ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 5, Special Issue 2, March 2016

4th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '16)

10th-11th March 2016

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

An Area Efficient, Low Power and High Speed Speculative Han-Carlson Adder

C.Dhanalakshmi[1], C. Manjula[2]

PG Scholar, Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India[1]

Asst. Professor, Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India[2]

ABSTRACT: Binary addition is one of the most important arithmetic function in modern digital VLSI systems. Adders are extensively used as DSP lattice filter where the ripple carry adders are replaced bytheparallelprefixaddertodecreasethedelay.Therequirementoftheadderisthatitisfastandsecondlyefficientintermso fpowerconsumptionandchiparea.Parallelprefixadder isatechnique forimproving thespeedoftheaddition.Parallelprefixaddersprovideagoodtheoreticalbasistomakeawiderangeofdesigntradeoffsinte rmsofarea,delayandpower.Thistechniqueismoresuitedforadderswithwiderwordlengths.Inthisproject,amodifiedPar allelPrefix speculativeHan-CarlsonAdderisintroducedwhichusesdiffereBinary addition is one of the most important arithmetic function in modern digital VLSI systems. Adders are extensively used as DSP lattice filter where the ripple carry adders are replaced by the parallel prefix adder to decrease the delay. The requirement of the adder is that it is fast and secondly efficient in terms of power consumption and chip area. Parallel prefix adder is a technique for improving the speed of the addition. Parallel prefix adders provide a good theoretical basis to make a wide range of design tradeoffs in terms of area, delay and power. This technique is more suited for adders with wider word lengths. In this project, a modified Parallel Prefix speculative Han-Carlson Adder is introduced which uses different stages of Brent-Kung and Kogge-Stone adders which reduces the complexity of the adder design.

KEYWORDS: Addition, digital arithmetic, parallel-prefix adders, speculative adders, speculative functional units, variable latency adders.

I. INTRODUCTION

VLSI binary adders are critically important elements inprocessor chips, they are used in floating-point arithmeticunits, ALUs, memory addresses program counter update andmagnitude comparator [1, 2]. Adders are extensively used asa part of the filter such as DSP lattice filter [3]. Ripple carryadder is the first and most fundamental adder that is capable
of performing binary number addition. Since its latency isproportional to the length of its input operands, it is not veryuseful. To speed up the addition, carry look ahead adder isintroduced. Parallel prefix adders provide a good results ascompared to the conventional adders.The adders with the large complex gates will be too slow forVLSI, so the design is modularized by breaking it into treesof smaller and faster adders which are more readilyimplemented. For large adders the delay of passing thecarry through the look-ahead stages becomes dominated andtherefore tree adders or parallel prefix adders are used. High speed adders depend on the previous carry to generate thepresent sum. In integer addition any decrease in delay willdirectly relate to an increase in throughput. In nanometerrange, it is very important to develop addition algorithm thatprovide high performance while reducing power.Parallel prefix adders are suitable for VLSI implementationsince they rely on the use of simple cells and maintainregular connection between them. We can define each prefix
Structures in terms of logic levels, fanout and wiring tracks.Zero or more inverters are added to each prefix cell output tominimize the delay based on this model, buffers areindividually sized to minimize the delay, buffers are used tominimize the fanout and loading on gates since high fanout causes poorperformance.we design an

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 5, Special Issue 2, March 2016

4th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '16)

10th-11th March 2016

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

extremely fast unreliable adder that produces correct results for the vast majority of input combinations. For brevity, we will call this adder an *Almost Correct Adder (ACA).*

