Middle-East Journal of Scientific Research 24 (S2): 46-52, 2016
ISSN 1990-9233

DOI: 10.5829/idosi.mejsr.2016.24.S2.115

Low Power Low Complexity Implementation of Direct Form Lmsadaptive Filter

M. Santhi and C.T. Nallammai

Department of ECE, Saranathan College of Engineering, Trichy, India

Abstract: Adaptive filtering is a two stage process whereby filter first estimates the statistical parameters of the relevant signal and then plugs the estimated result in a recursive manner for computing the filter parameters. Second stage can be implemented using least Mean Square (LMS) adaptive algorithm. Conventional LMS adaptive filter can be realized in either transpose or direct form Structure. Transpose formimplementation is by default gives the pipeline structure which is of Delayed LMS kind and it has slower convergence and register complexity. Direct form implementation with pipelining result in more adaptation delay, consumes more area and slower convergence. Direct form without pipelining is efficiently implemented with zero and two adaptation delays. Modified column bypassing multiplier is used in order to reduce the power consumption, Low complexity is achieved in zero adaptation delay by time multiplexed architecture where in during positive clock period, error computation block works and in negative clock period, weight update block works.It shares the multiplier between these two blocks. Thus it requires N multipliers instead of 2N multipliers where N is the filter order. Two Adaptation delay implementation requires 2N multipliers with 2N additions and has sixty two percentage of increase in maximum usable frequency when compare to zero adaptation at the cost of eleven times increase in area utilization.

Key words: LMS · Direct Form · Zero Adaptation Delay · Multiplier

INTRODUCTION

In typical Digital Signal Processing (DSP) fields like speech processing,communications, radar, sonar, seismology requires optimized system coefficients that need to be adjusted over time depending on the input signal. When the sampling frequency changes faster compared with the parameter, then better estimation for optimal system coefficients needs to be computed and adjust the system appropriately. In general, any filter structure, FIR or IIR, with the many architectural variations may be used as an adaptive digital filter (ADF). Comparing the different structural options, it is noted that for FIR filters as stated in [1] the direct form seems to be advantageous because the coefficient update can be done at the same time for all coefficients. For IIR filters the lattice structure seems to be a good choice as stated in [2] because lattice filters possess low fixed-point arithmetic round-off error sensitivity and a simplified stability control of the coefficients. From the published literature, however, it appears that FIR filters have been used more successfully than IIR filters.Basically, there are two distinct approaches for deriving the recursive algorithm for the operation of linear adaptive filters.

- Stochastic Gradient Approach
- Least Square Estimation

Stochastic gradient algorithms like LMS algorithm are model independent; it exhibits good tracking behaviour. In contrast, RLS algorithms of Least Square estimation are model-dependent, this, in turn, means that their tracking behaviour may be inferior to that of stochastic gradient family algorithm, It is needed to minimize the difference between the physical process and mathematical model responsible for generating the input data. LMS algorithm can be chosen due to its simplicity, robustness and satisfactory convergence performance as stated in [3] and [4]. Due to inner product computation in direct-form LMS adaptive filter, critical path is longer. When there is longer critical path, then it exceeds the sampling period and hence It is required to be reduced by pipelined

Corresponding Author: Dr. M. Santhi, Head of Department, Department of ECE, Saranathan College of Engineering, Trichy, India.

implementation. Conventional LMS algorithm is recursive in behaviour and thus does not support pipelined implementation. It is modified to a form called the delayed LMS (DLMS) algorithm as stated in [5] and [6], which allows pipelined implementation of the filter. According to this architecture, To update the next sample weights, the presently calculated error signal is utilized.

In A modular pipelined filter architecture is implemented based on a time shifted version of Delayed LMS algorithm. This pipelined architecture preserves the most desirable features of both lattice and traversal form adaptive filters. Time shifted DLMS algorithm obtained from normal DLMS algorithm can be implemented using pipelined fashion. Weight update and output can proceed in parallel with an array of identical processing elements. Speedup is more for this proposed architecture.

In [7], To reduce the delay D (Adaptation delay), low critical period (i.e., high throughput) and satisfying the requirements of systolic array realization, an efficient tree-systolic processing element and an optimized tree-level rule is designed in order to maintain minimum delay and high regularity under the constraint of maximum driving of feedback error signal.

