A Fast Hybrid Carry-Lookahead/Carry-Select Adder Design

Ohsang Kwon*
Sun Microsystems Inc.
901 San Antonio Rd.
Palo Alto, CA 94303
ohsang.kwon@eng.sun.com

Earl E. Swartzlander, Jr.
The University of Texas at Austin
Austin, Texas 78712
eswartzla@aol.com

Kevin Nowka
IBM Austin Research Lab
11400 Burnet Rd.
Austin, Texas 78758
nowka@us.ibm.com

ABSTRACT

In this paper, a parallel global carry generation is used for a hybrid carry-lookahead/carry-select adder design to reduce fanout load at the final multiplexor stage and relieve the speed requirement of compound adders. For compound adders in the hybrid scheme, a new logical decomposition is derived to minimize silicon area and to improve the speed. The new architecture is explained with a 64-bit adder design using dynamic CMOS circuit implementations. The 64-bit adder has the delay of 525ps in 0.225μm bulk CMOS technology.

Categories and Subject Descriptors

B.2.4 [**Hardware**]: Arithmetic and Logic Structures—*algorithms, performance*; B.7.1 [**Hardware**]: Integrated Circuits—*algorithms implemented in hardware, VLSI*

General Terms

Design, Algorithms, Performance

Keywords

Carry Lookahead Adder, CMOS, Domino Logic

1. INTRODUCTION

Carry lookahead adder(CLA) employs a fast tree structure in global and local carry generation. For speed improvement, carry select schemes can be combined with the CLA architecture[1][2][5].

In the hybrid scheme, two conditional addition results are pre-computed for each specific block and each global carry is used to select one of the two results. Therefore, the delay of the local carry generation is eliminated in the hybrid adder architecture. This paper proposes an improved architecture and fast CMOS circuits for the hybrid carry lookahead/carry select adder design.

For the global carry generation in the hybrid adder, a simple parallel scheme is proposed to relieve the fanout load at the final multiplexor stage. The parallelism does not require intermediate outputs in the lookahead logic and exhaustive duplication of lookahead cells is avoided in such a way that the majority of lookahead cells are shared among multiple lookahead trees.

For the carry-select scheme in the hybrid architecture, a new compound adder design is proposed. In the new compound adder, internal carry generation logic is shared to minimize area of the compound adder. The new compound adder is designed to be used with Ling's pseudo carry scheme to improve the speed of the carry lookahead logic[3].

2. A CARRY-LOOKAHEAD/CARRY-SELECT ADDER ARCHITECTURE

2.1 General Block Diagram

Figure 1 shows a general block diagram of the hybrid carry-lookahead/carry-select adder architecture. Usually, tree architectures are used for the fastest computation of the global carries[1][4][5]. Each global carry selects one of the two results of the corresponding compound adder.

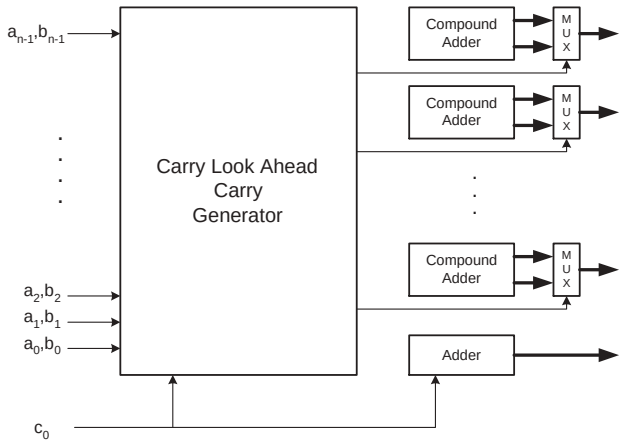
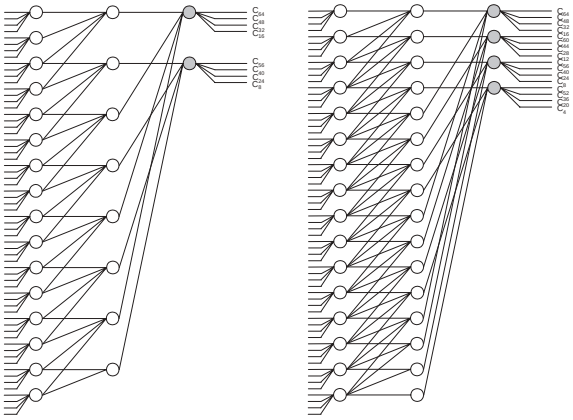


