Design of Efficient Han-Carlson-Adder

S. Sri Katyayani
Dept of ECE
Narayana Engineering College, Nellore

Dr.M.Chandramohan Reddy
Dept of ECE
Narayana Engineering College, Nellore

Murali.K
HoD, Dept of ECE
Narayana Engineering College, Nellore

Abstract— In digital VLSI systems binary addition is the most significance arithmetic function. To a great extent adders are used as DSP lattice filter where the ripple carry adders are substituted by the parallel prefix adder to reduce delay. The requirement of adder is that it is fast and it has area efficient and low power consumption. In this the parallel prefix adder is introduced as speculative Han- Carlson adder for differ binary addition of the most significant arithmetic function. In this method has wider word length and the parallel prefix adder is replaced as speculative Han-Carlson adder is introduced as variety stages of Brent-Kung and Kogge-Stone adders reduces the complexity of adder design. The design of Verilog code is simulated using ModelSim and implemented in Xilinx.

Index Terms— variable latency, adders, speculative adders, parallel-prefix adders.

I. INTRODUCTION

Adders are fundamental elements in arithmetic operations. Binary adders are utilized as a part of digital circuit for addition, subtraction operations and for floating point multiplication and division**.** Therefore adders are Fundamental components and improving their performance is one of the major challenges in digital designs. Computer arithmetic algorithm has established lower bounds on area and delay of n-bit adders, the former varies linearly with adder size, and the latter has an O (log2 (n) behavior.

High speed adders depend on entrenched parallel prefix structures including Brent-Kung [1], Kogge-Stone [2], Han-Carlson [3]and soon. These standard structures work with fixed latency. Better average performances can be achieved by using variable latency adders that have been recently proposed in literature speeding up of approximate circuits.

The approximated adder is enlarged with error detection network that asserts error signal when hypothesis fails. In this case (Mis prediction), another clock cycle is needed to obtain the correct result with the help of a correction stage. Since the addition time is one clock cycle when no error happens and two clock cycles when the speculation fails.

A first theoretical way to deal with expansion was proposed by Nowick [4] in asynchronous challenge, which executes a variable latency adder cutting the least levels of a Kogge-Stone adder.

In synchronous challenge, Verma [5] propose a variable latency speculative adder cutting the lower levels of a Kogge-Stone adder. The Kogge-Stone adder is frequently utilized when velocity is the essential worry has fan-out of 2.This comes at the expense of utilizing numerous propagate-generate cells and numerous wires that must be directed between stages.

The paper is described as follows .In Section II we recall the basic architecture of parallel-prefix adders. The stages in which variable latency speculative prefix adders can be subdivided, in Section III We recall brief

description of Kogge Stone speculative prefix-processing stage [2], we present the proposed Han-Carlson speculative topology. Detailed discussion about the error detection stage is also reported in this Section. The Section IV presents spatial and timing complexity of investigated architectures. Section V shows detailed implementation and synthesis results of the proposed adders.

II. PRELIMINARIES

A. Prefix Addition

The binary addition problem can be formulated as follows: given n-bit augends $A=a_{n-1,}\ a_{n-2\ldots}\ a_{0.}$ and an n-bit addend $B=b_{n-1,}\ b_{n-2,}\ b_{0.}$ generate the n-bit sum $S= s_{n-1,}\ s_{n-2,\ldots\ldots}\ s_0$ Give us a chance to show as Ci the carry out of the i-th bit The whole piece Si and the carry Ci can be computed as follows .

$$\begin{aligned} s_i &= a_i \oplus b_i \oplus c_{i-1} \\ Ci &= a_i b_i + a_i c_{i-1} + b_i c_{i-1} \end{aligned} \quad . \qquad (1)$$

In prefix addition we use three stages to register the sum- pre-processing, prefix-processing and post-preparing. In the pre-handling stage the generate gi and propagate Pi signal are figured as:

$$\begin{aligned} g_i &= a_i . b_i \\ p_i &= a_i \oplus b_i \end{aligned} \qquad (2)$$

