文章编号: 1001-2486 (2000) 03-0016-04

分布式算术及其在 FPGA 中的实现*

吴东, 王宇红, 卢焕章

(国防科技大学 ATR 国家重点实验室, 湖南 长沙　410073)

摘　要: 介绍了分布式算术的原理以及它在基于 LUT 的 FPGA 器件上的实现, 比较了串行分布式算术和并行分布式算术在面积开销和性能方面的差异。

关键词: 分布式算术; 查找表; 现场可编程门阵列

中图分类号: TN791　　文献标识码: A

Distributed Arithmetic and its Implementation in FPGA

WU Dong, WANG Yu-hong, LU Huang-zhang

(ATR State Key Laboratory, National Univ. of Defense Technology, Changsha 410073, China)

Abstract: The principle of distributed arithmetic and its implementation in LUT-based FPGA are introduced. The difference in area cost and performance between serial distributed arithmetic and parallel distributed arithmetic is also described.

Key words: distributed arithmetic; LUT; FPGA

分布式算术是一种用 VLSI 实现变量矢量和常量矢量点积运算的有效方法, 可以应用在包括传统的 FIR 滤波器和现代的小波变换等大量数字信号处理基本运算的 VLSI 实现中。最近随着 FPGA 广泛地应用于数字信号处理领域, 在 FPGA 上实现分布式算术成为非常现实的需求。

1　分布式算术

1.1　基本分布式算术

分布式算术针对的是一类变量矢量和常量矢量的点积运算, 也就是通常所说的乘法累加运算, 这类运算是滤波和相关算法的核心部分。典型的乘法累加运算如式 (1) 所示

$$y = \sum_{k=1}^{K} A_k \cdot X_k \tag{1}$$

其中 A_k 是固定的系数, X_k 是输入的数据。假设 $|X_k| < 1$。将 X_k 用二进制补码的形式表示为

$$X_k = -b_{k0} + \sum_{n=1}^{N-1} b_{kn} 2^{-n} \tag{2}$$

其中 b_{kn} 的值是 0 或者 1, b_{k0} 是符号位, $b_{k,N-1}$ 是最低位 (LSB)。将式 (2) 代入式 (1), 有

$$y = \sum_{k=1}^{K} A_k \left[-b_{k0} + \sum_{n=1}^{N-1} b_{kn} 2^{-n} \right] \tag{3a}$$

$$= \sum_{n=1}^{N-1} \left[\sum_{k=1}^{K} A_k b_{kn} \right] 2^{-n} + \sum_{k=1}^{K} A_k (-b_{k0}) \tag{3b}$$

由于 $b_{kn}(n = 0, 1, \cdots N-1)$ 的取值为 0 或 1, 而 A_k 是常数, 所以 $\sum_{k=1}^{K} A_k b_{kn}$ 有 2^K 个可能值, 并且它的值由 b_{kn} 决定。我们可以预先计算这 2^K 个值, 存入 LUT 中, 用 b_{kn} 作为 LUT 的地址信号, 通过查找存储器的方法直接获得乘法累加的结果。这一步是分布式算术中关键的一步。

将 LUT 的输出结果送到移位-累加器, 经过 $N-1$ 个周期, 就可以得到 $\sum_{n=1}^{N-1} \left[\sum_{k=1}^{K} A_k b_{kn} \right] 2^{-n}$。

* 收稿日期: 1999-10-07
作者简介: 吴东 (1971-), 男, 博士生。

而式（3b）的另一部分：$\sum_{k=1}^{K}A_k(-b_{k0})$ 也可以预先计算得到，同样存放在 LUT 中。最后经过了 N 个周期，得到了 y，中间过程所需要的 LUT 大小为 2×2^K 个存储单位。

实际上，分布式算术改变了原理性乘法累加的运算执行顺序，首先把 K 个输入数据的第 k 位与 A_k 乘法累加，再把 $k=1\cdots K$ 所得结果移位相加，得到最终结果。其中获得性能提高的关键在于利用了 A_k 是常数的特点，把乘法累加变成了 LUT 的查询操作。

1.2 针对存储器空间的优化

基本的分布式算术需要 2×2^K 个单位的存储器空间。从上面的描述可以看出 LUT 分为两个部分，这两个部分的内容分别是 $\sum_{k=1}^{K}A_kb_{kn}$ 和 $\sum_{k=1}^{K}A_k(-b_{k0})$ 的所有可能值。它们除了正负号相反之外，绝对值完全相同。由于 LUT 的输出结果要送到移位-累加器中，所以只要在累加器中增加一个控制加/减运算的信号，就可以使得存储器空间需求缩小为 2^K 个单位。

