GLOBAL JOURNAL OF ENGINEERING SCIENCE AND RESEARCHES

DESIGN OF KARATSUBA MULTIPLIER FOR LOW POWER FIR FILTER

A.Selvapandian[*1] and R.Balakumaresan[2]
Assistant professor - PSNA CET, Dindigul, Tamilnadu, India[*1,2]

ABSTRACT

In this paper the problem of optimizing the gate-level area in digit-serial MCM designs has been addressed and introduces high level synthesis algorithms and design architectures. The optimization algorithms and the digit-serial MCM architectures have efficiency in the design of digit-serial MCM operations and Finite Impulse Response filters. The conventional digit serial method increases power of the FIR architecture it has been optimized through the proposed Karatsuba's method of multiplication. Experimental results show the efficiency of the proposed optimization algorithm shows much reduction in power of finite impulse response filters.

Keywords:- Karatsuba, FIR, Filter etc.

I. INTRODUCTION

Multiple Constant Multiplications (MCM) is an efficient way of implementing several constant multiplications with the same input data. The coefficients are expressed using shifts, adders, and subtractor. By utilizing redundancy between the coefficients the number of adders and subtractors is reduced resulting in a low complexity implementation. However, for digit-serial arithmetic a shift requires a flip-flop, and, hence, the number of shifts should be taken into consideration as well. In this work the area, speed, power trade-offs for implementation of FIR filters using MCM and digit-serial arithmetic has been investigated. Also an algorithm has been introduced for reducing both the number of adders and subtractors as well as the number of shifts.

In Digital Signal Processing Finite Impulse Response (FIR) filters are of great importance since their characteristics in linear-phase and feed-forward implementations make them very useful for building stable high-performance filters. The multiplier block of the digital FIR filter where the multiplication of filter coefficients with the filter input is realized, has significant impact on the complexity and performance of the design because a large number of constant multiplications are required. This is generally known as the Multiple Constant Multiplications (MCM) operation and is also a central operation and performance bottleneck in many other DSP systems such as Fast Fourier Transforms (FFT), Discrete Cosine Transforms (DCTs), and error-correcting codes.

Although area, delay, and power-efficient multiplier architectures, such as Wallace and modified Booth multipliers, have been proposed, the full flexibility of a multiplier is not necessary for the constant multiplications, since filter coefficients are fixed and determined beforehand by the DSP algorithms. Hence, the multiplication of filter coefficients with the input data is generally implemented under a shift adds architecture, where each constant multiplication is realized using addition/subtraction and shift operations in an MCM operation. For the shift-adds implementation of constant multiplications, a straightforward method, generally known as digit based recoding, initially defines the constants in binary.

The algorithms designed for the MCM problem can be categorized in two classes: Common Sub expression Elimination (CSE) algorithms and Graph Based (GB) techniques.

The CSE algorithms initially extract all possible Subexpressions from the representations of the constants when they are defined under binary, Canonical Signed Digit (CSD), or Minimal Signed Digit (MSD). Then, they find the “best” subexpression, generally the most common, to be shared among the constant multiplications.

The GB methods are not limited to any particular number representation and consider a larger number of alternative implementations of a constant, yielding better solutions than the CSE algorithms.

II. EXISTING ALGORITHMIC METHOD FOR MCM

A) EXISTING CSE Algorithm

The CSE algorithms initially extract all possible Subexpressions from the representations of the constants when they are defined under binary, canonical signed digit (CSD), or minimal signed digit (MSD). Then, they find the “best” subexpression, generally the most common, to be shared among the constant multiplications.

As a simple example, consider the constant multiplications 29x and 43x.

Their decompositions in binary are listed as follows:

$29x = (11101)_{bin}x = x\_4 + x\_3 + x\_2 + x$

$43x = (101011)_{bin}x = x\_5 + x\_3 + x\_1 + x$

which requires six addition operations as illustrated

The exact CSE algorithm gives a solution with four operations by finding the most common partial products 3x = (11)binx and 5x = (101)binx when constants are defined under binary, as illustrated in Figure. 3.2(b). Whereas the exact

GB algorithm finds a solution with the minimum number of operations by sharing the common partial product 7x in both multiplications as shown in Figure. 3.2(c). Note that the partial product 7x = (111)binx cannot be extracted from the binary representation of 43x in the exact CSE algorithm .

