I J C T A, 9(16), 2016, pp. 7773-7782

Power Optimisation of FIR filters using an Advanced Number Representation

M. Likith Reddy*, K. Sai Rahul*, U. Nithin* and M. Valarmathi**

ABSTRACT

Power consumption is a key design limitation for several electronic digital systems. The power is increasing due to the continuous growth of the new circuit devices complexity. The integration of new applications and services in new devices is forcing the adoption of high frequencies and small technologies, which increases considerably the power consumption. New design techniques are, therefore, needed to limit the power consumption. This proposal targets in particular, the switching activity that represents the most important dynamic power consumption factor. Besides familiar binary and floating point number systems, there are also other alternative number systems such as level index number system (LI) and double base number system (DBNS) etc. available for design engineers. These kind of alternative number systems often have some intriguing property or properties, like efficient (sparse) number presentation, which may make them very attractive alternatives for certain applications. In this work The activity reduction is achieved through the usage of a sparse representation system, the Double Base Number System (DBNS).DBNS is a highly redundant and sparse number system. In fact, the main property of this representation system is its high sparseness. This property makes DBNS representation very interesting for the representation of digital filters coefficients since it allows the reduction of the number of operations required for the filter implementation. The DBNS system is a weighted number system which uses two bases instead of one. Numbers of this form are referred to as 2-integers. On extending the exponents within an arbitrary signed integer space, then it is possible to represent any real number, with arbitrary precision, using a single non-zero digit. The single digit can be mapped by its binary and ternary exponents, thus allowing an index calculus with which can be perform arithmetic using logarithmic-like computational units. Expressing the filter coefficients and input data in DBNS makes multiplication simple and fast. A conventional 8-tap filter is written in Verilog HDL and simulated in Model Sim. An optimized filter with DBNR of data is synthesized in Cadence Encounter and a power reduction of 56% is obtained, when compared to conventional filter.

Keywords: DBNS, FIR Filter, Power Optimization, Index Calculus, Digital Signal Processing.

1. INTRODUCTION

Complexity and power consumption are two major factors in limiting the design of digital systems. In many applications the computational complexity of algorithms crucially depends upon number of zeroes of the input data in corresponding number system. Number systems are often chosen to enable a reduction of complexity of the arithmetic operations; the most popular being signed digit number systems. The switching activity is directly proportional to the complexity of the filter. Traditional methods provide trivial solutions, but with some shortcomings. Having decent knowledge in DSP and filter design, the above stated problem initiated this work.

2. CONCEPT OF DBNS

2.1. DBNS

The representation of a given integer x into the form will be referred to as a double-base number system (DBNS)[11][12].

* Dept. of Electronics and Communication SRM University, Chennai Tamilnadu, India.

** Asst. Professor (Sr. G), Dept. of ECE SRM University, Chennai Tamilnadu, India.

$$X = \sum_{(i,j)} \left(2^i \times 3^j\right)$$

This type of representation of any number using the sum of 2-integers is called Canonical Representation. The main advantage of this representation is it being very much sparse in nature.

2.2. Representation of DBNS

The method to find a canonical representation to a large number is very complicated process, but to overcome this we propose a theory called Greedy Algorithm. In this algorithm the for a given number x, we find the closest number *w* such that $x \geq w$. Then the same process is applied recursively to $x - w$. we call this representation as Near canonic representation.More precisely, we try to find two non-negative integers *a*, *b* such that $2^a\ 3^b \leq x$, and among the solutions to this problem, $2^a\ 3^b$ is the largest possible value.

Generally, this follows the following algorithm

If ($x > 0$) then do

{

find the largest 2-integer $w <= x$;

write(w);

$x = x - w$;

greedy(x);

}

else exit;

This can even be represented graphically. In graphical representation vertical columns represent 2^i while horizontal rows represent 3^j where values of both (i, j) starts from 0. This is a 2-d representation. The darkened cells represent the non-zero active cells.

