Design and Implementation of Karatsuba Based FIR Filters for Signal Processing

N.Arunkumar[1], Prof. G.Kanagaraj,M.E., (Ph.D)[2]

[1]PG scholar, [2]Assistant Professor,
Department of ECE,
AVS Engineering College,
Salem-636003.

Abstract: Recursive combination of an algorithm based on Karatsuba multiplication is exploited to design a generalized transpose and Finite Impulse Response (FIR) Filter. Mid-range Karatsuba multiplication and Carry Save adder based on Karatsuba multiplication reduce time complexity for higher order multiplication implemented up to n-bit. The Karatsuba formula is used to speed-up the multiplication of large numbers by splitting the operands in two parts of equal length.As a result, we design modified N-tap Transpose and Parallel Symmetric FIR Filter Structure using Karatsuba algorithm for the automated processing of signal data. The mathematical formulation of the FFA Filter is derived. The proposed architecture involves significantly less area delay product (APD) then the existing block implementation. The proposed structure achieves more than a half of the power reduction by adopting with and without proposed techniques compared to the earlier design structure. our proposed architecture has been coded HDL and simulated using Xilinx 12.1.

Keywords: FIR, Xilinx, Karatsuba, Multiplication, Carry save adder

I. INTRODUCTION

FIR filter has wide application as the key component in any digital signal processing, image and video processing, wireless communication, and biomedical signal processing systems. Moreover, systems like Software Defined Radio (SDR) and multi-standard video codec need a reconfigurable FIR filter with dynamically programmable filter coefficients, interpolation factors and lengths which may vary according to the specification of different standards in a portable computing platform. Significant applicability of an efficient reconfigurable FIR filter motivates the system designer to develop the chip with low cost, power, and area along with the capability to operate at very high speed.

In any FIR filter, the multiplier is the major constraint which defines the performance of the desired filter. Therefore, over the past three decades, design of an efficient hardware architecture for fixed point FIR filter has been considered as the major research focus as reported in published literature. In this paper, we proposes the implementation of a programmable Finite Impulse Response (FIR) filter based on the use of the Modified Karatsuba formula. The Karatsuba formula is used to speed-up the multiplication of large numbers by splitting the operands in two parts of equal length.

The organization of the paper is as follows. In Section II, basic concepts along with booth recoded FIR algorithms proposed in the earlier literatures have been discussed. The proposed karatsuba multiplier is presented in Section III. An experimental results has been presented in Section IV. The implementation results along with discussions on the comparison of our results with other reported implementation have been provided in Section V. Finally, the paper is concluded in Section VI.

II. RELATED WORKS AND DESIGN OF FIR FILTER USING BOOTH RECODED MULTIPLIER

The radix-2 Baugh-Wooley multipliers are rather slow but fast multiplier that uses Booth recoding. A BR4 multiplier, has been considered, where the PPR is fed with less than half of the partial products of those in the BW2 multiplier, thereby providing a much shorter critical path. As opposed to the symmetric BW2 multiplier, in the BR4 topology, the two input operands are processed differently, since x is passed to the recoding logic that decides which multiples of y should be fed to the PPR. For the considered implementation, a carry-save adder with (m,2) compressors is used for the PPR and a carry-propagate adder is used for the VMA, as in the BW2 multiplier.

Table 1. Radix 4 booth recoding

Select Line	Operation
000	add 0
001	add multiplicand x
010	add multiplicand x
011	add 2*multiplicand x
100	subtract 2*multiplicand x
101	subtract multiplicand x
110	subtract multiplicand x
111	subtract 0

The non-zero partial product per input operand analysis on the PPG, previously shown for the BW2 multiplier, was also applied to the 8 × 8-bit BR4 multiplier. Booth multiplication that is based on string recoding can multiply a pair of two's complement numbers without concern about the signs of the numbers. In the radix-4 Booth multiplier, only (n +1)/(2) multiplicand multiples are required to be summed together due to the string recoding (Table 1) . In this approach, bxc stands for the largest integer less than or equal to the real number x.

