第 36 卷 第 1 期 计 算 机 工 程 2010 年 1 月
Vol.36 No.1 Computer Engineering January 2010
·开发研究与设计技术· 文章编号：1000—3428(2010)01—0251—02 文献标识码：A 中图分类号：TP342.21

5 加数并行加法器及其进位接口

刘 杰[1]，易茂祥[2]
(1. 阜阳师范学院物理与电子科学学院，阜阳 236041；2. 合肥工业大学应用物理系，合肥 230009)

摘 要：传统加法器在处理多操作数累加时，必须进行多次循环相加操作。针对该问题设计 5 操作数并行加法器及其高速进位接口。电路采用多操作数并行本位相加和底层进位级联传递的方式，在一定程度上实现多操作数间的并行操作，减少相加次数。模拟结果验证了该加法器的设计合理性，证明其能缩短累加时间、提高运算效率。

关键词：加法器；超前进位加法器；进位接口

Parallel Adder with Five Addend and Its Carry Interface

LIU Jie[1], YI Mao-xiang[2]
(1. School of Physics and Electronic Science, Fuyang Normal College, Fuyang 236041;
2. Department of Applied Physics, Hefei University of Technology, Hefei 230009)

【Abstract】Traditional adder must do several times of cyclic addition operation in processing of multi-operand accumlation. Aiming at this problem, this paper designs a five addend parallel adder and its high speed carry interface. Because the circuit uses parallel own department addition of multi-operands and bottom layer carry cascade connection transmission mode, the parallel operation between multi-operands is realized and addition times are reduced. Simulation results verify the design reasonability of the adder and prove that it can shorten addition time, enhance operation efficiency.

【Key words】adder; Carry Look-Ahead Adder(CLAA); carry interface

1 概述

在计算机中，数据的算术运算以加法运算为基础。加法器是微处理器和 DSP 的重要部件，通常位于关键路径，它是决定处理器性能的重要因素之一，直接影响处理器的速度[1]。因此，高速、低耗加法器的设计一直被众多研究者所关注。随着 64 位微处理器的出现，对处理器性能的要求越来越高，对快速加法器的需求随之增加，出现了长加法器。科技的快速发展使数据处理量陡增，大量数据需要进行累加或乘除等操作，所以，多操作数成为一个研究热点[2-4]。近 30 年来，出现了一些经典两数高速加法器及其变种[5-9]，如行波进位加法器(Ripple-Carry Adder, RCA)、跳跃进位加法器(Carry-Skip Adder, CSKA)[5]、进位选择加法器(Carry-Select Adder, CSLA)[6]、超前进位加法器(Carry Look- ahead Adder, CLA)等[7-9]。在上述加法器中，通常将 CLA 用于各种处理器。而对于多操作数而言，因为要产生更多的进位并传递，所以电路设计难度增大，进位的并行产生成为多操作数加法器的关键。

本文设计以 5 个 4 位加数为基础的多操作数并行加法器，探讨了 5 个加数并行相加的实施方案，给出扩展接口电路的设计。利用本文提出的接口扩充方案，可以得到多操作数并行长加法器。对本文设计的电路和传统加法器电路进行仿真，比较它们对 5 个加数相加的计算用时，结果证明本文方案可以实现 5 个加数的并行相加，提高了运算速度。

2 5 加数并行加法器及其进位接口

2.1 工作过程

设 $A=a_3a_2a_1a_0$，$B=b_3b_2b_1b_0$，$C=c_3c_2c_1c_0$，$D=d_3d_2d_1d_0$，$E=e_3e_2e_1e_0$ 分别为 5 个加数，$S=S_6S_5S_4S_3S_2S_1S_0$ 为 5 个加数的和，其逻辑运算方法如图 1 所示，具体步骤如下：(1)利用编码求出 S'_i，C_{i1} 和 $C_{i2}(i=0,1,2,3)$；(2)利用加法器求得 S''_i 和 $C''_i(i=0,1,\cdots,4)$；(3)利用超前进位加法求得最后的本位和 $S_i(i=0,1,\cdots,6)$。因为 5 个数相加有 2 种进位，即高进位 C_{i2} 和低进位 C_{i1}，所以要构成长加法器的接口，必须把 C_{22}，C_{31}，C_{32}，C''_3 和 C_3 作为并行加法器向高位模块进位的输出接口，而进位输入接口 C_{-00}，C_{-11}，C_{-22}，C_{-33} 和 C_{-1} 作为并行加法器低位模块上传进位的接口，与上述向高位进位的接口是一一对应的。

