Circuits, Systems, and Signal Processing (2022) 41:2173–2187
https://doi.org/10.1007/s00034-021-01881-9

An Efficient Block-Based Architecture for Reconfigurable FIR Filter Using Partial-Product Method

Prabhat Chandra Shrivastava[1] · Prashant Kumar[2] · Manish Tiwari[2] · Amit Dhawan[2]

Received: 5 January 2021 / Revised: 10 October 2021 / Accepted: 11 October 2021 / Published online: 19 November 2021

Abstract
Multipliers are the most demanding component of any filter. They not only dominate most of the chip area but also contribute to most of the computational delay. An efficient realization of filter thus requires optimization of the multipliers. In this paper, a high performance block reconfigurable finite impulse response filter structure is presented. Block-based realization improves the overall throughput of the structure. A novel partial-product-based structure is proposed for Multiply and Accumulation operation, instead of traditional multipliers-based structure. Pipelining and parallelism in the structure improves the throughput of the design. The reuse of the partial products reduces the number of adders and increases the speed, thereby reducing the area-delay product (ADP) and energy-per output (EPO), in comparison with the earlier reported structures. The comparative analysis is done with the help of ASIC synthesis results and the results show that the proposed architecture for filter order 16 and block input 8requires 28.91% less ADP and 25.72% less EPO than the earlier reported architecture.

Keywords Block processing · Decoders · FIR filter · Multiplierless · Reconfigurable architecture

✉ Prabhat Chandra Shrivastava
prabhatphd@gmail.com

Prashant Kumar
pkumar.mnnit@gmail.com

Manish Tiwari
mtiwari@mnnit.ac.in

Amit Dhawan
dhawan@mnnit.ac.in

1 Rewa Engineering. College, Rewa 486002, MP, India

2 Motilal Neharu National Institute of Technology Allahabad, Teliyarganj, Prayagraj, Uttar Pradesh 211004, India

1 Introduction

With the advent of software-defined radio (SDR) technology researchers [2, 7, 11, 14, 23, 24, 33] have suggested digital signal processing techniques to achieve highly flexible operation of the system. Due to flexible and multi-standard properties, the SDR [14] has gained much attention for the reconfigurable signal processing system. The conventional FIR filters may not be suitable for many high speed modern communication applications. Therefore, alternate FIR filters called reconfigurable FIR (RFIR) filters are used to meet the requirements such as channelization and spectrum sensing, bandwidth adjustments, high speed, etc. [2, 5–8, 15, 23, 24, 26, 33]. In particular, the RFIR filters are often applied in scenarios where low power utilization, lesser chip area and self-adjustment at higher speeds are required [5, 8, 15, 26, 33].

The MAC units are the heart of the digital filters that perform all the computations required by the digital filters. Most of the computation time in a MAC unit is attributed to the multiplications performed in the MAC unit. Further, the computational time taken by a MAC unit increases linearly with the length of the input data. To deal with these issues, several authors have suggested the multiplier-less architectures using Distributive Arithmetic (DA), Canonical Signed Digit (CSD) techniques, etc [1, 3–5, 9, 10, 12, 13, 16, 18–22, 27–32, 34, 35]. The encoding of filter coefficients using the CSD technique reduces the number of partial products and therefore reduces the complexity of the filter [10, 12, 31]. Chen and Chiueh [4] proposed an architecture where both order of the filters and the number of nonzero digits in each tap can be arbitrarily assigned. Using this architectural design, matched filters, pulse shaping filters, etc. could easily be configured. The Multiple Constant Multiplication (MCM) technique in [29] is used in tap filters to reconfigure multiplier blocks. In a reconfigurable multiplier block, additional multiplexers (MUX's) are used for data transmission to configure the multiplication with a finite set of filter coefficients. However, for precise filter design, more hardware is required in this technique. In [13, 30, 35], a multiplier-less design with CSD encoding of coefficients is presented. Using this design technique, any order of high-speed FIR filters can be easily realized. Based on the CSD coefficient encoding technique, a new filtering method known as the common sub-expression elimination (CSE) method is proposed in [9, 18]. In the CSE method, the number of adders required for the multiplication is reduced at the expense of a slight increase in critical path length. Peter T. et al. [29] proposed a new technique to save the resources as common partial products can be shared between different coefficient sets of RFIR filters.

When the input vector length is large, the DA-based designs are more suitable than the other MAC designs for reconfigurable filters [19].The DA technique is very popular among all multiplier-less designs due to its efficient low-power structure of higher-order digital filters [19–22]. It has been employed for image coding, vector quantization, discrete cosine transform, and adaptive filtering implementations, etc. [1, 3, 19, 21]. In [19], sharing-based look-up tables (LUTs) are used for the efficient architecture of the RFIR filter and DA units are used for bit slices of different weights. The systolic architectures for higher-order FIR filters are proposed by Meheret al. in [20, 21], which reduces the problem of large memory requirements. Since the coefficients of RFIR filters change in run-time, a RAM-based LUT is proposed in [16, 19] instead of a ROM-based LUT. Further, in [17] & [19], efficient architectures for RFIR filters are proposed. In [17], the Programmable shift method (PSM) and Constant shift method (CSM) methods are used for the realization of RFIR filter and it has been shown that there are significant improvements in the proposed structure as compared to the earlier results. Even though, the method presented in [17] is advantageous for conventional RFIR filters but the method turned out to be unsuitable for the design of realization structure for block RFIR filters. The increased number of input samples to be processed in block processing filters demands efficient methods to reduce the processing time and power consumption. MCM scheme with the transposed form of structure is employed in [25] to meet such requirements. The MCM scheme used in [25] results in a significant saving of computational time and offer relatively efficient realization for higher-order FIR filters structure for both fixed and reconfigurable applications.

