

基于 AltiVec 的指数运算的设计与实现

高建波, 张盛兵, 黄小平, 姚 涛, 卢 斌

(西北工业大学 航空微电子中心, 陕西 西安 710065)

摘 要: 基于 ALtiVec 技术, 对查找表的多项式拟合算法加以改进, 去除逻辑复杂的乘加操作, 仅通过移位、查表来实现本技术中指数的有限精度运算。同时, 通过误差计算, 在满足精度要求的情况下进一步缩小了 ROM 表的大小。

关键词: 有限精度; AltiVec; 指数运算

中图分类号: TP303

文献标识码: A

文章编号: 1000-7180(2010)09-0176-05

The Design and Implementation of Exponential Operation Based on AltiVec

GAO Jian-bo, ZHANG Sheng-bing, HUANG Xiao-ping, YAO Tao, LU Bin

(Aviation Microelectronics Center, Northwestern Polytechnical University, Xi'an 710065, China)

Abstract: This paper, based on AltiVec, improve the arithmetic of polynomial approximation based on Table _ Lookup, eliminate the complicated multiply _ add operations, and implement the finity _ precision exponential operation by shifting and Table _ Lookup. And, the ROM table can be shorten by computing the error.

Key words: finity _ precision; AltiVec; exponential operation

1 引言

AltiVec 技术是一种 SIMD 类型的处理器扩展技术, 提供一种对数据流快速处理的能力。AltiVec 技术可以显著提高通讯、多媒体, 以及其他的基于数据流并行的应用程序的处理速度。在 AltiVec 中, 除了基本的乘法、加法的基本操作, 还增加了倒数、倒数方根、指数、对数的浮点数据处理单元。文中基于 AltiVec 技术, 对其中的基本函数中的指数运算进行深入的分析, 在满足本处理器性能要求的条件下, 以最小的面积和功耗, 设计实现了指数运算^[1-3]。

2 相关研究

查找表法就是对于一个函数的所有可能参数的运算结果保存在一个表中, 直接查表得到结果^[4]。此方法电路简单, 而且速度快, 只有一个查表的时间。但是, 由于查找表太大而不利于实现, 比如 n bit

输入, n bit 输出, 表大小 $2^n \times n$ bit, 随着表示位数的增加查找表将以指数级增大, 使得电路面积无法估量。

一种比较先进的算法是把收敛速度较快的多项式拟合和查找表的方法相结合, 这种算法由 P. P. Tang 提出^[5], 原本是针对软件实现提出的, 后来用到了 Intel Pentium 系列微处理器的浮点运算上, 需要专用的乘加器, 可以达到相当高的运算精度, 并且迭代周期比较短, 电路面积一般。算法概述如下^[4]:

Step1: 选定一个整数 $L \geq 1$, 对于输入的参数 x , 减小到 r , $r \in [-\log 2/2^{L+1}, \log 2/2^{L+1}]$, 则

$$x = (m2^L + j)\log 2/2^L + r \quad (1)$$

其中 $j = 0, 1, 2, \dots, 2^L - 1$ 。

Step2: 由一个多项式计算 $\exp(r) - 1$, 记作 $p(r)$:

$$p(r) = r + a_1 r^2 + a_2 r^3 + \dots + a_n r^{n+1} \quad (2)$$

Step3: 重新构造指数计算公式:

收稿日期: 2009-07-16; 修回日期: 2009-09-28

基金项目: 国家自然科学基金项目(60773223)

$$\exp(x) = 2^m(2^{j/2^L} + 2^{j/2^L}p(r)) \quad (3)$$

具体算法执行:

- (1) 滤除特殊数据;
- (2) 对于输入的参数,进行如下操作:

首先按照下面的步骤计算 N :

$$N := \text{INTRND}(X \times \text{Inv_L})$$

$$N_2 := N \bmod 32$$

$$N_1 := N - N_2$$

其中 Inv_L 是 $32/\log 2$ 舍入到当前工作精度,注意 $N_2 \geq 0$,而不考虑 N 的符号.