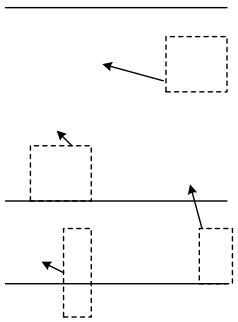

图 1 加法器的算法阵列

基金项目：安徽省高校省级自然科学研究基金资助项目(2006KJ042B)

作者简介：刘 杰(1970－)，男，博士研究生，主研方向：计算机体系结构，单片机及数字集成电路测试；易茂祥，副教授、博士研究生

收稿日期：2009-06-24 **E-mail：**liujie52@ah163.com

上述 5 操作数并行加法器和高速进位接口采用一个加法器阵列，对每次的运算数进行编码，并按权值进行排列，避免了进位串行产生和传递。进行多位扩展时，因为低权值并行加法器的进位输入信号能在高权值加法器运算过程中同步并行输入，所以保证了各权值位计算的独立性，不受进位传播的影响。因此，上述设计和扩展方案可以减少进位延迟带给整个加法器阵列的影响，从而减少进位从低位向高位传递的延迟时间，提高加法器的并行计算速度。

2.2 工作原理

2.2.1 一级进位和本位和

为了求出 S'_i, C_{i1} 和 $C_{i2}(i=0,1,2,3)$的逻辑表达式，根据 a_i, b_i, c_i, d_i, e_i 和 S'_i, C_{i1}, C_{i2} 的逻辑关系，采用正逻辑的方法进行化简，可以得到 S'_i, C_{i1} 和 C_{i2} 的表达式如下：

$$\begin{aligned}S'_i = {} & \overline{a_i}\,\overline{b_i}\,\overline{c_i}\,\overline{d_i}e_i + \overline{a_i}\,\overline{b_i}\,\overline{c_i}d_i\overline{e_i} + \overline{a_i}\,\overline{b_i}c_i\overline{d_i}\,\overline{e_i} + \overline{a_i}b_i\overline{c_i}\,\overline{d_i}\,\overline{e_i} + \\ & a_i\overline{b_i}\,\overline{c_i}\,\overline{d_i}\,\overline{e_i} + \overline{a_i}\,\overline{b_i}\,\overline{c_i}d_ie_i + \overline{a_i}\,\overline{b_i}c_i\overline{d_i}e_i + \overline{a_i}\,\overline{b_i}c_id_i\overline{e_i} + \\ & \overline{a_i}b_i\overline{c_i}\,\overline{d_i}e_i + \overline{a_i}b_i\overline{c_i}d_i\overline{e_i} + \overline{a_i}b_ic_i\overline{d_i}\,\overline{e_i} + a_i\overline{b_i}\,\overline{c_i}\,\overline{d_i}\,\overline{e_i} + \\ & \overline{a_i}b_ic_id_ie_i + a_i\overline{b_i}\,\overline{c_i}d_i\overline{e_i} + a_i\overline{b_i}c_i\overline{d_i}\,\overline{e_i} + a_ib_i\overline{c_i}\,\overline{d_i}\,\overline{e_i}\end{aligned}$$

$$\begin{aligned}C_{i1} = {} & \overline{a_i}\,\overline{c_i}d_ie_i + \overline{a_i}b_i\overline{c_i}e_i + a_i\overline{b_i}c_i\overline{d_i} + a_i\overline{b_i}\,\overline{c_i}e_i + \overline{a_i}b_i\overline{d_i}\,\overline{e_i} + \\ & b_i\overline{c_i}\,\overline{d_i}\,\overline{e_i} + a_i\overline{b_i}\,\overline{d_i}\,\overline{e_i} + \overline{a_i}\,\overline{b_i}c_i\overline{d_i} + \overline{a_i}\,\overline{c_i}\,\overline{d_i}e_i + \\ & b_i\overline{c_i}\,\overline{d_i}\,\overline{e_i} + a_i\overline{b_i}\,\overline{c_i}\,\overline{d_i}\end{aligned}$$

