Design and Analysis of Fast Carry-Propagate Adder Under Non-Equal Input Signal Arrival Profile

Vojin G. Oklobdzija
Department of Electrical and Computer Engineering
University of California
Davis, CA 95616
vojin@ece.ucdavis.edu
(916) 752-5634

Abstract

This paper examines the design of a fast carry-propagate adder under the condition of non-equal input signals arrival. This is a common case encountered in the fast parallel multipliers where a carry-propagate adder is deployed to produce the final product. It is shown that the rules used in the past are not valid and do not result in the fastest adder. We present the analysis of those conditions and provide the rules for the fast carry-propagate adder design.

1. Introduction

Usually, an adder is built under the assumption that all of the input signals arrive at the same time. In some cases, this is not true and a delay exists between input signals. One such example is the final adder which performs the final addition of the partial products in the parallel multiplier. Under this condition the speed of various adder configurations is different and it is observed that the use of Carry Lookahead Adder does not yield best results. This paper presents a study of how to build efficient adders under the condition of non-uniform signal arrival profile. One such example is shown in Fig. 1. [13].

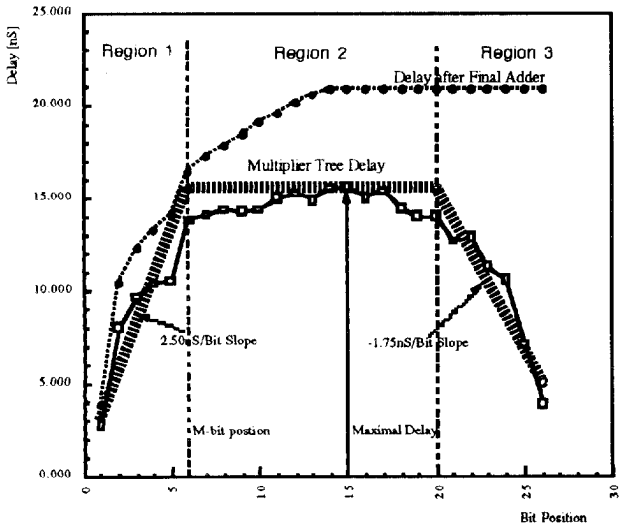

Fig. 1. *Selection of the adder types in the 3 regions of the multiplier*

Because non-uniform input arrival introduces a delay between each bit position, the objective is to minimize the delay between inputs and Cin signal for each Full Adder (FA) bit position. In this case, the delay between two bits is assumed to be constant regardless of their positions. Thus, the time representation of bit positions is a line with a positive or negative slope corresponding to the order of the bit arrival.

2. Analysis of Adder Behavior and Selection of the Adder for Non-Uniform Signal Arrival Profile

In this section we analyze the speed efficiency of conventional adders such as Ripple Carry Adders (RCA), Conditional Sum Adders (CSA), Carry Lookahead Adders (CLA), Carry Select Adders (CSLA) and Variable Block Adders (VBA) [1-4, 6].

2.1 Positive Slopes

If the delay difference between the two consecutive bits starting from LSB toward MSB is greater than Δ_{XOR}, RCA is as efficient as other schemes. We refer to this situation as the positive slope of the input signal delay profile. In this situation, each FA of a simple RCA will have sufficient time to compute the carry needed for the next FA stage. If the slope is lower than Δ_{XOR} per bit position then RCA is no longer the most efficient, and appropriate VBA should be used. For positive slopes, Variable Block Adder (VBA) or simple Ripple Carry Adder (RCA) is quite adequate for the fast addition. VBA [5-12] is based on the principle of carry skip blocks that pass the carry over groups of bits and the size of those groups is adjusted in order to minimize the critical path without using much additional logic. Non-uniform inputs introduce a different approach to building VBA because critical path depends on bit position.