INTRND 表示舍入一个数到最近的一个整数.

减得的结果用两个浮点数表示: R_1 和 R_2 , 首先 $\log 2/32$ 用两个数 L_1 和 L_2 表示:

$$R_1 := (X - N_1 \times L_1)$$

$$R_2 := -N \times L_2$$

为了完成这个步骤,把 N 分解成为 M 和 J :

$$M := N_1/32$$

$$J := N_2$$

- (3) 近似求得多项式:

$$R := R_1 + R_2$$

$$Q := R \times R \times (A_1 + R \times (\dots + R \times A_n) \dots)$$

$$P := R_1 + (R_2 + Q)$$

- (4) 重新构建

$$S := S_lead(J) + S_trail(J)$$

$$\exp := 2^M \times (S_lead(J) + S_trail(J) + S \times P))$$

在具体实现算法时,每一步中所用到的参数(临界值、系数值、 $S_lead(J)$ 、 $S_tail(J)$)都预先计算存储起来^[3].

3 算法改进

上述基于查找表的多项式拟合算法用来计算 e^x , 是全精度的计算. Altivec 技术中是有限精度的计算 2^x , 改进后的算法, 去除了复杂的乘加操作, 仅仅通过简单的移位、查表操作就可以实现 2^x 的有限精度(2^{-P})运算.

3.1 算法概述

- (1) 取 L 为大于 0 的整数, 使 x 缩小到 r , r 属于区间 $[-\frac{1}{2^{L+1}}, \frac{1}{2^{L+1}}]$, 则 x 可以表示为:

$$X = \frac{m \times 2^L + j}{2^L} + r \quad (4)$$

其中 $j = 0, 1, 2, \dots, 2^L - 1$. m 值为 x 的整数部分, $\frac{j}{2^L}$

为小数部分高几位, 而 r 则是更小的小数值.

- (2) 重新计算 2^x , 表达式为

$$2^x = 2^m \times 2^{\frac{j}{2^L}} \times 2^r \quad (5)$$

这样, 把 X 值分解为 3 部分, 各自单独计算, 最后相乘.

- (3) 由式(2)可知, 2^r 由一个多项式 $p(r)$ 来计算, 此多项式是整个算法中最复杂的部分, 其中包含了过多的乘加操作, 占据了整个操作的大部分时空, 主要用来修正低位数值. 在有限精度计算中, 可以去除这部分复杂的操作, 则重新构造后的公式如下:

$$2^x = 2^m \times 2^{\frac{j}{2^L}} \quad (6)$$

其中, m, j 值在计算过程中得到, L 为取定值, 均为整数.

3.2 具体执行

算法的具体执行步骤:

Step1: 滤出特殊数据

正无穷大, 结果为正无穷大;

负无穷大, 结果为 0;

正负 0, 结果为 1;

非数 NaN, 结果为 QNaN;

非规格化数据, 则置 0, 结果为 1 (Java 与 Not Java 模式结果相同);

最大值: 在大于等于此最大值时, 即产生上溢, 大于此值结果为正无穷大.

最小值: 在小于等于次最小值时, 小于此值结果为非规格化数:

Java 模式下置 assist exception, 结果为一不精确结果;

Not Java 模式下则为 0.

Step2: 得到 m, j 值

对输入 X 进行如下计算

$$N := \text{INTRND}(X \times 2^L)$$

$$N_2 := N \bmod 2^L$$

$$N_1 := N - N_2$$

INTRND 为取整运算, m, j 值与 N_1, N_2 相关, 得到 N_1, N_2 后

$$m = N_1/2^L$$

$$j = N_2$$

这一步的作用是得到输入的整数 m 与小数 $j/2^L$, 小数部分为一个具有 $1/2^{L+1}$ 误差的小数.