However, all these algorithms assume that the input data X is processed in parallel. On the other hand, in digit-serial Arithmetic, the data words are divided into digit sets, consisting Of d bits, that are processed one at a time. Since digit serial Operators occupy less area and are independent of the Data word-length, digit-serial architectures offer alternative low complexity. Designs when compared to bit-parallel architectures.

However, the shifts require the use of D flip-flops, as Opposed to the bit-parallel MCM design where they are free in terms of hardware. Hence, the high-level algorithms should take into account the sharing of shift operations as well as the sharing of addition/subtraction operations in digit-serial MCM design. Furthermore, finding the minimum number of operations realizing an MCM operation does not always yield an MCM design with optimal area at the gate level. Hence the high-level algorithms should consider the implementation cost of each digit-serial operation at the gate level.

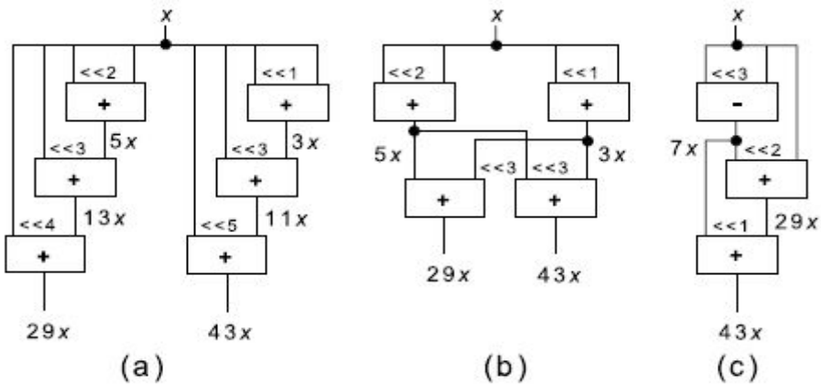


Figure.3.2. Shift-adds implementations of 29*x* and 43*x*. (a) Without partial product sharing and with partial product sharing. (b) Exact CSE algorithm. (c) Exact GB algorithm.

B) EXISTING GRAPH BASED METHOD

The GB methods are not limited to any particular number representation and consider a larger number of alternative implementations of a constant, yielding better solutions than the CSE algorithms.

Initially the gate-level implementation costs of digit-serial addition, subtraction, and left shift operations used in the shift-adds design of digit-serial MCM operations is determined. Then the exact CSE algorithm that formalizes the gate-level area optimization problem as a 0–1 integer linear programming (ILP) problem when constants are defined under a particular number representation. A new optimization model that reduces the 0–1 ILP problem size significantly and consequently, the runtime of a generic 0–1 ILP solver.

Since there are still instances which the exact CSE algorithm cannot handle, the approximate GB algorithm that iteratively finds the “best” partial product which leads to the optimal area in digit-serial MCM design at the gate level. In this project introduces a computer-aided design (CAD) tool called SAFIR which generates the hardware descriptions of digit-serial MCM operations and FIR filters based on design architecture and implements these circuits using a commercial logic synthesis tool. In SAFIR, the digit-serial constant multiplications can be implemented under the shift adds architecture, and also can be designed using generic digit serial constant multipliers.

Experimental results on a comprehensive set of instances show that the solutions of algorithms introduced lead to significant improvements in area of digit-serial MCM designs compared to those obtained using the algorithms designed for the MCM problem.

The digit-serial FIR filter designs obtained by SAFIR indicate that the realization of the multiplier block of a digit-serial FIR filter under the shiftadds architecture significantly reduces the area of digit-serial FIR filters with respect to those designed using digit-serial constant multipliers. Additionally, it is observed that the optimal tradeoff between area and delay in digit-serial FIR filter designs can be explored by changing the digit size d.