2.2. Negative Slopes

By construction, each FA of an RCA needs an available Cin to generate the sum and Cout, that is to say that RCA needs LSB to begin the addition and ripple through all Full Adders. In case of CSA, LSB is needed to perform selection, and produce the final result. For CLA, recurrent equation for generate and propagate blocks is:

$$G^*_n = G_n + G_{n-1}P_{n-1} + \ldots\ldots + G_0P_1P_2P\ldots.P_n$$

$$P^*_n = P_0P_1P_2P\ldots.P_n$$

Thus to generate or propagate a carry, CLA requires available P_0 and G_0, which themselves depend on LSB.

1058-6393/95 $4.00 © 1995 IEEE

Therefore, in the case of the negative slope, CSLA and CLA need LSB to begin the addition. Thus in such a case the delay will be the same as if all bits arrive at that same time with LSB.

In case of CSLA it is possible to use the delay between MSB and LSB and sum available bits. Those sums will be ready at the input of the multiplexers, and when LSB arrives, appropriate sum will be selected. Such configuration will not have to wait for the LSB before adding already available bits. Therefore CSLA represents the best configuration, and will be considered further in the section on adders for the input arrival profile with the negative slope.

For negative slopes, the building of efficient VBA will be presented. By building variable blocks in the CSLA, delay reduction introduced by multiplexers is possible.

3. Determination of the Adder Structure in the Region of the Signal Arrival Profile with a Positive Slope

For the arrival profile with a positive slope, the critical path in a VBA is different than for the one with uniform input arrival profile [6-10]. We will disregard the case of slopes steeper than Δ_{XOR}, because it is obvious that the use of RCA is the only alternative. Throughout this chapter, slopes will be assumed to be lower than Δ_{XOR}.

3.1. Determining the Size of the Block

For positive slopes, the carry skip block could be built as a tree of AND gates as shown in Fig.2.

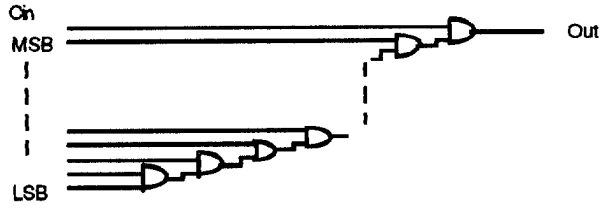

Fig. 2. Carry Skip Block Structure for the Positive Slope

Thus, as long as the delay between the arrival of two consecutive bits is smaller than Δ_{AND}, the time to produce carry from the block will be $\Delta_{OR}+\Delta_{AND}$ after the last bit arrival, otherwise it will be Δ_{OR} if Cin arrives after the last bit of the block. In most of the cases considered, Cin arrives before the latest bit. Thus, in this paper, we assume the skip delay to be equal to $\Delta = \Delta_{XOR}$.

3.2. Determination of the Maximum Propagation Time in VBA

Let :

n be the number of bits to add

m the number of blocks in an adder

kp (p=1..m) the number of bit positions in the pth block

t_p arrival time of the second bit in the pth block

t_q arrival time of the first bit in the qth block

T_p maximum propagation delay considering only the p firsts blocks

del worse case delay already at the step p

Bits will be numbered from MSB (i=1) to LSB (i=n), and blocks from the most significant one (p=1).

With respect to carry, four delay cases are possible [6-10]:

- carry generated and assimilated within a block
- carry generated within a block and propagated to the following block
- carry-in received and absorbed within a block
- carry skipping a block.

The worst case delay, or maximum propagation time, T, will be a combination of the last three delays.

For the uniform input arrival profile with a carry generated in the pth block and absorbed in the qth one, after skipping p-q-1 blocks, T is equal to:

$$T = \max_{p,q}\left[(k_p - 1) + d_{(p-q-1)} + (k_q - 1)\right]$$

where $1 \le q < p \le m$ and

$d_{(p-q-1)}$ is a delay required to skip p-q-1 blocks. Skipping a block requires a single Δ_{XOR} delay following the arrival of the last block's bit. That is to say, for all cases, the carry-in of the qth block arrives at least at:

$$t_q-\Delta_X+\Delta_{XOR}$$

(Δ_X: delay between two consecutive bits).

