Performance Comparison of VLSI Adders Using Logical Effort[1]

Hoang Q. Dao and Vojin G. Oklobdzija

Advanced Computer System Engineering Laboratory
Department of Electrical and Computer Engineering
University of California, Davis, CA 95616
http://www.ece.ucdavis.edu/acsel
{hqdao,vojin}@ece.ucdavis.edu

Abstract. Application of logical effort on transistor-level analysis of different 64-bit adder topologies is presented. Logical effort method is used to estimate delay and impact of different adder topologies and to evaluate the validity of the results obtained using logical effort methodology. The tested adder topologies were Carry-Select, Han-Carlson, Kogge-Stone, Ling, and Carry-Lookahead adder. The quality of the obtained estimates was validated by circuit simulation using H-SPICE for 1.8V, 0.18μm Fujitsu technology.

1 Introduction

Delay estimation is critical in development of efficient VLSI algorithms [2]. Unfortunately, delay estimates used are usually presented either in terms of gate delays or in terms of logic levels. Neither of these estimates allows us to properly evaluate different VLSI topologies. One such component, VLSI adder, is critical in the design of high-performance processors. Using gate delay is no longer adequate because gate delays are dependent on gate types, the number of inputs (fan-in), output load (fan-out), and particular implementation. Further, a particular VLSI implementation can use static or dynamic CMOS where logic function is usually packed into a complex logic blocks. Thus, the notion of logic gate and associated gate delay becomes artificial and misleading. In this analysis, we are evaluating the use of the logical effort method not only for the purpose of better delay estimation but also for evaluation of different adder topologies and their impact on design of VLSI adders.

The logical effort (LE) analysis [1] models the gate delay using gate characteristics and its loading and compares the gate delay to τ, the delay of a Fan-Out of 1 (FO1) inverter. This latter delay is normally known for a given technology and can serve to estimate the speed. When a gate is loaded, its delay varies linearly with the output load expressed in terms of fan-outs. LE also accounts for the effect of circuit

[1] This work has been supported by SRC Research Grant No. 931.001, Fujitsu Laboratories of America and California MICRO 01-063

topology, by including path branching in the model. The delay estimation using LE method is quick and sufficiently accurate.

In order to evaluate the efficiency and usefulness of LE we have chosen several diverse adder topologies and compared the estimated delay with the one obtained via simulation. The adders chosen for this analysis were: multiplexer-based adder (MXA) which is implemented as a static radix-2 4-bit adder with conditional-sum in the final stage [3]; Han-Carlson consisting of static and dynamic radix-2 adders [5][6]; Kogge-Stone, static and dynamic radix-2, and dynamic radix-4 adder [7][8]; Naffziger's implementation of Ling's adder [9] in a dynamic radix-4 topology [10]; and a Carry-Look-ahead (CLA) adder implemented in dynamic radix-4 topology [4].

The multiplexer-based adder (MXA) takes advantage of its simplicity and speed of transmission-gate multiplexer implementation [3]. The sums are generated conditionally in groups of 4 bits. The carries to these groups are formed using radix-2 propagates and generates. The generate path is critical, passing through 9 stages including the total of 7 multiplexers. Thus, the transmission-gate multiplexer speed is a dominant factor determining the speed of this adder.

The Han-Carlson and Kogge-Stone adders use similar radix-2 structure as MXA. However, they combine the carries with the half-sum signals in order to obtain the final results. Direct CMOS implementation of generate and propagate logic had been used, allowing usage of both static and dynamic gates. The Han-Carlson adder differs from the Kogge-Stone adder by not creating all the carries from the radix-2 structure. Instead, only even carries are created and odd carries are generated from even carries. Therefore, in terms of logic stages, Han-Carlson uses one extra stage while Kogge-Stone adder is equivalent in the number of stages to MXA.

Ling's adder obtains high performance by exploiting wired-OR gate property of emitter-coupled logic. With CMOS implementation, such advantage is lost. However, it was shown in [10] that high performance could be realized using radix-4 propagates and generates for carries and conditional sum.