T\B	1	2	4	8	16
1					
3					
9					
27					

2-D representation of number 88

3. REQUIRED AIRTHEMATIC TRANSFORMATIONS

The method for finding the near canonical representation plays an important role for performing basic arithmetic operations. Along with sparseness we should also see that non zero digits are not consecutive in our mapping representation allowing addition mapping as simple Boolean operation.

3.1. Reduction rules

While representing a number graphically, there are few combinations to be used which simplify the representation.

I. $2^i\ 3^j\ 2^{i+1}\ 3^j = 2^i\ 3^{j+1}$

	2^{i-1}	2^i	2^{i+1}	2^{i+2}
3^{j-1}				
3^j				
3^{j+1}				
3^{j+2}				

	2^{i-1}	2^i	2^{i+1}	2^{i+2}
3^{j-1}				
3^j				
3^{j+1}				
3^{j+2}				

This is used to eliminate 2 consecutive active cells in single column.

II. $2^i\ 3^j + 2^i\ 3^{j+1} = 2^{i+2}\ 3^j$

	2^{i-1}	2^i	2^{i+1}	2^{i+2}
3^{j-1}				
3^j				
3^{j+1}				
3^{j+2}				

	2^{i-1}	2^i	2^{i+1}	2^{i+2}
3^{j-1}				
3^j				
3^{j+1}				
3^{j+2}				

This is used to eliminate 2 consecutive active cells in single row.

III. $2^i\ 3^j + 2^{i+1}\ 3^j + 2^i\ 3^{j+1} = 2^{i+1}\ 3^{j+1}$

	2^{i-1}	2^i	2^{i+1}	2^{i+2}
3^{j-1}				
3^j				
3^{j+1}				
3^{j+2}				

	2^{i-1}	2^i	2^{i+1}	2^{i+2}
3^{j-1}				
3^j				
3^{j+1}				
3^{j+2}				

This is used to eliminate when immediate neighbour in column and row are present for a particular cell. This rule can be used in different cases

4. ARITHEMATIC USING DBNS

4.1. Addition

Let X and Y be two integers which can be represented by canonical form i.e. $2^i .3^j$. DBNS map over the other. We also use the reduction methods Also $2^{i+1}.3^j$ element does not exist then the addition can be performed overlying ones DBNS map over the other. Ideally we reduce it into minimal form for newer addition but, we use the reduction methods in order to reduce the calculation time.

	1	2	4	8
1				
3				
9				
27				

	1	2	4	8
1				
3				
9				
27				

	1	2	4	8
1				
3				
9				
27				

	1	2	4	8
1				
3				
9				
27				

The main concern for this is its boundary conditions. In few cases we might end up in a situation where after overlying 2 DBNS graphs over each other application of reduction rules might cause an overflow condition [3]. This condition can be overcome by adding an extra row and column to the right of the array. There are few other methods to overcome the overflow condition but, those methods have limitations.

4.2. Multiplication

Multiplication operation in DBNS using the graph involves multiplication of each and every active cell of one graph with other. Then the resultant cells are marked active, however reduction rules are applied for

	1	2	4	8
1				
3				
9				
27				

1 2 4 8

1				
3				
9				
27				

	1	2	4	8
1				
3				
9				
27				

	1	2	4	8
1				
3				
9				
27				

further simplification. In multiplication process there is good chance of overlapping/super-imposing of active cells. This condition can be reduced by shifting the super-imposed active cell by 1towards right side of the same row. Also it is very much possible for occurrence of the overflow condition during the multiplication process.