$$C_{i2}=b_ic_id_ie_i+a_ic_id_ie_i+a_ib_id_ie_i+a_ib_ic_ie_i+a_ib_ic_id_i$$

2.2.2 二级进位和本位和

根据图 1，得到 S''_i 和 $C''_i(i=0,1,\cdots,4)$的逻辑表达式，并进行逻辑优化，获得与或非表达式，即

$$S''_0 = S'_0 \oplus C_{-00} \oplus C_{-11} = \overline{\overline{S'_0}\,\overline{C}_{-00}\,\overline{C}_{-11} + \overline{S'_0}C_{-00}C_{-11} + S'_0\overline{C}_{-00}C_{-11} + S'_0C_{-00}\overline{C}_{-11}}$$

$$S''_1 = S'_1 \oplus C_{-22} \oplus C_{01} = \overline{\overline{S'_1}\,\overline{C}_{-22}\,\overline{C}_{-22} + \overline{S'_1}C_{-22}C_{01} + S'_1\overline{C}_{-22}C_{01} + S'_1C_{-22}\overline{C}_{01}}$$

$$S''_2 = S'_2 \oplus C_{02} \oplus C_{11} = \overline{\overline{S'_2}\,\overline{C}_{02}C_{11} + \overline{S'_2}C_{02}C_{11} + S'_2\overline{C}_{02}C_{11} + S'_2C_{02}\overline{C}_{11}}$$

$$S''_3 = S'_3 \oplus C_{12} \oplus C_{21} = \overline{\overline{S'_3}\,\overline{C}_{12}\,\overline{C}_{21} + \overline{S'_3}C_{12}C_{21} + S'_3\overline{C}_{12}C_{21} + S'_3C_{12}\overline{C}_{21}}$$

$$S''_4 = C_{22} \oplus C_{31} = \overline{C_{22}C_{31} + \overline{C_{22}}\,\overline{C_{31}}}$$

$$C''_0 = \overline{\overline{S'_0}\,\overline{C}_{-00} + \overline{S'_0}\,\overline{C}_{-11} + \overline{C}_{-00}\overline{C}_{-11}}$$

$$C''_1 = \overline{\overline{S'_1}\,\overline{C}_{-22} + \overline{S'_1}\,\overline{C}_{01} + \overline{C}_{-22}\overline{C}_{01}}$$

$$C''_2 = \overline{\overline{S'_2}\,\overline{C}_{02} + \overline{S'_2}\,\overline{C}_{11} + \overline{C}_{02}\overline{C}_{11}}$$

$$C''_3 = \overline{\overline{S'_3}\,\overline{C}_{12} + \overline{S'_3}\,\overline{C}_{21} + \overline{C}_{12}\overline{C}_{21}}$$

$$C''_4 = C_{22}C_{31}$$

2.2.3 超前进位和最终本位和

为了实现快速加法，采用超前进位方案计算，即可获得最终本位和。计算过程如下：

$P_0=S''_0 \oplus C_{-33}$, $P_1=S''_1 \oplus C''_0$, $P_2=S''_2 \oplus C''_1$, $P_3=S''_3 \oplus C''_2$, $P_4=S''_4 \oplus C''_3$, $P_5=C_{32} \oplus C''_4$

$G_0=S''_0C_{-33}$, $G_1=S''_1C''_0$, $G_2=S''_2C''_1$, $G_3=S''_3C''_2$, $G_4=S''_4C''_3$, $G_5=C_{32}C''_4$

$C_0=G_0+P_0C_{-1}$, $C_1=G_1+G_0P_1+P_0P_1C_{-1}$

$C_2=G_2+G_1P_2+G_0P_1P_2+P_0P_1P_2C_{-1}$

$C_3=G_3+G_2P_3+G_1P_2P_3+G_0P_1P_2P_3+P_0P_1P_2P_3C_{-1}$

$C_4=G_4+G_3P_4+G_2P_3P_4+G_1P_2P_3P_4+G_0P_1P_2P_3P_4+P_0P_1P_2P_3P_4C_{-1}$

$C_5=G_5+G_4P_5+G_3P_4P_5+G_2P_3P_4P_5+G_1P_2P_3P_4P_5+G_0P_1P_2P_3P_4P_5+P_0P_1P_2P_3P_4P_5C_{-1}$

$S_0=P_0 \oplus C_{-1}$, $S_1=P_1 \oplus C_0$, $S_2=P_2 \oplus C_1$, $S_3=P_3 \oplus C_2$