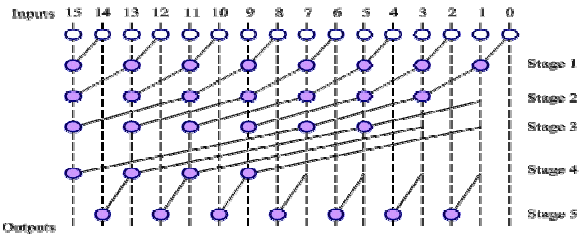

Fig 1. Graph representation of 16-bit Hybrid Han-CarlsonAdder

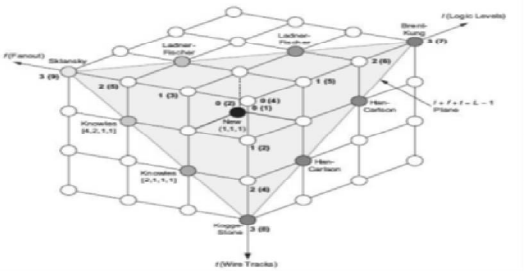

Fig 2. Taxonomy of prefix networks

II. PROPOSEDWORK

Design the Speculative Han-Carlson Adder. It differs from other adder in that it can be used for large word sizes. The proposed design reduces the number of prefix operation by using more number of Brent-Kung stages and lesser number of Kogge-Stone stages. This also reduces the complexity, silicon area and power consumption significantly.

Variable latency speculative prefix adders can be subdivided in five stages: pre-processing, speculative prefix-processing, post-processing, error 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.

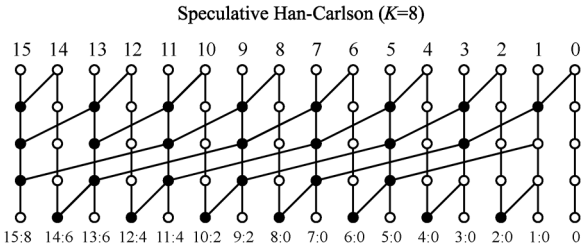

Fig. 3. Han-Carlson speculative prefix-processing stage.

Consider the *n*-bit addition of two numbers: $A = an-1, an-2,a0$ and $B = bn-1, bn-2,b0$ resulting in the sum, $S = sn-1, sn-2,s0$ and a carry, *C*out. The first stage in CLA computes the bit generate and bit propagate as follows:

$$gi = ai \cdot bi$$
$$pi = ai + bi, \quad (1)$$

Where*gi*is the bit generates and *pi* is the bit propagate. The schematic of *gi*and *pi* using CMOS and transmission gates design style. These are then utilized to compute the final sum andCarry bits, in the last stage as follows:

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 5, Special Issue 2, March 2016

4th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '16)

10th-11th March 2016

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

$si = pi \oplus ci,$
$Ci+1 = gi + pi \cdot ci,$ (2)

Where·, + and represent AND,OR, and XOR operations. It is seen that the first and last stages are intrinsically fast because they involve only simple operations on signals local to each bit position. However, intermediate stages embody the long-distance propagation of carries, as a result of which the performance of the adder hinges on this part [10]. These intermediate stages calculate group generate and group propagate to avoid waiting for aripple which, in turn, reduces the delay. These group generate and propagates are given by

$Pi: j = Pi:k \cdot Pk-1: j ,$
$Gi: j = Gi:k + Gk-1: j \cdot Pi:k.$ (3)

There are many ways to develop these intermediate stages, the most common being parallel prefix. Many parallel prefix networks have been described in the literature, especially in the context of addition. In this paper, we have used the Kogge-Stoneimplementation, Hans-Carlson,Sklansky, Brent-Kung implementation of CLA, and Kogge- Stone implementation of Ling adder. PG logic in all addersis generally represented in the form of cells. These diagrams known as cell diagrams will be used to compare a variety of adder architectures in the following sections. Here two cells are used for implementation of all the adders: grey cell and the black cell.

*Han-Carlson Adder:*The *Han-Carlson* trees are a family of networks between Kogge-Stone and Brent-Kung. The logic performs Kogge-Stone on the odd numbered bits and then uses one more stage to ripple into the even positions.
Kogge-Stone Adders: The main difference betweenKogge-Stone adders and other adders is its high performance. It calculates carries corresponding to every bit with the help of group generate and group propagates. In this adder the logic levels are given by log2*N*, and fan-out is 2.

$di = ai \oplus bi.$ (4)

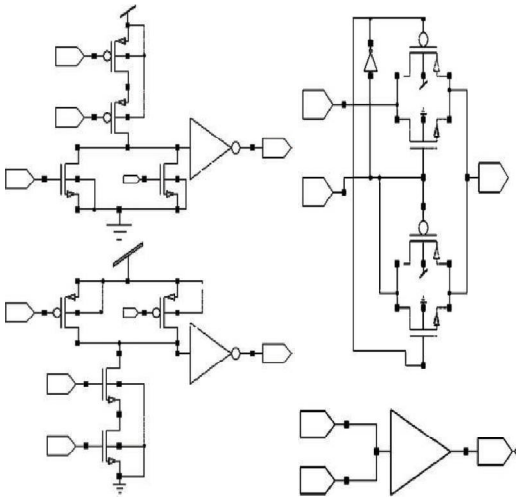

Figure 4: Block generate and propagate (Ling carry) using CMOS and transmission gate.

