www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 2 Issue 6 June , 2013 Page No. 2050-2054

DESIGN AND IMPLEMENTATION OF A PIPELINED BCD MULTIPLIER USING KARATSUBA –OFMAN'S ALGORITHM

*Nanditha H G, *Swaminadhan Rajula

nanditha.hg28@gmail.com,swaminadhanreddy@gmail.com

Amrita school of engineering, Amrita Vishwa Vidyapeetham, kasavanahalli, Bangalore-560035

Abstract: *Decimal multiplication is an important and very frequent operation in financial,commercial applications.The Dominant representation for decimal digits is the bcd encoding.The BCD multiplier serve as the key block of decimal multiplier.This paper presents the design and implementation of bcd multiplier using karatsuba Ofman's algorithm .The results indicates that the proposed pipelined bcd multiplier processed on virtex -6 FPGA device are efficient in terms of area and delay compared to other multipliers with the same number of digits.*

Keywords: BCD, Karatsuba-Ofman's algorithm, Double Dabble algorithm, FPGA.

1. Introduction

Decimal arithmetic became a requirement in the computation of many applications, like financial and commercial, where the results must match those obtained by human calculations. This is why over recent years decimal operations have become popular. The new revision of the IEEE 754-2008 standard for floating point arithmetic includes specification decimal multiplication format serves to satisfies the development of high performance.[1]

Erle and Schulte proposed designs for fixed point decimal multiplications in order to reduce the critical path delay.[2] Kenney, Schulte and Erle proposed the design that uses the ideas of doubling and Quintupling which increases the operating frequency.[3]

Carlos Eduardo Minchola Guardia proposed the architecture of an efficient pipelined multiplier. The design is based on carry save addition (CSA) techniques in order to compress the partial product tree [4]. The majority of hardware designs are synthesized on FPGA platform due to its high functionality ease of programming and low cost.

This work presents the algorithm, architecture and FPGA implementation of an efficient BCD pipelined multiplier. Several assessments are carried out and the synthesis results in terms of area, delay are published. The outline of the paper is as follows.

Section 2 describes an overview of karatsuba-Ofman's algorithm. Section 3 applies the karatsuba-Ofman's algorithm to decimal multiplication. Section 4 describes Binary to BCD conversion. Section 5 Presents the synthesis and implementation results. Section 6 conclusion and proposes future work.

2. Overview of karatsuba –Ofman's algorithm

The karatsuba- Ofman's algorithm is a fast multiplication algorithm, it reduces the multiplication of two n-digit numbers to at most $3k^{\log_2 3}$ single –digit multiplications in general. Algorithm multiplies two n-digit numbers in $k^{\log_2 3}$ elementary steps. The method is based on a divide and conquers approach.[5]

Let A and B be the decimal representation of two integers

$A = \sum_{i=0}^{k-1} a_i 10^i$

$B = \sum_{i=0}^{k-1} b_i 10^i$

To compute the product P=AB, the operands A and B can be decomposed into equal sized parts

3. Decimal multiplication using Karatsuba – Ofman's Algorithm

Decimal multiplication can be designed using Karatsuba- Ofman's algorithm, the middle term of equation (1) can be expressed as

$A_H B_L + A_L B_H = (A_H + A_L)(B_H + B_L) - A_H B_H - A_L B_L$

So the product P is now expressed as

$P = 10^{2n} A_H B_H + A_L B_L + 10^n((A_H + A_L)(B_H + B_L) - A_H B_H - A_L B_L$ (2)

Given that the multiplication dominates asymptotically the complexity of the algorithm. To calculate this expression, we must determine the partial products $P_1 = A_H B_H$, $P_0 = A_L B_L$, and $P_2 = (A_H + A_L)(B_H + B_L)$.

With P_0, P_1 P_2, the Karatsuba-Ofman's equation [5] is calculated as

$P = 10^{2n} P_1 + P_0 + 10^n (P_2 - P_1 - P_0)$

Therefore after the calculation of P_2, we have to determine P_2-(P1+ P_0). The critical path includes one multiplier and two additions. Instead of calculating operations following this order, we have reordered the factors of the equation to reduce the critical path as follows.

$P = 10^{2n} P_1 + P_0 - 10^n (P_1 + P_0) + 10^n (P_2)$

This implementation has a critical path of one n-digit multiplier, one 2n-digit adder, one 3n-digit adder and one3n-digit subtraction

A_H, A_L, B_H and B_L which represent the higher and lower order digits.

$A = A_H 10^n + A_L$

$B = B_H 10^n + B_L$

Therefore, the product P=AB can be expressed as

$P = 10^{2n} A_H B_H + 10^n \left(A_H B_L + A_L B_H\right) + A_L B_L$

To implement this we need four decimal multiplications of size n, one addition of size N and another addition of size 3n

4. Binary to BCD Conversion

The double dabble algorithm is used to convert binary numbers into decimal. The algorithm operates as follows:

Suppose the original number to be converted is stored in a register that is n bits wide. It takes a maximum of 4 bits in binary to store each decimal digit.

