Gray Encoded Harmonics Power Line Interference Cancelling Structure Using LMS and NLMS Adaptive Algorithms

Eduardo A. da Costa[1], Sérgio J. Almeida[1], Mônica L. Matzenauer[1], and Mateus B. Fonseca[2]

[1]Polytechnique Institute, Catholic University of Pelotas, Pelotas, Brazil
[2]Engineering Department, Federal University of Pelotas, Pelotas, Brazil,
e-mail: ecosta@ucpel.tche.br

ABSTRACT

This paper proposes the implementation of a Gray encoded structure for harmonics power line interference cancelling. The structure uses dedicated hardware architecture for the Least Mean Square (LMS) adaptive filtering algorithm, as well as its normalized version (NLMS). In the used scheme, from a 60Hz reference signal, the algorithms are able to estimate the superior harmonics, using after these results for the cancelling of interferences related to the signal of interest. In this work, the proposed adaptive filtering architectures and the harmonics generator block use a Hybrid encoding in its data buses, which is a compromise between the minimal input dependency presented by the Binary encoding and the low switching characteristic of the Gray encoding. For the Hybrid structures, new Hybrid multipliers were proposed, and the results showed that those multipliers are more efficient than the Binary ones, by presenting less power consumption in some cases. The implemented harmonic cancelling structure with the LMS and NLMS adaptive filtering architectures were validated and compared by using both Binary and Hybrid encoding. The efficiency of the implemented Hybrid structure for the cancelling of interferences was proved by reducing more power than the Binary one. By the results, we conclude that it could be practicable to implement a harmonic cancelling structure, based on LMS and NLMS adaptive filtering, operating on Hybrid encoding.

Keywords*:* LMS and NLMS adaptive filters; Gray encoding; arithmetic operators.

I. INTRODUCTION

In this work the main goal is the development of an efficient structure for cancelling harmonic power line interference application [1]. For the removal of sinusoidal interference, notch filters can be used with fixed coefficients tuned to the frequency of the interference at each harmonic. However, if the frequency of the interference is not known beforehand with accuracy, or even if the frequency is previously known, but some variations have occurred, the adaptive filter is certainly the best solution

The interference from the power line is the most common type of noise associated with bioelectric signals [2], and it becomes a serious problem in some applications, such as high resolution electrocardiograph [3]. In particular, the amplitude of the interference can be more significant in the first three harmonics, mainly due to magnetic fields originated from nonlinear characteristics of the propagation path, such as power transformer, or even fluorescent lamp reactors [1].

The structure proposed in this work is based on the interference canceller introduced in [1], where the higher harmonics are mathematically estimated by means of trigonometric relations. Since mathematic equations for the high order harmonics were simplified to fit on fixed-point representation, some constant multiplications were implemented with shift-add circuits. The structure is based on Least Mean Square (LMS), and Normalized mean Square (NLMS) adaptive filtering algorithms.

Our proposed structure is fully hardwired. It was described in VHDL, and synthesized using Cadence Encounter RTL Compiler tool with Nangate 45nm Open Cell library. It uses Hybrid encoding in the data buses, whose main idea is to split the operands in group of m-bits, encode each group using the Gray code (that potentially enable reduction of the switching activity into each group) and propagate the carry between the groups as in the Binary encoding [4]. We developed new signed Hybrid multipliers, which uses radix-2^m encoding. They are applied to both the harmonics generator block and the LMS and NLMS adaptive filtering architectures. We have implemented 18, and 36-bit regular radix-4 Hybrid array multipliers, as well as a particular case for irregular ($m=3$) operation.

Efficient FPGA-based architectures have been introduced [5]-[7] for harmonic interference cancel-

ling, based on LMS algorithm. However, the main idea has generally been the cancelling of interferences from the fundamental (50Hz or 60Hz) frequency. Due to the fast convergence speed, good stability, and accurate tracking capability characteristics, some FPGA-based NLMS architectures, have been proposed for different applications, such as acoustic echo cancellation [8], and interference suppression in communication channels [9]. In our work, both LMS and NLMS adaptive filter architectures are used as basis for an efficient structure for harmonics power line interference cancelling operating on both binary and hybrid encoding. To the best our knowledge, there exist no solutions that target hybrid encoded hardwired design for the harmonic interference cancelling application, based on mentioned adaptive filter architectures.

