Design of DA-Based FIR Filter Architectures Using LUT Reduction Techniques

A. Uma, P. Kalpana and T. Naveen Kumar

Abstract The multiplier-less techniques such as distributed arithmetic (DA) have gained large popularity for its high-speed processing. Architectures based on DA results in cost-efficient and area-efficient structures. This paper presents design and realization of various DA-based FIR filter architectures based on LUT reduction techniques of length $N = 4$ and also implemented using both shift accumulators and carry save shift accumulators. The larger LUT is subdivided into a number of LUTs to reduce the size of the LUT for higher order filter. FIR filter architectures designed include filter with LUT size of $2^N - 1$ words, filter with LUT size of 2^{N-1} words, filter with LUT breakup contains two $2^{N/2} - 1$ word LUTs, and also LUT-less filter but only has combinational blocks. These filter architectures have been synthesized for the target FPGA device and results are compared based on RTL area, device utilization, maximum operating frequency, and power consumption.

Keywords Distributed arithmetic (DA) · Carry save shift accumulator (CSSA) Shift accumulation (SA)

1 Introduction

Multiply and accumulate operation is very common in all digital signal processing algorithms such as finite impulse response (FIR) filter. As multipliers consume more area and power in multiply and accumulate (MAC) operation of FIR Filter, several multipliers-less schemes have emerged. Distributed arithmetic (DA) method is one of the multiplier-less techniques which uses memories (RAMs, ROMs) or LUTs to store precomputed values of coefficient operations [1]. DA is an efficient technique for performing multiply-and-add in which the multiplication is reorganized such that multiplication and addition are performed on data and single bits of the coefficients, at the same time [2]. The inner products containing many terms can

A. Uma (✉) · P. Kalpana · T. Naveen Kumar
Department of ECE, PSG College of Technology, Coimbatore, India
e-mail: umavithy22@yahoo.com

V. Nath (ed.), *Proceedings of the International Conference on Microelectronics, Computing & Communication Systems*, Lecture Notes in Electrical Engineering 453,
https://doi.org/10.1007/978-981-10-5565-2_20

be partitioned into a number of smaller inner products which can be computed and summed by using either DA or an adder. Hence, DA is widely used for the implementation of digital filters [3].

The advantages of DA are best exploited in data-path circuit designing. DA efficiently implements the MAC using basic building blocks (Look-up Tables) in FPGAs. These DA structures can be used even in adaptive filters. An adaptive filter which is commonly used in devices such as mobile phones, camcorders. This adaptive structure is a system with a linear filter that has a transfer function controlled by variable parameters and a means to adjust those parameters according to an optimization algorithm [4]. Adaptive filters are required for some applications because some parameters of the desired processing operation are not known in advance or are changing [5].

In conventional DA-based FIR filter, as filter size increases, the memory requirements of the implementation also grow exponentially. This in turn increases the look-up table (LUT) size. For example, a 128-tap DA-FIR filter will require a prohibitively large 2^{128} entries in the DA-based LUT [6]. The larger LUT is subdivided into a number of LUTs to reduce the size of the LUT for higher order filter. The FIR filter architectures are designed using four LUT reduction methods. They include filter with LUT size of $2^N - 1$ words, filter with LUT size of 2^{N-1} words, filter with LUT breakup contains two $2^{N/2} - 1$ word LUTs, and also LUT-less filter [4] but only has combinational blocks. This paper presents various DA-based FIR filter architectures with LUT reduction techniques and its implementation using both shift accumulator and carry save shift accumulator (CSSA). The results are compared with area, operating frequency, and power.

The paper is organised as follows: In Sect. 1, a brief introduction of DA-based filter structure is given. In Sect. 2, DA-based FIR filter structures and their operations are discussed. In Sect. 3, modified DA architectures and their hardware complexities derived through LUT reduction techniques are described. The synthesis and comparison results of DA-filter structures are described in Sect. 4. Summary and conclusions are given in Sect. 5.

2 DA-Based FIR Filter Structure

Many digital signal processing (DSP) applications require linear filters that can adapt to changes in the signals they process. Adaptive filters find extensive use in several DSP applications including acoustic echo cancellation, signal de-noising, sonar signal processing, clutter rejection in radars, and channel equalization for communications and networking systems [3]. DA is one method often preferred since it eliminates the need for hardware multipliers and is capable of implementing large filters with very high throughput [6].

