International Journal of Ethics in Engineering & Management Education
Website: www.ijeee.in (ISSN: 2348-4748, Volume 2, Issue 5, May 2015)

Design and Implementation High Speed with Compensation Circuit Using KOGGE Stone Adder Tree

RUBINA FIRDOUS
M.Tech student, VLSI Design & Embedded Systems
APPA Institute of Engineering & Technology
Gulbarga, Karnataka, India.
Rubina.firdous39@gmail.com

MAHESH R.K
Assistant Professor, Dept. of E&CE, VLSI Design & Embedded System APPA Institute of Engineering & Technology Gulbarga, Karnataka, India.
rkmahesh10@gmail.com

Abstract- **As the Kogge Stone Adder is a parallel prefix type of CLA. It can generate carry in 0 times and is far apart considered as the speedy adder and is a partly used in industry for high performance arithmetic circuit. Adders are the basic building blocks in digital integrated circuit based design. The research necessitates an investigation for the performances for these two adders in terms of design area and computational delay. By using the Quartus-II design software the design for Kogge Stone Adder is developed. The simulation result produce the vector waveform in which it then shows the computational delay in adder. Hence, this project is more significant which shows the adder being tested performing an excellent design area based and computational delay in different size of bits. This kogge stone adder use number of gates (XOR AND & OR). In previous related papers, the previous work has been done truncated multiplier in that paper use number of full adder and half adder, truncated adding the error.**

Key Words—Computer arithmetic, faithful rounding, fixed- width multiplier, tree reduction, truncated multiplier, kogge stone adder.

I. INTRODUCTION

Multiplication is large arithmetic operation, which are approximately high speed propagation delay, high power dissipation and large area needed [1]. To outcome about many researches works with low power design high speed multipliers. Two types of multiplication such as partial product and their sum, execute using two kinds of multiplication serial multiplication and parallel multiplication.

Serial multiplication algorithm procedure requires sequential circuit along with feedback structure.

Parallel multiplication algorithm frequently use combinational circuits, it doesn't include feedback structure.

Its maximum area consuming arithmetic operations, once selecting multiplication of two bit width digital signal numbers produce a product with twice of original bit width digital system, this one leads to unnecessarily high power dissipation, unnecessary large delay.

In general it required truncate partial product bits used to compress the area cost fixed width multipliers, N*N multiplication, a truncated multipliers work out only MSBs (most significant bits) part of 2N bit product plus use redundant correction/compensation circuits, to bring down the error of truncation

In earlier related papers, from error compensation circuits is use to reduce truncation error. Thus output will be précised.

We considers the tree reduction, truncation, and rounding from the partial product bits through design of fast parallel truncated multipliers.

Parallel multiplication is high speed multipliers, .Partial products step will have been partial product bits from the multiplicand and the multipliers. The intent of partial product reduction is to compress the partial product of two numbers and added up final addition. These over all pp bits are very large as many as possible parallel operation will have been done.

There are two important reduction trees.

- Dadda tree multiplier
- Wallace tree multiplier

Dadda tree multipliers executes only a couple of reduction, it is a least possible reduction strategy work out only when it required but it doesn't save carry addition.

Wallace tree multiplier every time compress, it diminishes as possible as tree reduction, as a result faster tree reduction because of a bit smaller carry propagation adder.

II.PREVIOUS WORK

II.ITRUNCATED MULTIPLIER

The result produced by truncated N*N multiplier is less than N+N bits. This truncated is use to cutting down the LSB part, due to cutting the LSB bit we get reduced area, power and delay and increased the speed. Here the truncated [3] is one method where as we are not much use the partial product bits in the part of least significant column. This shown in the below figure 1

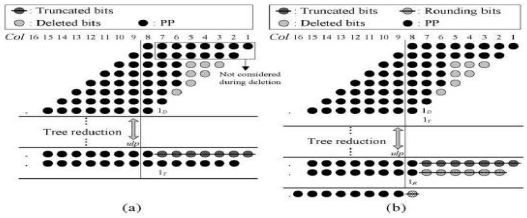

Figure 1. 8x8 truncated multiplication.(a) deletion,reduction and truncation. (b)deletion,reduction,truncation,and final addition.