进一步可将存储器空间缩小。对 X_k 作一个变换，令

$$X_k=\frac{1}{2}[X_k-(-X_k)] \tag{4}$$

其中

$$-X_k=-b_{k0}+\sum_{n=1}^{N-1}b_{kn}2^{-n}+2^{-(N-1)} \tag{5}$$

将式（5）和式（2）代入式（4）可以得到

$$X_k=\frac{1}{2}[-(b_{k0}-b_{k0})+\sum_{n=1}^{N-1}(b_{kn}-b_{kn})2^{-n}-2^{-(N-1)}] \tag{6}$$

令 $C_{kn}=b_{kn}-b_{kn}$，$n\neq 0$，$C_{k0}=-(b_{k0}-b_{k0})$，它们可能的取值为 ± 1，有

$$X_k=\frac{1}{2}\left[\sum_{n=1}^{N-1}C_{kn}2^{-n}-2^{-(N-1)}\right] \tag{7}$$

将式（7）代入式（1），得到

$$y=\frac{1}{2}\sum_{k=1}^{K}A_k\left[\sum_{n=1}^{N-1}C_{kn}2^{-n}-2^{-(N-1)}\right] \tag{8a}$$

$$=\sum_{n=1}^{N-1}Q(b_n)2^{-n}+2^{-(N-1)}Q(0) \tag{8b}$$

其中，$Q(b_n)=\sum_{k=1}^{K}\frac{1}{2}A_kC_{kn}$，$Q(0)=\sum_{k=1}^{K}\frac{1}{2}A_k$

在这里 $Q(b_n)$ 仍有 2^K 个可能值，但是因为 $C_{kn}=\pm 1$，如果不考虑正负号，则 $Q(b_n)$ 只有 2^{K-1} 个可能值。通过增加控制累加器加减运算的控制电路，就可以实现 LUT 大小为 2^{K-1} 个单元的分布式算术。

1.3 针对性能的优化

分布式算术优于原理性乘法累加运算之处在于性能。假设输入数据字长为 N，常矢量 A 有 K 个元素。基本的分布式算术的运算时间为 N 个周期，而原理性乘法累加运算时间为 K 个周期。只要 $K>N$，采用分布式算术就能得到优于原理性乘法累加的性能。

通过增加存储器的容量的方法，可以进一步提高性能。

将每一个输入数据 $X_k(N\text{ 位})$ 分割成为 L 个部分，其中 L 是 N 的因子。于是式（8a）中的 C_{kn} 可以表示为

$$C_{kn}=\sum_{i=0}^{L-1}C_{k,nL+i} \tag{9}$$

式（8a）变为

$$y=\left(\sum_{k=1}^{K}A_k\right)(-2^{-N})+\frac{1}{2}\sum_{k=1}^{K}\sum_{n=1}^{\frac{N}{L}-1}\sum_{i=0}^{L-1}C_{k,nL+i}2^{-(nL+i)}A_k$$

$$= 2^{-(N-L)}P_{IC} + \sum_{n=1}^{\frac{N}{L}-1} P(b_k)2^{-nL} \tag{10}$$

其中: $P(b_k) = \sum_{k=1}^{K}\sum_{i=0}^{L-1}\frac{1}{2}A_k 2^{-i}C_{k,nL+i}$,

$$P_{IC} = -2^{-L}\sum_{k=1}^{K}A_k$$

从上式看出, 在这里只需 $\frac{N}{L}$ 个周期就可以得到内积的结果, 当 $L = N$ 时, 只要一个周期就完成运算。当然性能提高 L 倍的代价是现在我们需要的存储器容量为 $\frac{1}{2}2^{KL}$ 个单位。

2 XC4000 系列 FPGA 上分布式算术的实现

2.1 设计原理

XC4000 系列 FPGA 是一种基于 LUT 的 FPGA。其内部的每个可重构逻辑块 (CLB) 含有 3 个查找表和 2 个触发器, 另外还有用于实现快速进位逻辑的专用硬件。因此 XC4000 系列 FPGA 中的 CLB 结构与分布式算术的算法结构非常吻合, 十分适合用来实现分布式算术。