Distributed arithmetic (DA) is an efficient technique for performing multiply-and-add where both the multiplication and addition is done at same time on coefficients which has single bit [6]. Distributed arithmetic is a bit level rearrangement of a multiply accumulate in order to eliminate the multiplications and also an efficient method for computing inner products between a fixed and a variable data vector [6]. The basic principle of the DA is the computed values are stored rather than carry out the computation. It reduces the size of parallel hardware implementation of multiply accumulate which is more preferred for FPGA. This feature can be extended to sum functions used in complex multipliers, Fourier Transforms. The basic DA technique which is bit serial has ROM look-ups an efficient technique to implement on Field Programmable Gate Arrays (FPGAs). The often encountered forms of computation in DSP are sum of product, dot product [7], and inner product which can be implemented using distributed arithmetic.

The block diagram of DA-based FIR filter is shown in Fig. 1. The DA based FIR filter has to perform an inner-product computation which produces the critical path [8]. The critical path is the longest delay free path which affects the speed of the architecture.

Let the inner product of FIR filter be given by

$$y = \sum_{k=0}^{N-1} w_k x_k \tag{1}$$

where w_k and x_k for $0 \leq k \leq N-1$ form the N-point vectors corresponding the current weights and most recent $(N-1)$ input, respectively [6]. Assume L be the bit

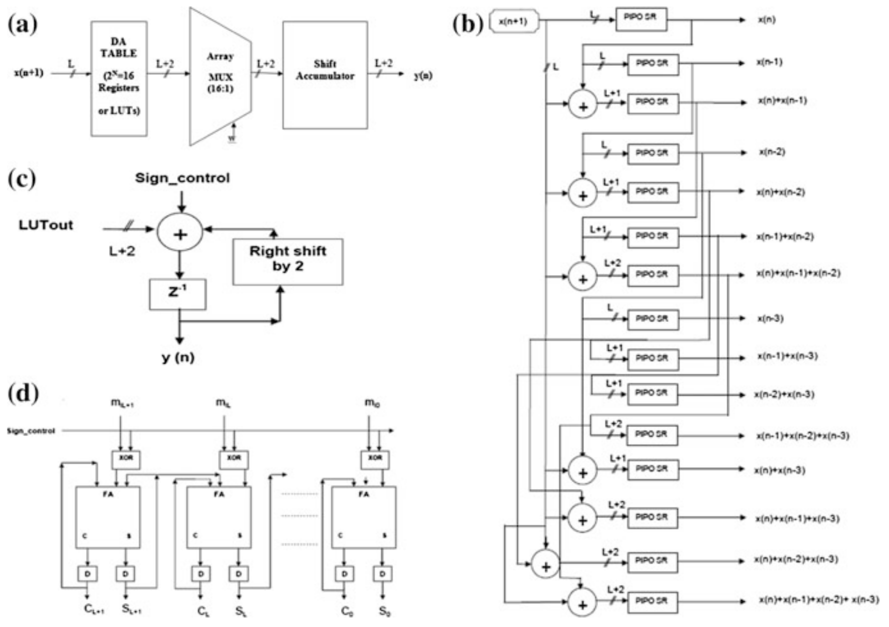

Fig. 1 **a** Block diagram of DA-based FIR filter, **b** generation of possible sum of input combinations (DA table), **c** shift accumulator, **d** carry save shift accumulator

width of the weight and w_k be a N-bits of two's complement number representation where $|w_k| < 1$ and w_k: $\{w_{k0}, w_{k1}, w_{k2}, \ldots, w_{k(N-1)}\}$ where w_{k0} is the sign bit.

$$w_k = -w_{k0} + \sum_{l=1}^{L-1} w_{kn} 2^{-l} \tag{2}$$

Substituting (2) in (1) and interchanging the order of summations over the indices k and l will give the reformulation of inner product to form distributed inputs as

$$y = -y_0 + \sum_{l=1}^{L-1} 2^{-l} \cdot y_l \quad \text{where} \quad y_l = \sum_{k=0}^{N-1} x_k \cdot w_{kl} \tag{3}$$

Any weight bit of the N-point w_k: $\{w_{k0}, w_{k1}, w_{k2}, \ldots, w_{k(N-1)}\}$ will either be one or zero, so the partial sum y_1 will have 2^N possible values. DA-based computation requires storing all possible values of partial sum y_1 in LUT of size 2^N words. The weight bit sequence $\{w_{k1}\}$ can be used as address vector to read out the corresponding partial sum from LUT, for computing the inner product. The hardware realization of inner product needs PIPO SR, shift accumulators, Vectored mux, etc. The calculation of inner product requires L cycles of shift accumulation, then read by LUT-read operations based on the number of bit slices $\{w_{k1}\}$ where $0 \leq 1 \leq L - 1$.