The CLA adder allows fast implementation, especially the dynamic radix-4 type [4]. CLA is a textbook example and it is most commonly used. However, with dynamic radix-4 implementation, its large transistor stack and many stages made it appear slow compared to other adders.

Using logical effort method for quick optimization, these adders were evaluated and compared in [12] and extended next with the inclusion of radix-2 Han-Carlson and radix-2 Kogge-Stone adders. Section 2 outlined the optimization conditions for the adders. The delay of adders using logical effort method was discussed in section 3. The results were compared with H-SPICE simulation in section 4. The conclusion of the work was given in section 5.

2 Optimization Conditions

All adders were optimized under the following conditions: maximum input size of 20µm, maximal allowable transistor size of 20µm and an equivalent load of 30µm-inverter. These conditions were set to get reasonable transistor sizes and loads to an adder.

The wiring capacitance was included. It was computed using the unit-length wiring capacitance and the 1-bit cell width. This width was determined from the preliminary layout of the most congested bit cell. The wire length was determined from the number of bits it spanned and the number of wires running in parallel.

Using logical effort method, the adders were optimized according to the critical paths that were estimated from the adder topology. Delay effort in other paths was computed from the critical one. The optimization process was applied recursively to update the branch factors along the critical path. It finished after all transistor sizes converged and the final result recorded the adder delay.

3 Delay Effort of Adders

The logical effort of gates was obtained from simulation. This adjustment was necessary for two reasons: first, pMOS and nMOS driving capability vary with technology, and secondly, better average per-stage delay can be achieved using the p-n ratio in the range of 1.4-1.6. Thus, we needed to repeat the gate delay simulation in order to accurately model the delay; the drain and source areas of transistors were

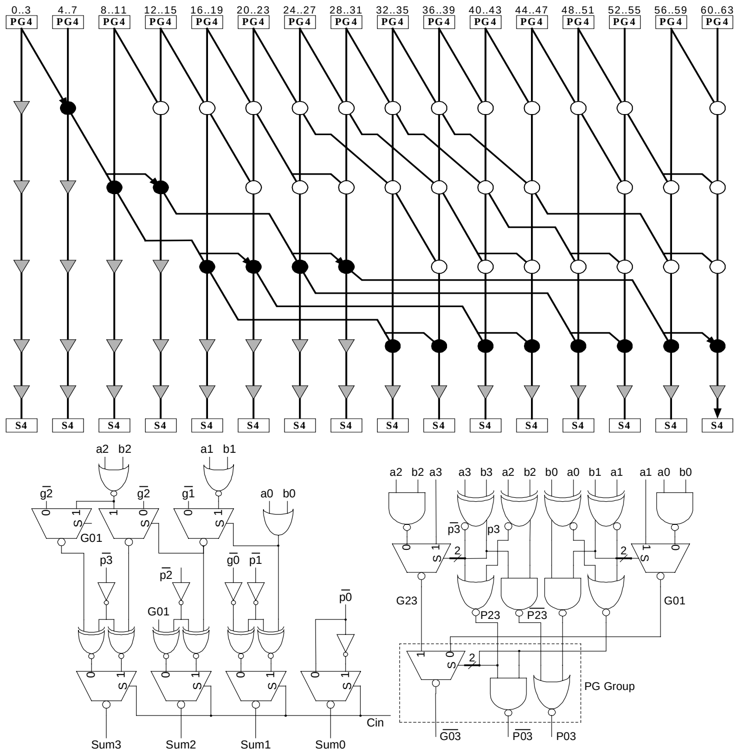

Fig. 1. Multiplexer-based carry select adder: diagram and circuits [3]

included to match better with real layout. We used p-n ratio of 1.5 for the performance reason. Nonetheless, all gates continued to show linear delay with fan-out. In addition, to accurately model the delay, the domino gates were broken into dynamic and static gates. First, the latter have different driving capability and needed to size differently. Second, domino gates can be very complex (for example, in CLA and Ling adder, group generates and group carries drive multiple inverters at different locations on its NMOS stack). Without such separation, it is very difficult to model its delay accurately.