Step3: 最后结果

由图 1 可见, 在 IEEE754 标准中, 单精度规格化数据在计算机中占 24 位, 最高一位为符号位 s , 接下

来 8 位为指数位 e (带 127 的偏移码), 最后 23 位为尾数 f (带前边一位隐含整数位 1). 可以表示为如下形式^[3]:

$$(-1)^s \times 2^e \times 1.f \quad (7)$$

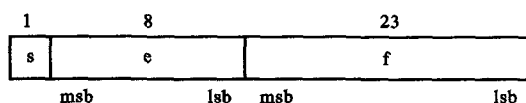


图 1 单精度浮点数格式

由式(6)可知, 计算 2^x 的公式已经简化为两部分: 2^m 和 $2^{j/2^L}$. 在此估计值中 m 为整数, $j/2^L$ 为小数, 结果如下:

符号位: 0 (2^x 大于 0);

阶码: m 值 (实际为 m 的移码);

尾数: 由 $2^{j/2^L}$ 得到 (结果为 1.XXXXX), 1 为隐含位, 小数部分则为尾数.

m 、 j 值由第 2 步简单计算可以得到; $2^{j/2^L}$ 指数运算比较复杂, 可以预先计算出来, 然后将其结果存在一个 ROM 表中, 以 j 为地址索引查表可以得到结果.

综上所述, 算法的执行如图 2 所示.

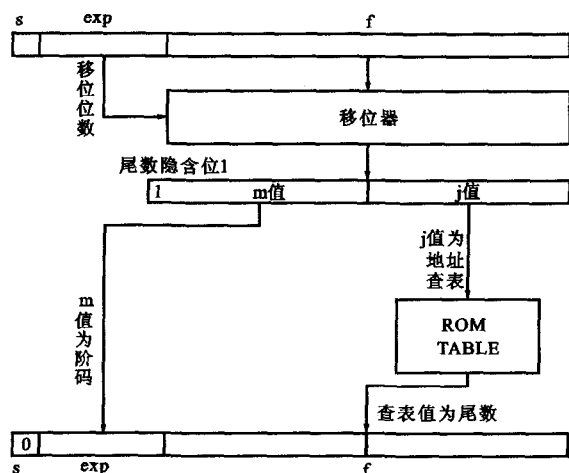


图 2 算法执行示意图

3.3 L 值的确定与误差分析

2^x 的计算已经由 $2^m \times 2^{j/2^L} \times 2^r$ 简化为 $2^m \times 2^{j/2^L}$, 这将是最后的估计值. 指数运算的绝对误差一般难以控制, 所以都采取相对误差来控制其精度. 误差公式:

$$\left| \frac{\text{estimate} - 2^x}{2^x} \right| \leq 2^{-P} \quad (8)$$

P 为精度要求的位数, 将计算公式代入得到

$$\left| \frac{2^m \times 2^{j/2^L} - 2^m \times 2^{j/2^L} \times 2^r}{2^m \times 2^{j/2^L} \times 2^r} \right| \leq 2^{-P} \quad (9)$$

约去 $2^m \times 2^{j/2^L}$ 得到

$$-2^{-P} \leq 2^{-r} - 1 \leq 2^{-P} \quad (10)$$

由不等式(10) 的得到 r 区间:

$$-\log_2(1 + 2^{-P}) \leq r \leq -\log_2(1 - 2^{-P}) \quad (11)$$

同时, 由算法分析可知

$$-2^{1/(L+1)} \leq r \leq 2^{1/(L+1)} \quad (12)$$

最后由式(11)、式(12) 得到

$$-\log_2(1 - 2^{-P}) \geq 2^{1/(L+1)} \quad (13)$$

由式(13) 可得到 L 的最小正整数

$$L \geq -\log_2(-\log_2(1 - 2^{-P})) - 1 \quad (14)$$

两者比较, L 取其中较大者:

$$L = \text{INTRND}[-\log_2(-\log_2(1 + 2^{-P})) - 1] \quad (15)$$

3.4 ROM 表的简化

单精度浮点数尾数是 23 位, 如果 ROM 表的每一项存放 23 位, 那么 ROM 表大小 $2^L \times 23\text{bit}$. 当然, 只要满足精度要求, 可以进一步缩小每一项的大小, 那么表总大小成倍数减小; 如果要提高精度, 则必须提高 L 的取值, 那么 ROM 表则会以指数级的速度增大. 在确定 P 、 L 值后, 可以通过计算来确定 ROM 表中每一项的最小 bit 位数. 式(10)中,