5. INDEX CALCULUS

The proposed representation of number makes operations such as multiplication and division simple [13]. The complexity of these operations is reduced to such an extent, they as plain as performing addition and subtraction [9][10]. The following equation represents the index calculus operation

$$a = (s_a, b_a, t_a)\ b = (s_b, b_b, t_b)$$
$$a \times b = (s_a . s_b, b_a + b_b, t_a + t_b)$$
$$\frac{a}{b} = (s_a . s_b, b_a - b_b, t_a - t_b)$$

Let us consider,

$a = 32$ $b = 27$

In DBNR, $a = 2^5 3^0$ $b = 2^0 3^3$

Multiplication $*b = (1, 5 + 0, 0 + 3) = (1, 5, 3) = 2^5 3^3 = 864$

Let,

$a = 54 b = 27$

In DBNR, $a = 2^1 3^3$ $b = 2^0 3^3$

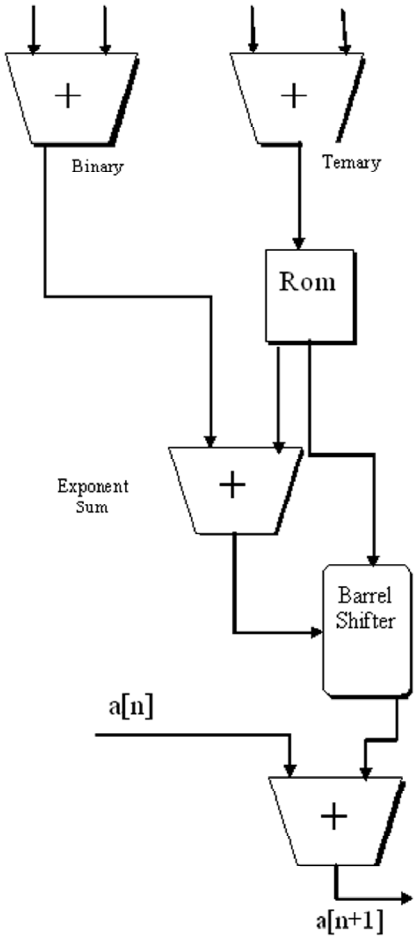
Division $a/b = (1, 1 - 0, 3 - 3) = (1, 1, 0) = 2$

6. IPSP

IPSP(Inner Product Step Processor)[6][8] is an important building block for the DSP processors. This mainly runs on the function of

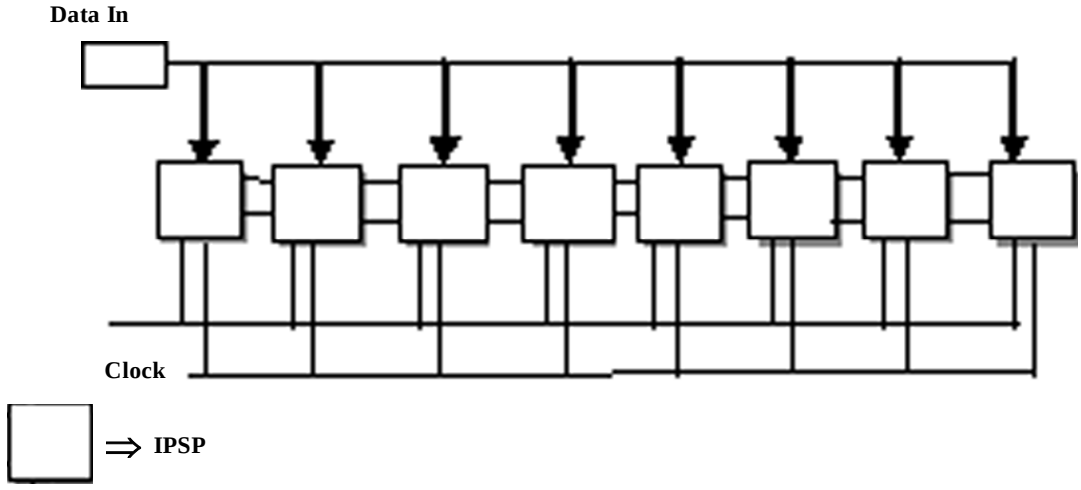
$$a[n+1] = a[n] + (d(n) \times c(n))$$

The mappings of x and y each produces binary exponent and ternary exponent. Then the binary and ternary exponents are summed separately. Instead of mapping back to DBNS, the resultant of summation of ternary exponents is converted to a floating point-type representation of $m.\ 2^n$. Then binary exponents are again added and the floating point is converted to fixed point using barrel shifter.