The block diagram representation of DA-FIR filter architecture is shown in Fig. 1a. It contains DA table, Array Multiplexer followed by shift accumulator. The various DA-based FIR filters are implemented and compared with both shift accumulator and carry save shift accumulator (CSSA). The matched LUT content should be selected by the weight vector in a bit serial vector. The detailed blocks of possible sum of input generation is shown in Fig. 1b.

DA Table Generation: The DA table stores $(2^N - 1)$ words using Parallel In Parallel Out Shift Registers (PIPO SR), whereas RAM-based LUT stores 2^N words [6]. For example, considering $N = 4$, the number of registers required is only 15, to store the precomputed possible sum of input sequences and seven adders are required to produce the seven new values of input sums.

Parallel In Parallel Out Shift Registers (PIPO—SR): A L-bit PIPO SR constructed by L number of D flip-flops. Once the register is clocked, all the data at the D inputs appear at the corresponding Q outputs simultaneously. N is the number of taps in FIR filter. In the process of constructing DA table, $2^N - 1 = 15$ PIPO registers with data bus sizes in the range from L to $L + 2$ are used as shown in Fig. 1b.

Bus Mux: Vectored Array Multiplexer is used to transfer the selected input bit combination to the output. The weight vector which are fed in bit serial fashion acts as select lines for the 16:1 vectored mux, for $N = 4$ and $L = 8$. The selected combination of inputs has data width size of $L + 2 = 10$ bits. The output of vectored mux is passed to input of adder-based shift accumulator or carry save shift accumulator [6].

Shift Accumulator: The shift register is shifted right at every bit clock cycle to feed the weight vector inputs serially to mux [6]. The corresponding LUT entries are also shifted and accumulated using L consecutive times to generate the output using adder-based conventional shift accumulator shown in Fig. 1c. The sign bit control is used to change the addition to subtraction for the sign bits.

Carry Save Shift Accumulator: The shift accumulation has large critical path and the longest delay free path that decides the ultimate speed of the architecture. The carry save accumulation serves as an alternate block for conventional shift accumulation and is shown in Fig. 1d. The bit-sliced vector *w* is fed as inputs from the least significant bit (LSB) to the most significant bit (MSB). The ex-or gate is used to pass all the bits of table output, and the sign bit is one.

The byte clock and the bit clock are the two clock signals used in the design. The byte clock synchronizes with sampling period of input sequence. The carry save accumulation block uses bit clock. The byte clock is used in the remaining circuits.

3 Modified DA-Based FIR Filter Architectures

It can be noted that as the filter size increases, the memory requirements of the implementation in Fig. 1 grow exponentially. This in turn increases the look-up table (LUT) size. This problem can be rectified by altering LUT sizes and use of combinational blocks. The DA-FIR filter architectures are modified and implemented based on shift accumulator and also carry save shift accumulator (CSSA) [9]. The LUT reduction technique is incorporated by breaking up the filter into smaller base DA filtering units that require tractable memory sizes and then summing up the outputs of these units. By incorporating additional Multiplexers and use of only combinational blocks such as adders and multiplexers in realizing inner-product block of DA-FIR filter further reduces the LUT size [10].

Using the LUT reduction techniques, the different DA-based FIR filters are

(1) DA-based filter with LUT size of 2^{N-1} words.
(2) DA-FIR filter with partitioned LUTs where breakup contains two $2^{N/2} - 1$ word LUTs.
(3) DA-FIR filter with No LUTs, but only having combinational blocks.

All the above architectures can reduce the memory requirement and also modified using both CSSA and shift accumulator.

3.1 Modified DA-Based Filter with LUT Size of 2^{N-1} Words

The conventional DA-based FIR filter is implemented with LUT size of 2^{N-1} words, and it uses shift accumulator. It is shown in Fig. 2. From the DA table, it is observed that lower half of LUT (locations where $w_3 = 1$) is the same with the sum of the upper half of LUT (locations where $w_3 = 0$) and $x(n-3)$, i.e., Lower Half of LUT outputs = Upper Half of LUT outputs + $x(n-3)$.