$$r = 2^{j/2^L} - s \quad (16)$$

把式(16) 代入式(10) 中, 得到:

$$-2^{-P} \leq 2^{j/2^L - s} - 1 \leq 2^{-P} \quad (17)$$

将式(17) 变换, 不等式中间只留下查表值 $2^{j/2^L}$, 得到:

$$2^S(1 - 2^{-P}) \leq 2^{j/2^L} \leq 2^S(1 + 2^{-P}) \quad (18)$$

ROM 表中每项最小几位可由式(18) 得到, 将 j 的每一个取值和与之对应的 S 极限取值 (一个 j 对应两个 S , 上、下限) 代入上式, 不等式做减法得到两组数据:

$$2^S(1 - 2^{-P}) - 2^{j/2^L}; 2^S(1 + 2^{-P}) - 2^{j/2^L}$$

这两组数据反映了 ROM 表中的估计值取值的最低精度要求, 最小值反映了 ROM 中满足 $-P$ 精度

表 4 综合结果

Time	Data required time	Data arrival time	slack
	2.34	2.34	0.00
Library	ports	nets	cells
Used	261	633	373
Area	Combinational area	Noncombinational area	total
/μm ²	46346.535	13348.870	59695.574

5 结束语

文中通过对 P. P. Tang 提出的基于查找表的多项式拟合算法改进,得到一种满足 ALtiVec 技术的 2^x 运算的实现方法.该设计满足了西北工业大学自主设计的“龙腾 R2”处理器 266MHz 的时钟要求,可在一拍内完成四个并行的单精度数据的指数运算, FLOPS 高达 1G,而电路面积只有 59696μm².

参考文献:

[1] 刘华平. 高性能附点除法与基本函数功能部件的研究 [D]. 北京:中国科学院,2003.
[2] 吉伟,黄巾,杨靓,等. 基于 FPGA 的 32 位浮点加法器的

设计[J]. 微电子学与计算机,2008,25(6):209-211.
[3] 谷理想,孙锋,于宗光. 一种 43 位浮点乘法器的设计[J]. 微电子学与计算机,2009,26(6):21-24.
[4] 潘洪亮. 浮点指数类超越函数的运算算法研究与硬件实现[D]. 西安:西北工业大学,2006.
[5] Ping Tak Peter Tang. Table - driven implementation of the exponential function in IEEE floating - point arithmetic [J]. Argonne National Laboratory, 1989, 15(2): 144 - 157.
[6] Ping Tak Peter Tang. Table - lookup algorithms for elementary functions and their error analysis[R]. Argonne National Laboratory. IL60439-4801:232-236.
[7] Cody W, Waite W. Software manual for the elementary functions[M]. Boston: Prentice - Hall, Englewood Cliffs, 1980.

作者简介:

高建波 男,(1982-),硕士研究生.研究方向为专用集成电路设计、微处理器体系结构.
张盛兵 男,(1968-),教授,硕士生导师.研究方向为计算机体系结构、VLSI 系统设计.
黄小平 男,(1980-),讲师.研究方向为专用集成电路设计、VLSI 系统设计.

(上接第 175 页)

[7] 叶青华,陈一辉. 一种 512Kbit 同步高速 SRAM 的设计 [J]. 固体电子学研究与进展,2004,24(3):355-358.
[8] Jan M Rabaey, Anantha chandrankasan. Digital integrated circuits - a design perspective[M]. 2 版. 北京:电子工业出版社,2004:208-209.

作者简介:

程 艳 女,(1984-),硕士研究生.研究方向为计算机体系结构及 ASIC 系统设计.
魏廷存 男,(1960-),教授,博士生导师.研究方向为数模混合信号 VLSI 设计技术以及平板显示驱动芯片技术等.
李 峰 男,(1978-),博士研究生.研究方向为片上系统与可重构计算.