It is worth noting that, there are continuously growing demands by modern-day applications to efficiently process of a large amount of information, e.g. large size images and videos have to be processed at minimal computational overhead. This needs to increase the block size L. The MCM technique used in [25], which uses multipliers, turns out to be disadvantageous because the use of multipliers not only increases the computational time but also increases the power consumption and chip area. Hence, if we use a multiplier-less design to realize the block processing filter along with pipelining then we may expect to get improved results. Motivated by this, we consider the problem of designing a multiplier-less architecture for RFIR filter using decoder-based partial-product generation technique.

In this paper, we have proposed a novel architecture for block RFIR filters using the partial-product technique. An efficient decoder-based method is used to generate the inner-partial product terms corresponding to the bit combination of input. Further, pipelining and parallelism are used in architecture to reduce the critical-path delay. Furthermore, block processing improves the throughput of the proposed architecture to a great extent. In Sect. 2, the mathematical formulations for 1-D reconfigurable FIR filter along with block processing are discussed. Section 3 illustrates the proposed architecture of the block RFIR filter. The theoretical and synthesis result performance analysis are discussed in Sect. 4. Finally, conclusions are given in Sect. 5.

2 Mathematical Formulation

In this section, we discuss the formulation of conventional RFIR filters and block processing RFIR filters. It may be noted that both the filters have similar mathematical expressions but the coefficient of the RFIR filters changes in run time. Due to this behaviour, RFIR is applicable for radio communication systems such as SDR where the users are able to change its parameter as required. The output for the 1-D FIR filter may be written as:

$$y(m) = \sum_{i=0}^{N-1} x(m-i).h(i) \tag{1a}$$

where N is order of the filter and $0 \le i \le N-1$, $x(m-i)$ delayed input samples and $h(i)$ is filter coefficient. Further, 1(a) may be written as

$$y(m) = [x(m)\ \ x(m-1)\ \ \ldots\ldots\ \ x(m-N+1)] \begin{bmatrix} h(0) \\ h(1) \\ . \\ . \\ h(N-1) \end{bmatrix} \tag{1b}$$

In the case of block processing, L samples are required to be processed in each iteration to produce L outputs. So the output of the block processing filter may be given as

$$\begin{bmatrix} y(0) \\ y(1) \\ . \\ . \\ y(N-1) \end{bmatrix} = \begin{bmatrix} x(m) & \ldots & x(m-(L-1)) & .. & x(m-(N-1)) \\ x(m-1) & \ldots & x(m-(L-1)-1) & .. & x(m-(N-1)-1) \\ . & \ldots & . & .. & . \\ . & \ldots & . & .. & . \\ x(m-(L-1)) & \ldots & x(m-(L-1)-(L-1) & .. & x(m-(N-L)-(L-1)) \end{bmatrix} \begin{bmatrix} h(0) \\ h(1) \\ . \\ h(L-1) \\ h(N-1) \end{bmatrix} \tag{1c}$$

Equation 1(c) may be also written as

$$Y_k = X_k.h \tag{1d}$$

where Y_k is the $(k+1)^{th}$ block output and X_k is $(k+1)^{th}$ input vector. Further Eq. 1(d) can be rewritten as

$$Y_k = \sum_{i=0}^{N-1} X_k^i.h(i) \tag{1e}$$

where X_k^i is the $(i+1)^{th}$ term of $(k+1)^{th}$ row of input matrix X_k and $h(i)$ is the $(i+1)^{th}$ term of filter coefficients. Now for higher-order filters, i.e.N is sub-divided into small order L. The number of blocks formed is given as $Q = N\big/L$ where $N \ge L$.The index i of Eq. 1(e) can be written as $i = l + qL$ where $0 \le q \le Q-1$ and $0 \le l \le L-1$. Here l represents the sample in each block. Therefore Eq. 1(e) can be further written as

$$Y_k = \sum_{l=0}^{L-1} \sum_{q=0}^{Q-1} X_{k-q}^l.h(l+qL) \tag{1f}$$

Let X_k be divided into sub-blocks p_k^I, and h be sub-divided into f_q in accordance with (1c), then the partial product generated due to multiplication of the sub-matrices, may be given as

$$g_k^q = P_{k-q} \cdot f_q \tag{1g}$$

and final output can be obtained by

$$y_k = \sum_{q=0}^{Q-1} g_k^q \tag{1h}$$

Writing 1(h) in Z-domain, we get a recurrence relation as,

$$Y(z) = p^0(z)[(z^{-1}(......(z^{-1}(z^{-1}f_{q-1} + f_{q-2}) + f_{q-3}) + ...) + f_1) + f_0] \tag{1i}$$

where $p^0(z)$ and $Y(z)$ are the z-domain representation of p_k^0 and Y_k respectively. Here z^{-1} indicates the delay of a block of data g_k^q. Any block transpose form can be derived using the relation (1i). Multiplications in (1 g) are replaced by a block of the decoder and AND/OR sub-blocks. Depending on the number of bits chosen in the input sample of length B bits, the culmination of partial products is done. These bits will determine the requirement of the probable sum of coefficients. For n-bit, we will have 2^n combinations that require B/n number of $n : 2^n$ encoders.