Now, instead of utilizing traditional carries, a new of carry, known as Ling carries, is produced where the *i*th Ling carry in [11] is defined to be
$ci = Hi \cdot pi,$ (5)

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 5, Special Issue 2, March 2016

4th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '16)

10th-11th March 2016

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

Where
$Hi = ci + ci-1.$ (6)
In this way, each *Hi* can be in turn represented by

$$Hi = gi + gi-1 + pi-1 \cdot gi-2 + \cdots + pi-1 \cdot pi-2 \cdot pi-3 \cdot \ldots p1g0. \quad (7)$$

We can see from (5) that Ling carries can be calculated much faster than Boolean carry. Consider the case of *c*4 and *H*4

$$c4 = g4 + p4 \cdot g3 + p4 \cdot p3 \cdot g2 + p4 \cdot p3 \cdot p2 \cdot g1 + p4 \cdot p3 \cdot p2 \cdot p1 \cdot g0, \quad (8)$$
$$H4 = g4 + g3 + p3 \cdot g2 + p3 \cdot p2 \cdot g1 + p3 \cdot p2 \cdot p1 .g0.$$

If we assume that all input gates have only two inputs, we cansee that calculation of *c*4 requires 5 logic levels, whereas thatfor *H*4 requires only four. Although the computation of carryis simplified, calculation of the sum bits using Ling carries is much more complicated. The sum bit, when calculated byusing traditional carry, is given to be

$$si = di \oplus ci - 1. \quad (9)$$

Substituting (5) into (9), we get that

$$si = di \oplus pi - 1 \cdot Hi - 1. \quad (10)$$

However, according to [12] the computation of the bits *si*can be transformed as follows:

$$si = Hi-1 \cdot di + Hi-1(di \oplus pi-1). \quad (11)$$

Equation (11) can be implemented using a multiplexer with*Hi*−1 as the select line, which selects either *di* or (*di* ⊕*pi*−1).No extra delay is added by Ling carries to compute the sumsince the delay generated by the XOR gate is almost equalto that generated by the multiplexer and that the time takento compute the inputs to the multiplexer is lesser than thattaken to compute the Ling carry.. Here, for *n*-bitaddition, Ling carry *Hi* and *Hi*+1 is given by

$$Hi = (G*i, P*i-1).(G*i-2, P*i-3). \cdots (G* 0, P*-1),$$
$$Hi+1 = (G* i+1, P* i).(G* i-1, P* i-2) .\cdots .(G* 1, P* 0), \quad (12)$$

Where
$$G*I = gi + gi-1, \quad P*I = pi \cdot pi-1. \quad (13)$$

$$H3 = g3 + g2 + p2 \cdot g1 + p2 \cdot p1 \cdot g0, \quad (14)$$
$$H4 = g4 + g3 + p3 \cdot g2 + p3 \cdot p2 \cdot g1 + p3 \cdot p2 \cdot p1 \cdot g0.$$

This can be further reduced by using (13) to

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 5, Special Issue 2, March 2016

4th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '16)

10th-11th March 2016

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

$$H3 = G*3 + P*2 \cdot G*1,$$
$$H4 = G*4 + P*3 \cdot G*2 + P*3 \cdot P*1 \cdot G*0. \quad (15)$$

This can be then further reduced by using the "." operatorto

$$H3 = (G*3, P*2).(G*1, P*0),$$
$$H4 = (G*4, P*3).(G*2, P*1).(G*0, P*-1). \quad (16)$$

This allows the parallel prefix computation of Ling adders using a separate tree [9] for even and odd indexed positions. Using this methodology, we implemented a 16-bit adder using the Kogge-Stone tree and then utilized that block to develop 32 and 64-bit adders. The gates and blocks used for this implementation were then modified using transmission gates. Cells other than gray and black cell that are used as components in Ling adder.