International Journal of Ethics in Engineering & Management Education
Website: www.ijeee.in (ISSN: 2348-4748, Volume 2, Issue 5, May 2015)

The previous non truncated multipliers procedure is same as the present truncated multiplier. In truncated use to removes some bits to reduced area.

The Truncated multiplier contents very few operations as given below.

- *Deletion operation, reduction operation and truncation*
- *Deletion operation, reduction operation truncation and finally addition*

II.1Deletion operation, reduction operation and truncation.

Deletion operation[E_d]

The deletion operation is the very first step used to remove all the unnecessary partial product bits from the LSB bits side. This shown in figure 1(a) the deletion operation is deleting a lot of partial product bits as very possible. Deletion Error E_d has to be in the order

$$-1/2ulp \leq E_d \leq 0$$

Here the addition correction Error *bias constant* of ¼ **ulp.**

Doing the deletion operation its upper part of partial product bits is not considered, so it got deleted. After the bias modification of deletion Error we get

$$-1/4upl \leq E_d \leq 1/4ulp$$

The deletion operation starts from column 3 of partial product. This method skipped starting two bits of partial product. This method is done by column –by – column after all deletion of partial product bits, so its reduction scheme2 use.

The deletion operation of reduction secheme2 implements the truncation .This truncation will be farther removes the first Row of [N-1] bits will be removed from Column [N-1] of the truncation.

This truncation Error [E_t] which is in the area of

$$-1/2ulp \leq E_t \leq 0$$

Therefore ¼ ulp truncation part introduce a different *bias constant.* As a result it adjusted truncated Error is

$$-1/4upl \leq E_t \leq 1/4ulp$$

II.2 Deletion operation, reduction operation truncation and finally addition

Rounding and final addition

Deletion, reduction and truncation operation all these operation are done, at last its added the partial product bits (MSB) part by using carry propagates addition(CPA) which is to generates a partial product of final bits. This shown in figure 1(b).

This final carry propagate addition (CPA), added for rounding bit operation 1/4 ulp is added a *bias constant.*

$$-1/4upl \leq E_r \leq 1/4ulp$$

Is the rounding Error [E_r] the total Error of truncated multiplier which is in the range of

$$-ulp < E = \{E_d + E_t + E_r\} \leq upl$$

Algorithm truncated

This design of multiplier which ***is 8*8*** bits, the methods of reduced bits are step by step. Which is the first step is deletion operation, removes the partial product bits in the least significant bits from the first stage 1. The total deletion Error is not more than 2^{-p-1}**.** Without increasing the Error it has been use the number of stages to reduce the final bit width. For this truncation the Error is less than the 1ulp.

This less power consuming, reduced the delay of various case in the multiplier, and also reduce area, because a carry propagate adder produce a product can also be short.

III. PROPOSED WORK

Here this paper kogge stone adder is implemented. By designing a kogge stone adder is high performance adder. Basically the arithmetic operations digital systems are multipliers and adders which are most commonly used in all the operation. The fastest and accurate operations are depends on the performance of adders. So these adders are used to speed up the partial product bits of addition operation and generate at same time of multiplication operation. Thus improve a speed through reduce area. There are several adders such as full adder, half adder, carry propagate adder which has been used for last paper (truncated multipliers) and kogge stone adder etc. A appropriate method for parallel adders so that delay [5] can be minimized, performance can be better. Therefore to reduce the calculation time, its faster ways by using Carry Look-Ahead (CLA) adder. Parallel prefix operation is used. For this paper we are using kogge stone adder because of this adder is faster than the carry propagate adder. Here the tree adders, generate parallel and fastest and increased area and power. For this most important advantage of design is that a carry reduce the number of levels. This is fastest adder compare to others. There are two stages such as propagation and generate.

Propagation. This is control the carry from lower bits to the higher bits.

$$\mathbf{P = X_i \text{ xor } Y_i}$$

Generate. This is control the carry generate.

$$\mathbf{G = X_i \text{ and } Y_i}$$

III.1 Kogge stone adder.

The kogge stone adder is also called as parallel prefix adder, [7]also generates a carry signals and generally consider as the faster adder design. It is high performance adder. These kogge stone adders are generating high speed with compute all in the parallel. Figure 2 shows the basic structure of kogge stone adder.