3 Proposed Architecture

In this section, we present the architecture of block reconfigurable FIR filter using partial-product methods for multiplication which is different than the conventional multiplication performed by the multiplier. The overall block diagram of the proposed filter architecture for a block size of L is shown in Fig. 1. It has a filter-coefficient storage block (FCSB), where all sets of coefficients are stored and used for further

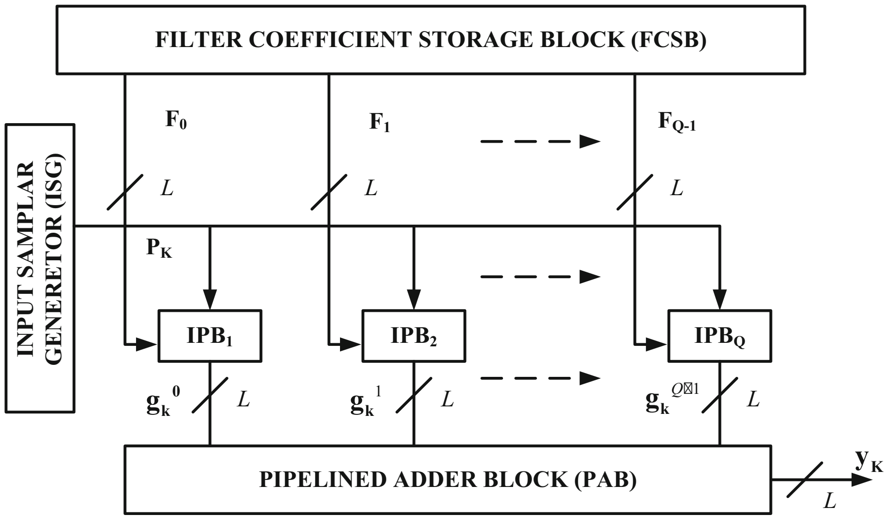

FILTER COEFFICIENT STORAGE BLOCK (FCSB)
F0
F1
FQ-1
L
L
L
INPUT SAMPLAR GENERETOR (ISG)
PK
IPB1
IPB2
IPBQ
gk0
gk1
gkQ-1
L
PIPELINED ADDER BLOCK (PAB)
yK
L

Fig. 1 Proposed structure for block RFIR filter

reconfigurable application of filters. These coefficients of the filter can be obtained by a single clock cycle. Input sampler generator (ISG) as depicted in Fig. 2 buffers the input samples and gives the required past input samples. The ISG receives inputs X_k during the kth cycle and produces L rows of p_k^0 in parallel and provides to the internal processing block (IPB) as shown in Fig. 3.

At the other side of IPB, the FCSB block provides weight-vectors of the filter simultaneously such that during the k^{th} cycle, the $(q+1)^{th}$ IPB receives the weight-vector f_q. The IPB is the heart of the proposed architecture. This architecture consists of L number of inner product cells (IPC) as shown in Fig. 4. The matrix multiplication occurs in each of the IPC blocks and gets the partial-products $g_k^0, g_k^1, \ldots, g_k^Q$.

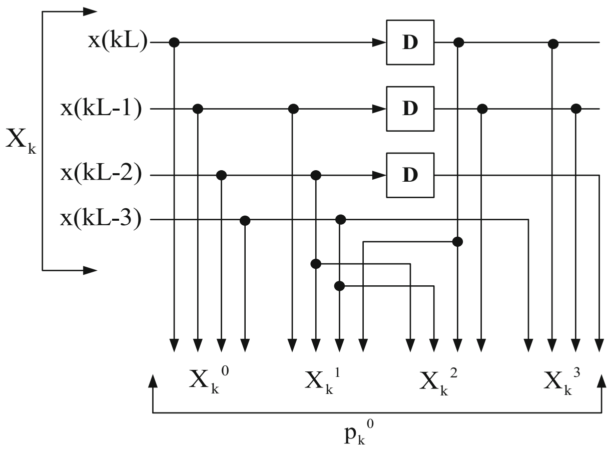


Fig. 2 Internal structure of Input sampler generator (ISG)

