Modified Carry Skip Adder for Reducing First Block Delay

Min Cha
ECE Department
University of Texas at Austin
Austin, TX 78712

Earl E. Swartzlander, Jr.
ECE Department
University of Texas at Austin
Austin, TX 78712

***Abstract*- The carry skip adder has low complexity and moderate delays. Within each block, ripple carry is used to produce the sum bits and a carry. When the carry of the first block is generated, skipping produces the next carry with two gate delays per block. The second block cannot pass carry signals until the block generates the first carry. A modified carry skip adder has been designed for reducing the first block delays. Carry lookahead logic is used for the first carry computation. Although the modified carry skip adder uses a few extra gates because of the CLA logic, it provides better speed than the conventional carry skip adder. It certainly works well with fixed-size blocks.**

I. INTRODUCTION

Adders are very important because they play several roles in a CPU. They are used not only for addition, but also for other operations such as subtraction, multiplication, division, and address computation[2]. Arithmetic operations are the slowest among processor operations and very often the adder delay defines the maximum frequency of operation of the chip. As a consequence, a fast adder can easily increase the overall chip performance.

The simplest adder architecture is the ripple carry adder. In this adder every block computes a 1-bit sum and provides the resulting output and the carry bit for the next 1-bit adder as shown in Fig.1. For this adder the worst-case delay increases linearly with the number of bits.

Various techniques have been proposed in order to improve the speed of large adders. In every case the target is to compute the input carry more quickly for the high order bits. Many time critical applications use carry-lookahead schemes (CLA) to derive a fast but high area adder. A carry lookahead adder, believed by many people to be the fastest known adder, is large because it employs several levels of Manchester carry chains[1]-[3].

The carry skip adder provides a compromise between a ripple carry adder and a CLA adder. The carry skip adder divides the words to be added into blocks. Within each block, ripple carry is used to produce the sum bit and the carry.

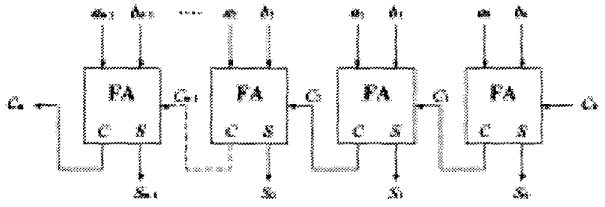

Fig. 1. Ripple carry adder.

When the carry of the first block is generated, the succeeding carries are generated with two gate delays per block[4]. Hence, the second block waits until the first ripple carry block generates a carry. So, we have examined a first block delay reduction algorithm which is the carry lookahead logic used for the first carry computation. Although the modified carry skip adder uses a few extra gates because of the CLA logic, it operates with small gate delay.

II. ADDER MODELS AND CARRY LOOKAHEAD LOGIC

In the following, full adder models and carry lookahead logic is developed for the conventional carry-skip adder and for the proposed adder. In this paper, we use two models. The full adder is the fundamental building block of most arithmetic circuits. The sum and carry outputs are described by the following equations:

$$S_i = a_i \bar{b}_i \bar{c}_i + \bar{a}_i b_i \bar{c}_i + \bar{a}_i \bar{b}_i c_i + a_i b_i c_i = a_i \oplus b_i \oplus c_i \quad (1)$$

$$c_{i+1} = \bar{a}_i b_i c_i + a_i \bar{b}_i c_i + a_i b_i \bar{c}_i + a_i b_i c_i = a_i b_i + a_i c_i + b_i c_i \quad (2)$$

Where a_i, b_i and c_i are the inputs to the ith full adder stage, s_i and c_{i+1} are the sum and carry outputs, respectively. Fig. 2 shows the 9-gate full adder. It operates with 6-gate delays from input (a_i, b_i) to sum out (s_i) and 5-gate delays from input (a_i, b_i) to carry out (s_i).

Fig. 3 shows the 8-gate modified full adder. There is no

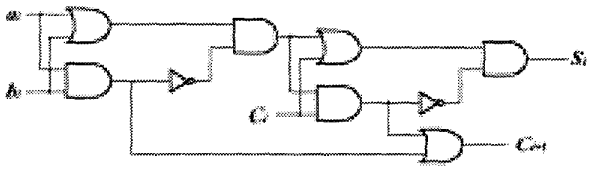

Fig. 2. The 9-gate full adder.

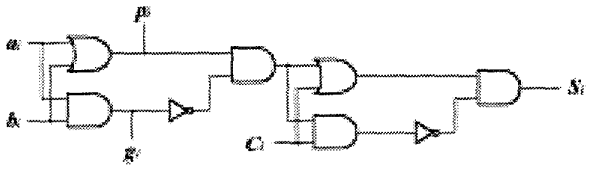

Fig. 3. The 8-gate modified full adder.