In the DA-based filter, LUT size is reduced to half of the memory LUT size in original DA table. This architecture includes one more 2:1 multiplexer and one adder additionally, which occupy very less area. This mux selects $x(n-3)$ input when the MSB bit of weight vector is high ($w_{31} = 1$), else 0. This is then added to output of main LUT to be sent to conventional shift accumulator to produce the output for inner-product computation of DA-FIR filter [7]. The architecture is modified by replacing the shift accumulator with carry save shift accumulator.

For instance, $N = 4$, this architecture requires 8 memory LUTs, one 8:1 mux, one 2:1 mux and one adder. For higher order filters, example of $N = 16$, this architecture requires 256 memory LUTs, one 256:1 mux, one 2:1 mux, and one adder, whereas in conventional DA-FIR, it needs $2^{16} = 65536$ memory LUTs and 65536:1 mux.

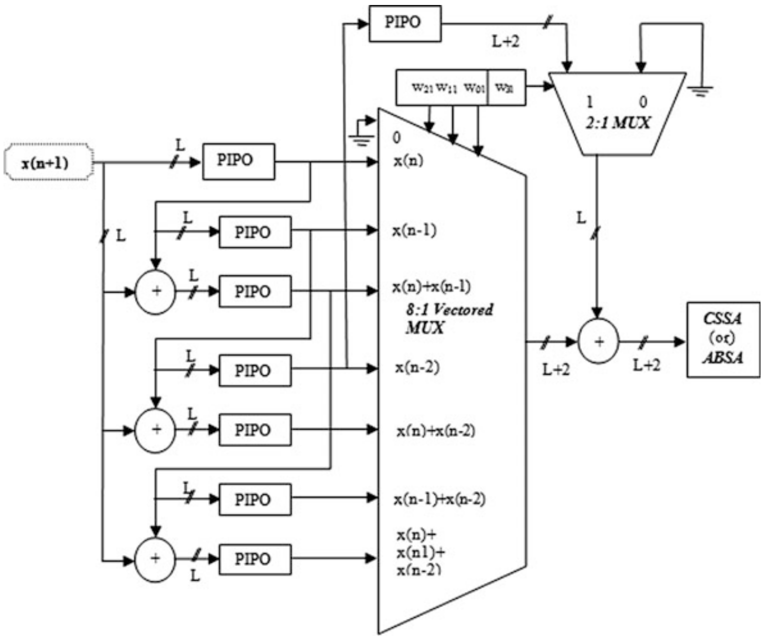

Fig. 2 DA-based filter with LUT size reduced by ½ for $N = 4$

3.2 Modified DA-Based Filter with Partitioned LUTs

The conventional DA-based filter is implemented with partitioned LUTs where breakup contains two $2^{N/2} - 1$ word LUTs. In this architecture, LUT is divided into LUT 1, LUT 2 so on. The N-tap filter is divided into p smaller filters each having q-tap DA base units. Here, it is assumed that N is not prime and also $N = p * q$. The total number of memory elements requirement to implement this structure is $p * 2^q$ [3]. Thus, there is a marginal decrease in throughput of this architecture.

For instance, if $N = 128$, $p = 32$, and $q = 4$ can be chosen, which would only require 512 memory elements. The number of clock cycles required for this implementation would be 21 clock cycles as compared with the single LUT implementation that would require 16 clock cycles [4]. Similarly for $N = 4$, p and q can be chosen as $p = 2$ and $q = 2$. This requires $2 * 2^2 = 8$ memory elements only for the implementation along with two 4:1 Multiplexers and one adder, as shown in Fig. 3. The partitioned LUT architecture is modified by replacing shift accumulator with CSSA and results are compared.

