Journal of Engineering and Applied Sciences 13 (Special Issue 5): 4733-4738, 2018
ISSN: 1816-949X

Implementation of Low Area FIR Filters Using Vedic Multiplication Algorithms

[1]R. Deepa and [2]S. Elango
[1]Department of EIE, [2]Department of ECE,
Bannari Amman Institute of Technology Sathyamangalam, Tamil Nadu, India

Abstract: Multiplication is the fundamental function in most of the lightweight systems. In most of the signal processing applications, the demand for low cost multiplier is the major concern. This research different multiplier algorithms like Urdhava Tiryagbhyam and Anurupye; it investigates the implementation of low pass Finite Impulse Response (FIR) filter. The functional unit of 8 bit multiplier and FIR filter is described using VHDL, simulated and synthesized for both FPGA and ASIC CMOS (180 and 90 nm) technology are reported in this research the implementation of Anurupye algorithm ensures the reduction of area and leakage power in comparison with Urdhava Tiryagbhyam architectures. The functionality of these circuits was checked in Xilinx Spartan6 XC6SLX45T-2CSG324 FPGA device and a number of lookup tables, delay was calculated.

Key words: Urdhava Tiryagbhyam, Anurupye, FIR filter, multiplier, FPGA, applications

INTRODUCTION

'Vedic' is derived from 'Veda'. It defines the storehouse of knowledge. Ancient system followed by indians is the vedic mathematics. It depends on 16 sutras and 16 Upa sutras. In this study two sutras were mainly used for multiplication; they are Urdhava Tiryagbhyam and Anurupye (Maharaja, 2009). Most commonly used Sutr a for the multiplication is the Urdhava Tiryagbhyam and mainly used in all kinds of multiplication. The operation of the multiplication of this sutra is like vertically and crosswise. The Anurupye sutra multiplies the numerical values that are adjacent to an index of 10 but not closure to the index of 10.

In digital signal processing, multipliers are one of the powerful tools used to perform operations like convolution, correlation, Discreet Fourier Transform (DFT), Fast Fourier Transform (FFT) and filter design. The FIR filters are digital filters, whose response is finite in duration. Since, they do not have feedback, this filter is called as non-recursive filter. For realizing the FIR filter, a recursive algorithm can also be used. It is simple to design and highly stable. FIR filter design is derived from the linear convolution operation (Chidgupkar and Karad, 2004).

The FIR filter impulse response consists of a finite number of zeroes for the duration 0 to N-1. Thus, for N-order impulse response of an FIR filter exists for N+1 samples and then decays to zero (Itawadiya *et al*., 2013). The output equation for the FIR filter is obtained from convolution of two sequences expressed as:

$$g(n) = i(n) \otimes h(n) g(n) = \sum_{N=0}^{N-1} i(n) * h(n-k)$$

Where:

h(n) = The input
g(n) = The output of the given signal to the FIR filter, respectively
I(n) = The impulse response, i.e., the filter coefficients

FIR filter architecture: In this, a non-recursive digital filter is used. No feedback is used and the response of the FIR filter is influenced by past and present input values. It shows the conventional 8-tap FIR filter architecture in Fig. 1. The output response of the filter is expressed as:

$$y(n) = \sum_{n=1}^{8} (x(n) \otimes h(n))$$

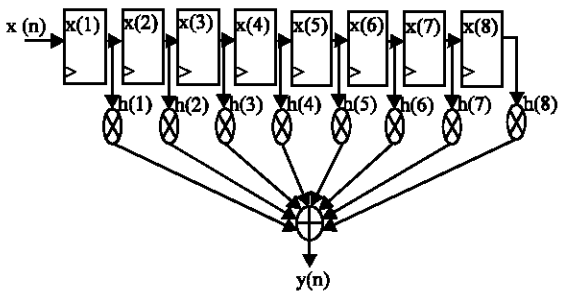


Fig. 1: Conventional FIR architecture

Coressponding Author: R. Deepa, Department of EIE, Bannari Amman Institute of Technology Sathyamangalam, Tamil Nadu, India

Eight tap filter architecture consists of a shift filter is register, multipliers and adders (Evans, 1994). The D flip flop is the basic component to develop the shift register. Architectures like vedic, baugh wooley, booth, wallace and array may be considered to perform the multiplication operation. Ripple carry, carry Save and Carry Select adder architectures are the various design options existing for FIR filter design.