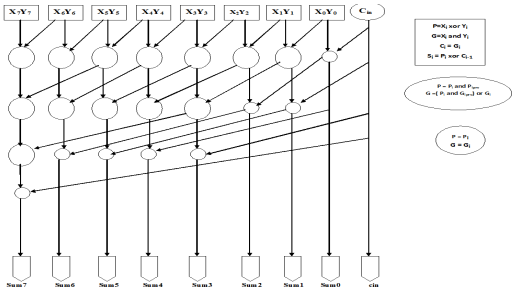

Figure 2: basic 8 bit kogge stone adder

The proposed kogge stone adder is a parallel prefix adder its is carry look adder form, and these generates the signals of carry in start bit zero (0) time, widely it consider as the fastest adder possible design. Most commonly it use for high performance adder in production.

International Journal of Ethics in Engineering & Management Education
Website: www.ijeee.in (ISSN: 2348-4748, Volume 2, Issue 5, May 2015)

The 8bit proposed kogge stone adder as shown in figure 3.

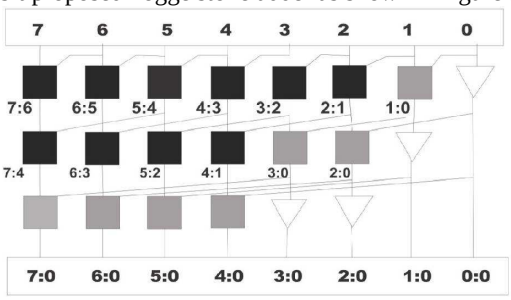

Figure 3: proposed diagram of 8 bit kogge stone adder

The kogge stone adder consists of three main components these are *black cell*, *gray cell* and *buffer*. The both signals generate and propagate are used in *black cells* this shown in figure, and only generate signal are used in gray cells this shown in figure but it needs the post processing stage. This buffers signal is used to balance the loading result.

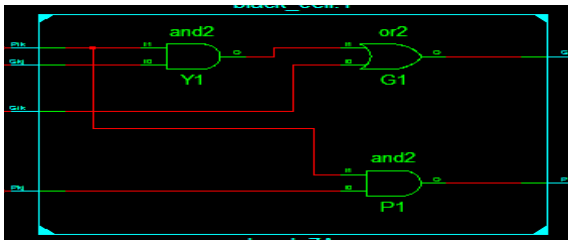

Figure 4: black cell

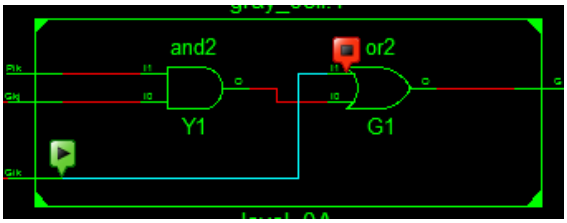

Figure 5: gray cell

Kogge stone adder design

There are five numbers of steps for doing kogge stone adder.

Level 1: The first stage is to generate propagation and also generation of signals for each and every bit.

Level 2: To create *black* and *grey cell equations*.

Level 3: In the every stage *grey cell* generated.

Level 4: Carry bits are directly generated by using the grey cell equation.

Level 5: The combination of carry bits and propagation bits for each and every stags produce the sum.

These are the equations of all the propagation, generation, sum, and lastly carry.

$$P = X_i \text{ xor } Y_i$$
$$P = P_i \text{ and } P_{iprv}$$
$$P = P_i$$
$$G = X_i \text{ and } Y_i$$
$$G = (P_i \text{ and } G_{iprv}) \text{ or } G_i$$
$$G = G_i$$
$$C_i = G_i$$
$$S_i = P_i \text{ xor } C_{i-1}$$

The most important advantage of this design is that the carry tree reduces the logic depth of the adder by basically generating the carries in parallel

The one significant concept in all tree adders is that every cell has its overall output. Example it generates left side of the second stage bit of 8-bit Kogge-Stone adder is calculated if only bit-6 and -7 is present. This only requires generate bit from bit-0 to -5, when Combined these two generated bits it gives the generate bit for the in general bit-0 to -7. The 6 and 7 generate bits are need take from the calculation, thus it is the important concept the calculation of the bit from 0 – 5 but alternate, so we can calculate bit 0 – 6 from generates, the result will be the same. Due to this generate bit as of 6 is included already but at the beings again we added as of calculation of generate bit of 0 – 6, does not change at the end of result.