下面利用分布式算术的原理在 FPGA 上设计一个简单的 FIR 滤波器。该滤波器为 16 阶, 参数 A_k 和输入数据 X_k 都为 8 位, 这里对设计作了简化, 假设数据为无符号整数, 从而省略了对符号位的处理与控制电路。使得电路呈现出规则的结构, 能更清晰地说明分布式算术 FPGA 实现的特点。

由上面对分布式算术原理的讨论可知, 分布式算术主要由 LUT 和移位-累加运算两个部分组成。

首先考虑在 FPGA 上实现 LUT, 虽然 FPGA 有丰富的 LUT 资源, 但是由于分布式算术对 LUT 容量的要求随 K 的增加而呈指数增长趋势, 采用单一的庞大的 LUT 的方案是不现实的。可以首先将一个大的 LUT 分割为几个小的 LUT, 然后将所有小 LUT 的结果相加。这种多个 LUT 的电路结构对 LUT 容量的需求将远远小于采用单个 LUT 的方案。其次, 考虑在 FPGA 上实现移位-累加运算, 由于在分布式算术中的移位操作是固定位数的移位, 可以通过抽头的方法来实现, 从而可以不增加硬件开销。在以上各步运算之间增加寄存器, 使之形成流水, 可以进一步提高运算的性能。

2.2 串行分布式算术设计

设计一个 $L = 1$ 的串行分布式算术运算。其原理图如图 1 所示。为了避免 LUT 过大, 将一个容量为 $2^{K-1} = 2^{15}$ 个单位的较大的 LUT 分解为一系列较小的 LUT 和一组寄存器、加法器。用 Xilinx Foundation 1.5 对该设计进行综合、布局、布线。得到结果, 该设计占用 68 个 CLB, 速度为 8M。

2.3 并行分布式算术设计

实现一个 $L = 2$ 的并行分布式算术算法, 它以消耗更多 CLB 资源为代价获得性能上的提高。其框图如图 2 所示。它占用 130 个 CLB, 速度为 16M。并行分布式算术算法性能在 $L = N$ 时达到最高, 称为全并行分布式算术。在我们的例子中, $N = 8$, 这时电路占用 400 个 CLB, 大约为串行分布式算术的 8 倍, 速度为 55M, 大约也是串行分布式算术的 8 倍。

3 结论

分布式算术的算法结构决定了它十分适合于用含丰富 LUT 和触发器资源的 FPGA 器件实现。大致上, 实现分布式算术的 CLB 资源开销与获得的性能成正比。在实际使用中, 可以根据特定应用的需要和约束条件, 通过在 CLB 面积资源和电路性能方面的权衡, 得到最有效的设计。

参考文献:

[1] XC4000E Data Sheet [R]. Xilinx, Inc.

[2] Gin-Kou Ma, Fred J. Taylor, Multiplier Policies For Digital Signal Processing [J]. IEEE ASSP Magazine, 1990 (1): 6-19