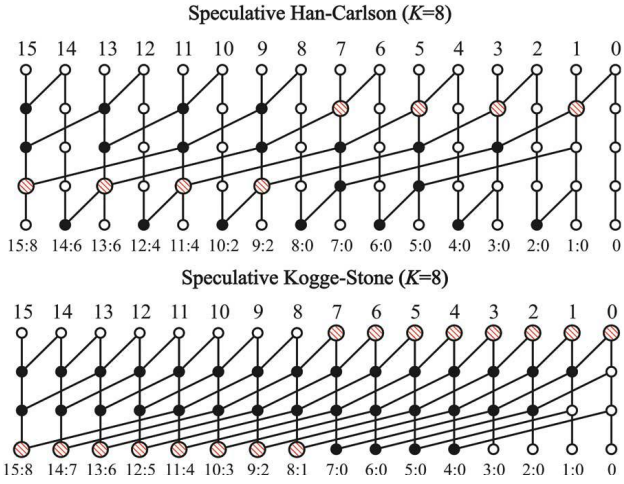

Fig 5.The nodes of the prefix-processing stage, whose outputs are needed tocompute the error signal, are named "checking nodes" and are highlighted asbig hatched dots.

III. ADDERS CHARACTERIZATION

3.1 Error Detection

This section presents a circuit that flags an error if the sumcomputed by the ACA is incorrect. This only occurs when thereis a chain of more than k propagates in the addenda. To checkfor the presence of an error, we must consider all chains of length$k$ + 1, and check if any of them contain solely propagates. Theexpression for error signal is stated as follows:

$$ER = \sum_{i=0}^{n-k-1} pipi + 1 \ldots . pi + k.$$

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 5, Special Issue 2, March 2016

4th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '16)

10th-11th March 2016

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

The critical path delay to compute error signal has complexity$O$ (log k + log (n − k)). Since k = O(log n), the error signalcomplexity can be reduced to O (log n). The critical path delay oferror detection has the same complexity as that of the critical pathdelay of a traditional adder; however, the error detector only requiressimple gates, such as AND, OR, etc.,. Experimentally, wereport that the delay of the error detection signal is approximatelytwo-thirds of the delay of a traditional adder.

3.2 Error Recovery

Once an error has been detected, one could simply employ atraditional correct adder to produce the sum. Instead, we have developeda novel error recovery technique that uses a computationinside the ACA to reduce both the critical path delay and hardwarearea.The matrix product $MiMi-1,Mi-k+1$ computesthe propagateand generate signals for the block between bitposition $i$and $i-k+1$. Thus, the ACA computes the propagateand generate signals for each k-bit block. If we divide the inputintegers into n/k blocks of k-bits, the values of propagate andgenerate for each block can be taken from the ACA. An n/k-bitcarry look-ahead adder then takes these values and computes thecarry for each of the blocks. Meanwhile, wecompute the propagate and generate signals for each bit in a block.
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. As it can be observed, 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.

Table 1: Delay and power of Han-Carlson Speculative adder

	Han- Carlson Adder
Delay	9.810 ns
Power	0.016W (or) 160mW

IV. VARIABLE LATENCY SPECULATIVE ADDER

We observe that for 16 bits, the delay of the ACAand error detection mechanisms are approximately equal and, individually, both are significantly less than the delay of a traditional adder. Based on this observation, we have designed the circuit shown in Fig. 6 whose clock period is slightly greater than the critical path delay of the error detection circuit. After one cycle, the circuit produces the result of the ACA and a bit indicating whether or not an error has been detected. If there is no error, then the circuit provides *SUM* □ as its output and will also set the *VALID* bit to 1. Since *STALL* is the complement of valid, the circuit will be ready to accept a new set of input addenda. If an error occurs, the valid bit will be set to 0 and the stall bit will be set to 1. After two cycles, the corrected sum value will be available and the valid bit will be set to 1. At this point, the circuit is ready to accept new inputs. We call this type of adder a *Variable Latency Speculative Adder (VLSA)*..

V. SYNTHESIS 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.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 5, Special Issue 2, March 2016

4th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '16)

10th-11th March 2016

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

5.1 The Optimal K Choice

Comparison between variable latency adder and the non-speculative Han-Carlson topology reveal that variable latency adders allow to reduce the minimum achievable delay. For instance, in the 64 bit case, the minimum achievable delay is about 280 ps for the non-speculative adder and reduces up to 225 ps in the variable latency architecture.