7. DBNS FILTER

In Diag.1 the main function of IPSP[1][3] is to reduce complexity created by multiplier in the conventional 8-Tap filter. During simulation we test IPSP in a multi-tap filter setting. The look up tables present are generated using custom programming package which has predefined values for a set of numbers, whereinone data conversion rom is required for DBNR [14][15] of input data and multiplier coefficients. Another rom is used for representing ternary exponents as floating point with binary exponent. The binary exponent adders are set such that barrel shifter provides a shift of ±16 places. Barrel shifter converts floating point into fixed point representation.



Diag. 1: Schematic of DBNS Filter using IPSP

8. IMPLEMENTATION REPORTS

8.1. Model Sim Simulation

A conventional8-Tap Finite Impulse Response (FIR)[5][7]filter is written in Verilog Hardware Description Language(HDL) using ModelSim Altera Started Edition 6.4a. The filter is successfully simulated with 8-bit fixed multiplier coefficients and a set of 8-bit input data‘s and the output is verified. A screenshot of simulation window

Of the conventional filter is shown in the fig 2.

An optimized 8-bit input filter is coded in Verilog HDL with IPSP as the top module. The filter is simulated in ModelSim with similar 8-bit multiplier coefficients and inputs. Fig 1 shows the simulation window of the proposed DBNS filter.

Comparing both the results, it can be concluded that both the filters functionality is similar.