Literature review: Sivasankar amd Nanjappan (2015) proposed the design of the high speed multiplier for digital FIR filter using ancient Urdhava Tiryagbhyam techniques. In this study Vedic multiplier uses the Urdhava Tiryagbhyam sutra. The concept of Urdhava multiplier is applicable for arithmetic circuits and multiplier architecture which results great improvement in DSP performance.

Kumar and Kumar (2016) proposed the transposed structure design of FIR filter using VHDL. The output of FIR filter is the convolution of two sequences with the tap length. The transposed structure is considered for the design of FIR filter to optimize the delay. The device is developed using modelsim and Xilinx Software and synthesized on FPGA.

Mehkarkar and Banarase (2014) proposed the high performance vedic multiplication based FIR filter. In this study vedic multiplication technique generates the intermediate products which eliminates the unwanted steps with zero. To design proposed high speed filter verilog hardware description language can be used.

Park *et al.* (2002) defines the sharing multiplication method for high performance filter design. For efficient filter design, the computation of vector scalar products are re-use which pay the way for improvement of speed and power in wallace tree based FIR filter.

MATERIALS AND METHODS

Vedic multiplication

Introduction: Vedic multiplier sutras are founded on ancient Indian vedic mathematics. Decimal multiplication of two numbers is performed by vedic sutras. The ancient Indian vedic mathematics uses the multiplication formula and this sutra is used in all cases of multiplication.

Urdhava Tiryagbhyam (UT) Sutra : Urdhava Tiryagbhyam algorithms, the calculations are processed vertically and crosswise (Dhillon and Mitra, 2008). The repeated addition is basically called as multiplication where the delay value is directly proportional to the number of bits (digits). For an n×n multiplier, partial products are generated in parallel and partial product summation is

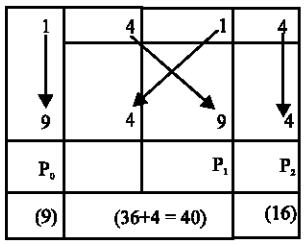


Fig. 2: Decimal multiplication using Urdhava: P_0, 9 carry 0; P_1 40+0,| 0 carry 4; P_2 16+4, 0 carry 2; P_3 0 (Thus, the product is 2009)

$$\begin{array}{rcl} & A_1 & A_0 \\ * & B_1 & B_0 \\ \hline & WXYZ & \end{array}$$

Fig. 3: Multiplication of two binary numbers using Urdhava Tiryagbhyam

achieved using this sutra is shown in Fig. 2. For the implementation of binary numbers using these sutras, a carry select adder is used (Kumar, 2013). To illustrate the steps involved in the Urdhava Tiryagbhyam Sutra, consider the values 41 and 49 for multiplication. Initial carry is assumed as zero. The operation is based on UT method. Carry of the operation will be propagated to the subsequent stages.

The 2×2 bit multiplication using Urdhava Tiryagbhyam algorithm for two binary numbers is shown in Fig. 3. Let $A_1A_0$and B_1B_0 are the two binary numbers used for the multiplication in the following four steps:

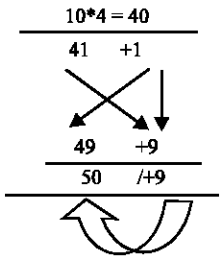
- $Z = A_0*B_0$ (LSB-vertical)
- The sum provides the product second bit. (LSB and MSB-Crosswise) $Y = A_1*B_0+A_0*B_1$
- The carry obtained from the step 2 is added with the multiplication of Most Significant Bit (MSB) (vertical) gives the product (Sundari *et al.*, 2013). $X = A_1*B_1$
- The carry obtained in step 3 is considered as a result W (Kerur *et al.*, 2011)