The synthesized hybrid harmonic cancelling structures were validated, so that the interferences from the ECG signal, in 60 Hz and its high order harmonics (120, 180, 240 Hz), could be efficiently filtered. The proposed Hybrid harmonics cancelling structures, based on LMS and NLMS adaptive filters proved to be efficient in terms of power consumption reduction, when compared with the Binary ones.

This paper contains as contributions:

• An optimized Hybrid Harmonics cancelling, based on LMS and NLMS adaptive filters (in [10] only the structure based on LMS adaptive filter was taken into account);

• Synthesis results using Cadence Encounter RTL Compiler tool with Nangate 45nm Open Cell library. Note that in [10], only results for a FPGA-based structures were presented.

The rest of the paper is organized as follows: Section II presents adaptive filter background, and the proposed interference canceller. Section III presents an overview of the encoding techniques as well as the new hybrid multipliers. The proposed hybrid harmonics generator block, and hybrid LMS and NLMS filter architecture are introduced in Section IV. The experimental results and the validation of the structure are given in Section V and finally, Section VI concludes the paper.

II. ADAPTIVE FILTER STRUCTURES

The adaptive filtering is today a powerful tool applied to the signal processing. Among the most popular adaptive algorithms the LMS and NLMS can be highlighted. The low complexity of these algorithms, makes them the most popular in terms of applications. The adaptive filter application is generally categorized in four classes as the identification, inverse modeling, prediction, and interference cancelation [11]. In this work, the last class is addressed, and its block structure is shown in Figure 1.

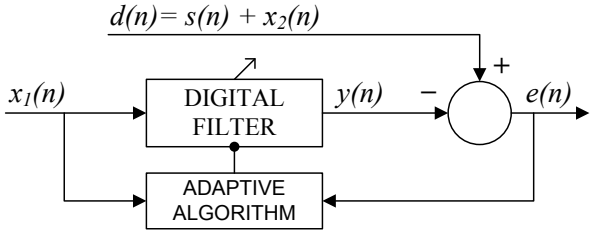


Figure 1. Adaptive system: interference cancelling

An adaptive system consists of an adaptive algorithm that adjusts the values of the weights of a digital filter. The FIR (Finite Impulse Response) filter is widely used in adaptive structures, due to its inherent stability, and it is the best choice for real-time applications [12],[13]. In Figure 1, $d(n)$ is the sum of the desired signal $s(n)$ with the corrupted by additive noise signal $x_2(n)$. A distorted signal $x_1(n)$, but correlated with $x_2(n)$, is also available. The adaptive system in this class of application will produce an output, $y(n)$, that closely resembles $x_2(n)$; therefore, the output $e(n)$ will closely resemble $s(n)$.

A. Least Mean Square (LMS) Algorithm

The LMS adaptive algorithm is a convenient method of adapting the coefficients of a finite impulse response (FIR) filter [11]. This algorithm is described in (1), where $\mathbf{w}(n) = [w_0(n), w_1(n), .., w_{N-1}(n)]^T$ are coefficients and $\mathbf{x}(n) = [x(n), x(n-1), ..., x(n-N+1)]^T$ are the input data samples currently in filter memory, $d(n)$ is the desired response, N is the filter length, and μ is the algorithm step size.

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu\left(d(n) - \mathbf{w}^T(n)\mathbf{x}(n)\right)\mathbf{x}(n) \qquad (1)$$

The LMS algorithm [11] uses the instantaneous value of the squared error in order to estimate the mean square error. The updating of the coefficients vector to the LMS algorithm solution is presented in (2), where $\mathbf{w}(n+1)$ is the actual vector updated from the previous coefficients vector $\mathbf{w}(n)$, and the term μ term establishes the step of adaptation and the speed of convergence of the algorithm to the optimal coefficients vector in the filtering process. As higher the step of adaptation μ, faster will be the convergence of the coefficients. However, the value of μ depends on some criteria that can assure the stability of the LMS algorithm, such as the filter order and the power of the input signal.