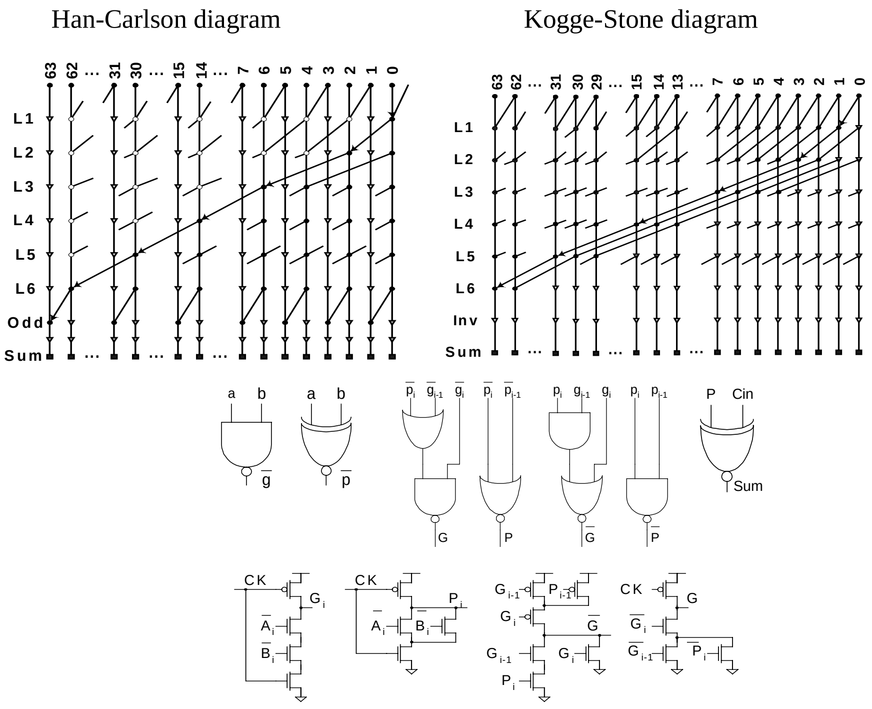

Fig. 2. Radix-2 Han-Carlson and Kogge-Stone adders: diagrams and circuits [5][6][7]

3.1 Results

The static radix-2 MXA consists of 9 stages and was implemented using static CMOS (Fig. 1). The radix-2 structure was chosen so that 2-input gates could be used. The generate signals were implemented with transmission-gate multiplexers, which were controlled by propagate and their complementary signals. In [3], single-ended propagate signals were implemented and inverters were needed to generate the complement signals. To avoid this delay penalty, complementary propagate signals were generated directly. The critical path was from bit-1 propagate through generate paths to the MSB sum. Along this path, the fan-out was slightly larger than 2. The logical effort optimization achieved the total delay of 55.8τ (11.4FO$_4$).