Anurupye sutra: This sutra is mainly used for finding the product of two numbers. The common bases are 20, 50, 100, etc., (multiples of power of 10). The Anurupye sutra multiplication is almost done based on the index 10 and it is not closer to the index 10 (10, 100, 1000, etc.,) (Pande *et al.*, 2013). Consider the multiplication of two

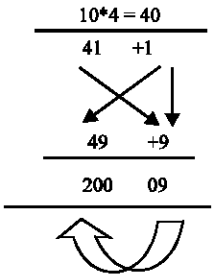
decimal numbers (41*49). In the Anurupye sutra, the 40 is taken as the base which is 4*10 instead of 100. Here, the factor is 4. The steps to be followed in Anurupye sutra are as follows: Consider the number 40 as the base. Register the compliments from the base adjacent to the given number:

10*4 = 40	
41	+1
49	+9

On the Right Hand Side (RHS), cross adds the MSB diagonally with the LSB to get the value as 50. Multiply the two LSBs and store it.



In the Left Hand the Side (LHS), the value 50 in step 2 is multiplied with the factor 4 to get the product, i.e., 50*4 = 200. Take the sum of the LHS with the carry, 200+0 = 200. Thus, the answer obtained from the Anurupye sutra in the above calculations as 41*49 = 2009.



RESULTS AND DISCUSSION

In the Urdhava Tiryagbhyam Sutra, a is the multiplier, b is the multiplicand and p is the product where other variables are the intermediate signal. For example in an 8×8 bit multiplier, the multiplier a value is 41 and the multiplicand b value is 49 is applied to the Urdhava Tiryagbhyam multiplier functional block and it produces the product output pas 2009. The output is a desired 16-bit number. Figure 4 shows an 8×8 bit multiplier. It is designed using Urdhava Tiryagbhyam and simulated using Xilinx ISIM simulator and it is functionally verified. In the Anurupye multiplier, the same input stimulus is applied to the multiplier, i.e., a = 41 and b = 49. Considering the base value as 40 instead of 100. The resultant output produces the value as 2009 and it is similar to the Urdhava Tiryagbhyam multiplier. Figure 5 shows an 8×8 bit multiplier designed using Aunrupye multiplier and it is simulated using Xilinx ISIM simulator. Both Urdhava Tiryagbhyam multiplier and the Aunrupye multiplier are functionally equal.

Both the Urdhava Tiryagbhyam multiplier and the Aunrupye multiplier are functionally equal. In this study, imulated the results of an 8-tap FIR filters using Anurupye multiplication algorithm and it is shown in Fig. 6.

FPGA synthesize result: The multiplier and FIR filter results were obtained from Xilinx Synthesis Technology (XST) with reference to Xilinx Spartan6 XC6SLX45T-2CSG324 device are shown in Table 1 and 2. Entities in terms of number of Look-Up Tables (LUT) and delay were analyzed (Patel *et al*., 2016; Saha *et al*., 2001, 2011).

ASIC synthesize result: The multiplier and FIR filter results were synthesized using cadence RTL compiler for both 180 and 90 nm CMOS technology and it is shown in Table 3 and 4. Entities in terms of area, leakage power, dynamic power and delay were reported and analyzed.

From the ASIC synthesize 90 nm technology results for FIR filters, the comparison of area in (μm^2) and leakage power of the multipliers are plotted in the graph and it is presented in Fig. 7 and 8. Figure 7 shows that Anurupye multiplication based FIR occupies less area compared to FIR using Urdhava Tiryagbhyam, it saves 16% of chip area when its implemented as IC.