Figure 1: Carry-Lookahead/Carry-Select Adder

*This work was done when the author was with the University of Texas at Austin

2.2 Parallel Carry Generation

Figure 2 (a) shows a 8-bit spaced global carry generation. It has two standard 16-bit spaced global carry generation trees where one of them is a simply 8-bit shifted version of the other. It has more uniformly distributed fanout load in the global carry generation at the expense of more redundant lookahead cells than the spanning tree network of Lynch, *et al.* Figure 2 (b) shows a 4-bit spaced global carry generation using a 4-way parallelism.



(a) 2-way Parallel (b) 4-way Parallel

Figure 2: Parallel Carry Generation

In general, as the level of parallelism is increased, the compound adder design is simplified. The extreme case of such parallelism is Kogge-Stone tree, where there are 16 global carry generation trees for a 64-bit adder and the resulting global carry spacing is one bit[6]. In this case, only a simple XOR is required to calculate the final sum bits and compound adders are not needed. However, area and power consumption overhead of multiple tree network is the limiting factor in having such exhaustive parallelism.

2.3 Logical Decomposition for a Compound Adder

Let a_i and b_i denote the two operand bits at i-th bit position. g_i and p_i are defined as *generate* and *propagate* signals at the bit position i, respectively.

$$p_i \equiv a_i + b_i \tag{1}$$
$$g_i \equiv a_i \cdot b_i \tag{2}$$

Let $g_{i:j}$ denotes that a carry is generated in a group of bits from i to j inclusive and propagated to bit position $(i+1)$[1]. $p_{i:j}$ is similarly defined as a group propagate signal denoting that the input carry of bit position j is propagated to bit position $(i+1)$.

Now, assume that a carry lookahead architecture is used for the compound adder. Only two-level lookahead hierarchy is assumed for convenience. In the hierarchy of the carry lookahead tree, for the i-th bit position, there is a block in the first level lookahead stage where the i-th bit belongs. Denote $f1(i)$ as the LSB(Least Significant Bit) position of the block. Then, the carry-in bit of the i-th bit position can be represented as below.

$$\begin{aligned} c_i &= g_{i-1:0} + p_{i-1:0}c_0 \\ &= g_{i-1:f1(i)} + p_{i-1:f1(i)}g_{f1(i)-1:0} + \\ &\quad p_{i-1:f1(i)}p_{f1(i)-1:0}c_0 \end{aligned} \tag{3}$$

Now, the two conditional carries are represented as below.

$$c_i^0 \equiv g_{i-1:f1(i)} + p_{i-1:f1(i)}g_{f1(i)-1:0} \tag{4}$$
$$\begin{aligned} c_i^1 \equiv\ & g_{i-1:f1(i)} + p_{i-1:f1(i)}(g_{f1(i)-1:0} \\ & + p_{f1(i)-1:0}) \end{aligned} \tag{5}$$

The conditional sum bits, s_i^0 and s_i^1, are obtained through the final exclusive-OR stage as follows.

$$p_i' \equiv (a_i \oplus b_i) \tag{6}$$
$$s_i^0 = (c_i^0 \oplus p_i') \tag{7}$$
$$s_i^1 = (c_i^1 \oplus p_i') \tag{8}$$

3. HIGH-SPEED CIRCUIT IMPLEMENTATION: A 64-BIT ADDER DESIGN

The 2-way parallel global carry generation and the 8-bit compound adder are explained with a 64-bit adder design example. Delayed-reset dynamic CMOS circuits are used for high speed implementation[8]. For further speed improvement, Ling's pseudo carries are employed[3][5][7]. The first level lookahead logic and the 8-bit compound adder are modified to accommodate the use of Ling's pseudo carries.