IV SIMULATION RESULTS

The previous and proposed designs result are simulated in Xilinx 14.7 and the outputs are shown below. Experimental results shows of kogge stone adder bits delay, power.

Simulation result of previous paper.

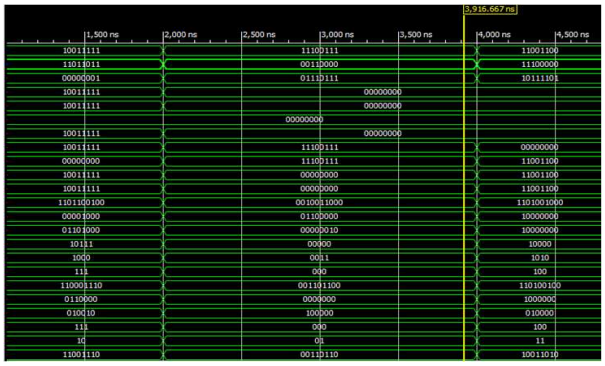

Figure 6: truncated multiplier

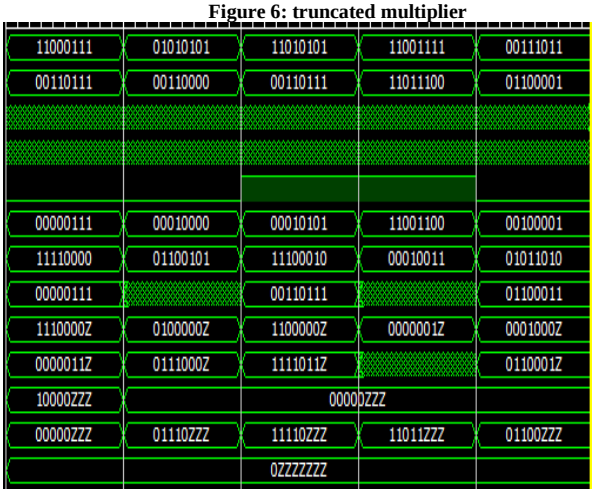

Figure 7: proposed kogge stone adder

International Journal of Ethics in Engineering & Management Education
Website: www.ijeee.in (ISSN: 2348-4748, Volume 2, Issue 5, May 2015)

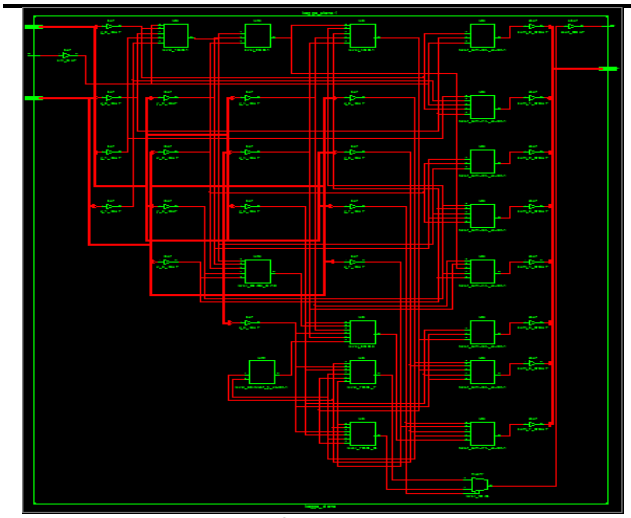

Figure 8: koggre stone adder of Routing Technique and RTL schematic

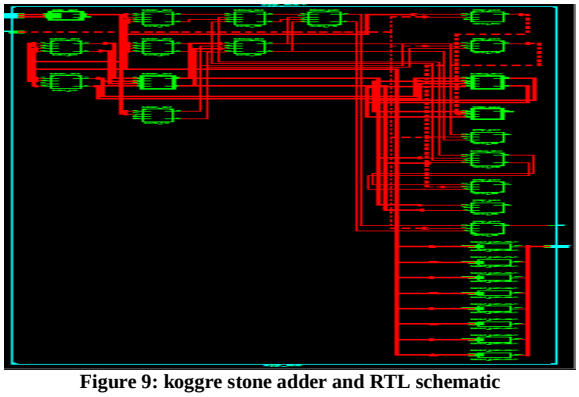

Figure 9: koggre stone adder and RTL schematic

V .RESULTS AND DISCUSSION

All the circuits mentioned above are compared with power, delay, and area and the table 1 show the present and previous method of parameters compare to present parameters are more accurate than the previous. Power results are shown I figure 11 delay results are shown in figure 10. here we can conclude that proposed work results are efficient with respect to power, delay and finally area than the last method truncated multiplier. The results are simulation by using Xilinx 14.7 tool.