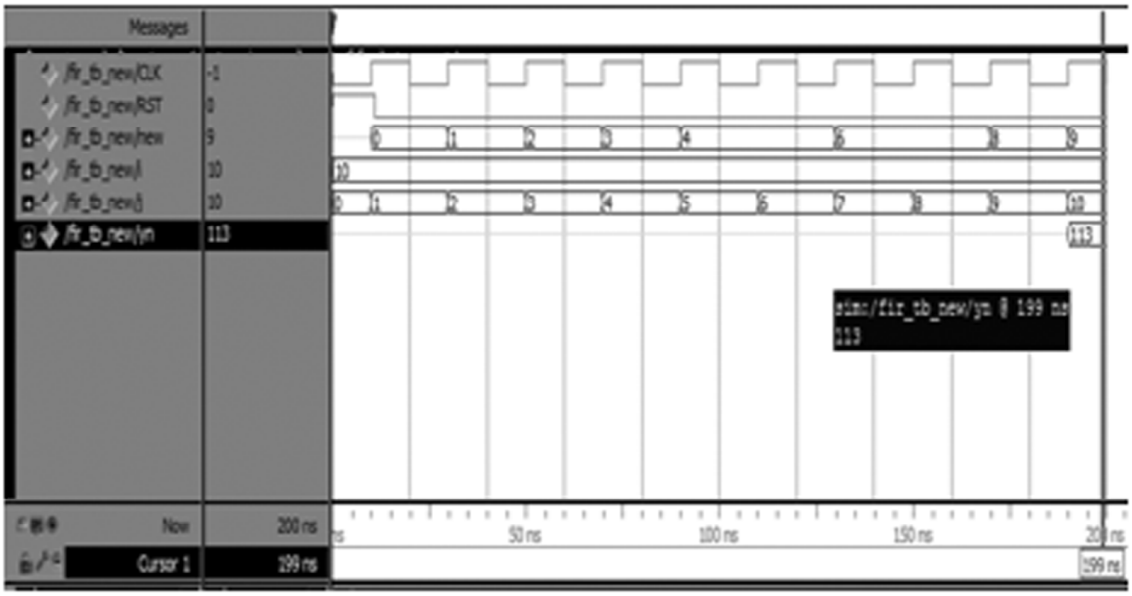


Figure 1: Simulation Window of DBNS Filter

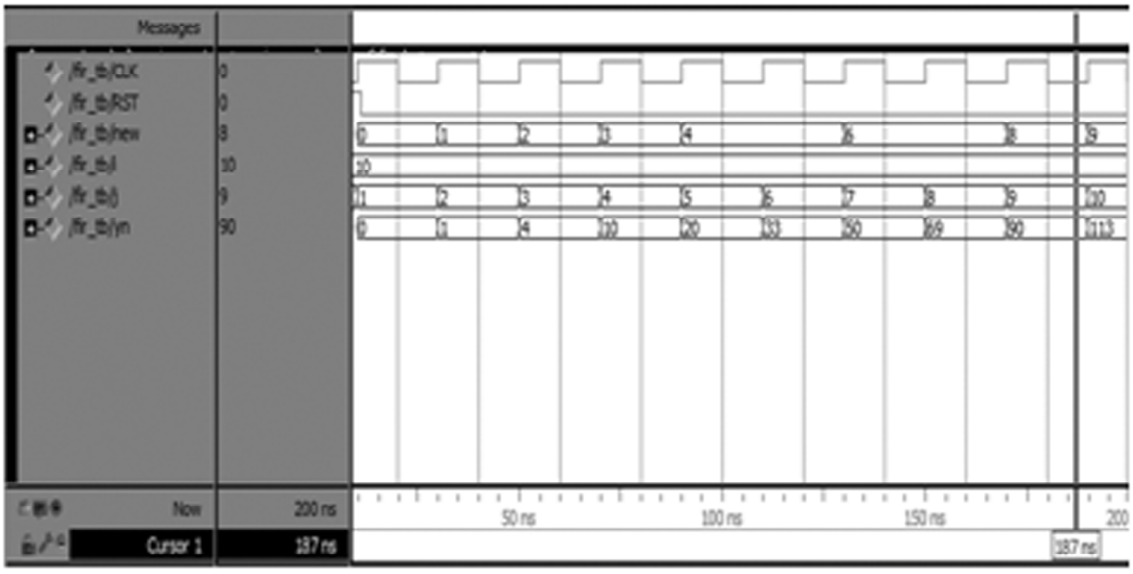


Figure 2: Simulation Window of FIR Filter

8.2. Cadence Encounter Synthesis

In order to compare the power optimization of the proposed filter, power analysis is done in Cadence Encounter tool.The considered filters are respectively an 8 bit-input filter with conventional multipliers and another 8 bit-input filter with Index Calculus, both with 8 taps.

The schematics for both the filters are shown below in fig 3 and fig 4.