3.1 Carry Generation

For a hierarchical carry generation scheme for the 64-bit adder, G4 and P4 are defined as a group generate and a group propagate of a group of 4-bits. With the use of an identity, $g_i = p_i \cdot g_i$, the following equations are obtained for i=0, 4, 8, ... , 56 and 60.

$$\begin{aligned} G4_{i/4} &\equiv g_{i+3:i} \\ &= g_{i+3} + p_{i+3}g_{i+2} + p_{i+3}p_{i+2}g_{i+1} + \\ &\quad p_{i+3}p_{i+2}p_{i+1}g_i \\ &= p_{i+3}(g_{i+3} + g_{i+2} + p_{i+2}g_{i+1} + p_{i+2}p_{i+1}g_i) \\ &= p_{i+3}G4^*_{i/4} \end{aligned} \tag{9}$$
$$G4^*_{i/4} \equiv g_{i+3} + g_{i+2} + p_{i+2}g_{i+1} + p_{i+2}p_{i+1}g_i \tag{10}$$
$$P4_{i/4} \equiv p_{i+3}p_{i+2}p_{i+1}p_i \tag{11}$$
$$P4^*_{i/4} \equiv p_{i+2}p_{i+1}p_{i+0}p_{i-1} \tag{12}$$

Here, $G4^*$ and $P4^*$ are pseudo group generates and pseudo group propagates in the Ling's pseudo carry scheme[3][5]. $G16$, $P16$, $G16^*$ and $P4^*$ are similarly defined for 16-bit groups as in Quach *et al.*[5] except for differences in group partitioning. Using the group generates and group propagates, the 16-bit spaced global carries can be obtained by

the following equations.

$$\begin{aligned} C_{16} &= G16_0 + P16_0C_0 = p_{15}(G16_0^* + P16_0^*C_0) \\ &= p_{15}C_{16}^* \qquad (13) \\ C_{32} &= G16_1 + P16_1G16_0 + P16_1P16_0C_0 \\ &= p_{31}(G16_1^* + P16_1^*G16_0^* + P16_1^*P16_0^*C_0) \\ &= p_{31}C_{32}^* \qquad (14) \\ C_{48} &= G16_2 + P16_2G16_1 + P16_2P16_1G16_0 \\ &\quad +P16_2P16_1P16_0C_0 \\ &= p_{47}(G16_2^* + P16_2^*G16_1^* + P16_2^*P16_1^*G16_0^* \\ &\quad +P16_2^*P16_1^*P16_0^*C_0) \\ &= p_{47}C_{48}^* \qquad (15) \end{aligned}$$

From the above equations, global carries can be simply obtained from Ling's pseudo global carries. Therefore, pseudo global carries are generated in the lookahead network since the use of pseudo carries simplifies the first stage of lookahead network. The use of pseudo carries can be easily handled in the new compound adder design to get the correct sum bits without hurting critical path delay.

Now, Using Equation (10), Equation (12) and the identity of $g_i = g_i \cdot p_i$, the pseudo generate and propagate for the first 4-bit block are reformulated as follows.

$$\begin{aligned} G4_0^* &= g_3 + g_2 + p_2g_1 + p_2p_1g_0 \\ &= (g_3 + g_2) + p_2p_1(g_1 + g_0) \qquad (16) \\ P4_0^* &= p_2p_1p_0p_{-1} \\ &= (p_2p_1)(p_0p_{-1}) \qquad (17) \end{aligned}$$

Here, p_{-1} is used to maintain the consistency with other groups though it can be omitted for the first 4-bit block.

Figure 3 shows a dynamic CMOS circuit implementation of the first lookahead stage. The generation of p_i and g_i is merged in the first lookahead logic and the newly derived logic equations with the compound dynamic CMOS circuit styles provides a simpler and faster circuit. The new circuit is faster by 67% in G4 generation than the direct implementation. Similar circuits are used for the following 16-bit group generates and group propagates.