5.2 Comparison with Kogge-Stone Variable LatencySpeculative Adder

Fig. 7 shows the comparison between proposed speculative adder and Kogge-Stone one. Also in this case, we report the performance of non-speculative adders, in order to identify theregion where the speculative approach is effective.As an example, focusing on 64-bit adders, for lower than 350 ps, the proposed Han-Carlson speculative adder is the best choice in terms of silicon area and power consumption. Moreover, it allows to reduce the minimum achievable to 225 ps, with a 18% improvement respect to Kogge-Stone non-speculativeadder and a 11% improvement respect to Kogge-Stonespeculative adder.

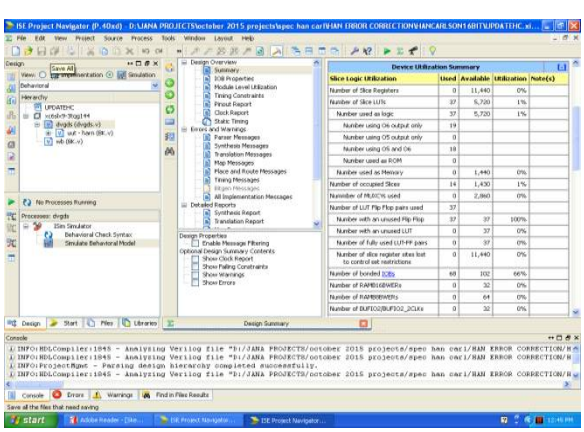

Fig: 7(a)

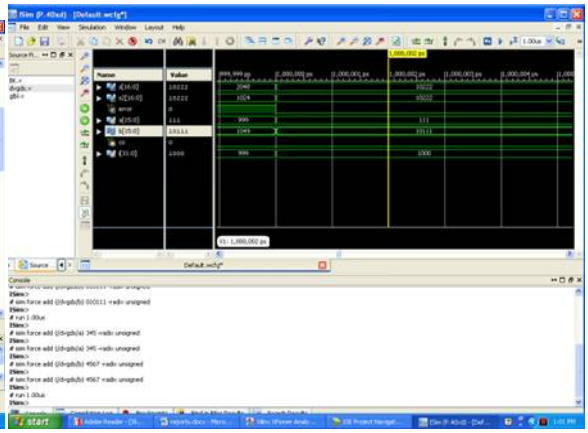

Fig: 7(b)

Fig. 7(a). Area report of Han-Carlson Adder

7(b). error correction report of Han- Carlson Adder

VI. CONCLUSION

In this paper a novel variable latency Han-Carlson parallel prefix Speculative adder for high-speed application is proposed. An accurate error detection network is implemented to reduce the error probability compared with kogge stone adder. Variable latency adder performance mainly depends on prefix-processing stage. Speculative latency adder reduces the number of black cells. Compared with traditional, non-speculative, adders, our analysis demonstrates that variable latency Han-Carlson adders show sensible improvements when the highest speed is required; otherwise the burden imposed by error detection and error correction stages overwhelms any advantage. Additional work is required to extend the speculative approach to other parallel-prefix architectures, such as Brent-Kung, Ladner-Fisher, and Knowles.

REFERENCES

[1] I. Koren, Computer Arithmetic Algorithms. Natick, MA, USA: A KPeters, 2002.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 5, Special Issue 2, March 2016

4th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '16)

10th-11th March 2016

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

[2] R. Zimmermann, "Binary adder architectures for cell-based VLSI andtheir synthesis," Ph.D. thesis, Swiss Federal Institute of Technology,(ETH) Zurich, Zurich, Switzerland, 1998, Hartung-GorreVerlag.
[3] R. P. Brent and H. T. Kung, "A regular layout for parallel adders,"IEEE Trans. Comput., vol. C-31, no. 3, pp. 260–264, Mar. 1982.
[4] P. M. Kogge and H. S. Stone, "A parallel algorithm for the efficientsolution of a general class of recurrence equations," IEEE Trans.Comput., vol. C-22, no. 8, pp. 786–793, Aug. 1973.
[5] J. Sklansky, "Conditional-sum addition logic," IRE Trans. Electron.Comput., vol. EC-9, pp. 226–231, Jun. 1960.