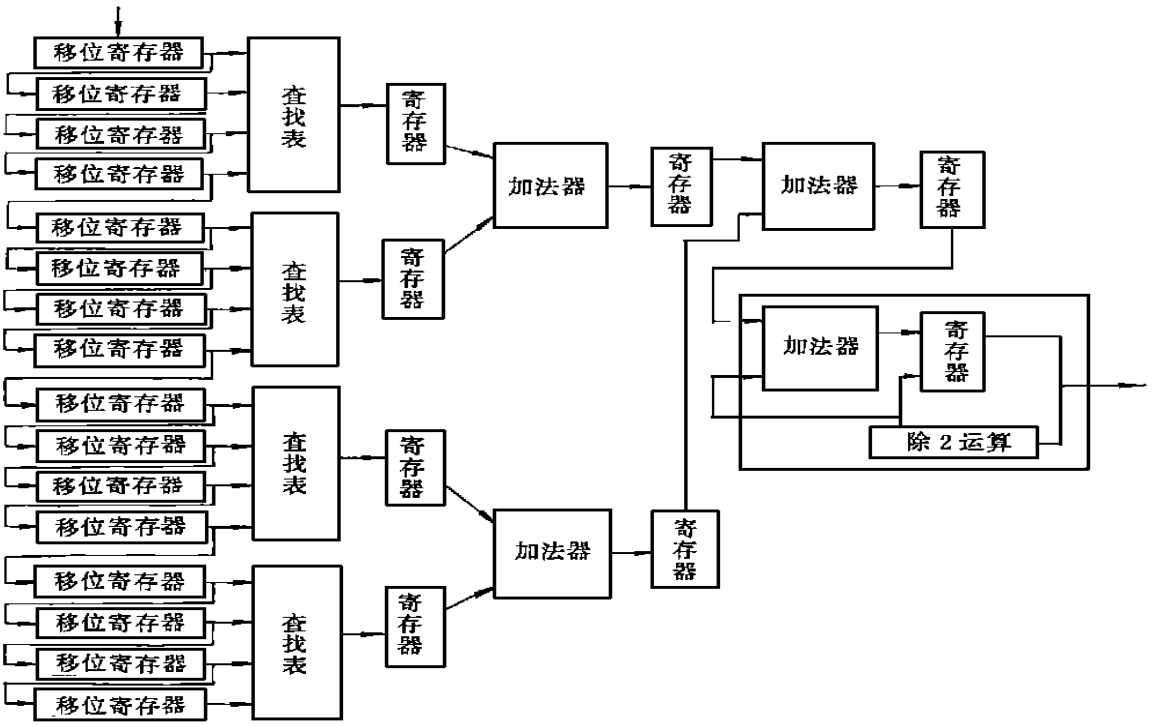

图 1　16 阶 FIR 串行分布式算术

Fig. 1　16-tap FIR SDA diagram

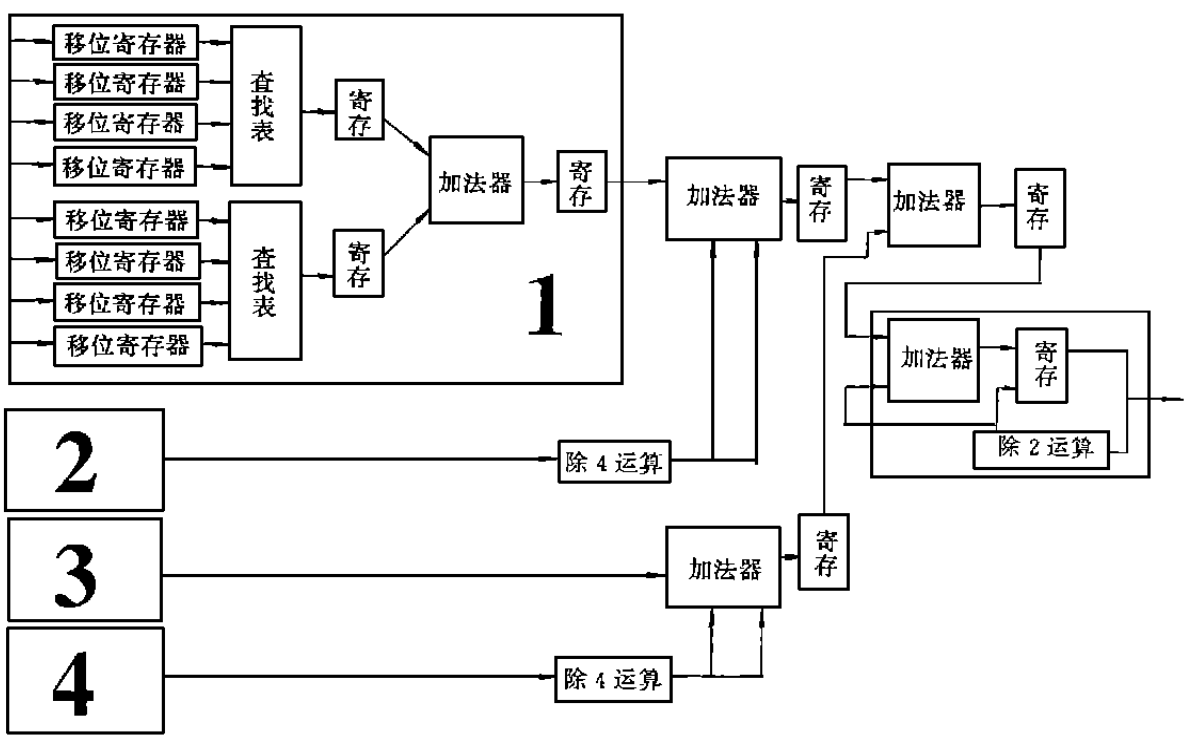

图 2　16 阶并行 FIR 分布式算术 *(L = 2)*

Fig 2　16-tap FIR PDA *(L = 2)*