The condition g_i =1 means that a carry is generated at bit i. While the condition P_i=1 means that a carry is propagated through bit i .The concept of generate and propagate can be extended to a block of contiguous bits, from bit k to bit i (with k< i) as follows:

$$\begin{aligned} g_{[i:k]} &= \begin{cases} g_i & \text{if } i = k \\ g_{[i:j]} + p_{[i:j]} g_{[l:k]} & \text{otherwise} \end{cases} \\ p_{[i:k]} &= \begin{cases} p_i & \text{if } i = k \\ p_{[i:j]} p_{[l:k]} & \text{otherwise} \end{cases} \end{aligned} \qquad (3)$$

Where: $i \geq l \geq j \geq k.$

The condition g[i:k] means that a carry is generated in the block k-1 , while the condition p[i:k] means that a carry is propagated through the block. Thus, for any bit i the carry C_i can be expressed as:

C_i=g[i:0]+p[i:0]c-1 (4)

Where C-1 is the information convey of the n-bit adder. In the accompanying, for straight forwardness, we accept that Ci-1=0, so that above equation 4 follows as:

C_i=g[i:0] (5)

The block generate and propagate terms are registered in the prefix-preparing phase of the adder. To that reason, the (g[i:k] ,p[i:k]) couples are communicated with the assistance of the prefix operator characterized as takes follows

(6)

$$(g_{[i:k]}, p_{[i:k]}) = (g_{[i:j]}, p_{[i:j]}) \bullet (g_{[l:k]}, p_{[l:k]}) = (g_{[i:j]} + p_{[i:j]} g_{[l:k]}, p_{[i:j]} p_{[l:k]})$$

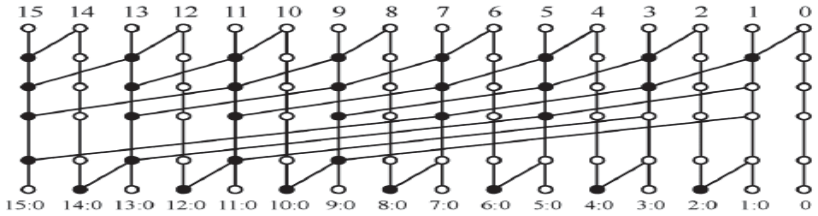

Figure 1: Han-Carlson parallel-prefix topologies. n=16

III. VARIABLE LATENCY SPECULATIVE PREFIX ADDERS

Variable latency speculative prefix adders can be subdivided in five phases: pre-handling, speculative prefix-processing, post-processing, blunder detection and error correction. The error correction stage is off the critical path, as it has two clock cycles to obtain the exact sum when speculation fails.

A. Pre-Processing

In the pre-preparing stage the generate gi and propagate Pi signs are computed as in Equation-(2**)**

B. Speculative Prefix-Processing

The speculative prefix-processing stage is one of the fundamental contrasts and the contrasted standard prefix adders reviewed in past area. Rather than figuring all the g[i:0] and p[i:0] required in (4) to get the accurate c cluster values, just a subset of square generate and propagate signs is computed, in the post processing stage surmised convey qualities are acquired from this subset. The yield of the speculative prefix-processing stage will likewise be utilized as a part of the error detection and in the error correction stages examined in the following.

The basic assumption behind speculative prefix-processing stage is that carry signals propagate for no more than K bits, with k<n and k=O (log (n)). This assumption is corroborated by the analyses in [5] that demonstrate that having a propagate chain longer that log (n) is a very rare event.

1) Kogge-Stone Topology: The Kogge-Stone speculative prefix-processing stage has been proposed in [4], [5] and can be obtained by pruning the last levels of a traditional Kogge-Stone adder. In the example appeared in Figure. 2 the last level of an n=16 bit Kogge-Stone adder is pruned. As it can be seen, for i>8 the length of propagate chains stretches out for 8 bits, bringing about a speculative prefix-processing stage with k=8.