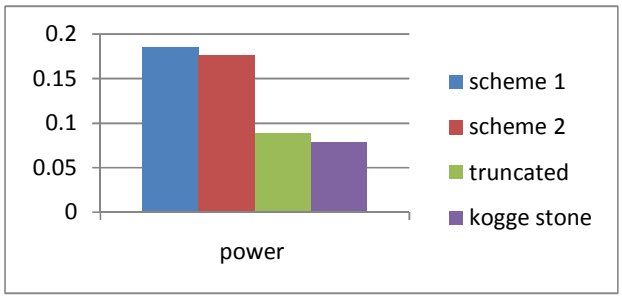

Figure 10: Power Dissipation (nwatts)

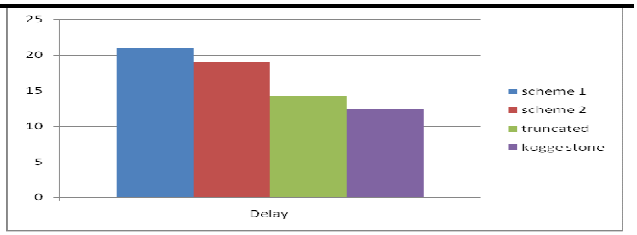

Fig 11: Delay (nsec)

Parameters	Scheme1	Scheme2	Truncated	Kogge stone adder
Power(nw)	0.185	0.176	0.088	0.078
Delay(nsec)	21	19	14.234	12.385

Table 1: Comparison of Power and Delay

V. CONCLUSION

In this paper the 8-bit kogge stone adder has been implemented using with the gates (XOR, AND & OR). The structure for the multiplication of 8 bits and this multiplier is to change into an adder by using a kogge stone adder. The proposed to 8bits and based on the N inputs bit by bit form that is least significant bit to most significant bit in the manner that each stages generate carries which have to give a next higher significant stages, these operation dependent only to a lower bit of significant stage. This kogge stone adder is very fast and high speed adder compare to the previous work done. It use less number of gates (XOR, AND & OR). It is high performance design. Reduce delay, power, and area. The kogge stone adder is very less area produced and also delays.

REFERENCES

[1]. J. M. Jou, S. R. Kuang, and R. D. Chen, "Design of lowerror fixed-width multipliers for DSP applications," IEEE Trans. Circuits Syst. II, s Analog Digit. Signal Process., vol. 46, no. 6, pp. 836–842, Jun. 1999.

[2]. L.-D. Van and C.-C. Yang, "Generalized low-error areaefficient fixed width multipliers," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 52, no. 8, pp. 1608–1619, Aug. 2005.

[3]. M. J. Schulte and E. E. Swartzlander, Jr., "Truncated multiplication with correction constant," in VLSI Signal Processing VI. Piscataway, NJ:IEEE Press, 1993, pp. 388–396.

[4]. E. J. King and E. E. Swartzlander, Jr., "Data-dependent truncation scheme for parallel multipliers," in Proc. 31st Asilomar Conf. Signals, Syst. Comput., 1997, pp. 1178– 1182.

[5]. CH. Chimpiraiah, E.V.Vijay, "An Efficient Architecture for Parallel Adder", International Journal of VLSI and Embedded Systems Vol. 03, Issues 04; Sep-Oct 2012.

[6]. D.H.K Hoe, C. Martinez, and J.Vundavalli, "Design and Characterisation of Parallel Prefix Adders using FPGAs", IEEE 43rd Southern Synopsium on System Theory, March 2011.

[7]. Kogge P and Stone H, "A Parallel Algorithm for the Efficient Solutions of a General Class of Recurrence Relations", IEEE Transactions on Computers, Vol. C-22, No.8, 1973.

[8]. Pakkiraiah Chakali, Madhu Kumar Patnala," Design of High Speed Kogge-Stone Based Carry Select Adder", International Journal of Emerging Science and Engineering, Vol.1, Issue-4, February 2013.