In the case of positive slope, the delay in the k<u>th</u> block will be:

$$(t_q - \Delta_x + 1) + (k_q - 1)$$

If the time required for the carry to ripple within pth block and skip p-q-1 blocks is higher or equal than t_q-$\Delta_X+\Delta_{XOR}$, the propagation delay for the carry from p<u>th</u> block to the k<u>th</u> block is :

$$T = \max_{p,q}\Big\{ \max\left[t_p + k_p + (p-q-1)\ ,\ t_q - \Delta_x + 1\right] + \max(k_q - 1\ ,\ 1) \Big\}$$

$$t_p + 1 + (k_p - 1) + (p - q - 1) + (k_q - 1).$$

Finally, the maximum propagation delay for a positive slope is:

Let us assume that MSB arrives at t=0 and LSB at t=- δ. The shortest time to add n bits in case of a positive slope is $2\Delta_{XOR}$. Indeed, regardless of the size of the block, the last adder, whose inputs arrive at t=0, requires $2\Delta_{XOR}$ to calculate the Sum, as long as Cin arrives before Δ_{XOR} (see Fig.3.). The goal is to determine optimal block sizes

such that Cin is available before Δ_{XOR}, that is to say to keep $T_p \leq \Delta_{XOR}$.

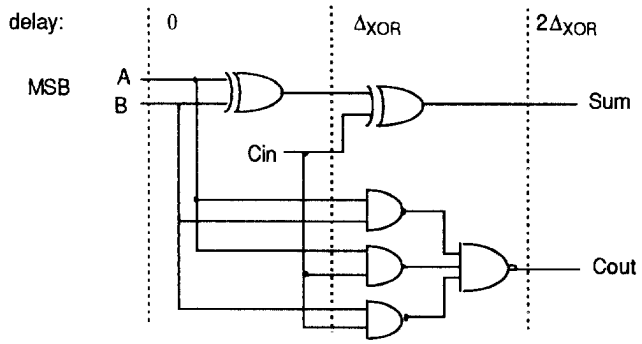

Fig.3. Delays within a Full Adder

With the definition of T seen above, we can calculate delay assuming a slope of $1/2\Delta_{XOR}$. First blocks sizes are 4, 3, 2, 1. In this case we can notice that for each block, $T_p=\Delta_{XOR}$. For a slope $\frac{1}{r}\Delta$, with $r > 2$, the maximum delay will be higher than $2\Delta_{XOR}$. The best configuration in this case is:

r, r, r, r-1, r-2, ..., 4, 3, 2, 1 (1)

This configuration requires blocks of r bits, thus not introducing more delay between T_p and T_{p+1}. In a block p of size r, where $p>r$, carry begins to ripple from the second bit (position i) of this block and skip all blocks until block 1 (worst case):

$$T_p = (r-1) - i.\frac{1}{r} + (p-2)$$

If the block $p+1$ has the same length, then $T_{p+1} = T_p$. That is to say that when the size of a block is r, all the sizes following that one will be r without adding delay. Experiments show that the best configuration to reach blocks of size r is: r-1, r-2, ..., 3, 2, 1. A corresponding worse case delay is:

$$T_p = \max_{p,q}\left\{\max\left[t_p + (k_p - 1) + (p-q-1)\ ,\ t_q - \Delta_x + 1\right] + \max(k_q - 1\ ,\ 1)\right\}$$

$$1 \leq q < p \leq m$$

Three cases may occur:

- T_p < del, that is to say that the block p doesn't introduce an extra delay. In this case it can be enlarged as long as delay *del* is not increased. It's better to have larger blocks because skip delay is not linked to the size.
- T_p = *del*, then the size of block p will be chosen as kp .
- T_p > *del*, in this case block size p will be chosen as kp -1.