In general, one has $K=n/2^P$, where P is number of pruned levels; the number of levels of the speculative stage is correspondingly reduced from $\log_2$ (n) to $\log_2$ (k) (assuming that K is a force of two).

$$(g,p)_{[i:0]} \quad \text{for: } i \le k = 1$$
$$(g,p)_{[i:i-k+1]} \quad \text{otherwise} \qquad (7)$$

2) Han-Carlson Topology: Han-Carlson adder contains a good trade-off between fan out, number of logic levels and number of black cells. Because of this, Han-Carlson adder can achieve equal to speed execution admiration to Kogge-Stone adder, at lower power utilization and territory. In this manner it is fascinating to execute a speculative Han-Carlson adder.

Moved by these reasons, we have generated a Han-Carlson speculative prefix-processing stage by removing the final rows of the Kogge-Stone part of the adder. As an example, Figure. 2 show the Han-Carlson adder of Figure 1 in which the two Brent-Kung rows at the initial and toward the end of the graph are unaltered, while the last Kogge-Stone row is pruned. This yields a speculative stage with K=8=n/2.in general, one has K ==n/2p. where P is the quantity of pruned levels; the number of levels of the speculative Han-Carlson stage lessens from 1+ log(n) to1+ log(k) (assuming that is a power of two).

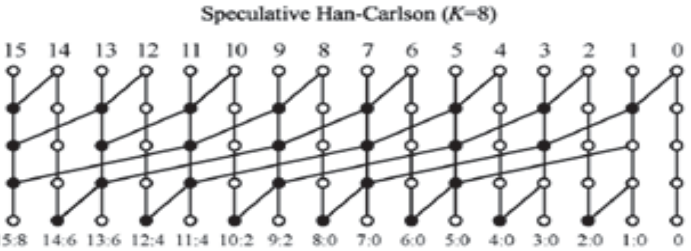

Figure.2: Han-Carlson speculative prefix processing stage.

As it can be shown in Figure. 3, the length of the propagate chains is K=8 only for i=9, 11, 13,15 , while for i=10,12,14 the propagate chain length is K+1=9. In casual, the computed propagate and generate signals for the speculative Han-Carlson design are:

$$\begin{matrix}(g,p)_{[i:0]} & for: i \le k \\ (g,p)_{[i:i-k+1]} & for: i > k, i\ odd \\ (g,p)_{[i:i-k]} & for: i > k, i\ even\end{matrix} \quad (8)$$

As it will be apparent in the following, having the propagate lengths equal to k+1 for half of the outputs greatly simplifies the error detection.

C. *Post-Processing*

In the post-processing stage we firstly compute the approximate carries, $\tilde{c}_i$, and then use them to find the approximate sum $\tilde{s}_i$, bits as follows:

$$\tilde{s_i} = p_i + \tilde{c_{i-1}} \quad .(9)$$

Similarly to (5), the approximate carries are obtained as the generate signals available in the last level of the prefix-processing stage. We have:

$$\tilde{c_i} = \begin{cases} g_{[i:0]} & for: i \le k-1 \\ g_{[i:i-k+1]} & otherwise \end{cases} \quad \text{(Kogge stone)(10)}$$

And

$$\tilde{c_i} = \begin{cases} g_{[i:0]} & for: i \le k \\ g_{[i:i=k+1]} & for: i > k, i\ odd \\ g_{[i:i-k]} & for: i > k. i\ even \end{cases} \quad \text{(Han Carlson)(11)}$$

D. *Error Detection*

The conditions in which at least one of the approximate carries is wrong (mis prediction) are signaled by the error detection stage. In case of mis prediction, an error signal is asserted by error detection stage and the output of the post-processing stage is deleted.