In [13], a modified DLMS algorithm is proposed which provides a faster convergence rate with less computational complexity compared to theconversion based DLMS algorithm. The proposed algorithm uses the error signal from each stage of the adaptive FIR filter independently to update the value of the corresponding coefficient. Actually DLMS algorithm was proposed to overcome the throughput problem of the LMS algorithm. It is noted that the error e(n) in LMS is generated after carrying out one multiplication and N sequential additions as N is the order of the filter. This N sequential addition limits the throughput rate of the LMS algorithm. To achieve the throughput, D the delay is introduced in the error feedback path. This inserted delay D results in slower convergence. Hence to improve the convergence rate, conversion based DLMS was proposed in [2]. But this results in increase in hardware complexity.

In [10], a fine grained pipeline DLMS algorithm i.e., pipelining within the processor is proposed results in increased output latency. The pipelined LMS filter is implemented by inserting pipelining registers as stated in [3] along the filter feed forward path FIR filter and error feedback path and this can reduce the critical path. To reduce the latency, Binary-tree adder structure is used in FIR filter where in the binary adder structure is used in place of direct form FIR which reduces mD from N-1 to log2N. This structure is widely used in non-transposed form FIR filter, where mD is the number of pipeline delays in the feedback loop. Using DLMS algorithm, registers can be inserted into error feedback path before adaptation loop and use these delays as a means of pipelining the LMS filter. This process involves the determination of number of delays needed for a fully pipelined version of the circuit, once this has been determined, retiming technique is then used.

Proposed Work: In conventional LMS algorithm, all weights are updated concurrently in everycycle to compute the output. It has its implementation in transpose form and in directform.

In direct form implementations, critical path is longer due to an inner productcomputation in obtaining the filter output. This is mainly based on the assumption thatarithmetic operation starts only after the complete input operand words are available.Thus addition in multiply-add operation is proceed only after the completion ofmultiplication and thus the critical path of the multiply-add operation is

$$T_{MA}=T_{MULT}+T_{ADD} \quad (1)$$

where T_{MULT} and T_{ADD} are the time required for multiplication and additionrespectively. Hence the critical path of direct form adaptive filter without pipelining isestimated as

$$T = 2T_{MULT} + (N+1)\, T_{ADD} \quad (2)$$

But, the convention LMS algorithm does not support pipelined implementation. Therefore it is modified to a form called Delayed LMS as stated in [8] which allowspipeline implementation. Transpose Form is inherently of delay LMS kind where the adaptation variesacross the sequence of filter weights. Several efficient DLMS algorithms reported inthe literature are FPDLMS, Conversion based DLMS, Time Shifted DLMS, ModifiedDLMS, Systolic with optimized tree rule. In FPDLMS, Direct form implementation efficiently utilizes the resources byusing the binary adder tree structure and increases the throughput. Conversion basedDLMS algorithms improves the convergence rate but at the expense of an increased computational complexity and a lower throughput rate than the original DLMSalgorithm. Modified DLMS algorithm in comparison to conversion

based DLMSprovides higher throughput rate for a similar convergence rate with less computational complexity. Pipelined Systolic type DLMS algorithm called as time shifted version ofDLMS has been implemented as parallel version of DLMS which uses N identicalprocessing units which achieves a computational speedup proportional to the order ofthe filter and it is easily expandable. Another Efficient DLMS implementation consistsof pipelined inner product computation unit for calculation of feedback error and apipelined weight update unit which consists of N parallel multiply accumulators forfilter order N and has less adaptation delay.

All these efficient DLMS implementation provides less adaptation delay intranspose form as well as in direct form but with two adverse effects. First, the registercomplexity and hence power dissipation increases, Second, the large pipeline depth. Inall these, critical path and arithmetic operation involved in inner product calculation isignored. Also various multipliers like Booth, Wallace and Array are used to generatethe inner product during error computation and weight update. These multiplierarchitectures are considered to have the switching activity even if the bit coefficient iszero and this ultimately results in unnecessary power dissipation. Direct-Form LMS adaptive filter is implemented with zero adaptation delay andhence the value of m=0, this result in no pipeline registers after the feedback errorcalculation in the error computation block.

In this direct form implementation, an efficient inner product calculation isproposed under the assumption that arithmetic operations start immediately as soon asthe LSBs of the operands are available. This result in reduced critical path whencompare to transpose form delayed LMS since it requires an additional signal-pathdelay line for weight updating and the registers on the adder-line to compute the filteroutput are at least twice the size of the delay line of the direct-form LMS adaptivefilter.