III. PROPOSED KARATSUBA MULTIPLIER

If we want to multiply large numbers, it’s possible to use other techniques than the traditional multiplication, which is time expensive. In this case the Karatsuba Multiplication Algorithm is one of the choices. But this method is one of the fastest as well as mostly uses method in processors, because this method required only three multiplications and most of time this method is recursive in nature. For convenience, we will focus on long integers in binary representation.
Divide each number into two halves.

- $x=x_H r^{n/2}+x_L$
- $y=y_H r^{n/2}+y_L$

Then:

$xy=(x_H r^{n/2}+x_L)y_H r^{n/2}+y_L$

$=x_H y_H r^n+(x_H y_L+x_L y_H)r^{n/2}+x_L y_L$

Worked Example:

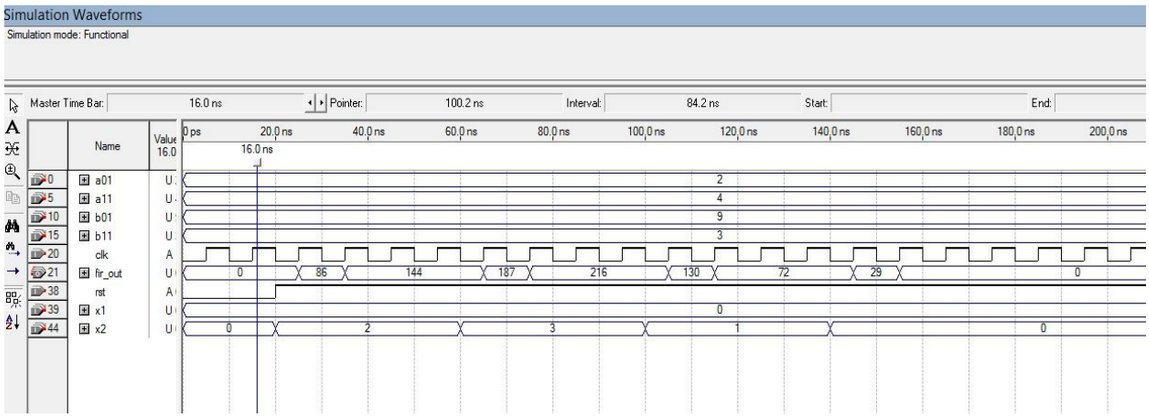
a1=12*43
- Subproblems:

a2=1*4=4
d2=2*3=6
e2=(1+2)(4+3)–a2–d2=11
- Answer:$4*10^2+11*10+6=516$

From above it is clear that we required three multiplication and thus increase the speed of multiplication.It is similar to Urdhva Tiryakbhyam, but it reduce multiplication steps because A and B multiplication is already performed and we have to take value in to C as an input. When this technique is recursively applied to ultidigit numbers, a point is reached in the recursion when the overhead of addition and subtraction makes it more efficient to use the usual O (n 2) multiplication algorithm to evaluate the partial products. We call this point "Karatsuba threshold". The most efficient overall method therefore relies on a combination of Karatsuba and conventional multiplication. Namely, we'll have n log3/log2 digit products for operands of length n, not n
2 like in the traditional multiplication. It's convenient to see this algorithm in terms of a ternary tree. Each node has three children that compute the partial products and at each level the input length is divided with two. The leaves perform the classical multiplication.