$S_4=P_4 \oplus C_3$, $S_5=P_5 \oplus C_4$, $S_6=C_5$

3 功能仿真与测试

通过上述分析，使用 MAX+plus Ⅱ提供的各种原理图库对提出的 5 个 4 位并行加法器进行设计。为保证电路的正确性，必须对其进行测试验证。根据测试原理对输入端进行数据设置，获得如图 2 所示的仿真波形。

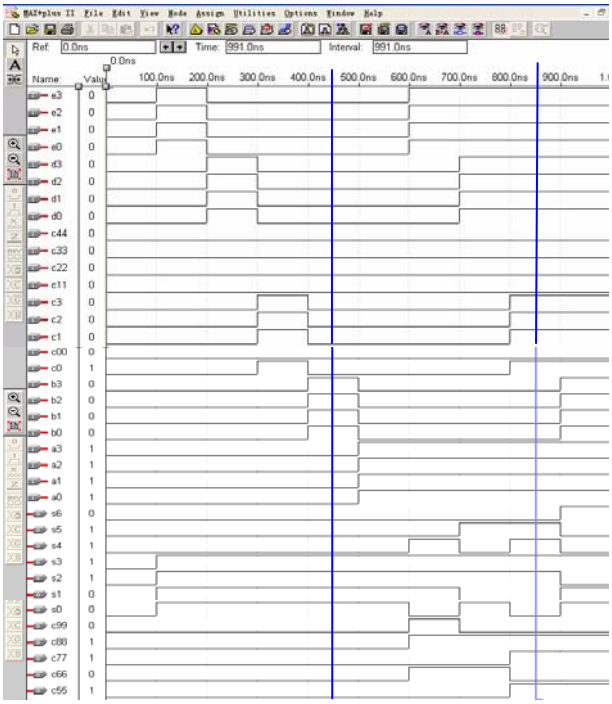

图 2 仿真波形界面

在图 2 上任意取一个脉冲区间，例如，在 400 ns~500 ns 范围内(图 2 中左边竖线)，S_0, S_1, S_2, S_3, S_4, S_5, S_6, C_{55}, C_{66}, C_{77}, C_{88}, C_{99} 的结果分别 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0。在 800 ns~900 ns 范围内，S_0, S_1, S_2, S_3, S_4, S_5, S_6, C_{55}, C_{66}, C_{77}, C_{88}, C_{99} 的结果分别为 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0，这些数据与理论值完全吻合。其他的测试数据与理论值也完全吻合，仿真结果表明该设计方案是正确的。

4 性能分析

为了说明本文提出的 5 加数并行加法器极大提高了 5 个操作数的计算速度，对其与传统超前进位加法器进行比较。

通过 MAX+plus Ⅱ软件对 2 种加法器的时延关系进行时间分析，获得的时延结果分别如图 3 和图 4 所示。由图 3 可知，超前进位加法器最快运算时间为 6.0 ns，最慢运算时间为 13.1 ns。因为超前进位加法器一次只能处理 2 个数据相加，所以当对 5 个 4 位二进制数进行运算时，需要逐个相加，即需要连续进行 4 次相加过程。由于超前进位加法器的每次运算时间相同，因此只要知道一次运算时间就可以简单地推算出实现 5 个 4 位二进制数运算所用的时间。因此，对 5 个 4 位二进制加数进行相加时，最快运算用时为 6.0 ns×4=24.0 ns，最慢运算用时为 13.1 ns×4=52.4 ns。由图 4 可知，建议的 5 加数并行加法器的最快计算用时是 7.5 ns，最慢计算用时是 37.5 ns。由数据对比可知，本文方案最快运算用时不到超前进位加法器最快运算用时的 1/3，最慢运算时间则减少了 14.9 ns。超前进位加法器对 5 个 4 位二进制加数进行相加

(下转第 259 页)