3.3 Modified DA-Based Filter with No LUTs

The conventional DA-based filter with no LUTs or LUT less is implemented using shift accumulator and then modified with carry save shift accumulator. By the same

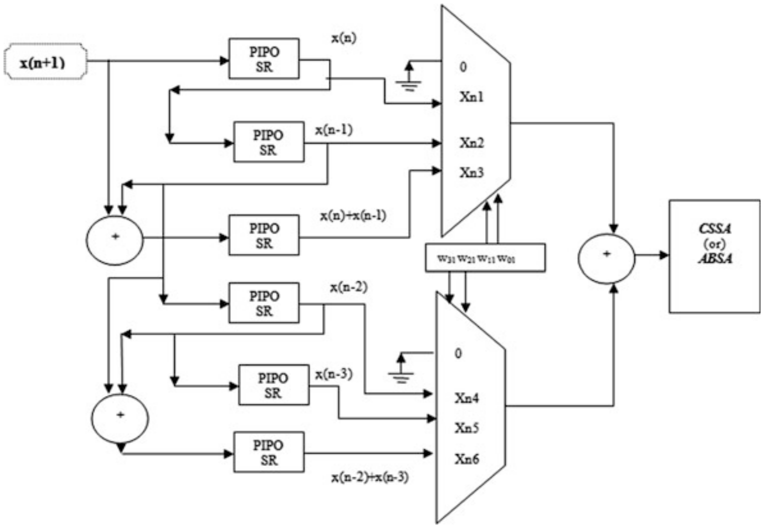

Fig. 3 DA-based filter with partitioned LUT for $N = 4$

LUT reduction procedure, the final LUT-less DA-FIR filter architectures can be realized [1]. For $N = 4$, it requires four 2:1 Multiplexers and two stages of three adders as shown in Fig. 4.

Similarly, for $N = 8$, it requires 8, 2:1 Multiplexers and 3 stages of 7 adders, where first, second, and third stage contains 4, 2, and one number of adders, respectively. So, in general, for any N-tap DA-FIR filter, it requires N, 2:1 Multiplexers, and ($N/2 + N/4 + \cdots + 1$) number of adders. For the use of combination logic circuit, the filter performance will be affected. But when the taps of the filter is a prime, 4-input LUT units with additional multiplexers and full adders can be used to get the trade-off between filter performance and small resource usage [1].

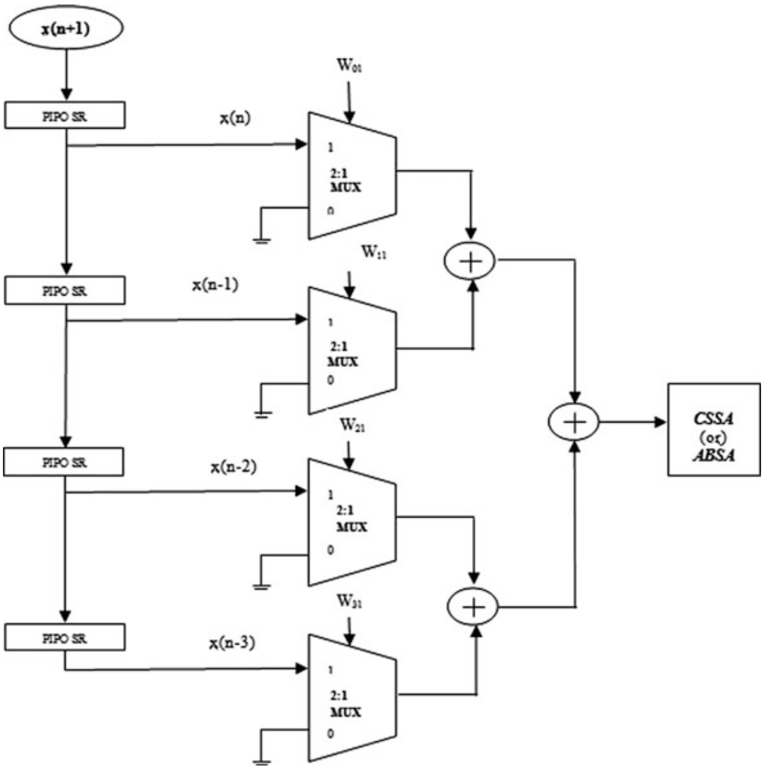

Fig. 4 DA-based filter with no LUT

4 Synthesis and Comparison Results of DA-Filter Structures

The conventional DA-based FIR filter architectures (for $N = 4$ and $L = 8$) with shift accumulator and modified architecture where shift accumulator replaced with carry save shift accumulator are implemented. The design is coded in Verilog HDL and simulated and synthesized for the target device of Xilinx's FPGA SPARTAN-3E (XC3S500E-FG320). The comparison results of inner-product block replaced by conventional DA and other architectures based on shift accumulator and CSSA are shown in Tables 1 and 2. The comparison results are tabulated based on maximum operating frequency area and power consumption for the given clock frequency = 10 MHz, bit clock frequency = 80 MHz. The default activity factor = 12.5% and capacitive load of 35 pF has been taken.