IV. RESULTS AND SIMULATION

A) SIMULATION OF KARATSUBA ALGORITHM



B) POWER REPORT GENERAL, CSE, GB & KARATSUBA METHOD

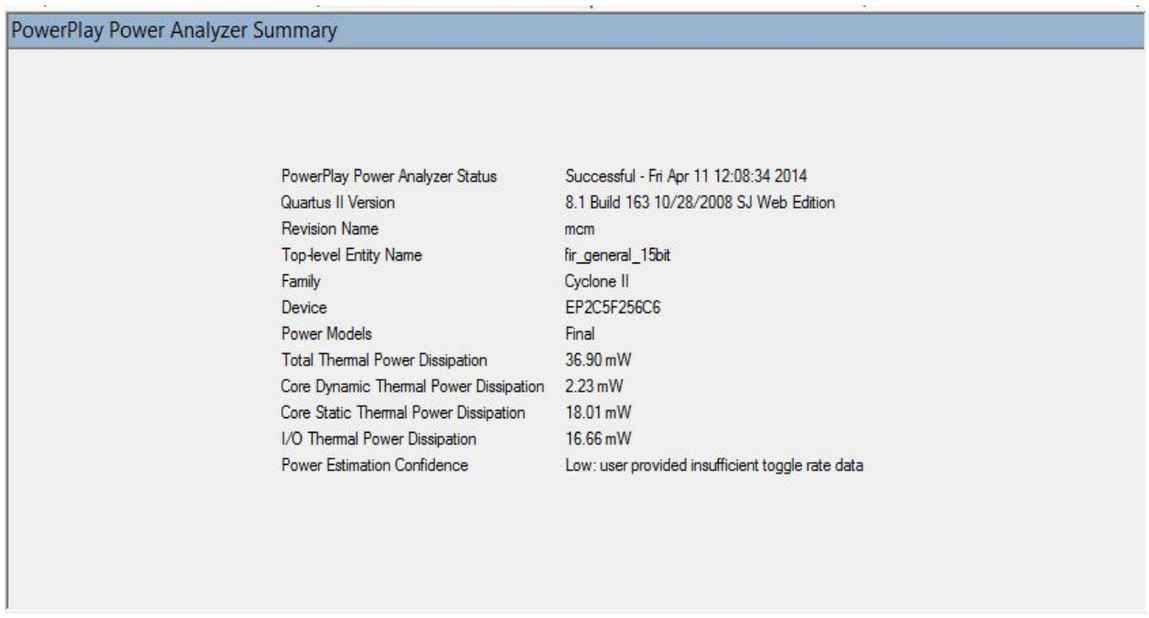
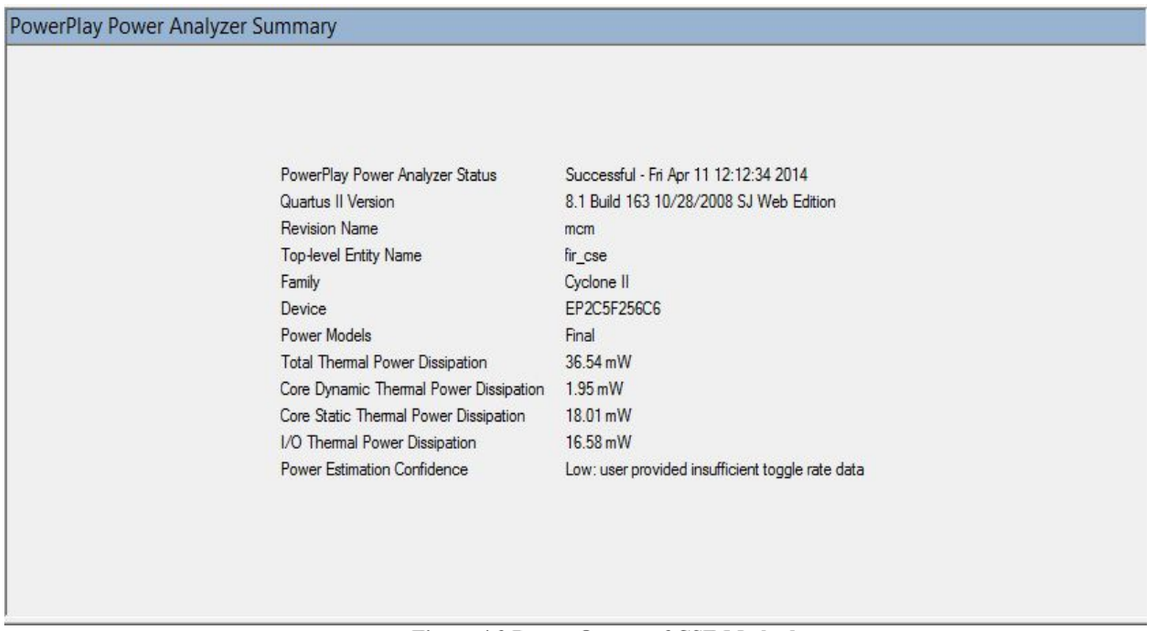
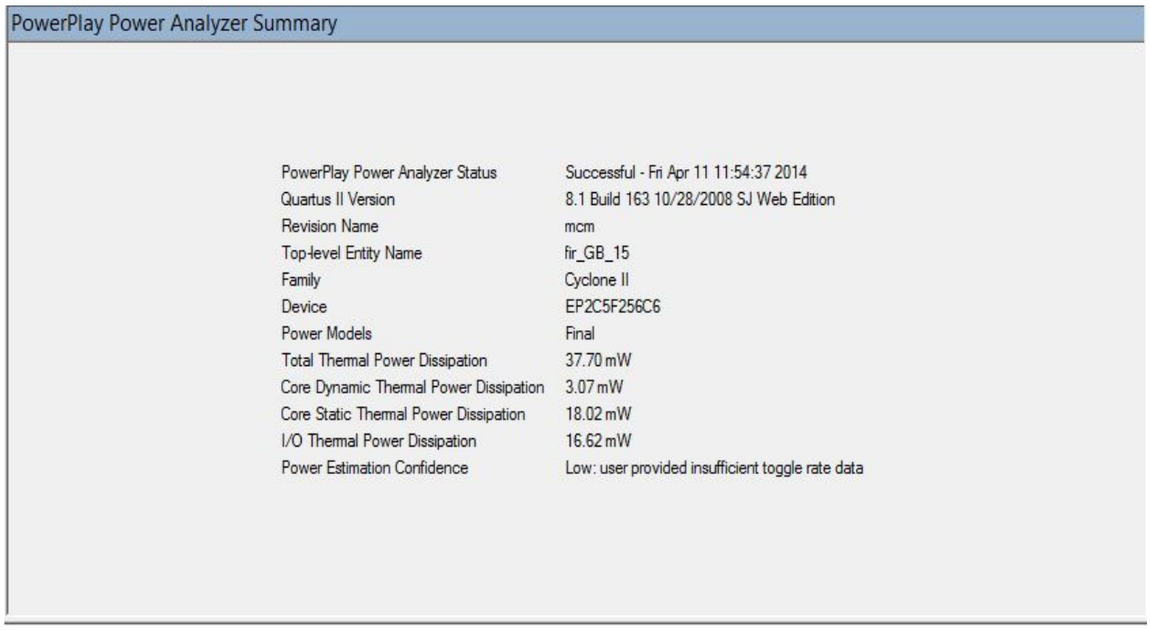


