International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 13
October 2017

A New Design for Variable Latency Speculative E.C&D Han-Carlson Adder

K . KARTHIK
M.Tech (VLSI)
Dept of E.C.E
Chaitanya Institute of Technology and Science,
Warangal, Telangana,India
Email: karthichinna452@gmail.com

B. RAJESHWAR
Assistant Professor
Dept of E.C.E
Chaitanya Institute of Technology and Science,
Warangal, Telangana,India
Email: rajeshwar713@gmail.com

ABSTRACT: Variable latency adders have been recently proposed in literature. In variable latency adder unwanted interconnections also reduced compared with kogge-stone topology. Kogge-Stone adder consists of large number of black cells and many wire tracks. A variable latency adder employs speculation: the exact arithmetic function is replaced with an approximated one that is faster and gives the correct result most of the time, but not always. In order to detect the error, error detection network is also used. The approximated adder is augmented with an error detection network that asserts an error signal when speculation fails. Speculative variable latency adders to reduce average delay compared to traditional architectures.

This paper proposes a novel variable latency speculative adder based on Han-Carlson parallel- prefix topology which proposes the error detection network that reduces error probability compared to previous approaches. Several variable latency speculative adders, for various operand lengths, using both Han-Carlson and Kogge- Stone topology, have been synthesized using Xilinx 14.3. Obtained results show that proposed variable latency Han-Carlson adder used in high-speed application.

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

I.INTRODUCTION

VLSI binary adders are critically important elements in processor chips, they are used in floating-point arithmetic units, ALUs, and memory addresses program counter update and magnitude comparator [1, 2]. Adders are extensively used as a part of the filter such as DSP lattice filter.

Ripple carry adder is the first and most fundamental adder that is capable of performing binary number addition. Since its latency is proportional to the length of its input operands, it is not very useful. To speed up the addition, carry look ahead adder is introduced. Parallel prefix adders provide good results as compared to the conventional adders. The adders with the large complex gates will be too slow for VLSI, so the design is modularized by breaking it into trees of smaller and faster adders which are more readily implemented. For large adders the delay of passing the carry through the look-ahead stages becomes dominated and therefore tree adders or parallel prefix adders are used.

High speed adders depend on the previous carry to generate the present sum. In integer addition any decrease in delay will directly relate to an increase in throughput. In nanometer range, it is very important to develop addition algorithm that provide high performance while reducing power. Parallel prefix adders are suitable for VLSI implementation since they rely on the use of simple cells and maintain regular connection between them. We can define each prefix.

Structures in terms of logic levels, fan-out and wiring tracks. Zero or more inverters are added to each prefix cell output to minimize the delay based on this model, buffers are individually sized to minimize the delay, buffers are used to minimize the fan-out and loading on gates since high fan-out causes poor performance. We design an 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).

II.EXISTING METHOD

The existing adder uses parallel prefix adders and speculates the output, where the exact output is replaced with the speculating one. This adder speculates the output instead of giving exact output. This has been considered the fastest when compared with the previous adders. This adder comes with an error which has to be detected and then corrected. A network has been designed for assessment of the output with an error, at this point one more clock cycle is used to obtain the error less output. The addition time is considered one clock cycle when there is no error and the clock cycle is considered to be two when the speculation fails and error is present.

This adder is main based on the critical path. The output of the unit depends upon all the previous bits, so the output of the adder depends upon all the n input bits of the adder.

A first based on guessing approach to addition was suggested by Nowick in (not happening at the same time) contest, which puts into use a changeable delay something that adds cutting the lowest levels of a Kogge Stone adder. In (two or more things happening at the same time) contest, Verma propose a changeable delay based on some guessing adders; here the based on guessing addition is done in the same way as, cutting the lower levels of a Kogge-Stone, something that adds. In a changeable delay carry-select something that adds is introduced, where the adder is broken-up in different windows, each one contains a Kogge-Stone adder.

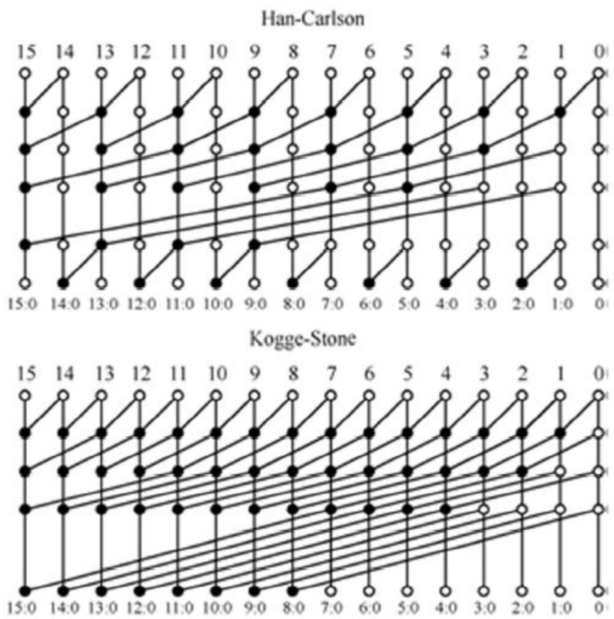


Fig 1. 16 bit Han Carlson and Kogge Stone Topologies

Variable Latency Speculative Adder

The general prefix addition can be done as follows: given *n*-bit augends *A* and *B* which generates the *n-bit* sum, *ci* is the carry out of *i*-th bit. The sum *si* and carry *ci* can be calculated as following