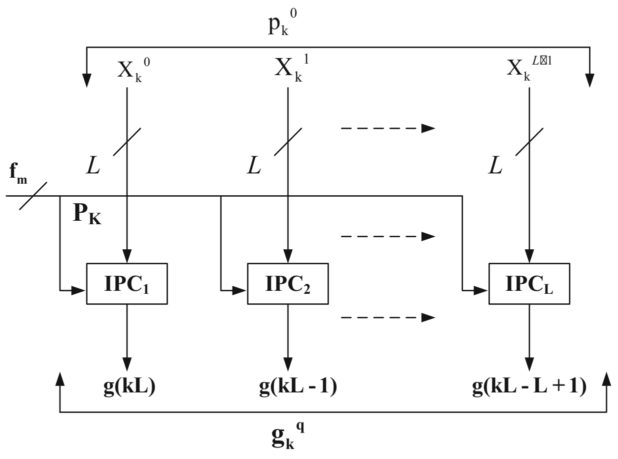


Fig. 3 Internal Processing Block (IPB)

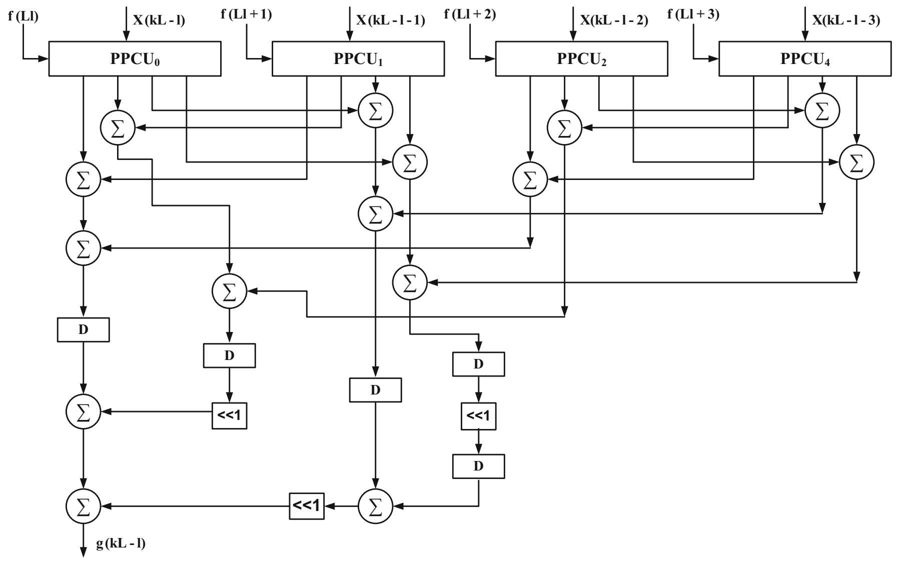


Fig. 4 $(k+1)$th IPC block

The IPC block consists of a partial product computational unit (PPCU), as depicted in Fig. 5, which is used for the multiplication of the filter coefficient vector f_q and the output of the ISG block, i.e.P_k^q. Here, we have used 2×4 decoder and AND-OR unit (AOU) for the partial product. Finally, the output of the partial product is added with the help of pipelined adder block (PAB). The architecture of AOU and PAB are shown in Fig. 6 and Fig. 7, respectively.

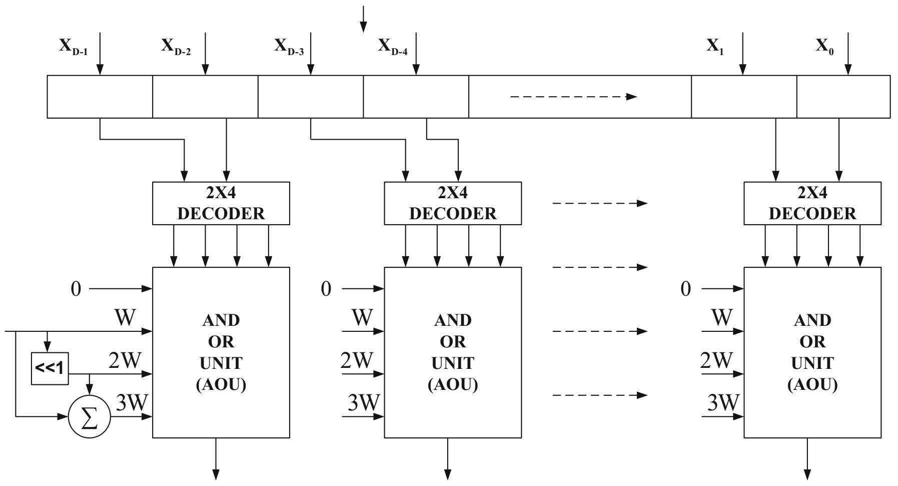
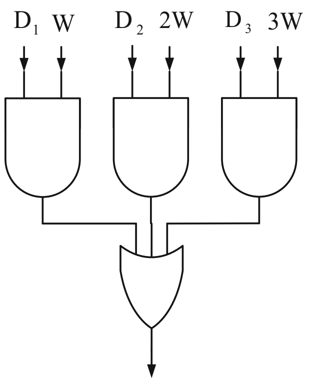


Fig. 5 Internal Structure of a PPCU block

Fig. 6 AND/OR block (AOU)