Table 1: Comparison of multipliers in FPGA technology

	Xilinx Spartan6 XC6SLX45T-2CSG324	
Types of multiplier	No. of slice LUTs required	Delay (nsec)
Urdhava Tiryagbhyam	120	24.717
Anurupye	43	18.884

Table 2: Comparison of multiplier implemented FIR filter

	Xilinx Spartan6 XC6SLX45T-2CSG324	
Types of multiplier	No. of slice LUTs required	Delay (nsec)
FIR using Urdhava Tiryagbhyam	1752	69.791
FIR using Anurupye	1009	66.829

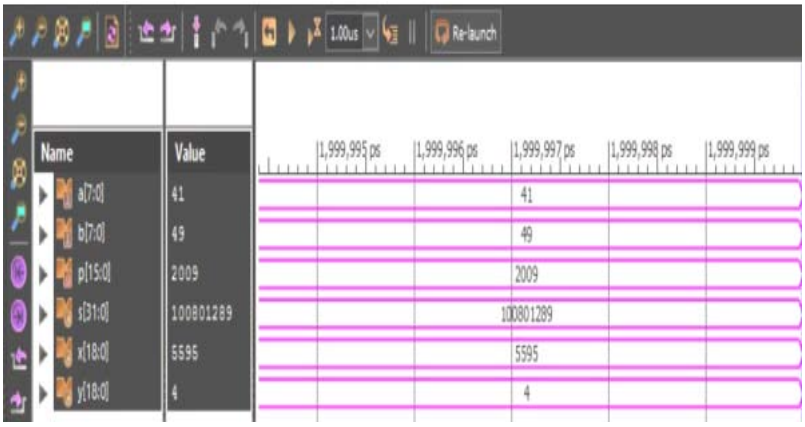


Fig. 4: Urdhava Tiryagbhyam multiplier result

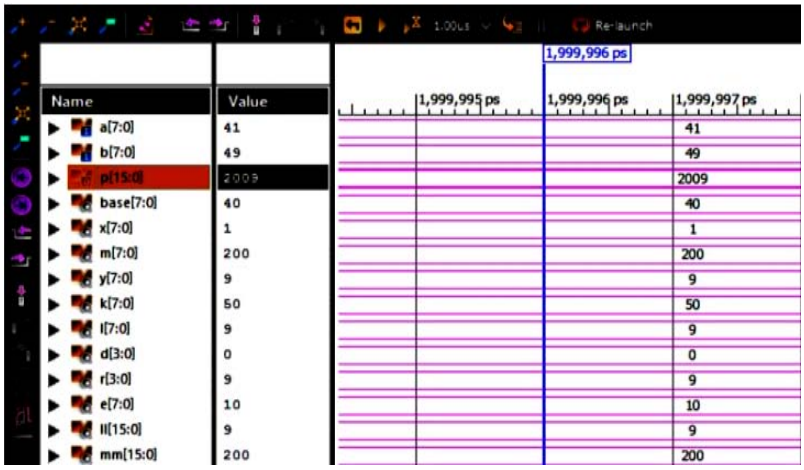


Fig. 5: Aunrupye multiplier result

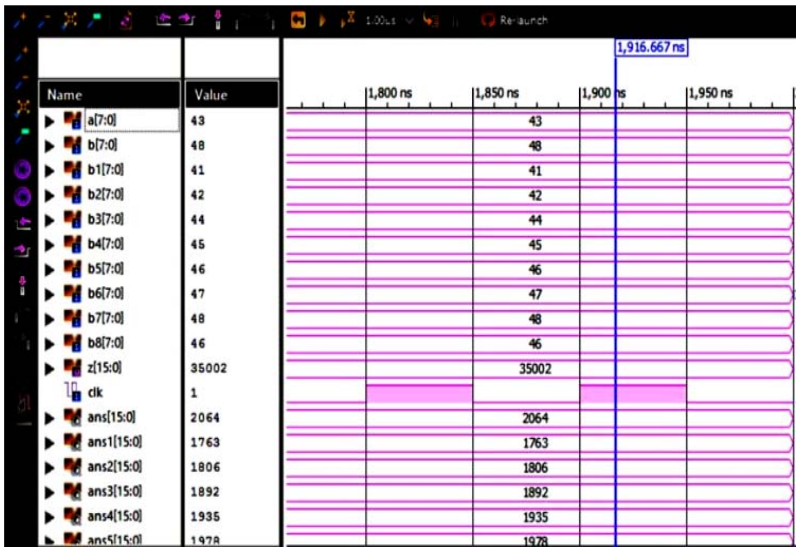


Fig. 6: Simulation result of FIR filter using Anurupye multiplication algorithm

Table 3: Comparison of multipliers in ASIC technology

Technology (nm)	Parameter	Urdhava	Anurupye Tiryagbhyam
180	Area(μm²)	5415	4244
	Leakage power (nW)	265.205	170.801
	Dynamic power (nW)	323100.819	410544.391
	Total power (nW)	323366.023	410715.192
	Delay (ps)	8397	9301
90	Area(μm²)	1651	1304
	Leakage power (nW)	9123.151	6371.147
	Dynamic power (nW)	51394.006	86761.845
	Total power (nW)	60517.157	93132.992
	Delay (ps)	7779	7081

Table 4: Comparison of multiplier implemented FIR filters in ASIC technology

Technology (nm)	Parameter Tiryagbhyam	Urdhava	Anurupye
180	Area (μm²)	81141	68218
	Leakage power (nW)	3446.487	2491.439
	Dynamic power (nW)	7066511.894	7021775.932