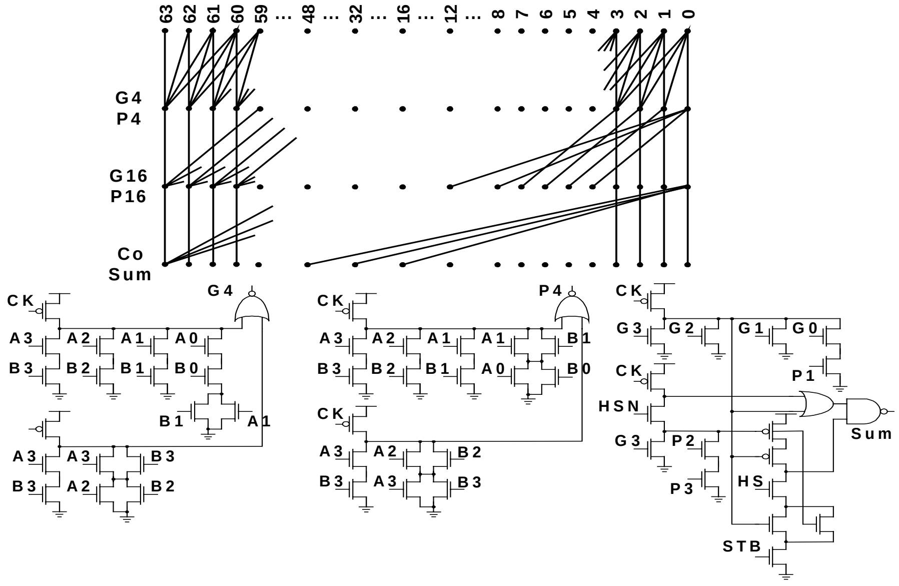

Fig. 3. Radix-4 Kogge-Stone adder: diagrams and circuits [7]

The radix-2 Han-Carlson adder (Fig. 2) realizes even carries with propagate and generate signals of even bits. The odd-bit carries are generated at the end using even carries. The critical path goes from bit 1 through the generate path to the MSB sum, traversing 10 stages. The propagate paths had the equal number of stages but they were loaded less heavily than the most critical generate path. The fan-out along the critical path was less 2. The total delay was 62.5τ (12.8FO$_4$) and 55.8τ (11.4 FO$_4$) for static and dynamic implementation.

The radix-2 Kogge-Stone adder is similar in architecture to the Han-Carlson. The difference is that propagate and generate signals of all bits are created in Kogge-Stone adder (Fig. 2). This results in 9 stages, one less as compared to Han-Carlson adder. The cost, however, was in twice as many gates for propagate and generate signals and doubling of the number of wires. The critical path went through the generate signals, traversing 9 stages. The fanout was also less than 2. The total delay after optimization was 57.6τ (11.8FO$_4$) and 42.6τ (8.7FO$_4$) for static and dynamic implementation. The delay is better compared to Han-Carlson adder.

The dynamic radix-4 Kogge-Stone adder was implemented in only 6 stages, by using redundant logic in propagate and generate stages and strobe signals for final sum (Fig. 3). The cost was very high input and internal loading and large amount of wiring between stages. In addition, dynamic stages that followed were slow NOR gates. The critical path went through the generate path from bit 0 to the MSB sum. The total delay is 30.1τ (6.2FO$_4$). This is the best delay seen – showing the advantage of using fewer stages over its complexity.

The dynamic radix-4 CLA was realized in 16 stages or 8 domino gates (Fig. 4). The critical path was from bit 0 through the generate path and higher-bit carries to the MSB of the sum. Fan-out of 3 was observed along generate and carry paths. The total delay is 54.3τ (11.1FO$_4$) due to more loading and longer wires.