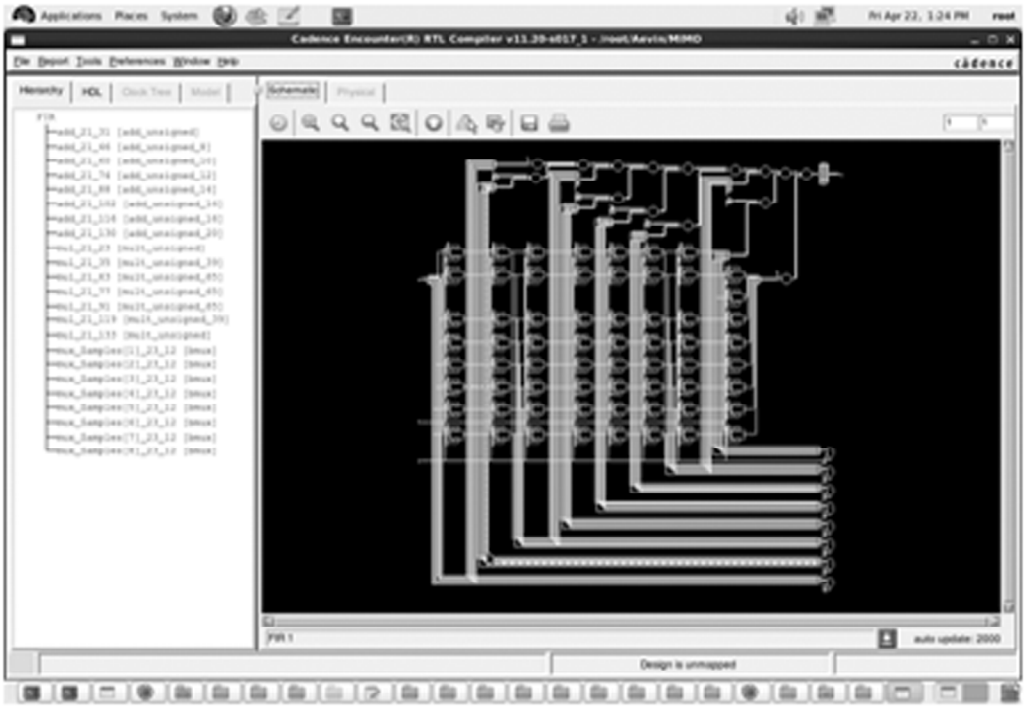


Figure 3: Schematic of a conventional Filter

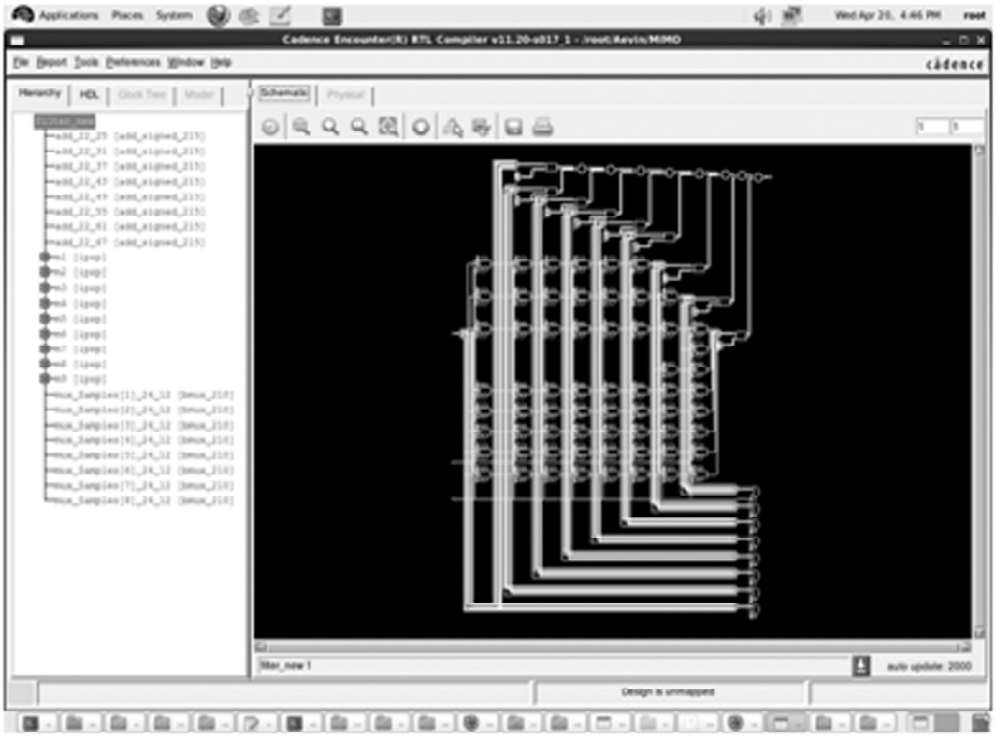


Figure 4: Schematic of DBNS Filter using IPSP

Implementation results regarding average power consumption are tabulated in the table 1. The results show a sizable amount of variation of 56% by the optimized filter in comparison with conventional filter which depicts the sparseness of DBNS representation.

The below illustrated reports fig.5, fig.6 depict that, even in the event of reduction of power consumption by 56.377% not much hindrance is observed in the delay parameter of the above discussed filter.