In order to reduce the weight update block complexity, step-size is considered asa power of 2 fraction, then the multiplication with step-size can be implemented byrewiring without any hardware or time delay. When error computation and weight update are performed in two pipelinedstages and then the critical path is the maximum of error computation or weight updateand not the sum of both as like in other architectures.Direct-Form LMS with two adaptation delay is implemented to have moremaximum usable frequency (MUF) when compare to zero adaptation which hasminimum of MUF. Minimum MUF supports high data rate applications and for lowdata rate applications two adaptation delay can be used.To consider the power dissipation in multiplier architecture, the switchingactivity of the component used in the design can be ignored if the bit coefficient is zeroi.e., corresponding row or column of adders need not be activated as stated in [8], thusif multiplicand contains more zeros, high power reduction can be achieved.

Zero Adaptation Delay: There are two computation blocks in Direct-Form ZeroAdaptation LMS filter shown in Figure 1, 2 which are Error computation andweight update block. In both the blocks, area intensive components are multipliers,weight registers and tapped-delay line. Adder tree and subtraction of error computation and adders of weight update block are different and they consume small area in thecircuit.

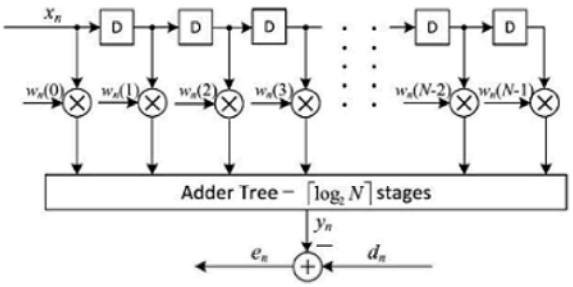

Fig. 1: Error Computation Block

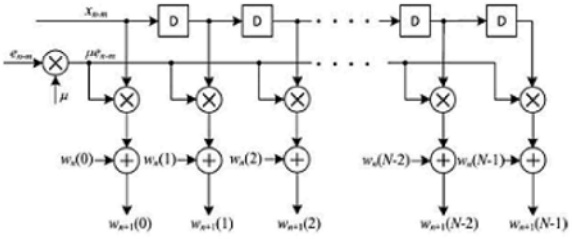

Fig. 2: Weight Update Block

Error computation block and weight update block computation are performed inthe same cycle for zero adaptation. According to direct form structure (Non-pipelined),error computation and weight update cannot occur concurrently and hencemultiplication of both these phases could be multiplexed by the same set of multipliersand same registers could be used by means of performing error computation in firsthalf cycle and weight update in the second half cycle.

The proposed zero-adaptation delay structure for a direct form N-tap LMSadaptive filter consists of N multipliers. From a common tapped delay line, The input

samples are fed to the multipliers. The N weight values (stored in N registers) and theestimated error values (after right-shifting by a fixed number of locations to realize multiplication by the step size ì) are fed to the multipliers as the other input through a2:1 multiplexer. The proposed structure requires N adders for modification of Nweights and an adder tree to add the output of N multipliers for computation of filteroutput. N 2:1 de-multiplexers are used to move the product values either towards theadder tree or weight-update circuit. Clock signal is used as the control signal for all multiplexers and de-multiplexers.

During rising edge of the clock pulse, the registers in the delay line are clocked and it remains unchanged for a complete clock period because to take one new sample, the structure requires one complete clock cycle. During the positive half of each clock period, filter output is computed from multipliers through the multiplexors which drives the weight values stored in different registers. Via Demultiplexers, the product words are then fed to the adder tree.

Using adder tree, filter output is computed and the error value is computed by a subtractor. Then the computed error value is right-shifted to obtain ìe_n and is broadcasted to all N multipliers in the weight-update circuits. To break the recursive loop of LMS adaptive filter, a delay is required and it could be inserted either after the adder tree (results in increased critical path by TADD), after the en computation or afterìe_n computation (after ìe_n result in reduced register width).

The first half-cycle of each clock period ends with the computation of ìe_n and during the second half cycle, the ìe_n value is fed to the multipliers though the multiplexors to calculate ì$e_n x_n$ and de-multiplexed out to be added to the stored weight values to produce the new weights. During the second half of a clock period computation is completed once a new set of weight values is computed. For the computation of filter output and for subsequent error estimation, the updated weight values from the previous clock pulse are used in the present positive cycle of the next clock period. When the next clock cycle begins, the new weight values are updated in the weight registers. Therefore, the weight registers are also clocked at the positive edge of each clock pulse.