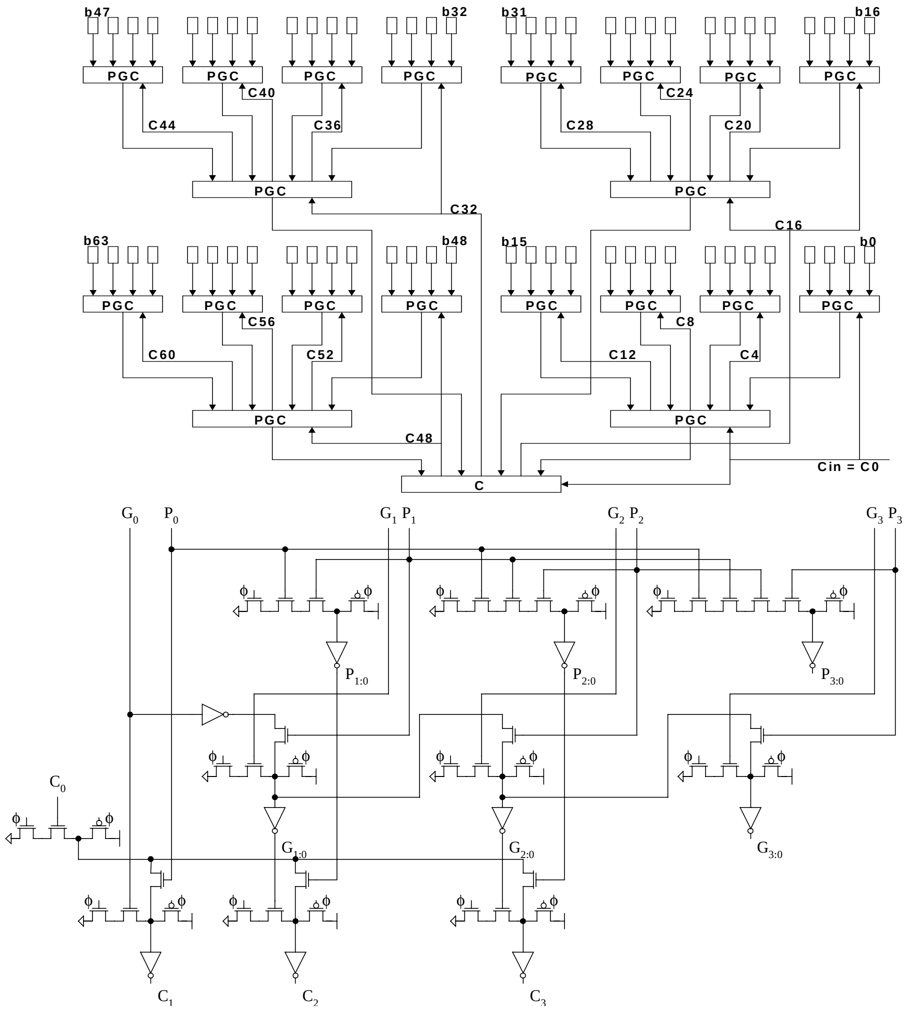

Fig. 4. Radix-4 CLA adder: diagrams and circuits [4]

Naffziger's implementation of modified Ling's adder [10] utilizes Ling pseudo-carries and propagate signals [9] in order to generate long carries and the conditional-sum adder for local carries (Fig. 5). The critical path was chosen through the long carry to the MSB Sum and it was realized in 9 stages, due to larger gate and wire loading. Local carry and sum paths have more stages than the critical path. They were implemented with faster gates to avoid becoming critical. The total delay is 43.9τ (9.0FO$_4$).

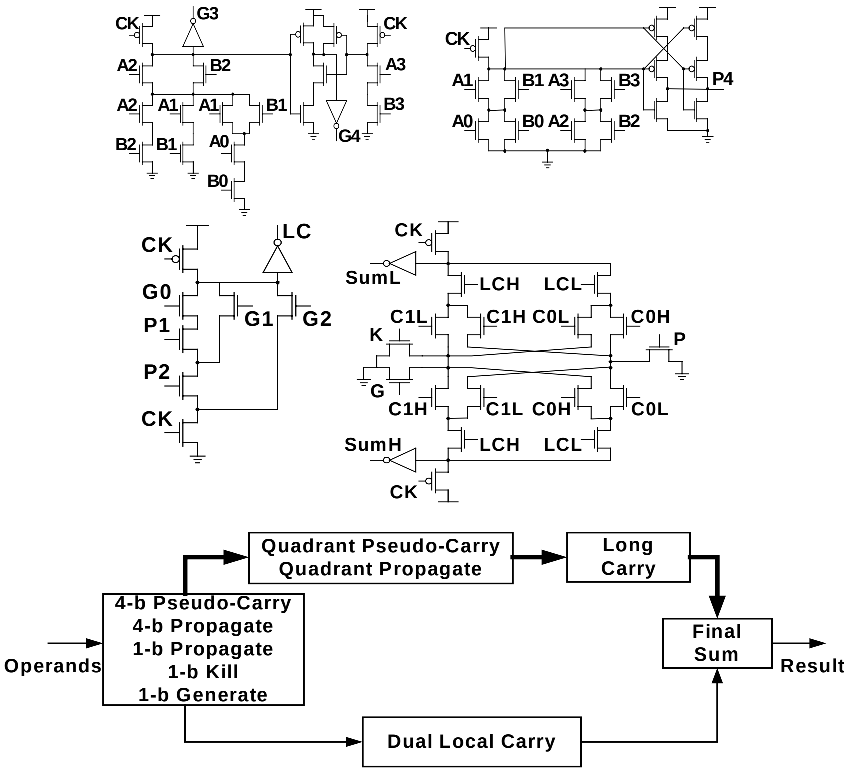

Fig. 5. Radix-4 modified Ling adder: diagrams and circuits [7]

