器系数由类属参数传入而不以常数参数传入的原因是:常数只能从设计实体的内部得到赋值且不能再改变,而类属的值可以由设计实体外部提供,因此设计者可以从外面通过类属参量的重新设定而容易地改变该模块的内部电路结构,即在替换时只需将各乘法系数通过类属参量传入模块便可实现不同的优化乘法器,图 5 所示为替换 8 位小数乘法 $y=31/255x$ 的优化乘法器。153×31/255 的优化乘法仿真波形如图 6 所示,其中乘法系数 multi_find_g 为 0.121 569,即 31/255;被乘数 x_i 为 153;位宽 n 为 8,乘法器内部 4 个加法运算结果 A_1, A_2, A_3, A_4 分别为 11100101_b, 100001011_b,

100011110_b, 100101000_b, 最终 8 位运算结果 z_o 为 18.5,即 10010.100_b,与图 2 运算结果一致。

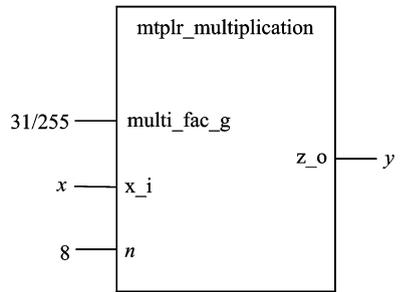


图 5 替换小数乘法 $y=31/255x$ 的优化乘法器

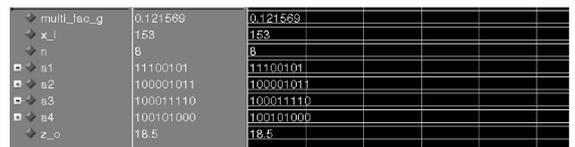


图 6 小数乘法 153×31/255 的优化仿真波形

3 优化效果验证

为了精确测试优化指标,以某含有大量数字滤波器 and 数字信号处理器的射频模块作为测试对象进行优化。测试工具为 Sequence Design 公司的 Power Theater,作为标准功耗计算工具,它可以对系统的前端 RTL 代码计算出准确的功耗和面积。表 1,2 为优化前射频模块功耗与面积;表 3,4 为该射频模块经一般优化后的功耗与面积,即使用优化算法优化各乘法器,但优化逻辑存在于各乘法器中;表 5,6 为该射频模块经本文所述设计方案优化后的功耗与面积,即模块内部所有小数乘法器均被

表 1 优化前射频模块功耗

功耗类别	静态功耗/ μ W	动态功耗/mW	总和/mW
内部寄存器	16.6	2.12	2.13
内部缓存	0.663	0.025 8	0.026 5
内部存储单元	9.49	0.536	0.545
其他内部功耗	77.6	5.2	5.28
内部功耗总和	104	7.88	7.98
时钟功耗	4.3	2.71	2.71
总功耗	109	10.6	10.7

表 2 优化前射频模块逻辑单元数和面积

类别	标准逻辑单元	非标准逻辑单元	总和
逻辑单元数	95 249	713	95 962
面积/mm ²	0.402	1.077	1.479

表 3 射频模块经一般优化设计方案优化后的功耗

功耗类别	静态功耗/ μ W	动态功耗/mW	总和/mW
内部寄存器	15.4	1.98	2.00
内部缓存	0.612	0.022 3	0.022 9
内部存储单元	9.49	0.492	0.501
其他内部功耗	66.68	4.57	4.64
内部功耗总和	92.182	7.06	7.16
时钟功耗	4.3	2.71	2.71
总功耗	96.48	9.78	9.87

表 4 射频模块经一般优化设计方案优化后的逻辑单元数和面积

类别	标准逻辑单元	非标准逻辑单元	总和
逻辑单元数	88 654	536	89 190
面积/ mm^2	0.388	1.069	1.457

表 5 射频模块经本文所述优化设计方案优化后的功耗

功耗类别	静态功耗/ μ W	动态功耗/mW	总和/mW
内部寄存器	10.1	1.55	1.56
内部缓存	0.365	0.012	0.012
内部存储单元	9.01	0.362	0.371
其他	33.68	4.01	4.04
内部功耗总和	44.093	5.934	5.98
时钟功耗	4.3	2.71	2.71
总功耗	48.365	8.64	8.69