Proc. 43rd IEEE Midwest Symp. on Circuits and Systems, Lansing MI, Aug 8-11, 2000
0-7803-6475-9/00/$10.00 ©IEEE 2000

gate for carry output. It operates with 6-gate delays from input (a_i, b_i) to sum output (s_i) and 1-gate delay from input (a_i, b_i) to propagate and generate signals (p_i, g_i). It is used for carry lookahead logic. Fig. 4 shows carry lookahead logic that has 14 gates and operates with 2 gate delays from input (p_i, g_i) to carry output and block generate signal. The carry output, block generate and propagate signals are described by the following equations[4]:

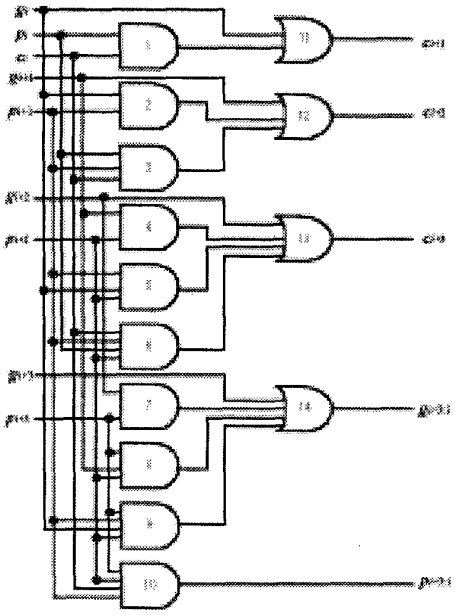

Fig. 4. Carry lookahead logic.

$$g_i = a_i b_i \tag{3}$$

$$p_i = a_i + b_i \tag{4}$$

$$c_{i+1} = g_i + p_i c_i \tag{5}$$

$$c_{i+2} = g_{i+1} + p_{i+1}c_{i+1} = g_{i+1} + p_{i+1}g_i + p_{i+1}p_i c_i \tag{6}$$

$$\begin{aligned} c_{i+3} &= g_{i+2} + p_{i+2}c_{i+2} \\ &= g_{i+2} + p_{i+2}g_{i+1} + p_{i+2}p_{i+1}g_i + p_{i+2}p_{i+1}p_i c_i \end{aligned} \tag{7}$$

$$g_{i+3:i} = g_{i+3} + p_{i+3}g_{i+2} + p_{i+3}p_{i+2}g_{i+1} + p_{i+3}p_{i+2}p_{i+1}g_i \tag{8}$$

$$p_{i+3:i} = p_{i+3}\, p_{i+2}p_{i+1}p_i \tag{9}$$

III. Carry Skip Adder and Modified Carry Skip Adder

A carry skip adder has a low area requirement and a reasonable time performance. The basic idea behind a carry skip adder is the observation that the carry propagates unchanged through a bit position with different values for the a_i and b_i bits. The block-carry-in for each block is initialized to 0. Each block computes a block-propagate that indicates whether the block-carry-in will skip to the next block. The block-carry-in bits are chained through skip logic as shown in Fig. 5. An AND gate is used to form the block propagate signal.

In the conventional carry skip adder, the first and last blocks are simple ripple carry adders, whereas the $[n/k]$-2 intermediate blocks are ripple carry adders augmented with three gate skip circuits. The delay of a carry skip adder is the

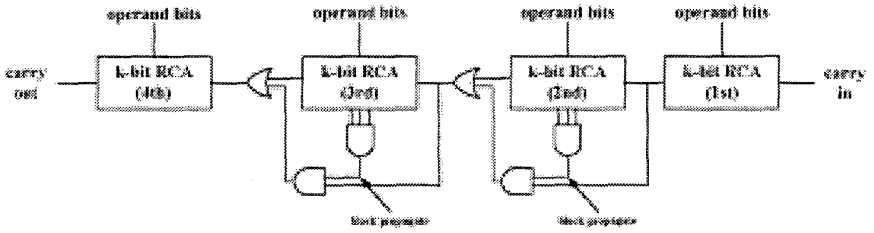

Fig. 5. Carry skip adder.

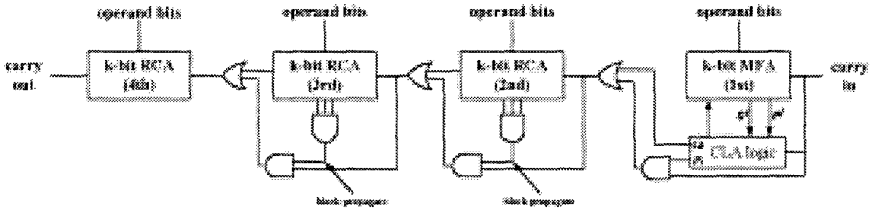

Fig. 6. Proposed carry skip adder.

sum of $2k$+3 gate delays to produce the carry in the first block, 2 gate delays through each of the intermediate blocks, and $2k$+1 gate delays to produce the most significant sum bit in the last block[4].

$$Delay_{skip} = 2k + 3 + 2(\frac{n}{k} - 2) + 2k + 1 = 4k + 2\frac{n}{k} \qquad (10)$$

If the block width is k, $Delay_{skip}$ is the total delay of the carry skip adder with a single level of k bit wide blocks. As shown in Fig. 5 and Eq. (10), $Delay_{skip}$ depends on the delay time of the first block and the number of blocks.