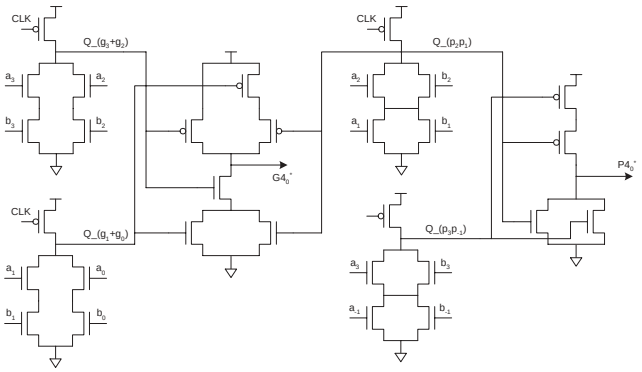


Figure 3: Dynamic CMOS Circuit for the First Lookahead Stage

So far, only 16-bit spaced global carry generation has been explained for the convenience. Other 8-bit offseted global carries can be obtained with the same mechanism. They are also 16-bit spaced global carries for another 64-bit group which consists 56 LSB bits of input operands and 8 dummy bits of '0' as seen in Figure 2(a).

3.2 Compound Adder

For a 8-bit block with a LSB(Least Significant Bit) at bit position i, the carry-in at the $(i+7)$-th bit position can be represented as follows.

$$\begin{aligned} C_{i+7} &= g_{i+6:i} + p_{i+6:i}C_i \\ &= (g_{i+6:i+3} + p_{i+6:i+3}g_{i+2:i}) + \\ &\quad (p_{i+6:i+3}p_{i+2:i})p_{i-1}C_i^* \\ &= (g_{i+6:i+3} + p_{i+6:i+3}g_{i+2:i}) + \\ &\quad (p_{i+6:i+3}p_{i+2:i-1})C_i^* \qquad (18) \end{aligned}$$

Let $S_{7+i}^{(0)}$ and $S_{7+i}^{(1)}$ denote the corresponding conditional sum bits at bit position $(i+7)$, respectively. Then

$$\begin{aligned} S_{i+7}^{(0)} &= (g_{i+6:i+3} + p_{i+6:i+3}g_{i+2:i}) \oplus p'_{i+7} \qquad (19) \\ S_{i+7}^{(1)} &= (g_{i+6:i+3} + p_{i+6:i+3}(g_{i+2:i} + p_{i+2:i-1})) \\ &\quad \oplus p'_{i+7}. \qquad (20) \end{aligned}$$

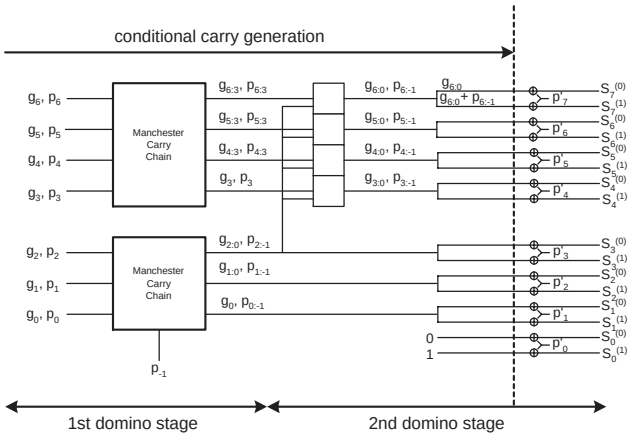


Figure 4: A New 8-bit Compound Adder

Figure 4 shows a block diagram of the new 8-bit compound adder. As shown in the figure, carry generation circuits are shared for both cases, which leads to area-efficiency. Moreover, the last two logical stages can be merged in one domino stage to minimize the delay and area as shown in Figure 5. In the figure, k and q are needed for dynamic CMOS XOR implementation. They are logical complements of p and g, respectively, and are defined as follows.

$$\begin{aligned} k_i &\equiv \overline{a_i} \cdot \overline{b_i} \qquad (21) \\ q_i &\equiv \overline{a_i} + \overline{b_i} \qquad (22) \end{aligned}$$

Their group signal definitions go in parallel with those of g and p through duality.