1) ***Kogge-Stone:*** The error condition for convey can be gotten from (5), (10) and utilizing the properties of propagate generate signals as:

$$e_i = \begin{cases} o & for: i \le k-1 \\ p[i:i-k+1]g[i-k:0] & otherwise \end{cases} \quad (12)$$

Thus, the error signal can be expressed as

$$E_{KS} = \sum_{i=k}^{n-1} p[i:i=k+1]g[i-k:0] \quad (13)$$

Where the symbol represents the logical OR. It is important to note that Equation 13 is a necessary and sufficient error condition that requires the calculation of. Unfortunately, these terms are really not registered by the speculative prefix-processing stage (dodging the calculation of these terms is the key thought of speculative adders). Therefore, Equation 13 is supplanted by the accompanying looser relation:

$$\tilde{E}_{KS} = \sum_{i=k}^{n-1} p[i:i-k+1] \quad (14)$$

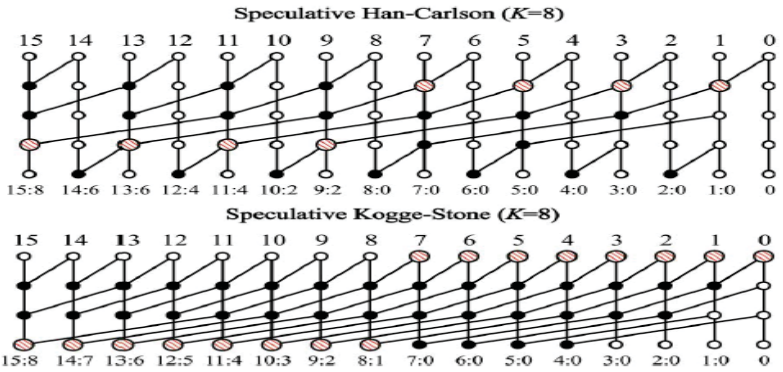

Figure. 3: The nodes of the prefix-processing stage, whose outputs are needed to compute the error signal, are named checking nodes" and are highlighted as big hatched dots, for the topologies.

As it can be obtained, in Kogge-Stone some of the checking cells are at the last level of the graph; their output signals are available after three black cells delay. In Han-Carlson the critical checking cells are in the second last level of the graph and are also available after three black cells delay, in spite of the larger number of levels of the Han-Carlson prefix-processing stage .From the above observations, it can be concluded that error detection is sensibly simplified and potentially faster in Han- Carlson, compared to Kogge-Stone.As an additional note, the need of driving the gates of the error detection stage increases the fan-out of the checking cells, slowing the speculative prefix-processing stage.

E .Error Correction

The blunder remedy stage registers the careful convey signals (5), to be utilized as a part of instance of mis prediction. The mistake remedy stage is made by the levels out of the prefix-handling stage pruned to acquire the Speculative adder. The Figure.4 shows the error correction stage of the proposed speculative Han-Carlson adder; the error correction for Kogge-Stone topology can be obtained similarly. It can be observed that the inclusion of the error correction stage increases the fan-out of some of the cells of the speculative prefix-processing stage, with adverse effect on adder speed.

F.Post-Processing

The approximate carries are already available at the output of the prefix-processing stage. The post-processing, according to (9), is equal to the one of a non-speculative adder and consists of n xor gates.