However, if the slope is lower than $1/2\Delta_{XOR}$, T_p may be chosen to be higher than *del*. In this case blocks will be built following equation (1). Optimal block sizes for a 36-bit adder are shown in Table 1.

Table 1. *Optimal Block sizes for a 36 bits adder*

Slope	Configuration	Delay
$2/3\Delta_{XOR}$	15-9-6-4-2	$2\Delta_{XOR}$
$1/2\Delta_{XOR}$	10-10-6-4-3-2-1	$2\Delta_{XOR}$
$1/3\Delta_{XOR}$	3-3-3-3-3-3-3-3-3-3-2-1	$(2+2/3)\Delta_{XOR}$
$1/4\Delta_{XOR}$	2-4-4-4-4-4-4-4-3-2-1	$(2+2)\Delta_{XOR}$
$1/5\Delta_{XOR}$	1-5-5-5-5-5-4-3-2-1	$(2+3.2)\Delta_{XOR}$

For slopes up to Δ_{XOR}, RCA is sufficient and the delay (after MSB arrival),will be $2\Delta_{XOR}$.

For slopes between Δ_{XOR} and $1/2\Delta_{XOR}$, configurations given by the proposed algorithm keep delay at $2\Delta_{XOR}$.

For slopes less than $1/2\Delta_{XOR}$, delay is higher than $2\Delta_{XOR}$.

4. Determination of the Adder Structure in the Region of the Signal Arrival Profile with a Negative Slope

The problem with the signal arrival profile with the negative slope is that carry can not start propagating until the LSB has arrived.

4.1. Use of VBA in CLSA Arrangement

Carry Select Adders are perform two operations: the first one computes two additions in parallel, one assuming the carry in is 0, the other assuming it is 1; the second selects the appropriate result based on the Cin signal which is done using a multiplexer (Fig.4.). For long blocks, CSLA could be built as shown in Fig.4 (rightmost block). In this way, the carry out of the block comes from OR and AND gates, not from a multiplexer. CSLA has results already available and selects the appropriate results after LSB arrival.

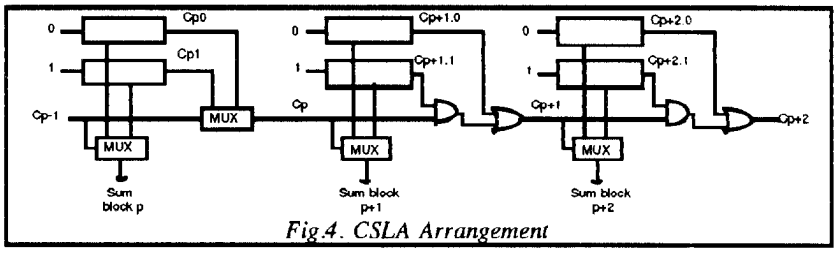

Fig.4. CSLA Arrangement

4.2. Determining Optimal Configuration

Let us assume that $\Delta=\Delta_{XOR}=2\Delta_{MUX}$.

$d_n = 2\sqrt{n\Delta}$ is the delay equation for one level VBA if n >= 4 [6].

- k_p length of the block p,
- i current bit position,
- T_p delay within the block p.
- δ_p total delay introduced from block 1 until p.

For each Full Adder, input signals arrive first and pass the first XOR before the arrival of Cin. The result of Sum and Cout will be obtained at the same time, that is to say that Sum delay equation d_n will be the same as the carry.

Therefore, it is appropriate to choose $2\sqrt{x}$ as a delay model within a block.

Because each block (VBA+MUX) adds a delay of $\frac{1}{2}\Delta$ (multiplexer selection) to the final delay, the aim is to use the smallest number of blocks possible. Thus each block must be as big as possible to have an available result before receiving the Cout of the former block. We need to determine the biggest size for a block p so that:

$$\delta_p \leq \delta_{p-1} + \tfrac{1}{2}\Delta.$$

The principle of this algorithm consists of building the adder block by block, beginning at the least significant bit. For each block, the delay d_X introduced by the VBA addition is smaller than the delay of the Cout of the former block. This delay is equal to Δ plus a multiple of $\frac{1}{2}\Delta$ equal to the number of former blocks. The first additional delay Δ is the time required by the first Full Adder to compute the carry out of LSB addition.