	Conventional Flter	*Optimized Filter*
Leakage Power(nW)	29852.787	119758.206
Dynamic Power(nW)	436853.244	83842.293
Total Power(nW)	466706.031	203600.499

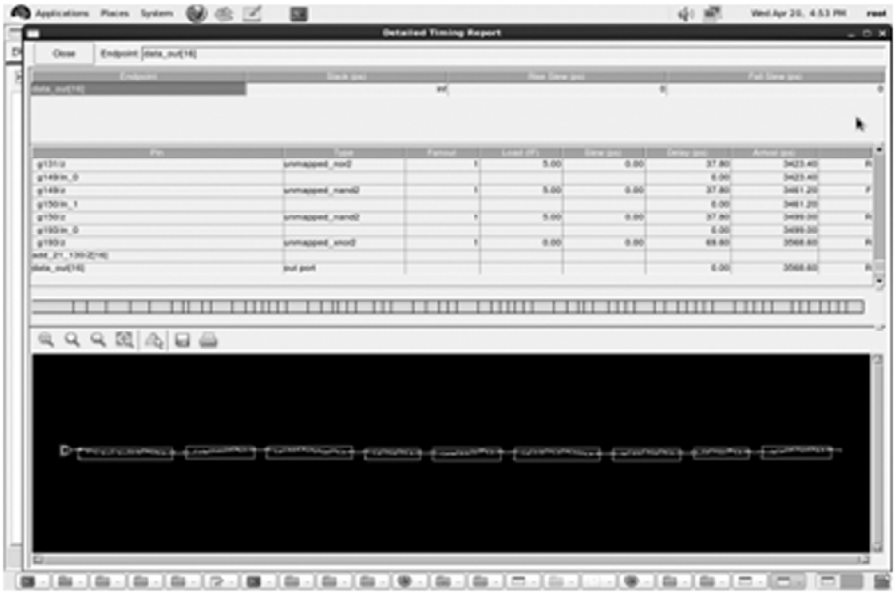


Figure 5: Conventional Filter

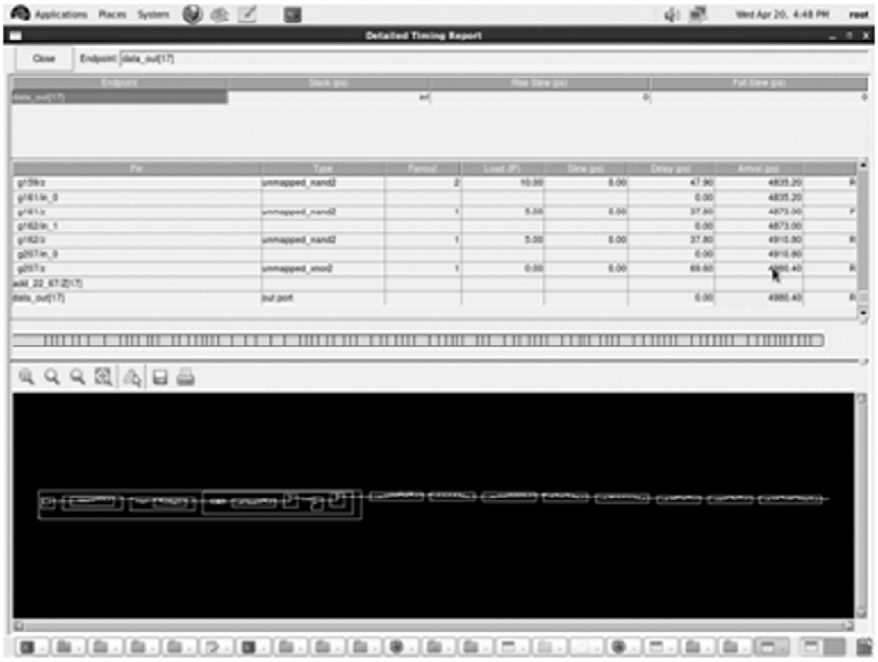


Figure 6: Optimized Filter

9. CONCLUSION

This work provides an optimum solution for reducing the power consumption of a conventional Finite Impulse Response filter with fixed multiplier coefficients. This is achieved by representing the coefficients and input data in Double Base Number System and performing the calculations in index calculus. An implementation advantage of this system is that the index additions and subtractions are reduced in complexity as the binary and the ternary operations are completely independent.