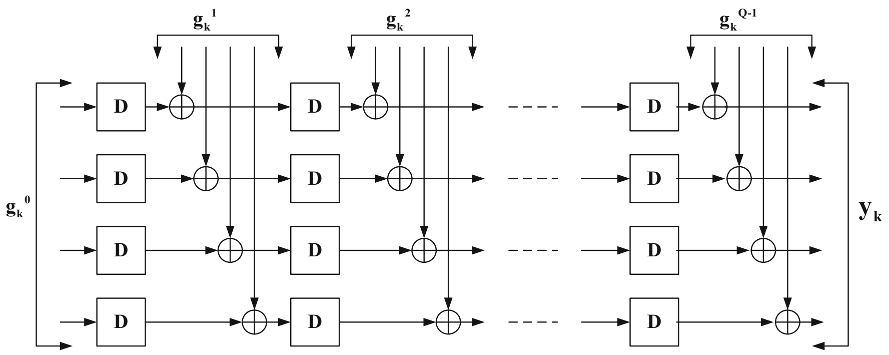


Fig. 7 Pipelined Adder Block (PAB)

4 Results and Discussion

In this section, hardware and time complexities of the proposed architectures are discussed. The theoretical comparisons of the proposed structure with the structures reported in [17, 25] are shown in Tables 1 and 2. Further, a comparison of the synthesis results is summarized in Table 3.

4.1 Theoretical Comparison of Proposed Architecture

The proposed architecture is realized using a single ISG unit, one FCSB, Q IPB units, and a single PAB unit. Let B' be the width of coefficients and B be the width of input samples X_k. The ISG requires $(N-1)$ registers of width B bit. The PAB block requires $L(Q-1)$ adders and the same amount of registers. The proposed structure involves $L(N-1)+NL(B/2)$ adders, $B(N-1)+(B^{'}+B)(N-L)$ FFs (flip flops) and $NL(B/2)$ decoders to process L samples in every cycle where the duration of the cycle is $T_{DEC}+T_{AND}+T_{OR}+4T_A$. The general comparison of hardware and time complexities for the proposed structure, CSM-based architecture[17] and block RFIR

Table 1 General comparison of hardware and time constraints

Structure	Flip-flop	Adder	Multiplier	MUX (2:1)	Decoder (2:4)	Cycle period	Throughput
Mahesh et al.[17]	$(B+B')(N-1)$	$N((B'/3)+1)+2$	0	$(7B^{'}/3)(B+2)$	0	$4T_1 + ((\log_2(B'/3)T_{FA}$	$1/CP$
B K Mohanty et al. [25]	$B(N-1)+$ $(B+B'+2)(N-L)$	$L(N-1)$	NL	0	0	$T_2 + (\log_2(L)T_{FA}$	L/CP
Proposed	$B(N-1)+$ $(B+B')(N-L)$	$L(N-1)+NL(B/2)$	0	0	$NL(B/2)$	$T_{DEC} + T_{AND} + T_{OR} + 4T_A$	L/CP

Table 2 Theoretically estimated hardware and time complexities of proposed and existing structures for $B = 8$ and $B' = 16$

Structures	Filter-length	FF	Adder	Multiplier	Mux (2:1) Bit-level	Decoders (2:4)	Cycle period (T)	Throughput
Mahesh [17]	16	360	114	0	7104	0	$4T_{MUX} + 4T_A + 3T_{FA}$	$1/T$
	32	768	226	0	14,208	0	$4T_{MUX} + 4T_A + 3T_{FA}$	$1/T$
	64	1536	450	0	28,416	0	$4T_{MUX} + 4T_A + 3T_{FA}$	$1/T$
Mohanty [25] L = 4	16	432	60	64	0	0	$T_M + T_A + 2T_{FA}$	$4/T$
	32	976	124	128	0	0	$T_M + T_A + 2T_{FA}$	$4/T$
	64	2064	252	256	0	0	$T_M + T_A + 2T_{FA}$	$4/T$
Proposed Structure L = 4	16	408	316	0	0	256	$T_{DEC} + T_{AND} + T_{OR} + 2T_A$	$4/T$
	32	920	636	0	0	512	$T_{DEC} + T_{AND} + T_{OR} + 2T_A$	$4/T$
	64	1944	1276	0	0	1024	$T_{DEC} + T_{AND} + T_{OR} + 2T_A$	$4/T$

$T_1 = T_{MUX} + T_A,\ T_2 = T_M + T_A$

Table 3 Comparison of synthesis results of the proposed structures and the suggested structures of Mahesh [17] & Mohanty [25] for the filter of order (*N*) 16, 32 & 64 and block size (*L*) 4 & 8 respectively. Bit-width of filter coefficients is *B*' = 16 and bit-width of input samples is *B* = 8. Power Consumption is estimated at maximum clock pulse (MCP)