$$s_i = (a_i) xor (b_i) xor (c_{i-1})$$

$$c_i = a_i b_i + a_i c_{i-1} + b_i c_{i-1}$$

Three stages are used to compute the sum; pre processing, prefix processing and post processing. In the pre processing stage the generate *gig* as propagate *pi* signal are computed as follows

$$gi = ai.bi \quad (4)$$

$$p_i = (a_i) xor (b_i) \quad (5)$$

The pre processing and post processing stages are having only simple operations on signals local to each bit position; hence the adder's performance mostly depends on the prefix processing stage. Fig 1, shows the topologies of Kogge Stone and Han Carlson adders. There the black balls represent the prefix operator and white balls represent the simple placeholders. Kogge Stone adder is composed by log2*(n)* levels and also fan-out of two is present at the each level using a large count of black balls and wire tracks. A good tradeoff between fan-out and number of logic levels is given by Han Carlson as well as the tradeoff between the number of logic and the black balls is given by Han Carlson.

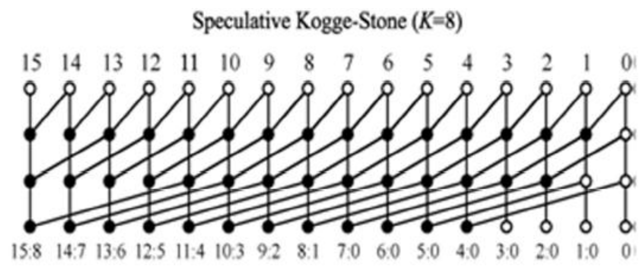


Fig 2. Kogge Stone speculative prefix processing stage where the last row of 16 bit Kogge Stone adder is been pruned which results in speculative adder with K = 8.

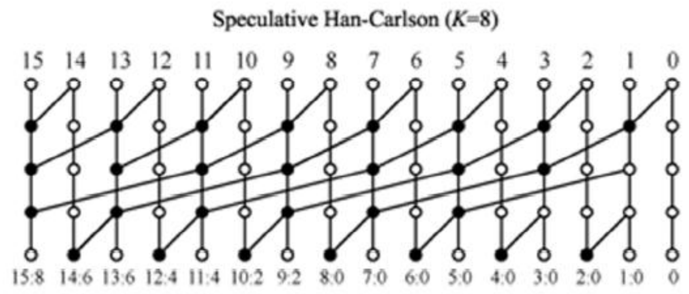


Fig 3. Han Carlson speculative prefix preprocessing stage where the last row of Han Carlson adder is been pruned to achieve speculative adder with K = 8.

The Kogge-Stone adder is generally used when speed is the first (or most important) concerns the most, as it uses the lowest possible number of logic levels and each cell in the adder tree has fan-out of two. This comes at the cost of using many spread-create cells and many wires that required to be routed between stages. As it is shown in the fig 1, it requires 6 stages to achieve output.

The Han Carlson adder is basically made of two adders, the outside rows of the Han Carlson adder are Brent Kung and similarly inside rows are Kogge Stone graphs. The Han Carlson in fig 1 is made of Brent Kung where single level at the beginning and single level at the bottom are Brent Kung graph, the number of levels are 1+log2(*n*).

The normal stage has been divided in 3 stages where as the variable latency speculative prefix adder can be divided in to 5 stages which are as follows; pre processing, speculative prefix processing, post processing, error detection and error correction stages. In the speculative stage the last rows of the adders are been pruned thus replacing the exact output with the speculative one as shown in the fig 2 and fig 3. To overcome this issue two stages have been designed they are error detection and error correction stages as shown in the fig 4. The error correction stage is have 2 clock cycle in the case of misprediction and when speculation fails. The pre processing stage *gi* is been generated and as well as *pi* is been propagated.

In the error detection stage the conditions where at least one of the propagation carries is wrong are signaled by the error detection block. It has 2 clock cycles in case of any mis-prediction the error signal is backed up by error detection stage and output of post processing stage is discarded. The error correction stage will give the right output in the next stage. The error signal of either Kogge Stone or Han Carlson is calculated by or-ing the *pi* and *gi*. The error rate analysis i.e, the value of the error probability is fundamental to understand the degradation of average time by misprediction for that Han Carlson and Kogge Stone adders have been simulated by using the Monte Carlo approach with 1% relative error and 99% confidence level. The black balls are employs with AO gates. After the error signal has been detected by the error detection the error signal has to be corrected which is done by help of error correction stage.