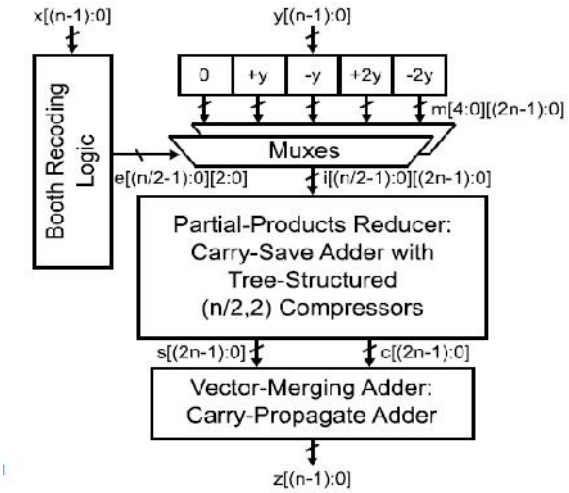

Fig.1 Structure of a signed bit radix-4 Booth-recoded multiplier.

The optimization technique does not require any hardware overhead and it enables the possibility of scaling the power consumption of the filter at runtime, while ensuring the full baseline performance of any programmed filter whenever it is required. The analyzed FIR filters were fabricated in a 28nm FD-SOI test chip and measured at a near threshold, 600mV supply voltage.To derive a set of coefficients that are optimized for power consumption, we use the algorithm proposed in the flowchart of Fig: all the design requirements, such as passband ripple (Ap), stopband attenuation (As) and passband frequency edge (Fp) initially define the coefficient values of the reference filter. These initial coefficients satisfy the baseline specifications and provide an estimation of the power consumed by the multipliers using the results provided.

With the coefficients of the reference filter as a starting point, the coefficients are perturbed to drive the multipliers towards lower power consumption. In particular, at each optimization step, the algorithm considers a range of values around each of the coefficients and the value that results in the lowest power consumed by the multiplier is preferred. For the examples, shown in this paper, the considered range of values (±_) is defined manually through experiments and it is kept constant for all optimization steps in the algorithm for the derivation of the coefficients.

While the algorithm is rather insensitive to the choice of _, a sufficiently large _ value is only required to avoid local minima. However, as this minimum value highly depends on the multiplier structure, their bitwidth, the number of taps in the filters as well as on the baseline coefficients, it cannot be easily defined analytically, but it is simply derived through manual experiments to show a lower bound on the achievable gains.

When the coefficients are perturbed for lower power consumption, the frequency response of the filter changes, and therefore, relaxed specifications should be defined to accept only the coefficients that still provide sufficient filter quality. Hence, the most power-efficient coefficients that meet the relaxed specifications are chosen among the considered range of values at each optimization step and this optimization is repeated until a local minimum for the power consumption is found.4 The obtained coefficients are then programmed instead of the original reference FIR coefficients.

The optimization technique yields a well-controlled quality degradation, since the relaxed specifications are a-priori well defined and are always met. In addition, both the frequency responses of the filter, as well as the dynamic power consumptions can easily be estimated during the filter optimization process without the need for time consuming power simulations on the whole FIR filter, since the implemented multiplier topology (used as basic building block) has been previously characterized.

The representations of the multiplicand and product are not specified; typically, these are both also in two's complement representation, like the multiplier, but any number system that supports addition and subtraction will work as well. As stated here, the order of the steps is not determined. Typically, it proceeds from LSB to MSB, starting at i = 0; the multiplication by 2i is then typically replaced by incremental shifting of the P accumulator to the right between steps; low bits can be shifted out, and subsequent additions and subtractions can then be done just on the highest N bits of P.[1] There are many variations and optimizations on these details.

The algorithm is often described as converting strings of 1s in the multiplier to a high-order +1 and a low-order −1 at the ends of the string. When a string runs through the MSB, there is no high-order +1, and the net effect is interpretation as a negative of the appropriate value.