3.2 Comparison

Table 1 summarized the delay of adders using logical effort analysis. The delays are expressed in terms of inverter delay τ and FO$_4$. The adders with fewer stages are consistently faster. Figure 6 shows the total delay and number of stages. The delay was found to be linearly proportional to the number of stages in the critical path. It was capitalized into 1.2FO$_4$ and 0.6FO$_4$ per stage, respectively, for static and dynamic implementation.

Table 1. Adder delays using logical effort method

Type	Adder	# Stages	LE (τ)	# FO_4
Static	*MXA*	9	55.8	***11.4***
	KS	9	57.6	***11.8***
	HC	10	62.5	***12.8***
Dynamic	*KS-4*	6	30.1	***6.2***
	KS-2	9	42.6	***8.7***
	Ling	9	43.9	***9.0***
	HC	10	47.9	***9.8***
	CLA	16	55.8	***11.4***

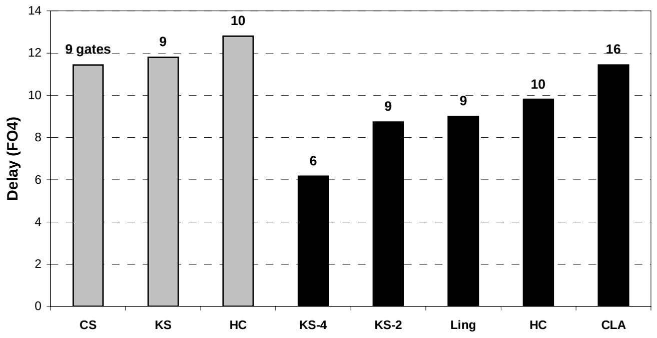

Fig. 6. Total delay from logical effort method and number of stages

4 Simulation Results

The worst-case delay of each adder's critical path was simulated with H-SPICE using the 0.18μm, 1.8V CMOS at 27°C temperature. The results obtained were presented in Table 2.

The results obtained using H-SPICE simulations are fairly consistent with the logical effort analysis in term of relative performance among adders. That is a good indicator and it confirms our belief that LE estimates should replace number of stages or gate counts as delay estimates when developing VLSI algorithms. Figure 7 showed the delays obtained using H-SPICE and a relative difference with logical effort results. The delay of adders remained dependent on the number of stages. In addition, the per-stage delay difference was degraded to $1.4FO_4$ and $0.8FO_4$ for static and dynamic implementation, respectively.

Some inconsistency was observed between logical effort result and H-SPICE for MXA, which had larger errors compared to Kogge-Stone and Han-Carlson. The main error came from larger delay in the multiplexers than modeled. Because pMOS-to-nMOS ratio of 1.5 was used, the rising signal was faster than the falling signal. So, multiplexer did not fully switch until the rising control to the multiplexer. Therefore, the multiplexer delay was always determined by the slow rising signal. It corresponded to the worst-case delay, not the average. Large errors were also seen in radix-4 dynamic adders. They used high-stack nMOS and had many branches. Therefore were harder to model accurately, especially on parasitic delay.