The comparison results show that LUT-less DA-filter has less power consumption and area when compared to other structures and also it can be used for high-speed applications.

Table 1 Comparison of maximum operating frequency, power, and area for DA-based filters using CSSA

Features and architectures	Power (mW)	Maximum operating frequency (MHz)	No. of slice (FFs)	No. of slices	No. of 4 input LUT
Conventional DA-FIR filter	91	252.972	146	121	147
Filter with LUT of 2^{N-1} words	87	256.805	79	67	80
Filter with partitioned LUTs	88	254.388	60	55	69
Filter with no LUTs	83	446.429	51	46	54

Table 2 Comparison of maximum operating frequency, power, and area for DA-based filters using shift accumulator

Features and architectures	Power (mW)	Maximum operating frequency (MHz)	No. of slice (FFs)	No. of slices	No. of 4 input LUT
Conventional DA-FIR filter	95	252.972	155	130	167
Filter with LUT of 2^{N-1} words	87	256.805	88	76	99
Filter with partitioned LUTs	88	254.388	69	64	88
Filter with no LUTs	86	255.820	42	37	35

5 Conclusion

The conditional signed carry save accumulation which replaces the conventional adder-based shift accumulation for DA-based inner-product computation, in order to reduce the sampling period and to decrease the total area occupied, has been realized. By LUT reduction techniques, modified DA-FIR filter architectures such as filter with partitioned LUTs, LUT size of 2^{N-1} words, and also LUT-less filters have been realized. The comparison results show that LUT-less DA has low power and even it can be used for high-speed applications. For higher order filters, the LUT reduction based DA-FIR filter architectures can be adopted to reduce the resource utilization. Hence, it is shown that modified DA architectures are hardware efficient for FPGA implementation.

References

1. J. Xie, J. He, G. Tan, FPGA realization of FIR filters for high-speed and medium-speed by using modified distributed arithmetic architectures. Micro Electron. J. 365–370 (2010)
2. Y.-H. Chen, J.-N. Chen, T.-Y. Chang, C-W. Lu, High-throughput multistandard transform core supporting MPEG/H.264/VC-1 using common sharing distributed arithmetic. IEEE Trans. Very Large Scale Integration (VLSI) Syst. **22**(3), Mar 2014
3. J. Xie, P.K. Meher, J. He, Hardware-efficient realization of prime-length DCT based on distributed arithmetic. IEEE Trans. Comput. **62**(6), June 2013
4. D.J. Allred, H. Yoo, V. Krishnan, W. Huang, D.V. Anderson, LMS adaptive filters using distributed arithmetic for high throughput. IEEE Trans. Circ. Syst. I Reg. Pap. **52**(7), 1327–1337 (2005)
5. M.S. Prakash, R.A. Shaik, Low-area and high-throughput architecture for an adaptive filter using distributed arithmetic. IEEE Trans. Circ. Syst. Ii Express Briefs **60**(11), Nov 2013
6. S.Y. Park, P.K. Meher, Low-power, high-throughput, and low-area adaptive FIR filter based on distributed arithmetic. IEEE Trans. Circ. Syst. II Express Briefs **60**(6), June 2013
7. R. Guo, L.S. DeBrunner, Two high-performance adaptive filter implementation schemes using distributed arithmetic. IEEE Trans. Circ. Syst. Ii Express Briefs **58**(9), Sept 2011
8. E. Özalevli, 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. Circ. Syst. I Regul. Pap. **55**(2), Mar 2008
9. B.K. Mohanty, P.K. Meher, A high-performance energy-efficient architecture for FIR adaptive filter based on new distributed arithmetic formulation of block LMS algorithm. IEEE Trans. Sig. Process. **61**(4), 15 Feb 2013
10. S.Y. Park, P.K. Meher, Efficient FPGA and ASIC realizations of DA-based reconfigurable FIR digital filter. IEEE Trans. Circ. Syst. Ii Express Briefs doi:10.1109/TCSII.2014.2324418