III. PROPOSED KARATSUBA ALGORITHM

The Karatsuba algorithm [9], [10] is a fast multiplication algorithm which was designed by Anatoly Karatsuba in 1960 and published in 1962. It reduces the multiplication of two n-digit numbers to at most $N\log_2 3$ ($N1.59$ single-digit multiplications in general). It is therefore faster than the classical algorithm, which requires $N2$ single-digit products. For example, the Karatsuba algorithm requires 310 = 59,049 single-digit multiplications to multiply two 1024-digit numbers (n = 1024 = 210), whereas the classical algorithm requires (210)2 = 1,048,576. Considering two 16-bits wide binary variables, X and Y for the example. The numbers are broken down as follows into the two significant halves:

$$X = X_H \times 2^8 + X_L \quad (1)$$
$$Y = Y_H \times 2^8 + Y_L \quad (2)$$

where, X_H and Y_H represent the two most significant halves of X and Y respectively while X_L and Y_L represent their two least
significant halves. Hence,

$$XY = (X_H \times 2^8 + X_L) \times (Y_H \times 2^8 + Y_L) \quad (3)$$

The above expression shows the multiplication of the two binary numerals and when expanded they yield four multiplication terms, each N-bits wide as shown below:

$$A = X_H Y_H \quad (4)$$
$$B = X_H Y_L + X_L Y_H \quad (5)$$
$$C = X_L Y_L \quad (6)$$
$$XY = A \times 2^{16} + B \times 2^8 + C \quad (7)$$

where, A, B and C represent the three combinatorial expressions. A and C each require one multiplier while B requires two and C each require one multiplier while B requires two.

So, the long multiplication technique here requires a total of four 8-bit multipliers for proper implementation. However, going by the Karatsuba approach, the multiplication terms shall be reduced by substituting one stage of multiplier with adders and subtractors as follows:

$$B' = (X_H + X_L) \times (Y_H + Y_L) - A - C = B \quad (8)$$

Here, B' represents an alternative manner of representing the same term B with the calculation of just one multiplication term as opposed to the two terms in case of B.

$$XY = A \times 2^{16} + B' \times 2^8 + C \quad (9)$$

The above expression depicts the numerical representation of the Karatsuba algorithm involving the calculation of three multiplicands as opposed to the four required in case of any classical approach.

Mathematically, the Karatsuba algorithm thus effectively reduces the operational overheads of the multiplier in terms of area and power. However, the algorithm alone doesn't really simplify the design of the multiplier core in the design.

IV. EXPERIMENTAL RESULTS

The proposed circuit are simulated and synthesized by using modelsim and xilinx12.1 which occurs low area than the existing. The experimental results are given in Table 1 and the simulation results of layout and the waveforms are shown in the fig.2a and 2b. Then the synthesis result of the proposed are shown in fig.3.

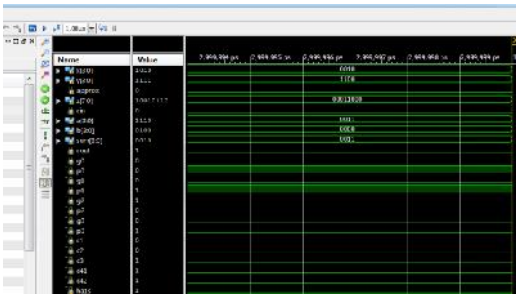

Fig.2a simulation results

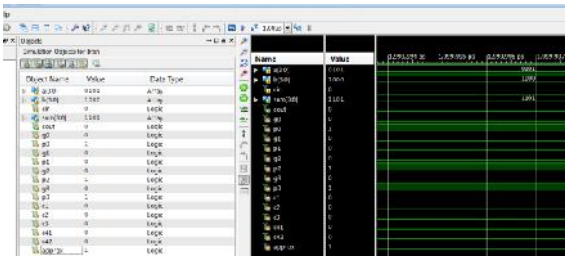

Fig. 2b output waveform of proposed

Device Utilization Summary (estimated values)			[-]
Logic Utilization	Used	Available	Utilization
Number of Slices	17	960	1%
Number of 4 input LUTs	31	1920	1%
Number of bonded IOBs	17	66	25%