Table 2. Logical effort and simulation delay results

Type	Adder	# Stages	LE (FO4)	HSPICE (FO4)	HSPICE (ps)	Diff. (%)
Static	**KS**	9	11.8	10.9	853.0	-8.04
	MXA	9	11.4	12.8	1003	10.99
	HC	10	12.8	13.3	1036	3.49
Dynamic	**KS-4**	6	6.2	7.4	581	17.11
	KS-2	9	8.7	9.2	717	4.82
	Ling	9	9.0	9.5	742	5.34
	HC	10	9.8	9.9	772	0.85
	CLA	16	11.4	14.2	1107	19.3

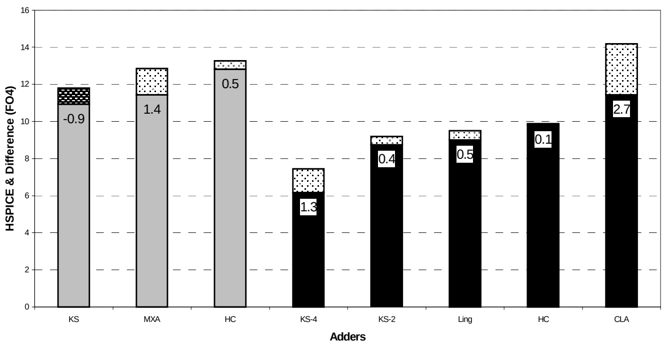

Fig. 7. Total delay with H-SPICE and delay difference

Nonetheless, the relative performance among adders did not vary significantly. It was realized that having less stages in critical path helped to improve delay. Although less stage meant more complex gates that translated into worse per-stage delay, such delay degradation was offset by more delay reduction due to fewer stages.

5 Conclusion

Use of Logical Effort method for performance comparison of different adder topologies was presented with wire capacitance included. Obtained results were consistent with simulation and are encouraging. They show that incorporating Logical Effort into the analysis of VLSI adders can help find better adder topologies.

References

1. I. Sutherland, B. Sproull, D. Harris, "Logical Effort: Designing Fast CMOS Circuits," Morgan Kaufmann Publisher, 1999.
2. V. G. Oklobdzija, E. R. Barnes, "Some Optimal Schemes for ALU Implementation in VLSI Technology", Proceedings of 7th Symposium on Computer Arithmetic, June 4-6, 1985, University of Illinois, Urbana, Illinois.
3. A. Farooqui, V. G. Oklobdzija, "Multiplexer Based Adder for Media Signal Processing," 1998 Symposium on Circuits and Systems.
4. A. Naini, D. Bearden, W. Anderson, "A 4.5nS 96-b CMOS Adder Design", in Proc. CICC, Feb. 1992, pp. 25.5.1–25.5.4.
5. S. K. Mathew et al., "Sub-500ps 64-b ALUs in 0.18µm SOI/Bulk CMOS: Design and Scaling Trends," Journal of Solid-State Circuits, Nov. 2001.
6. T. Han, D. A. Carlson, "Fast Area-Efficient VLSI Adders," 8th IEEE Symposium on Computer Arithmetic, Como, Italy, pp. 49–56, May 1987.
7. P. M. Kogge, H. S. Stone, "A Parallel Algorithms for the Efficient Solution of a General Class of Recurrence Equations", IEEE Transactions on Computers, Vol. C-22, No 8, Aug. 1973. p. 786–93.
8. J. Park et al., "470ps 64-Bit Parallel Binary Adder," 2000 Symposium on VLSI Circuits Digest of Technical Papers.
9. H. Ling, "High Speed Binary Adder", IBM Journal of Research and Development, Vol. 25, No 3, May 1981, p. 156.
10. Naffziger, S., "A Sub-Nanosecond 0.5 um 64 b Adder Design", 1996 IEEE International Solid-State Circuits Conference, Digest of Technical Papers, San Francisco, February 8-10, 1996. p. 362–3.
11. R. P. Brent, H. T. Kung, "A Regular Layout for Parallel Adders," IEEE Trans., C-31(3), pp. 260–264, Mar 1982.
12. H. Q. Dao, V. G. Oklobdzija, "Application of Logical Effort Techniques for Speed Optimization and Analysis of Representative Adders," 35th Annual Asilomar Conference on Signals, Systems and Computers, Pacific Grove, California, November 4–7, 2001.
13. V. G. Oklobdzija, "High-Performance System Design: Circuits and Logic", IEEE Press, 1999.