Designs	Filter Length (N)	MCP (ns)	Area (μm^2)	Power consumption (mw)	EPO (pJ)	ADP (μm^2 x ns)	Reduction with respect to synthesis results of Mahesh [17]	
							ADP	EPO
Mahesh [17]	16	1.27	67,949	28.70	36.49	86,295.23	-	-
	32	1.29	133,015	55.80	71.98	171,589.35	-	-
	64	1.30	264,342	104.40	135.72	343,644.60	-	-
Mohanty et al. [25] L = 4	16	1.30	123,489	59	19.17	40,133.92	53%	47.5%
	32	1.30	247,350	108.20	35.17	80,388.75	53%	51.1%
	64	1.30	495,437	201.10	65.35	161,017.02	59%	51.8%
Proposed Structure L = 4	16	1.11	96,640	49.28	13.67	26,817.60	68%	62.5%
	32	1.17	199,765	91.29	26.70	58,431.26	66%	62.9%
	64	1.22	396,560	176.67	53.88	120,950.80	65%	60.3%
Mohanty [25] L = 8	16	1.40	232,089	100.9	17.65	40,615.57	52%	51.6%
	32	1.40	476,503	186.40	32.62	83,388.02	51%	54.6%
	64	1.40	957,186	366.20	64.08	167,507.55	51%	52.8%
Proposed Structure L = 8	16	1.17	190,578	89.67	13.11	27,872.03	67.7%	64%
	32	1.27	381,121	172.45	27.37	60,502.95	64.7%	62%
	64	1.32	765,689	330.49	54.53	126,338.68	63.2%	60%

MCP: Minimum Clock Period, ADP: Area-Delay-Product (Area x MCP/L), EPO: Energy-Per-Output (Power x MCP/L) architecture proposed in [25] are listed in Table 1. The CSM-based architecture [17] is efficient if the coefficients are reoccurring, that way a lot of repetition in multiplication can be saved. Although [17] is implemented without using any multipliers, but it uses a large number of MUX resulting in large usage of area. It may also be noted that there is a reduction in the cycle period of the proposed structure as it depends on the bit-width B of input samples and B' of coefficient sample. Architecture proposed in [17] does not support block processing thus lowering the throughput of the filter. The filter presented in [25] is lagging behind in performance due to the large usage of multipliers. As the order of the filter increases, the complexity also increases. Theoretically estimated hardware and time complexities of proposed and existing structures [17, 25] for $B = 8$ and $B' = 16$ are presented in Table 2.

4.2 Synthesis Results and Performance Comparison of Proposed Architecture

To validate the proposed design, we have coded the proposed designs in Verilog HDL for filter orders 16, 32 & 64 and block input samples 4 and 8, respectively. Here, bit-width of input samples is taken $B = 8$ and bit-width of filter coefficients is taken $B' = 16$. Coded designs are synthesized by Synopsys design compiler using TSMC 65 nm CMOS standard library. Synthesis results of the proposed structures and the structures of [17, 25]are listed in tabular form in Table 3. From Table 3, it can be observed that the proposed structures require lesser area-delay product (ADP) and energy per output (EPO) than the structures of [17, 25] respectively. As shown in Table 3, for the block size $L = 4$, the proposed structure for filter sizes 16, 32, and 64 requires 68%, 66%, and 65% lesser ADP and 62.5%, 62.9% and 60.3% less EPO, respectively, as compared to the structure discussed in [25]. Similarly, for block size $L = 8$ and the same order of filters, the proposed structure has 67.7%, 64.7%, and 63.2%, less ADP and 64%, 62% and 60% less EPO, respectively, as compared to the structure discussed in [25]. It is also evident from Table 3 that the structure proposed in [25] is superior to the one in [17], and hence, our proposed structure which offers much higher performance than [25] is much more advantageous than structures in [17] and [25]. The detailed comparison of reduction in ADP, EPO is shown in the bar chart of Fig. 8a, and b respectively.

5 Conclusion

In this paper, an efficient architecture for a reconfigurable FIR filter has been proposed. A partial-product-based method has been used for the MAC design. Decoder- and AOU-based structures are used to generate the inner partial-products. Further, multistage pipelining is used to reduce the critical-path delay. The analysis of different structures indicates the efficiency of the proposed structure in terms of area as well as operating speed. Usage of decoders greatly reduces the complexity of the structure. Critical-path delay of the structure is independent of size of the filter therefore, it can be observed from the synthesis results that MCP is almost constant with size of the