Fig.3 synthesis report of proposed architecture

S.No	Parameter	Existing	Proposed
1	Slice	19	17
2	LUT	33	31

Table 1: comparison of existing and proposed results

V. PERFORMANCE ANALYSIS

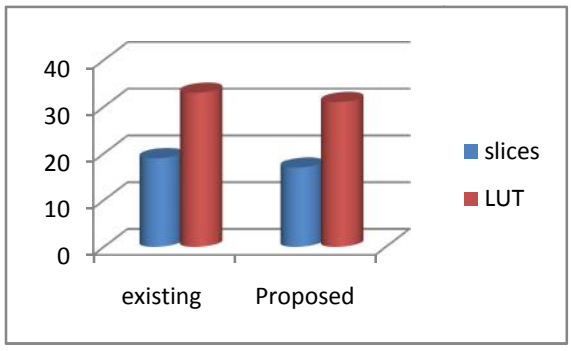

Fig.4 performance analysis

The Figure given below is shown that there is a considerable reduction based on no of transistors and the performance chart has been shown below in Fig.4

VI. CONCLUSION

In this work, the authors analyzed a new technique for FIR filter implementation based on the use of the Karatsuba formula. The analysis shows that the Karatsuba filter architecture is composed by three sub-filters of reduced dynamic range, three adders and two hardwired shifters for the implementation of the input and output conversions. Evaluations of area, power and speed have been obtained by implementing a set of experiments showing the advantages of the Karatsuba implementation over the traditional transposed implementation. Thus the experimental results showed the proposed Karatsuba algorithm to fault-tolerant applications that require FIR filtering. Which gives a low area overhead and an efficient system. In particular, the earnings are relevant both in power and delay especially for high dynamic ranges.

REFERENCES

[1] B. K. Mohanty and S. K. Patel, "Area–Delay–Power Efficient Carry- Select Adder," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 61, no. 6, pp. 418-422, June 2014.

[2] B. Shao and P. Li, "Array-Based Approximate Arithmetic Computing: A General Model and Applications to Multiplier and Squarer Design," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 62, no. 4, pp. 1081-1090, April 2015.

[3] A. Raha, H. Jayakumar, and V. Raghunathan, "Input-Based Dynamic Reconfiguration of Approximate Arithmetic Units for Video Encoding," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 24, no. 99, pp. 1-1, May 2015.

[4] M. S. Khairy, A. Khajeh, A. M. Eltawil and F. J. Kurdahi, "Equi-Noise: A Statistical Model That Combines Embedded Memory Failures and Channel Noise," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 61, no. 2, pp. 407-419, Feb. 2014.

[5] R. Ye, T. Wang, F. Yuan, R. Kumar and Q. Xu, "On reconfiguration oriented approximate adder design and

its application," Proceedings of IEEE/ACM International Conference on Computer Aided Design (ICCAD), 2013, pp. 48-54.

[6] A. K. Verma, P. Brisk and P. Ienne, "Variable Latency Speculative Addition: A New Paradigm for Arithmetic Circuit Design," Proceedings of Design, Automation and Test in Europe (DATE), 2008, pp. 1250-1255.

[7] H.R. Mahdiani, A. Ahmadi, S.M. Fakhraie and C. Lucas, "Bio-Inspired Imprecise Computational Blocks for Efficient VLSI Implementation of Soft-Computing Applications," IEEE Transactions on Circuits and Systems-I: Regular Papers, vol. 57, no. 4, pp. 850-862, April 2010.

[8] D. Shin and S.K. Gupta, "A Re-design Technique for Datapath Modules in Error Tolerant Applications," Proceedings of 17th Asian Test Symposium(ATS), 2008, pp. 431-437.

[9] N. Zhu et al. "Design of Low-Power High-Speed Truncation-Error- Tolerant Adder and Its Application in Digital Signal Processing," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 18, no. 8, pp. 1225-1229, Aug. 2010.