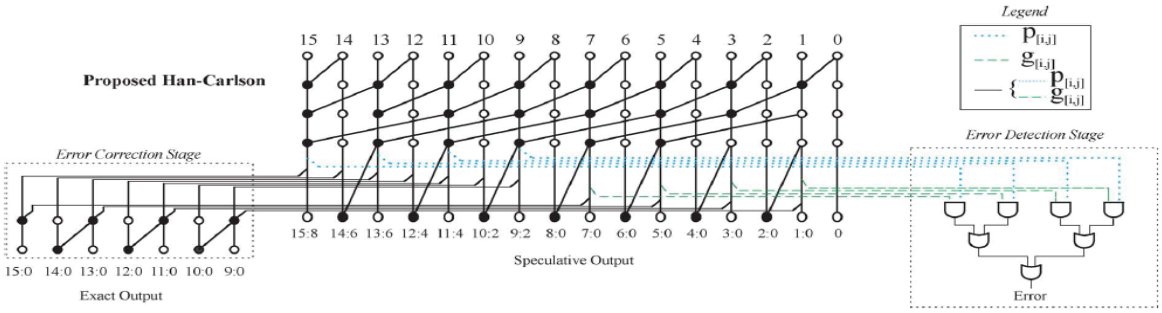

Figure4: Error correction, detection stages for the proposed speculative Han-Carlson adder of Figure. 2

IV. ADDERS CHARACTERIZATION

In this section we provide a characterization of the spatial and timing complexity of the investigated variable latency speculative adders, using either Han-Carlson or Kogge-Stone topologies. Results for non-speculative adders are also reported, for comparison. This will be achieved with the help of simplistic hypotheses on area and speed of employed gates, with the aim of obtaining an analytic comparison (albeitApproximated) between the various topologies. Accurate values of area, speed and power. Results of error rate analysis will also be reported at the end of this section.

A. Spatial and Timing Complexity

In order to estimate adder complexity, we make some simplistic hypotheses. We assume that the spatial complexity of speculative prefix processing and error correction is proportional to the number of employed black cells (AO gates). Error detection spatial complexity is simply estimated assuming that it is composed by a set of AND gates. According to the model proposed in [5] we assume as unit gate a basic 2-input gate, such as AND gates and OR gates.

B. Error Rate Analysis

The value of error probability is fundamental to understand the degradation of average addition time caused by mis prediction. In order to evaluate error probabilities, the proposed speculative Han-Carlson and the Kogge-Stone topologies have been simulated by using a Monte Carlo approach with a 1% relative error and a 99% confidence level .Input vectors have been chosen uniformly distributed [4],[5].

V. RESULTS

Verilog descriptions of the proposed variable latency speculative adders, and of their non-speculative counterpart. It is not easy to compare performances (in terms of power, speed, and area) of different designs, since they strongly depend on timing constraint used during synthesis. The adders we described in following discussions are CLA (Carry Look Ahead Adder) and HCA (Han Carlson Adder).

	CLA	KOGGE-STONE	HCA
Delay	**22.95ns**	**21.82ns**	**20.82ns**
Number of Slices	**1%**	**1%**	**1%**
Number of 4 i/p LUT's	**1%**	**1%**	**1%**
Number of I/O	**23%**	**21%**	**21%**

Table 1: comparison table of Han-Carlson Speculative adder and carry look Ahead adder

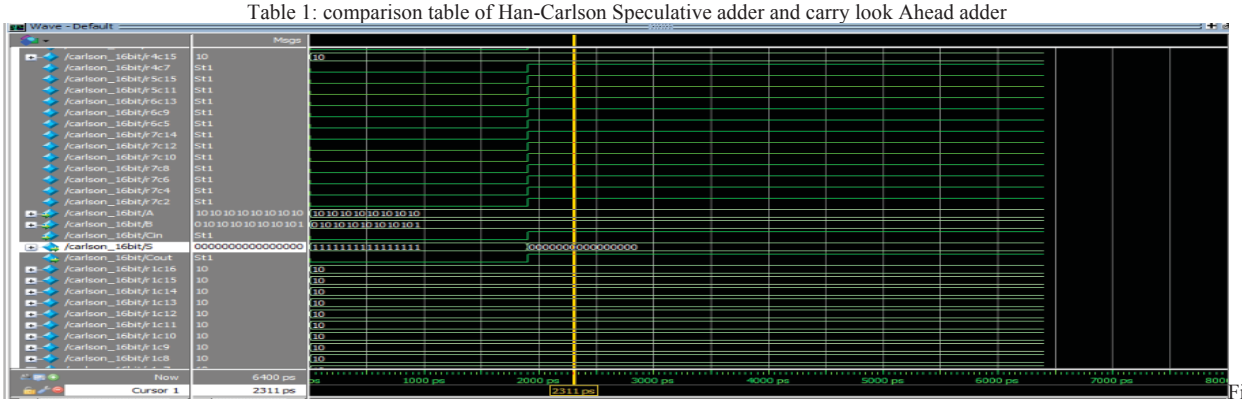

Figure 5: shows it shows the Simulation and synthesis of Han-Carlson adder using ModelSim altera10.1bfor16 bit binary variable