3.3 Performance

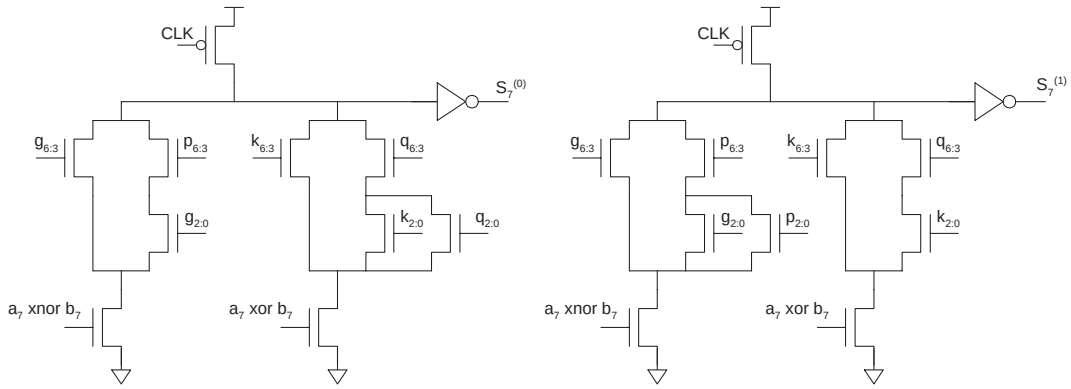


Figure 5: The Second Domino Stage For The New 8-bit Compound Adder: MSB

Table 1: Cumulative Delay Of 64-Bit Adder Designs

	a new scheme	a conventional scheme
architecture		
global carry spacing	8-bit(2-way parallel)	16-bit(standard)
compound adder	8-bit(new)	16-bit(recursive)
Ling's pseudo carry	yes	no
global carry generation		
LA1	133ps(G4),124ps(P4)	222ps(G4),143ps(P4)
LA2	282ps(G16),247ps(P16)	368ps(G16),284ps(P16)
LA3	424ps(C56)	491ps(C48)
compound adder		
MCC	318ps($g_{6:3}$)	380ps
stage2	406ps	493ps
final MUX	524ps(S63)	592ps(S63)

The new 64-bit hybrid adder was simulated using 0.225 μm bulk CMOS technology. It has a delay of 524ps on the critical path at $1.62V$ and $85°C$ operating condition.

Table 1 shows that the new design is 15% faster than the conventional design in global carry generation. The main speed improvement comes from the first lookahead stage where the new design employs Ling's pseudo carry scheme with a new simplified dynamic CMOS circuit. In compound adder designs, the 8-bit compound adder is fast enough to match the delay of the global carry generation.

4. CONCLUSIONS

In this paper, a new fast carry lookahead/carry select adder architecture is proposed. By duplicating some parts of global carry generation network with an offset, the fan-out load of the global carries can be significantly reduced and the design of compound adders is eased. To relieve the area burden and speed requirement in the carry-select scheme, a new fast area-efficient compound adder design is proposed. The compound adder is designed to accommodate the use of Ling's pseudo carries. The proposed 64-bit dynamic CMOS adder design has a 13% speed improvement compared with a conventional design in 0.225μm CMOS technology.

5. REFERENCES

[1] T. Lynch and E. Swartzlander, Jr., "A Spanning Tree Carry Lookahead Adder," *IEEE Trans. on Computers*, vol. 41, no. 8, pp. 931-939, August, 1992.

[2] Vivit Kantabutra, "A Recursive Carry-Lookahead/Carry-Select Hybrid Adder," *IEEE Trans. on Computers*, vol. 42, no. 12, pp. 1495-1499, Dec., 1993.

[3] H. Ling, "A High-speed binary adder," *IBM J. Res. Develop.*, vol. 25, pp. 156-166, May, 1981.

[4] R. P. Brent and H. T. Kung, "A regular layout for parallel adders," *IEEE Trans. on Computers*, vol. C-31, pp. 260-264, 1982.

[5] Nhon T. Quach and Michael J. Flynn, "High-Speed Addition in CMOS," *IEEE Trans. on Computers*, vol. 41, pp. 1612-1615, Dec., 1992.

[6] Jaehong Park, Hung C. Ngo, Joel A. Silberman and Sang H. Dhong, "470ps 64bit Parallel Binary Adder," *Symposium on VLSI Circuits Digest of Technical Papers*, pp. 192-193, 2000.

[7] Samuel Naffziger, "A Sub-nanosecond 0.5μ 64b Adder Design," *ISSCC Digest of Technical Paper*, pp. 362-363, 1996.

[8] Kevin J. Nowka and Tibi Galambos, "Circuit Design Techniques for a Gigahertz Integer Microprocessor," *International Conference on Computer Design*, pp. 11-16, 1998.