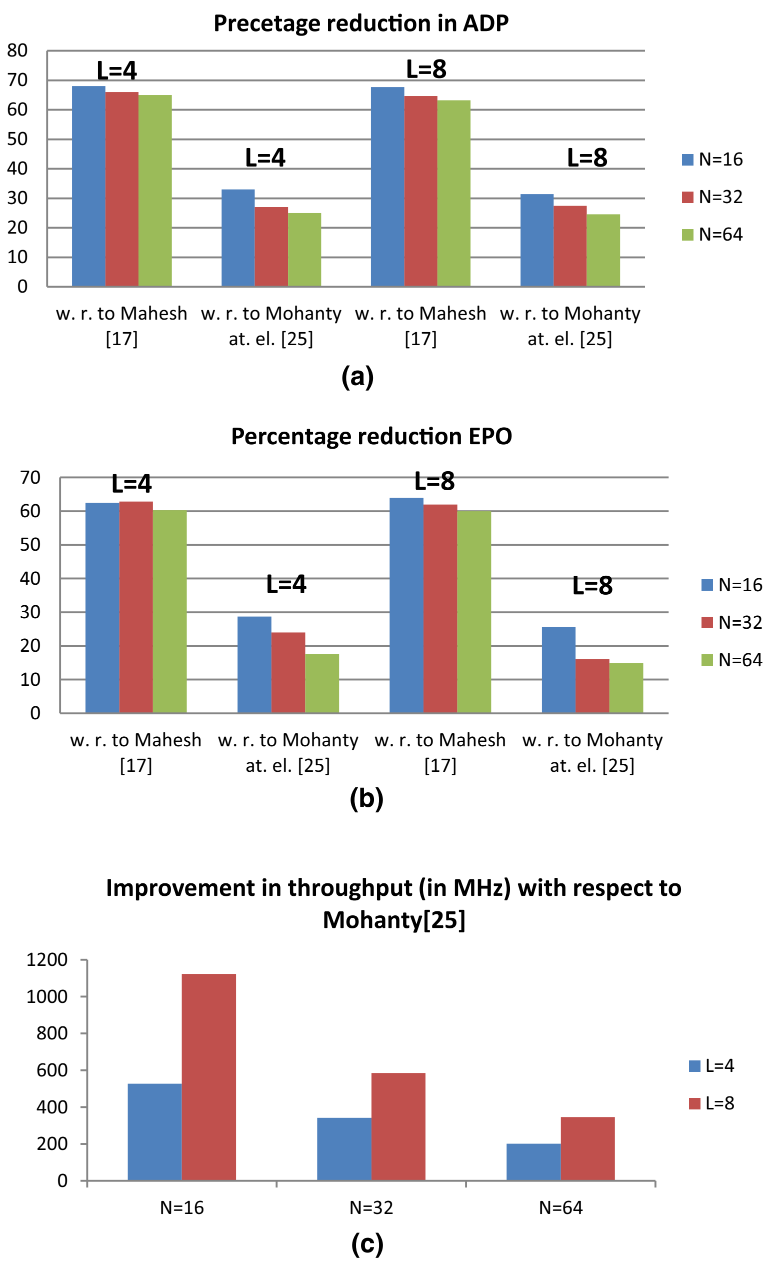


Fig. 8 **a** Reduction in ADP, **b** Reduction in EPO, **c** Improvement in throughput

filters. Increase in the block size will increase the area but will not affect the performance of the filter. It may also be noted that the proposed method is a generalized approach for low complexity reconfigurable filters.

Acknowledgements The authors would like to give thanks to Editor-in-Chief and to the anonymous reviewers for their valuable comments and suggestions to improve the quality of this paper.
We further declare that, all data is provided in full in the results section of this paper and this manuscript has no associated data.

References

1. D.J. Allred, H. Yoo, V. Krishnan, W. Huang, D.V. Anderson, LMS adaptive filters using distributed arithmetic for high throughput. IEEE Trans. Circuits Syst. I. Reg. Papers **52**(7), 1327–1337 (2005)
2. E. Buracchini, The software radio concept. IEEE Commun. Mag. **38**, 138–143 (2000)
3. H.Q. Cao, W. Li, VLSI implementation of vector quantization using distributed arithmetic. Proc. IEEE Int. Symp. Circuits Syst. **2**, 668–671 (1996)
4. K. H. Chen, T. D. Chiueh, Design and Implementation of A Reconfigurable FIR Filter. IEEE International Symposium on Circuits and Systems, 2003 (ISCAS '03), Bangkok, Thailand, 25–28 (2003).
5. X. Chenghuan, C. He, Z. Shunan, W. Hua, Design and implementation of a high speed programmable polyphase FIR filter, in Proc. 5th Int. Conf. Applicat.-Specific Integr. Circuit, vol. 2., pp. 783–787 (2003).
6. C. S. Choi, H. Lee, An reconfigurable FIR filter design on a partial reconfiguration platform, Communications and Electronics, 2006. ICCE '06. First International Conference on. IEEE, 352–355 (2006).
7. S. J. Darak, S. K. P. Gopi , V. A. Prasad, E. Lai, Low-complexity reconfigurable fast filter bank for multi-standard wireless receivers. IEEE Trans. Very Large Scale Integr. (VLSI) Syst., 22(5), 1202–1206 (2014).
8. S. S. Demirsoy, I. Kale, A. G. Dempster, Efficient implementation of digital filters using novel reconfigurable multiplier blocks, in Proc. 38th Asilomar Conf. Signals Syst. Comput. Conf. Rec., vol. 1. Pacific Grove, CA, USA, pp. 461–464 (2004).
9. R.I. Hartley, Sub-expression sharing in filters using canonic signed digit multipliers. IEEE Trans. Circuits Syst. II **43**(10), 677–688 (1996)
10. Y. M Hasan, L. J. Karem, M. Falkinburg, A. Helwig, M. Ronning, Canonic signed digit Chebyshev FIR filter design. IEEE Signal Process. Lett., 8(6), 167–169 (2001).
11. Hentschel, G. Fettweis, Software Radio Receivers, in CDMA Techniques for Third Generation Mobile Systems, Dordrecht, The Netherlands: Kluwer Academic,pp. 257–283,1999.
12. R. M. Hewlitt, E. S. Swartzlantler, Jr., Canonical Signed Digit Representation for FIR Digital Filters, in Proc. IEEE Worksh. Signal Process. Syst., pp. 416–426 (2000).
13. K. T. Hong, S. D. Yi, K. M. Chung, A high-Speed Programmable FIR Digital Filter Using Switching Arrays, in Proc. IEEE Asia Pacific Conf. Circuits Syst., pp. 492–495 (1996).
14. M. Isohookana, T. Kokkonen, P. Leppanen, A. Nykanen, J. Pyhtila, J. Reinila, J. Sillanpaa, and V. Tapio, Software radio-an alternative for the future in wireless personal and multimedia communications, in Proc. IEEE Int. Conf. Personal Wireless Commun., pp. 364–368 (1999).
15. R. Jia, H.-G. Yang, C.Y. Lin, R. Chen, X.-G. Wang, Z.-H. Guo, A computationally efficient reconfigurable FIR filter architecture based on coefficient occurrence probability. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., 35(8), 1297–1308 (2016).
16. M. Kumm, K. Moller, P. Zipf, Dynamically reconfigurable FIR filter architectures with fast reconfiguration, in Proc. 8th Int. Workshop ReCoSoC, pp. 1–8, (2013).
17. R. Mahesh, A. P. Vinod, New reconfigurable architectures for implementing FIR filters with low complexity. IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., 29(2), 275–288 (2010).
18. R. Mahesh, A. P. Vinod, A new common subexpression elimination algorithm for realizing low-complexity higher order digital filters. IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 27(2), 217–229 (2008).
19. P. K. Meher, S. Y. Park, High-Throughput Pipelined Realization of Adaptive FIR Filter Based on Distributed Arithmetic, in Proc. IEEE/IFIP 19th Int. Conf. VLSI-SOC, pp. 428–433, (2011).