Two Adaptation Delay: The proposed two adaptation delay Direct-Form LMS adaptive filter also contains two blocks as mentioned for the zero adaptation. It consists of three pipelined stages, where the first stage refers to error computation block's first level adder tree and the rest of the error computation block comprises the next pipeline stage. Weight update block comprises the third pipeline stage. In this architecture, weight update uses delayed input and the delay value is 2.

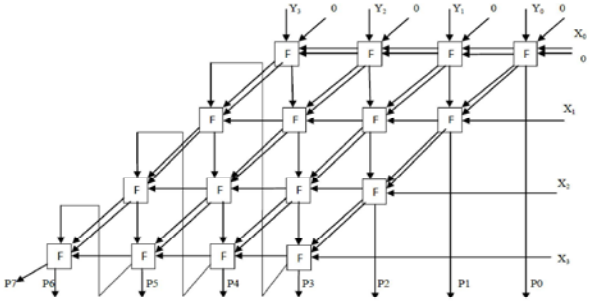

Fig. 3: Column-By Pass Structure

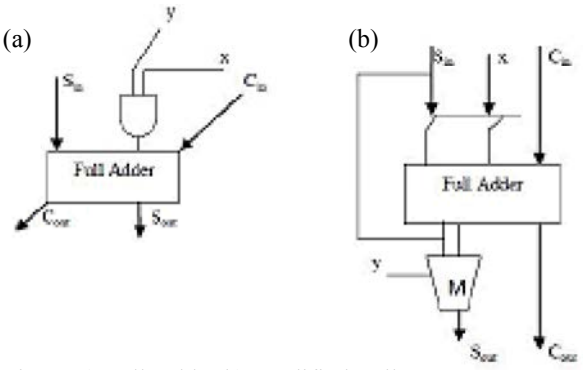

Fig. 4: a) Full Adder b) Modified Full

Here filter coefficients are multiplied with input samples and are then added by means of first level adder for two inner product addition and the results are delayed and then given as input to the adder tree.

Column By-Passing Multiplier: In order to reduce the power consumption and removal of extra correction circuit during inner product calculation, column-bypassing multiplier is used. Column–bypassing means turning off some columns in the multiplier array whenever certain multiplicand bits are zero and its operation is shown in the Figure 3. In this technique, Full Adder used in general multiplication shown in Figure 4 (a) and with the cell in Figure 4 (b) called the Full Adder Bypassing (FAB) cell. The transmission gates in the FAB cell lock the inputs of the full adder to prevent any transitions. In FAB cell, the transmission gates lock the inputs of the full adder to prevent any transitions when multiplicand bit is 0 and a multiplexer propagates the sum input to thesum output. When multiplicand bit is 1, the sum output of the full adder is passed.

Time Multiplexed Zero Adaptation Delay: The implementation of time multiplexed zero adaptation delay direct form LMSadaptive filter is described using the following Figure 5

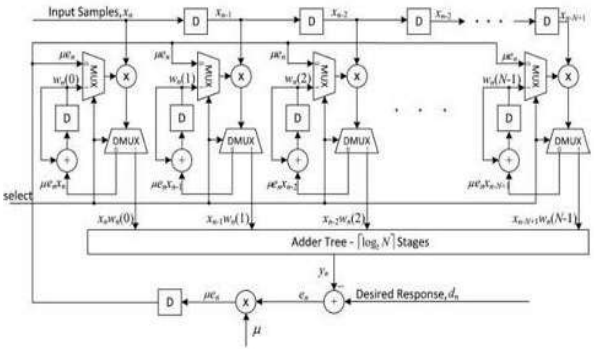

Fig. 5: Zero Adaptation Delay Time Multiplexed Structure

Control signal for the Multiplexer and de-multiplexer is the clock signal. Themultiplexer output is either given to error computation or weight update. Multiplierused is the same for both error and weight update. De-multiplexer output is either givento Adder tree used for error computation or to current weight update.

Experimental Results: Time multiplexed Zero Adaptation delay and two adaptation delay of Linear FIR LMS Adaptive filter are simulated in Xilinx Spartan 2 family with the target device as xc2s50-6tq144. Before designing the filter, multiplier with array and column by pass architecture are implemented and simulated with the same FPGA device.