	Total power (nW)	7069958.381	7024267.371
	Delay (ps)	24591	28579
90	Area (μm²)	25254	21431
	Leakage power (nW)	137152.760	107151.207
	Dynamic power (nW)	1542063.922	1658207.601
	Total power (nW)	1679216.682	1765358.808
	Delay (ps)	19007	19319

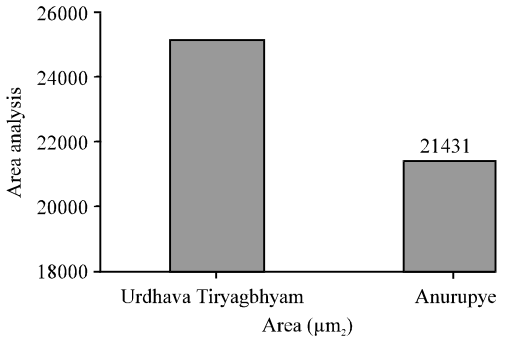


Fig. 7: Area analysis

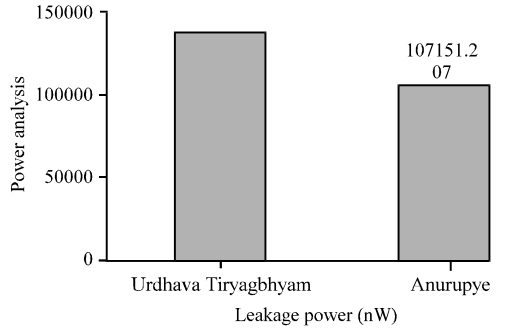


Fig. 8: Power analysis, leakage power (nW)

Figure 8 shows that Anurupye multiplication based FIR consumes less leakage power compared to FIR using Urdhava Tiryagbhyam, it saves 12% of leakage power saving.

CONCLUSION

In this research, the vedic multiplication algorithms like Urdhava Tiryagbhyam, Anurupye and its FIR filter implementation were discussed. Algorithms were described using VHDL and synthesized in both FPGA and ASIC technology with respect to the Xilinx Spartan6 XC6SLX45T-2CSG324FPGA Device, CMOS (18, 90 nm) Technology, respectively. The performance evaluations of the algorithms which include the parameters like area, power, delay and LUT were reported. Synthesize results shows that the Anurupye algorithm results in reduction of area and leakage power compared to the Urdhava Tiryagbhyam algorithm for both 180 and 90 nm CMOS technology. In case of FPGA synthesize results with FIR implementation; the number of LUT required and propagation delay was reduced in the Anurupye algorithm.