III.PROPOSED METHOD

In the proposed method the adders are been replaced with the SQRT adder, it is also known as SQRT CSLA which uses the binary to excess converter instead of RCA with cin = 1 to achieve lower delay and less area. The common way of implementation of CSA is that it computes the (n + 1) bit sum of any two n-bit numbers. Here instead of using a couple of RCA blocks we are using the modified SQRT CSLA which developed using just single carry adder with BEC converter which also replaces RCA block for cin = 1 to improve the performance of the adder

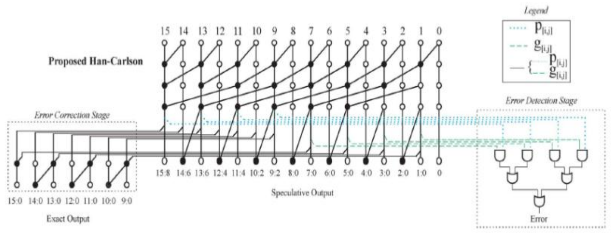


Fig 4: Speculative Han Carlson adder with the Error Detection and Error Correction stage

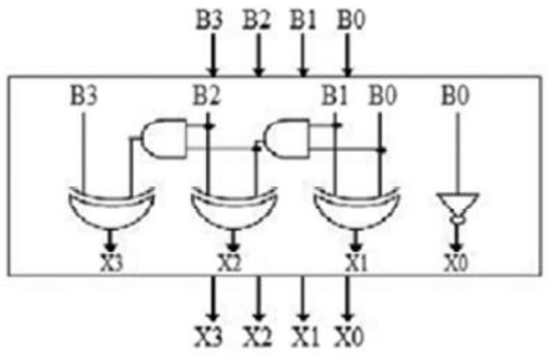


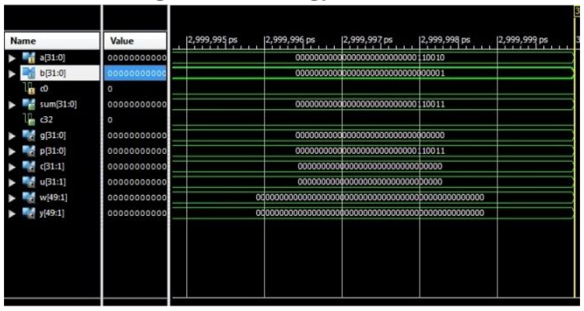
Fig 5: Binary to Excess one converter (BEC) circuit

The BEC replaces the RCA it shows better performance. The synthesized results are given in the below section as well as the simulated output. When the output of the Kogge Stone, Han Carlson and speculative

Han Carlson is compared the speculative Han Carlson performs better and when the SQRT adders are includes it almost show 5.77%improveent in the performance.

IV.RESULTS

Fig 6: Technology schematic



```
Name        Value          |2,999,995 ps  |2,999,996 ps  |2,999,997 ps  |2,999,998 ps  |2,999,999 ps
a[31:0]     00000000000    00000000000000000000000000110010
b[31:0]     00000000000    00000000000000000000000000000001
c0          0
sum[31:0]   00000000000    00000000000000000000000000110011
c32         0
g[31:0]     00000000000    00000000000000000000000000000000
p[31:0]     00000000000    00000000000000000000000000110011
c[31:1]     00000000000    0000000000000000000000000000000
u[31:1]     00000000000    0000000000000000000000000000000
w[49:1]     00000000000    0000000000000000000000000000000000000000000000000
y[49:1]     00000000000    0000000000000000000000000000000000000000000000000
```

Fig 7: output waveform

V.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.

VI.REFERENCES

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

[2] R. Zimmermann, “Binary adder architectures for cell-based VLSI and their synthesis,” Ph.D. thesis, Swiss Federal Institute of Technology, (ETH) Zurich, Zurich, Switzerland, 1998, Hartung-Gorre Verlag.

[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 efficient solution 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.

[6] T. Han and D. A. Carlson, “Fast area-efficient VLSI adders,” in *Proc. IEEE 8th*

Symp. Comput. Arith. (ARITH), May 18–21, 1987, pp. 49–56.

[7] R. E. Ladner and M. J. Fischer, "Parallel prefix computation," *J. ACM*, vol. 27, no. 4, pp. 831–838, Oct. 1980.

[8] S. Knowles, "A Family of Adders," in *Proc. 14th IEEE Symp. Comput. Arith.*, Vail, CO, USA, Jun. 2001, pp. 277–281.

[9] S.-L. Lu, "Speeding up processing with approximation circuits," *Computer*, vol. 37, no. 3, pp. 67–73, Mar. 2004.

[10] T. Liu and S.-L. Lu, "Performance improvement with circuit-level speculation," in *Proc. 33rd Annu. IEEE/ACM Int. Symp. Microarchit. (MICRO-33)*, 2000, pp. 348.

Authors:

K. KARTHIK studying M.Tech (VLSI) from Chaitanya Institute of Technology and Science, Warangal, Telangana,India.

B. RAJESHWAR working **as** Assistant Professor in Dept of E.C.E from Chaitanya Institute of Technology and Science, Warangal, Telangana, India.