Table 2. *Optimal Block Sizes for a 100-bit Adder*

Slope	Configuration	Delay after LSB arrival
$-\frac{1}{3}\Delta$	1-1-2-3-5-12-34-42	4.5 Δ
$-\frac{1}{2}\Delta$	1-1-2-5-12-45-29	4 Δ
$-\Delta$	1-2-5-20-72	3 Δ
-2Δ	1-2-9-84	2.5 Δ

In comparison, adding 100 bits using CLA would require $\log_2(100)$=6.65 Δ.

Table 3. *Delay for Different Adder Sizes*

No of bits	Slope:	$-\frac{1}{3}\Delta$	$-\frac{1}{2}\Delta$	$-\Delta$	-2Δ	CLA
16		3.5	3	2.5	2,5	4
32		4	3.5	3	2.5	5
64		4.5	3.5	3	2.5	6

6. Conclusion

Methods presented in this paper permit the building of efficient adders for non uniform input arrival profile with positive and negative slopes. These methods can be applied to increase efficiency of the final adder in a parallel multiplier [13]. The evaluation of the method was made only theoretically, therefore the future work has to be supported by simulation on selected examples.

Acknowledgment

I gratefully acknowledge Nicolas de Guillebon of ESIEE, France for his analysis and contributions to this work.

References

[1] Earl E. Swartzlander, *"Computer Arithmetic"* Vol. 1 & 2, IEEE Computer Society Press, 1990.

[2] K. Hwang, *"Computer Arithmetic : Principles, Architecture and Design"*, John Wiley and Sons, 1979.

[3] O.J. Bedrij, *Carry-Select Adder*, IRE Trans. on Electronic Computer, June 1962.

[4] J. Sklansky, *"Conditional-Sum Addition Logic"*, IRE Transactions on Electronic Computers, Vol. EC-9, No.2., June 1960.

[5] S. Majerski, *On Determination of Optimal Distributions of Carry Skips in Adders*, IEEE Trans. Electron. Computers, February 1967.

[6] V. G. Oklobdzija, E. R. Barnes, *"Some Optimal Schemes for ALU Implementation in VLSI Technology"*, 7th Symposium on Computer Arithmetic ARITH-7, June 4-6, 1985, Urbana, Illinois.

[7] V. G. Oklobdzija, E. R. Barnes, *"On Implementing Addition in VLSI Technology"*, IEEE Journal of Parallel Processing and Distributed Computing, No.5, p. 716 1988.

[8] E.R. Barnes, V.G. Oklobdzija, *New Multilevel Scheme for Fast Carry-Skip Addition*, IBM Technical Disclosure Bulletin, April 1985.

[9] E.R. Barnes, V.G. Oklobdzija, *New Scheme for VLSI Implementation of Fast ALU*, YO884-0064, Vol.28, No.3, IBM Technical Disclosure Bulletin, August 1985.

[10] E.R. Barnes, V.G. Oklobdzija, *Method for Fast Carry-Propagation for VLSI Adders*, IBM Technical Disclosure Bulletin, YO883-0225, Vol.28, No.4, September 1985.

[11] P.K. Chan, M.D.F Schlag,C.D.Thomborson, V.G. Oklobdzija, *Delay Optimisation of Carry-Skip Adders and Block Carry-Lookahead Adders*, Proc. of the 10th IEEE Symposium on Computer Arithmetic, June 91.

[12] S. Tirini, *Optimal Group Distribution in Carry-Skip Adders*, Digital Western Res. Lab. Res., Feb 89.

[13] V.G. Oklobdzija, D. Villeger, *Improving Multiplier Design by using Improved Column Compression Tree and Optimized Final Adder in CMOS Technology*, in press, IEEE Transactions on VLSI, 1995.