The modified carry skip adder as shown in Fig. 6. CLA logic is used to reduce the first block's delay. It costs a few gates more than the basic carry skip adder, but it can significantly reduce the delay of the first block. If the first block is less than 4 bits wide, it is not necessary to use all 14 gates of the CLA logic. Because 5 gates are needed for only c_{k+1} generation (k=2) and 9 gates for c_{k+1} and c_{k+2} generation (k=3), the CLA logic is not a critical factor in the complexity.

IV. Comparison with other adders

There are four cases to compare with other adders such as the ripple carry adder, the conventional carry skip adder, and the carry lookahead adder as shown in Table 1. The first case is an 8-bit addition that is divided into 2-bit blocks except for the carry lookahead adder that has a block size of 4 bits. The second case is a 16-bit addition realized with five 3-bit blocks and a 1-bit block (most significant bit). The third case is a 32-bit addition with eight 4-bit blocks. The 64-bit addition is four 5-bit blocks each for the MSB and LSB and four 6-bit blocks in the middle. Table 1 shows the delay and complexity of adders, where Δ_{sum} means delay time from input to final summation and Gates is the total complexity.

TABLE I
DELAYS AND GATES OF ADDERS

Adder size	Skip Block size		Ripple Carry	Carry Look-ahead	Carry Skip	Modified Carry Skip
8	4×2-bit	Δ_{sum}	20Δ	9Δ	16Δ	13Δ
		Gates	72	99	78	83
16	5×3-bit +1-bit	Δ_{sum}	36Δ	10Δ	22Δ	15Δ
		Gates	144	200	156	164
32	8×4-bit	Δ_{sum}	68Δ	13Δ	32Δ	25Δ
		Gates	288	401	306	318
64	8×5-bit +4×6-bit	Δ_{sum}	132Δ	14Δ	44Δ	37Δ
		Gates	576	802	606	622

TABLE 2
DELAY TIME OF ADDERS

Adder size	Ripple Carry	Carry Lookahead	Carry Skip	Modified Carry Skip
8	1.9 ns	1.0 ns	1.4 ns	1.4 ns
16	3.5 ns	1.1 ns	2.0 ns	1.8 ns
32	6.7 ns	1.4 ns	3.0 ns	2.6 ns
64	13.1 ns	1.5 ns	4.2 ns	3.7 ns

To demonstrate the performance of the modified carry skip adder, four type adders are implemented using the Quicksim. Table 2 shows delay time of adders in the Quicksim simulation. These results show that the modified carry skip adder performs rather well. Especially, the 16-bit and 32-bit designs show that the performance of the modified carry skip adder much better than the conventional carry skip adders.

V. Conclusion

Carry skip adders have low complexity and moderate delays. Within each block, ripple carry is used to produce the sum bits and a carry. When the carry of the first block is generated, skipping produces the next carry with two gate delays per blocks. The second block cannot pass carry signals until the block generates the first carry. A modified carry skip adder has been designed for reducing the first block delays. Carry lookahead logic is used for the first carry computation. To verify the performance of proposed carry skip adder, it is compared with different adders. Although the modified carry skip adder uses a few extra gates because of the CLA logic, it provides better speed than the conventional carry skip adder. It certainly works well with fixed-size blocks. Carry skip adders with variable block size are faster than adders with fixed-size blocks and may not benefit as much from speeding the formation of the first carry. The modified skip adder can be useful in the small bit addition.

References

[1] Thomas Lynch and Earl E. Swartzlander, Jr., "A Spanning Tree Carry Lookahead Adder," *IEEE Transactions on Computers*, vol. 41, pp. 931-939, 1992.

[2] V. Kantabutra, "Designing optimum one-level carry-skip adders," *IEEE Transactions on Computers*, vol. 42, no. 6, pp. 759-764, June 1993.

[3] V. Kantabutra, "Designing optimum carry-skip adders," *Proceedings 10th IEEE Symposium on Computer Arithmetic*, pp. 146-153, 1991.

[4] Earl E. Swartzlander, Jr., "Computer Arithmetic," in Allen B. Turker, Jr., Ed. *The Computer Science and Engineering Handbook*, Boca Raton, FL: CRC Press, 1997, pp. 462.