20. P. K. Meher, Hardware-Efficient Systolization of DA-Based Calculation of Finite Digital Convolution. IEEE Trans. Circuits Syst. II, Exp. Briefs, 53(8), 707–711, (2006).
21. P.K. Meher, S. Chandrasekaran, A. Amira, FPGA realization of FIR filters by efficient and flexible systolization using distributed arithmetic. IEEE Trans. Signal Process. **56**(7), 3009–3017 (2008)
22. S. N. Merchant, B. V. Rao, Distributed Arithmetic Architecture for Image Coding, in Proc. IEEE Int. Conf. TENCON'89, pp. 74–77, (1989).
23. J. Mitola, *Object-Oriented Approaches to Wireless Systems Engineering", Software Radio Architecture* (Wiley, New York, 2000)
24. J. Mitola, The software radio architecture. Commun. Mag. IEEE **33**, 26–38 (1995)
25. B. K. Mohanty, P. K. Meher, A high-performance FIR filter architecture for fixed and reconfigurable applications. IEEE Trans. Very Large Scale Integr. (VLSI) Syst., 24(2), 444–452 (2016).
26. N. Moreano, E. Borin, C. De Souza, G. Araujo, Efficient Datapath Merging for Partially Reconfigurable Architectures. IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., 24(7), 969–980 (2005).
27. E. Ozalevli, W. Huang, P.E. Hasler, D.V. Anderson, A reconfigurable mixed-signal VLSI implementation of distributed arithmetic used for finite-impulse response filtering. IEEE Trans. Circuits Syst. I Regul. Pap. **55**(2), 510–521 (2008)
28. S. Y. Park, P. K. Meher, Efficient FPGA and ASIC Realizations of a DA-based reconfigurable FIR digital filter. IEEE Trans. Circuits Syst. II Exp. Briefs, 61(7), 511–515 (2014).
29. T. Peter, C. Hoe James, P. Markus, Time-multiplexed multiple-constant multiplication. IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., 26(9), 1551–1563, (2007).
30. T. Rissa, R. Uusikartano, J. Niittylahti, Adaptive FIR filter architectures for run-time reconfigurable FPGAs," in Proc. 2002 IEEE Int. Conf. Field Programm. Technol., 52–59, (2002).
31. M. Tamada, A. Nishihara, High-Speed FIR digital filter with CSD coefficients implemented on FPGA," in Proc. ASP DAC'01, pp. 7–8, (2001).
32. P. Tummeltshammer, J. C. Hoe, M. Puschel, Time-multiplexed multiple-constant multiplication. IEEE Trans. Comput. Aided Design. Integr. Circuits, 269, 1551–1563 (2007).
33. A.P. Vinod, E. Lai, Low power and high speed implementation of FIR filters for software defined radio receivers. IEEE Trans. Wireless Commun. **5**(7), 1669–1675 (2006)
34. S.A. White, Applications of distributed arithmetic to digital signal processing: a tutorial review. IEEE ASSP Mag. **6**(3), 4–19 (1989)
35. T. Zhangwen, Z. Zhanpeng, Z. Jie, M. Hao, A high-speed, programmable, CSD coefficient FIR filter. IEEE Tran. Consumer Electr. **48**(4), 834–837 (2002)