表 6 射频模块经本文所述优化设计方案优化后的逻辑单元数和面积

类别	标准逻辑单元	非标准逻辑单元	总和
逻辑单元数	84 654	261	84 915
面积/ mm^2	0.248	1.064	1.312

优化乘法器模块 $\text{mtplr_multiplication}$ 替换,且优化逻辑不引入各乘法器。可以看出,优化前模块的功耗为 10.7 mW,逻辑单元数为 95 962,面积为 1.479 mm^2 ;经一般联合优化设计方案优化后,以上参数分别为 9.87 mW,89 190,1.457 mm^2 ,分别降低 7.76%,7.06%,1.49%;经过本文所述设计方案优化后,以上参数分别为 8.69 mW,84 915,1.312 mm^2 ,分别降低 18.79%,11.51%,11.29%,优化效果明显。

为了进一步测试本文所述优化设计方案的硬件优化效果,分别对优化前的射频模块和经一般联合优化设计方案优化后的射频模块以及本文所述优化设计方案优化过的射频模块分别进行 FPGA 测试。测试采用 Stratix IV 家族 EP4SE820F43C3 型号 FPGA 作为测试平台。

测试采用 Quartus II 作为 FPGA 的编译、综合工具,该工具来自于 ALTERA 公司,并且作为业内普遍认可的一种 FPGA 开发工具。优化前射频模块对该 FPGA 的逻辑占用率为 5.6%,生成寄存器总数为 18 175,存储单元占用率为 6.5%;经一般联合优化方案优化过的射频模块对该 FPGA 的逻辑占用率为 5.1%,生成寄存器总数为 16 805,存储单元占用率为 5.9%,分别降低 8.9%,7.5%和 9.2%;经本文所述优化方案优化过的射频模块对该 FPGA 的逻辑占用率为 4.6%,生成寄存器总数为 12 600,存储单元占用率为 5.1%,分别降低 17.9%,30.7%和 21.5%。FPGA 测试结果对比明显,证明本文所述优化设计方案解决了一般优化设计中优化逻辑自身被引入系统的问题,提升了系统优化效果。

4 结束语

本文提出了一种针对小数乘法器的低功耗算法,对射频模块的功耗分析和 FPGA 测试结果表明该算法对含有大量乘法运算的系统优化效果十分显著,而且解决了目前低功耗设计中算法自身的逻辑单元被引入系统从而降低系统优化效果的问题。

参考文献:

- [1] Graillat S, Langlois P, Louvet N. Compensated horner scheme [R]. University of Perpignan, France, 2005:10-26.
- [2] Wong A C W, Kathiresan G, Chan C K T, et al. A 1V wireless transceiver for an ultra low power SoC for biotelemetry applications[C]// 33rd European Solid State Circuits Conference. Abingdon: Toumaz, 2007:127-130.
- [3] 肖玮,涂亚庆,刘良兵,等.一种频率估计的倍频等长信号加权融合算法[J].数据采与处理,2012,27(1):74-79.
Xiao Wei, Tu Yaqing, Liu Liangbing. A long signal of a frequency estimated multiplier weighted fusion algorithm[J]. Journal of Data Acquisition and Processing, 2012,27(1):74-79.
- [4] Malvar H S, Hallapuro A, Karczewicz M. Low complexity transform and quantization in H.264/AVC [J]. IEEE Transactions on Circuits and Systems for Video Technology, 2003;13(7): 598-603.
- [5] Kang S M. Elements of low power design for integrated systems[C]// Proceedings of the 2003 International Symposium on Low Power Electronics and

- Design. Seoul, Korea:[s. n.], 2003: 205-210.
- [6] 罗柏文,万明康,于宏毅. 两种基于自适应相位补偿的FDOA估计算法[J]. 数据采集与处理, 2012, 27(1): 20-26.
- Luo Baiwen, Wan Mingkang, Yu Hongyi. Two kinds of estimation algorithm based on adaptive phase compensation FDOA [J]. Journal of Data Acquisition and Processing, 2012, 27(1): 20-26.
- [7] Samueli H. An improved search algorithm for the design of multiplierless FIR filters with power-of-two coefficients[J]. IEEE Transactions on Circuits and System, 1989, 36(7): 1044-1047.
- [8] Yoo H, Anderdon D V. Hardware-efficient distributed arithmetic architecture for high-order digit filters [C]// Proc IEEE International Conference on Acoustics, Speed and Signal Processing. [S. l.]: IEEE, 2005: 125-128.
- [9] Brickell E B. A fast modular multiplication algorithm with application to two key cryptography[C]// Proceedings of Crypto 82. New York: Plenum, 1982: 51-60.

作者简介:袁博(1982-),男,博士研究生,研究方向:大规模集成电路设计, E-mail: vias.yuan@tom.com; 刘红侠(1968-),女,博士,教授,博士生导师,研究方向:微电子学与固体电子学。