1. The algorithm iterates n times.
2. On each iteration, the entire scratch space is left-shifted one bit.
3. Before the left-shift is done, any BCD digit which is greater than 4 is incremented by 3
4. The increment ensures that a value of 5, incremented and left-shifted, becomes 16, thus correctly "carrying" into the next BCD digit. **5. Synthesis and Implementation Results**

The circuits have been implemented on a Xilinx Virtex-6, device xc6vlx760 ff1760, with speed grade–2 using timing constraints. For synthesis and implementation XilinxISE 13.1 tool have been used respectively.[6]

Results in terms of area and delay for each N by M decimal multiplier with and without pipeline stage are given in table1; table2.Finally table3 provides comparisons between multiplications on

FPGA virtex-6.simulation results of 8x8 BCD karatsuba –Ofman's multiplier with and without pipelining are shown in figure1, figure2.comparision of different devices in terms of area delay

Of 8x8 karatsuba multiplier and normal multiplier is shown in figure3.[7]

N	M	#Pipe	#IOBs	#Slices	Delay(ns)
8	8	7	32	197	25.58

Table 1: Delay and area of 8x8 BCD karatsuba –Ofman's multiplier without pipelining.

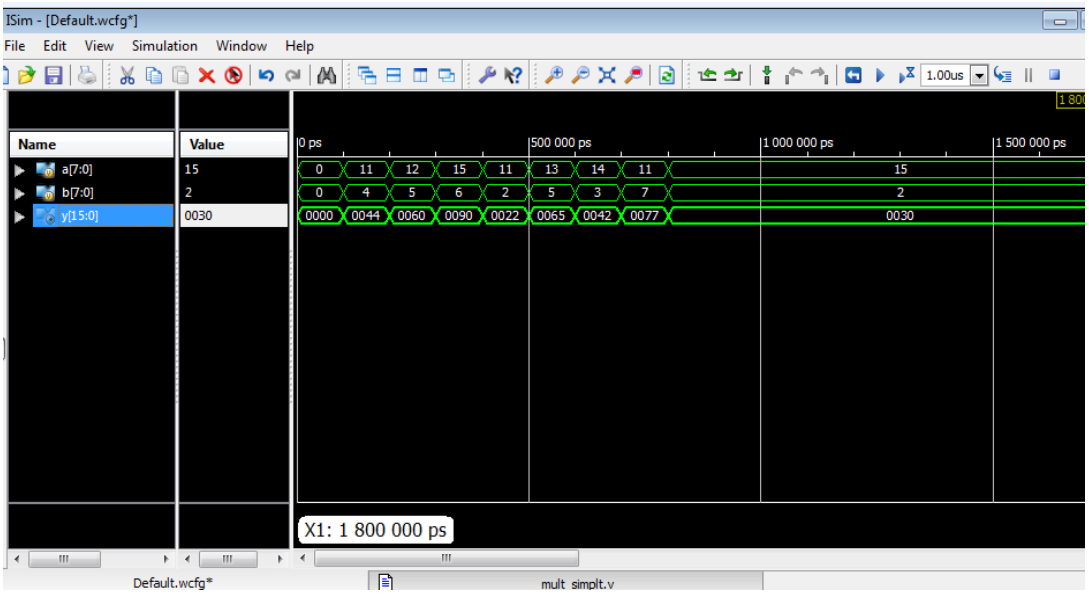


Figure 1: Simulation results of 8x8 BCD karatsuba –Ofman's multiplier without Pipelining.

N	M	#Pipe	#LUTs	#Slices	#FFs	Delay(ns)	Minimum Input arrival Time(ns)	Maximum Output req time(ns)
8	8	7	116	16	16	25.585	15.748	11.432

Table 2: Delay and area of 8x8 BCD karatsuba –Ofman's multiplier with pipelining.