Simulation Setup:

Input Word Length:	4 bits
Filter Order:	16
System Coefficients:	1,2,3,4,5,6,7,8,9,10, 11,12,13,14,15,0
System Coefficient Length:	4 bits
Inner Product Word Length:	8 bits
Step Size:	1/16
Input Signal x:	Random noise
Multiplier:	4 bits
Multiplicand:	4 bits
Simulation Tool:	Xilinx 9.2i
FPGA Family:	Spartan 2
FPGA Device:	xc2s50-6tq144

Both Zero and Two adaptation delay are implemented withthe same filter order.

Array and Column Bypass Multiplier:

Fig. 6: Array Multiplier Simulation

```
Release 9.2i - XPower SoftwareVersion:J.36
Copyright (c) 1995-2007 Xilinx, Inc.  All rights reserved.
Design:       norMult.ncd
Preferences:  norMult.pcf
Part:         2s50tq144-6
Data version: PRELIMINARY,v1.0,07-31-02

Power summary:                          I(mA)     P(mW)
---------------------------------------------------------------
Total estimated power consumption:                   11
                                          ---
                          Vccint 2.50V:       2        5
                          Vcco33 3.30V:       2        7
                                          ---
                               Inputs:        0        0
                                Logic:        0        0
                              Outputs:
                                Vcco33        0        0
                              Signals:        0        0
                                          ---
                  Quiescent Vccint  2.50V:       2        5
                  Quiescent Vcco33  3.30V:       2        7

Thermal summary:
---------------------------------------------------------------
  Estimated junction temperature:                    25C
```

Fig. 7: Array Multiplier Power Report

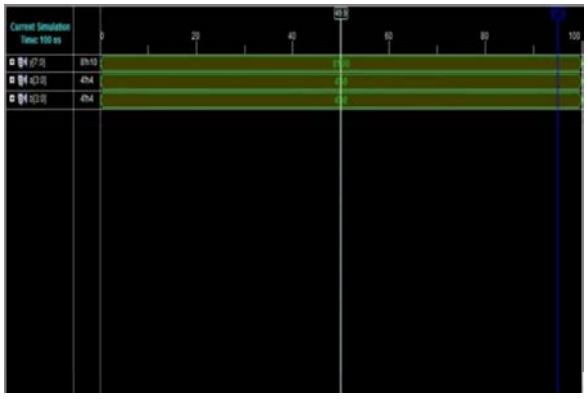

Fig. 8: Column Bypass Simulation

```
--------------------------------------------------------------------
Release 9.2i - XPower SoftwareVersion:J.36
Copyright (c) 1995-2007 Xilinx, Inc.  All rights reserved.
Design:       multiplier_4bit.ncd
Preferences:  multiplier_4bit.pcf
Part:         2s50tq144-6
Data version: PRELIMINARY,v1.0,07-31-02

Power summary:                          I(mA)     P(mW)
--------------------------------------------------------------------
Total estimated power consumption:                   7
                                          ---
                        Vccint 2.50V:        0        0
                        Vcco33 3.30V:        2        7
                                          ---
                              Inputs:        0        0
                               Logic:        0        0
                             Outputs:
                                Vcco33        0        0
                             Signals:        0        0
                                          ---
                 Quiescent Vcco33  3.30V:        2        7

Thermal summary:
--------------------------------------------------------------------
   Estimated junction temperature:                  25C
```

Fig. 9: Column Bypass Power Report

Time Multiplexed Zero Adaptation Delay Lms Filter:

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	3	16640	0%
Number of Slice Flip Flops	6	33280	0%
Number of 4 input LUTs	6	33280	0%
Number of bonded IOBs	93	519	17%
Number of GCLKs	1	24	4%

Fig. 10: Device Usage-Area Utilization of Zero Adaptation

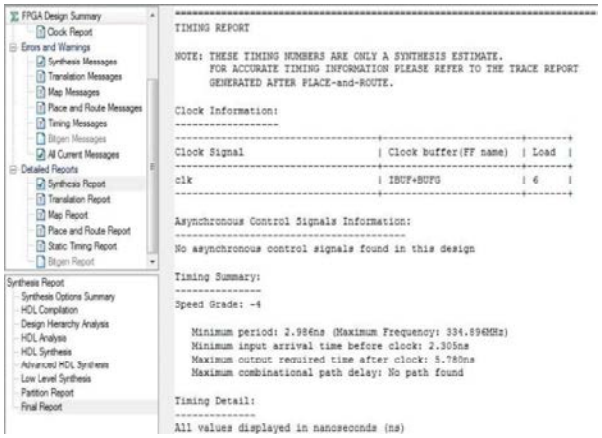

Fig. 11: Zero Adaptation Delay Timing Report

Two Adaptation Delay Adaptive Lms Filter

Input: Same as in zero adaptation are used here.