REFERENCES

Chidgupkar, P.D. and M.T. Karad, 2004. The implementation of vedic algorithms in digital signal processing. Global J. Eng. Educ., 8: 153-158.

Dhillon, H.S. and A. Mitra, 2008. A reduced-bit multiplication algorithm for digital arithmetic. World Acad. Sci. Eng. Technol., 19: 719-724.

Evans, J.B., 1994. Efficient FIR filter architectures suitable for FPGA implementation. IEEE. Trans. Circuits Syst. II. Analog Digital Signal Process., 41: 490-493.

Itawadiya, A.K., R. Mahle, V. Patel and D. Kumar, 2013. Design a DSP operations using vedic mathematics. Proceedings of the International Conference on Communications and Signal Processing (ICCSP), April 3-5, 2013, IEEE, Melmaruvathur, India, ISBN:978-1-4673-4865-2, pp: 897-902.

Kerur, S.S., P. Narchi, C.N. Jayashree, H.M. Kittur and V.A. Girish, 2011. Implementation of vedic multiplier for digital signal processing. Intl. J. Comput. Appl., 16: 1-6.

Kumar, H. and K. Kumar, 2016. Transposed structure design of FIR filter using VHDL. HCTL. Open Intl. J. Technol. Res., 19: 1-7.

Kumar, H., 2013. Implementation and analysis of power, area and delay of array, urdhava, nikhilam vedic multipliers. Intl. J. Sci. Res., 3: 1-5.

Maharaja, J.S.S.B.K.T., 2009. Vedic Mathematics. Motilal Banarsidass Publisher, Bungalow Road, India,.

Mehkarkar, S.M. and S.J. Banarase, 2014. Implementation of high speed FIR filter based on ancient vedic multiplication technique. Intl. J. Emerging Technol. Adv. Eng., 4: 763-770.

Pande, N.A., V. Niranjane and A.V. Choudhari, 2013. Vedic mathematics for fast multiplication in DSP. Intl. J. Eng. Innov. Technol., 2: 245-247.

Park, J., W. Jeong, H. Choo, H.M. Meimand and Y. Wang *et al*., 2002. High performance and low power FIR filter design based on sharing multiplication. Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED'02), August 14, 2002, ACM, Monterey, California, USA., ISBN:1-5811-3475-4, pp: 295-300.

Patel, D.K., R. Chouksey and M. Saxena, 2016. Design of fast FIR filter using compressor and carry select adder. Proceedings of the 3rd International Conference on Signal Processing and Integrated Networks (SPIN), February 11-12, 2016, IEEE, Noida, India, ISBN:978-1-4673-9198-6, pp: 460-465.

Saha, P., A. Banerjee, A. Dandapat, P. Bhattacharyya, 2001. ASIC design of a high speed low power circuit for factorial calculation using ancient Vedic mathematics. Microelectron. J., 42: 1343-1352.

Saha, P., A. Banerjee, P. Bhattacharyya and A. Dandapat, 2011. High speed ASIC design of complex multiplier using vedic mathematics. Proceedings of the 2011 IEEE Symposium on Students Technology (TechSym), January 14-16, 2011, IEEE, Kharagpur, India, ISBN:978-1-4244-8941-1, pp: 237-241.

Sivasankar, M. and G. Nanjappan, 2015. Design of high speed multiplier for digital FIR filter using ancient urdhava tiryagbhyam techniques. Intl. J. Eng. Sci., 1: 30-36.

Sundari, R.P.M., D. Subathra and M.S. Dhanalaxmi, 2013. Enhancing multiplier speedin fast fourier transform based on vedic mathematics. Intl. J. VLSI. Des. Commun. Syst., 4: 135-144.