Figure 4.2 Power Output of GENERAL Method


PowerPlay Power Analyzer Summary

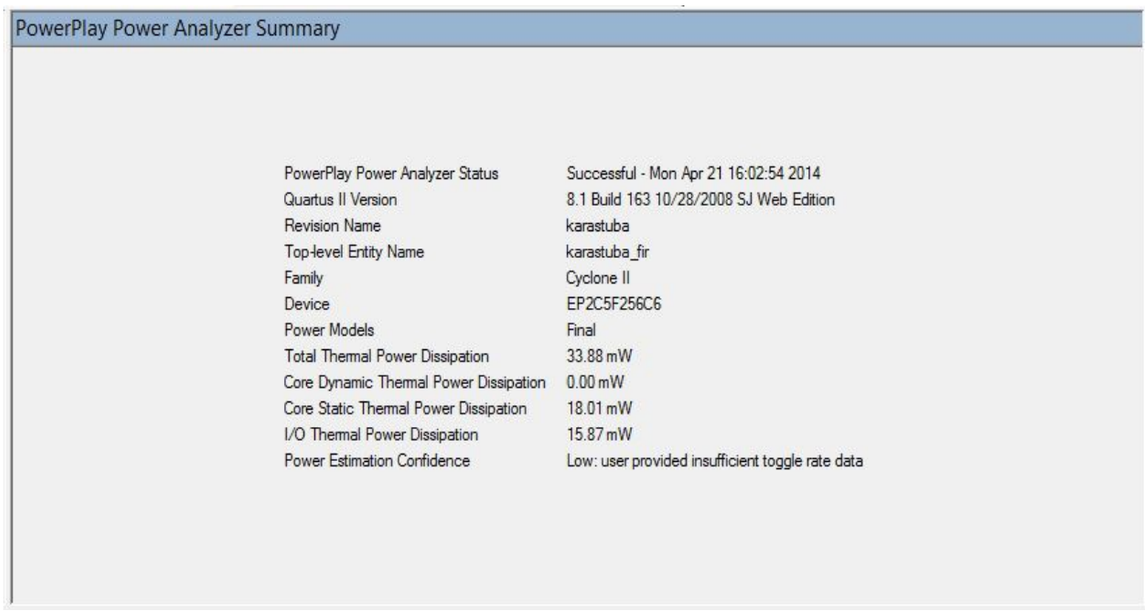
PowerPlay Power Analyzer Status	Successful - Fri Apr 11 12:12:34 2014
Quartus II Version	8.1 Build 163 10/28/2008 SJ Web Edition
Revision Name	mcm
Top-level Entity Name	fir_cse
Family	Cyclone II
Device	EP2C5F256C6
Power Models	Final
Total Thermal Power Dissipation	36.54 mW
Core Dynamic Thermal Power Dissipation	1.95 mW
Core Static Thermal Power Dissipation	18.01 mW
I/O Thermal Power Dissipation	16.58 mW
Power Estimation Confidence	Low: user provided insufficient toggle rate data

Figure 4.3 Power Output of CSE Method


PowerPlay Power Analyzer Summary

PowerPlay Power Analyzer Status	Successful - Fri Apr 11 11:54:37 2014
Quartus II Version	8.1 Build 163 10/28/2008 SJ Web Edition
Revision Name	mcm
Top-level Entity Name	fir_GB_15
Family	Cyclone II
Device	EP2C5F256C6
Power Models	Final
Total Thermal Power Dissipation	37.70 mW
Core Dynamic Thermal Power Dissipation	3.07 mW
Core Static Thermal Power Dissipation	18.02 mW
I/O Thermal Power Dissipation	16.62 mW
Power Estimation Confidence	Low: user provided insufficient toggle rate data

Figure 4.4 Power Output of GB Method


PowerPlay Power Analyzer Summary

PowerPlay Power Analyzer Status	Successful - Mon Apr 21 16:02:54 2014
Quartus II Version	8.1 Build 163 10/28/2008 SJ Web Edition
Revision Name	karastuba
Top-level Entity Name	karastuba_fir
Family	Cyclone II
Device	EP2C5F256C6
Power Models	Final
Total Thermal Power Dissipation	33.88 mW
Core Dynamic Thermal Power Dissipation	0.00 mW
Core Static Thermal Power Dissipation	18.01 mW
I/O Thermal Power Dissipation	15.87 mW
Power Estimation Confidence	Low: user provided insufficient toggle rate data