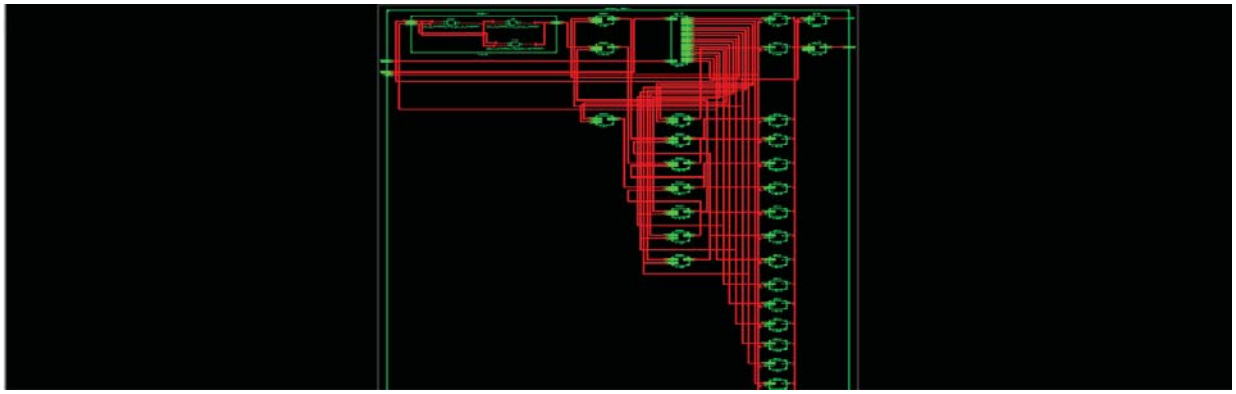

Figure 6: RTL schematic of Han Carlson adder

VI. CONCLUSION

In this paper a novel variable inertness Han-Carlson parallel prefix speculative adder for fast application is proposed. A new, more accurate, error detection network is introduced, which allows reducing the error probability compared to the previous approaches. Compared with traditional, non-speculative, adders, our analysis exhibits that variable inertness Han-Carlson adders show sensible upgrades when the most elevated velocity is required; generally the weight forced by error detection and error correction stages over powers any advantage. Additional work is required to extend the speculative approach to other parallel-prefix architectures, such as Brent-Kung, Lander-Fisher, and Knowles.

REFERENCES

[1] R.P Brent and H.T.Kung ".A regular layout for parallel adders," IEEE Trans.Computers, volume. C-31, no. 3, pp. 260.264, Mar. 1982.

[2] P. M. Kogge and H. S. Stone, "A parallel algorithm for the efficient solution of a general class of recurrence equations,"IEEE Trans. Computer., vol. C-22, no. 8, pp. 786–793, Aug. 1973.

[3] T. Han and D. Carlson, .Fast area-efficient VLSI Adders, In Proc. 8th Symp.Comp. Arithmetic, Sept. 1987, pp. 49.56.

[4] S. M. Nowick, "Design of a low-latency asynchronous adder using speculative completion," IEE Proc. Computer. Digit. Tech., vol.143, no.5, pp. 301–307, Sep. 1996.

[5] A. K. Verma, P. Brisk, and P. Ienne, "Variable LatencySpeculativeAddition: A New Paradigm for Arithmetic Circuit Design,"inProc.Design, Auto. Test Eur. (DATE '08), Mar. 2008, pp. 1250–1255.