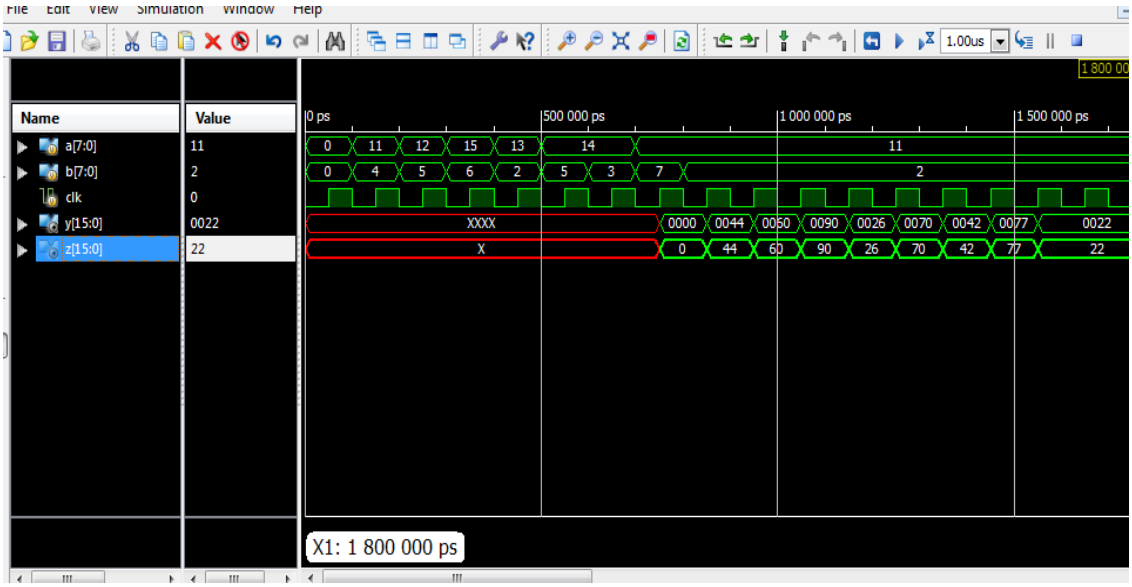
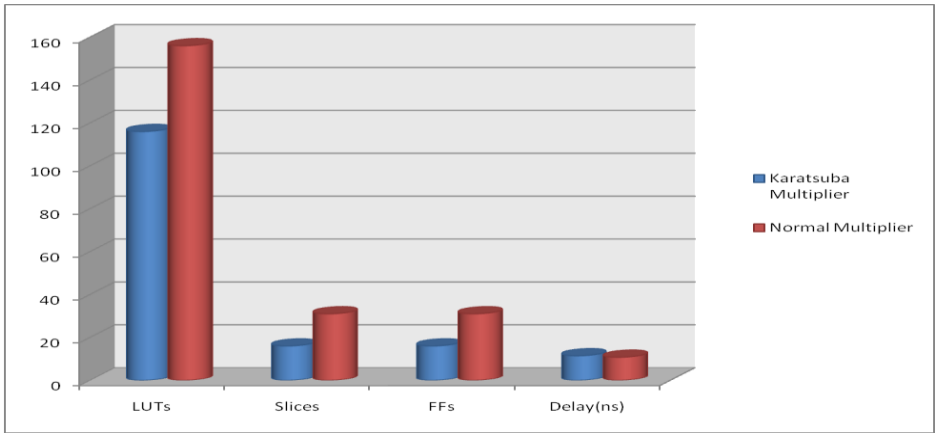


Figure 2: Simulation results of 8x8 BCD karatsuba –Ofman's multiplier with Pipelining.

Device Virtex-6 (XC6VLX760Ffl760-2)	N	M	#Pipe	#LUTs	#Slices	#FFs	Delay(ns)
Karatsuba Multiplier	8	8	7	116	16	16	11.432
Normal Multiplier	8	8	7	156	31	31	10.755

Table 3: Final Comparison



Graph: Comparison of device utilization and delay of 8x8 karatsuba multiplier and normal multiplier.

6. Conclusion and proposes Future work.

This paper presents the design of BCD pipelined karatsuba-Ofman's multiplier and implementation on VIRTEX-6 FPGA.The design proposed are analyzed to carry out the performance in terms of area and delay.

Karatsuba–Ofman's Algorithm is a fast multiplication algorithm. It reduces the multiplication of to at most$k^{\log_2 3} = k^{1.58}$. This algorithm is based on divide and conquers approach. It improves the efficiency of state of the art of decimal multipliers.

Future work plans to include the proposed multiplier in operations based on logarithmic function.

References:

1. IEEE Standard for Floating-Point Arithmetic, 2008. IEEE Std 754-2008.
2. M. A. Erle and M. J. Schulte. "Decimal multiplication via carry-save addition". In Proc. IEEE Int Application-Specific Systems, Architectures, and Processors Conf, pages 348–358, 2003.
3. Kenney, M.J.Schutle, M.A. Erle, "A High frequency Decimal Multiplier", 2009.
4. Carlos Eduardo Minchola Guardia "Implementation of a fully pipelined BCD Multiplier"2012.
5. H. C. Neto and M. P. Vestias, "Parallel Decimal Multipliers and Squarers using Karatsuba–Ofman's algorithm", 15th Euromicro Conference on Digital system Design, 2012.
6. Xilinx Inc. Xilinx Inc. Virtex-6 Libraries Guide for VHDL design, v11.1 edition, June 2009.
7. Xilinx Inc. Xilinx Inc. Xilinx ISE Design Suite 11.1 Software Manuals, v11.1 edition, June 2009.