Figure 4.5 Power Output of KARATSUBA Method

C) PARAMETER ANALYSIS FOR GENERAL, CSE, GB & KARATSUBA METHOD

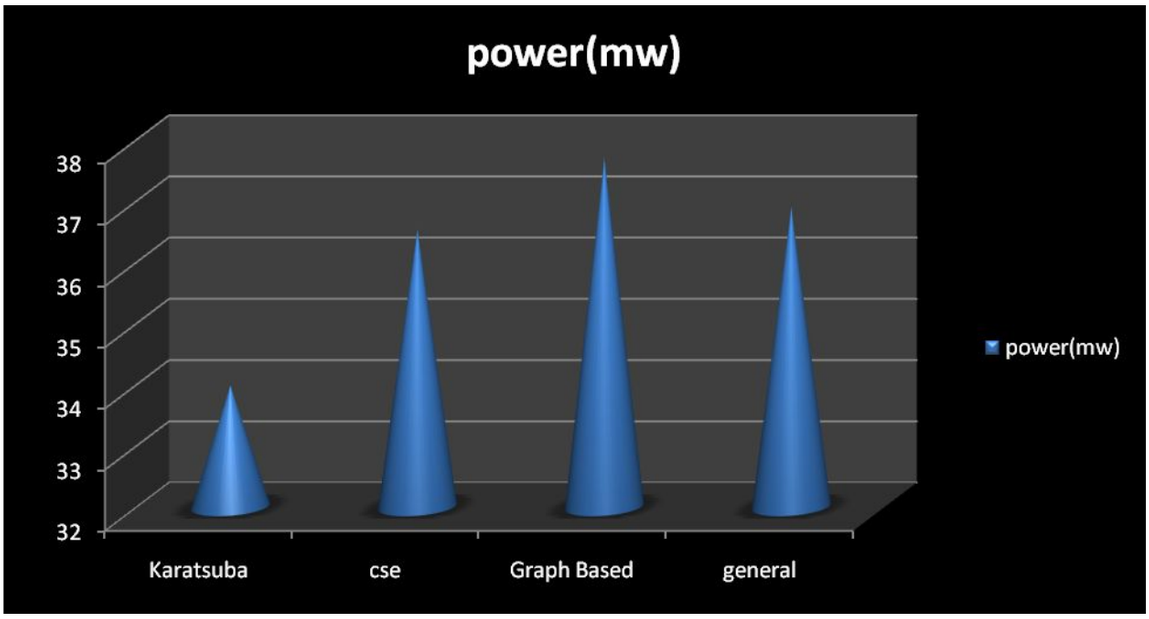


Figure 4.12 Parameter Analysis

COMPARISON OF POWER WITH DIFFERENT MULTIPLIERS

Multiplier	Power
Karatsuba	33.99 mW
Common Sub-Expression Elimination	36.54 mW
Graph Based	37.7 mW
General	36.9 mW

V. CONCLUSION

In this paper, we presented a Karatsuba-based Multiplier for FIR Filter applications.The multipliers have significantly lower power compared with previous designs. They also provide excellent performance and energy efficiency compared with software implementations. Thus the results show that the various methods have analyzed and the computation of all results is analyzed in terms of power.

REFERENCES

1. *Abdelgawad.A and M.Bayoumi (2007) "High speed and area-efficient multiply accumulate (MAC) unit for digital signal processing applications," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), Vol. 43, No.3, pp. 3199–3202*

2. *Hatamian.M., and G. L. Cash,(1986) "A 70-MHz 8-bit x 8-bit parallel pipelined multiplier in 2.5-μm CMOS," IEEE J. Solid-State Circuits, Vol. JSSC-21, No. 4, pp. 505–513.*

3. *Hoang. T. T., M. Själander, and P. Larsson-Edefors, (2009) "Double throughput multiply-accumulate unit for FlexCore processor enhancements," presented at the IEEE Int. Symp. Parallel Distrib. Process. (IPDPS), Reconfigurable Archit. Workshop (RAW), Rome, Italy, Vol. 27, No.9,*

4. *Hoang. T. T., M. Själander, and P. Larsson-Edefors, (2009) "High-speed, energy-efficient 2-cycle multiply-accumulate architecture," in Proc. IEEE Int. SOC Conf. (SOC), Vol. 38, No.14, pp.119–122*

5. *Kuang.S.-R., and J.-P. Wang, (2010) "Design of power-efficient configurable booth multiplier," IEEE Trans. Circuits Syst. I, Reg. Papers, Vol. 57, No. 3, pp. 568–580.*

6. *Shiann-Rong Kuang, Member, IEEE, Jiun-Ping Wang, and Cang-Yuan Guo, (2009) "Modified Booth Multipliers With a Regular Partial Product Array," IEEE transactions on circuits and systems—ii: express briefs, Vol. 56, No. 5.*

7. *Själander. M., and P. Larsson-Edefors, (2009) "Multiplication acceleration through twin precision," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., Vol. 17, No.12, pp. 1233–1246.*