Apart from Digital Signal Processing the Double Base Number Systems can also be used in Cryptographic techniques[4] due to its unusual sparsity. The efficiency of this number system can further be improved by using multiple bases[2].

REFERENCES

[1] Jiajia Chen, Chip-Hong Chang, Feng Feng, Weiao Ding, Jiatao Ding, "Novel Design Algorithm for Low Complexity Programmable FIR Filters based on Extended Double Base Number System.", IEEE Transactions on Circuits, pp. 224-233, vol. 62, Issue 1, 01st Oct 2014.

[2] V. S. Dimitrov, G. A. Jullien, and R. Muscedere, Multiple-Base Number System: Theory and Applications. Boca Raton, FL, USA: CRC, 2012.

[3] N. T. A. El-Kheir, M. S. El-Kharashi, and M. A. EL-Moursy, A low power programmable FIR filter using sharing multiplication technique, in Proc. IEEE Int. Conf. IC Design Tech., Austin, TX,USA, May 2012, p. 4.

[4] J. Adikari, V. S. Dimitrov, and L. Imbert, Hybrid binary-ternary number system for ellipic curve cryptosystems, IEEE Trans. Comput., vol. 60, no. 2, pp. 254, Feb. 2011.

[5] R. Mahesh and A. P. Vinod, Newreconfigurable architectures for implementing FIR filters with low complexity, IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 29, no. 2, p. 275, Feb. 2010.

[6] J. Chen and C. H. Chang, High-level synthesis algorithm for the design of reconfigurable constant multiplier, IEEE Trans.Comput.-Aided Design Integr. Circuits Syst., vol. 28, no. 12, pp. 1844, Dec. 2009.

[7] Radhia Kacem, Nadia Khouja, Khaled Grati, Adel Ghazel, Low Power Implementation Of Digital Filters Using DBNS Representation And Sub-expression Sharing, International Conference on Signals, Circuits and Systems, pp. 1-6, 7-9 Nov. 2008.

[8] Mikko Pinknali, Ari Paasio, Mika Laiho, Implementation Alternatives of a DBNS Adder, 9th international workshop on Cellular Neural Networks and their Applications, May 28-30, pp. 138-141, 2005.

[9] V. Berth, L. Imbert, G.A. Jullien, More on Converting Numbers to the Double-Base Number System, Research Report LIRMM 04031, October 2004.

[10] Valerie Berthe, Laurent Imbert, On Converting Numbers to the Double-Base Number System, Advanced Signal Processing Algorithms, Architectures and Implementations XIV, volume 5559 of Proceedings of SPIE, pages 70-78. Denver, CO, USA, August 2004.

[11] Vassil S. Dimitrov and Graham A. Jullien, Loading the bases: A new number representation system and applications, IEEE Circuits and Systems Magazine, Vol. 3, pp. 6-23, 2003.

[12] J. Eskritt, R. Muscedere, G.A. Jullien, V.S. Dimitrov and W.C. Miller, A 2-Digit DBNS Filter Architecture, IEEE workshop on Signal Processing, 447, 2000.

[13] R. Muscedere, G. A. Jullien, V. S. Dimitrov and W. C. Miller, Non-linear signal processing using index calculus DBNS arithmetic, in print, SPIE 2000 Conference on Advanced Algorithms and Architectures for Signal Processing, San Siego, CA

[14] V. S. Dimitrov, G. A. Jullien, and W. C. Miller, Theory and applications of the double-base number system, IEEE Transactions on Computers 48, pp. 1098, October 1999.

[15] V.S. Dimitrov, G.A. Jullien, W.C. Miller, Theory and Applications of Double Base Number System, IEEE Symposium on Computer Arithmetic, pp. 44-51, 1997.