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	490	16640	2%
Number of Slice Flip Flops	70	33280	0%
Number of 4 input LUTs	833	33280	2%
Number of bonded IOBs	336	519	64%
Number of GCLKs	1	24	4%

Fig. 12: Device Usages – Area Utilization of Two Adaptations

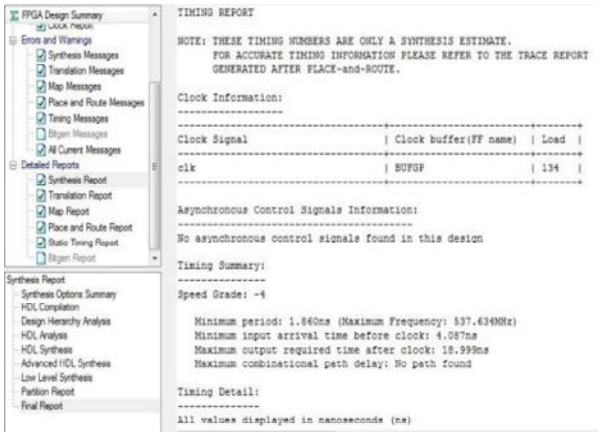

Fig. 13: Two Adaptation Delay Timing Report

CONCLUSION

As per the results obtained, Column By-Passing Multiplier architecture has less power when compare to array multiplier architecture which makes it suitable for use in adaptive filters as shown in Table 1.With this column-Bypassing multiplier, Direct-Form zero adaptation and two adaptation delay LMS adaptive filters are implemented.

Table 1: Comparison of Array and Column Bypass Multiplier

Architecture	Power Consumption
Array Multiplier	11mw
Column ByPass Multiplier	7mw

With this column-Bypassing multiplier, Direct-Form zero adaptation and two adaptation delay LMS adaptive filters are implemented. Based on the critical path analysis from Figures 10,1112,13, zero adaptation has less complexity when compare to Two Adaptation. Also this zero adaptation has minimum maximum usable frequency and thus can be used for high data rate applications as shown in Table 2.

Table 2: Comparison of Zero and Two Adaptation delay:

Architecture	Critical Path Delay	Multiplier	MUF (MHZ)
Zero Adaptation	2.986 ns	N	334.896
Two Adaptation	1.86 ns	2N	537.64

REFERENCES

1. Yi, Y., R. Woods, L.K. Ting and C.F. Cowan, (Jan 2005), High speed FPGA-based implementations of delayed-LMS filters, Trans. Very Large Scale Integr. (VLSI) Signal Process., 39(12): 113-131.

2. Harrison, W.A, J.S. Lim and E. Singer, (Feb 1986), A new application of adaptive noise cancellation, IEEE Trans. Acoust., Speech, Signal Process., 34(1): 21-27.
3. Meher, P.K. and M. Maheshwari, (May 2011), A high-speed FIR adaptive filter architecture using a modified delayed LMS algorithm, in Proc. IEEE Int. Symp. Circuits Syst., pp. 121-124.
4. Meher, P.K. and M. Maheshwari, (May 2011), A high-speed FIR adaptive filter architecture using a modified delayed LMS algorithm, in Proc. IEEE Int. Symp. Circuits Syst., pp. 121-124.
5. Meher, P.K. and S.Y. Park, 2014. Area-delay-power efficient fixed-point LMS adaptive filter with low adaptation-delay, Trans. Very Large Scale Integr. (VLSI) Signal Process.
6. Park S.Y. and P.K. Meher, (Jun 2013), Low-power, high-throughput and low area adaptive FIR filter based on distributed arithmetic, IEEE Trans. Circuits Syst. II, Exp. Briefs, 60(6): 346-350.
7. Van, L.D. and W.S. Feng, (Apr 2001), 'An efficient systolic architecture for the DLMS adaptive filter and its applications', IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process., 48(4): 359-366.
11. Uwe Meyer-Baese, 2014. Digital Signal Processing with Field Programmable Gate Arrays' Springer-Verlag Berlin Heidelberg.