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu\mathbf{x}(n)e(n) \qquad (2)$$

B. Normalized Least Mean Square (NLMS) Algorithm

The normalized LMS algorithm presents the property of adjusting the signals related to the power of the reference signal. This is done according to the equation of coefficients adjustment presented in (3). The μ parameter represents the step size for the adjustment of the convergence speed of the NLMS algorithm.

This technique of normalization presents the main benefits: i) the convergence of the coefficients into the optimal filtering values is immune of variations in the power of the reference signal; ii) the speed of the adaptation of the coefficients is increased as compared with the LMS algorithm.

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu \frac{\mathbf{x}(n)}{\mathbf{x}(n)^T \mathbf{x}(n)} e(n) \qquad (3)$$

C. The Interference Canceller Structure

The interference canceller structure (ICS) is based on [14], and it includes one harmonics generator block, and four 2-coefficient dedicated fixed-point adaptive filter (one for each harmonic), based on LMS or NLMS algorithm, as can be seen in Figure 2. Note that in the ICS of [16] only the NLMS adaptive filter was considered. Moreover, the hybrid structure for the harmonics cancelling was not taken into account, as in this work.

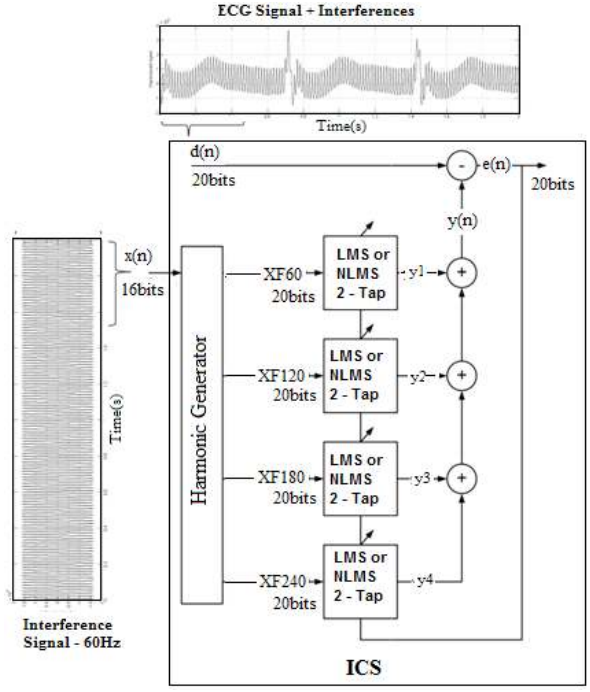


Figure 2. The proposed interference canceller structure (ICS) based on LMS and NLMS adaptive filters

It should be observed that the proposed structure is fully modular, so that the number of blocks to be used depends on the number of harmonics to be cancelled.

The output of the filter is the error signal $e(n)$ that is obtained from the subtraction of the desired signal $d(n)$ and the signal $y(n)$ (obtained from the summation of each adaptive filter output). The signal $e(n)$ represents the estimated error used by the adaptive algorithm blocks in order to adjust the coefficients until the components of the interference have been suppressed from the desired signal.

The filters are implemented with only 2-tap, because according to [11], it is enough to represent a sinusoidal signal. The proposed structure is fully modular, so that the filters are easily replied according to the number of harmonics to be canceled. The output of the filter $e(n)$ is obtained subtracting $y(n)$ from $d(n)$, $y(n)$ is the summation of each adaptive filter output. All the signals of Figure 2 will be operated on Hybrid code.

III. HYBRID ENCODING OVERVIEW

One of the most promising encodings that can be used to reduce switching activity is the gray code since only one bit changes between consecutive values. Therefore, for highly correlated signals the switching activity can be reduced significantly [15].

In [16], the process of implementing arithmetic operators that operate directly upon gray code inputs was investigated. However, it was observed that the combinational logic required by the arithmetic modules is large. Furthermore, it was not possible to develop a regular structure for the operators such that different word sizes could be accommodated. In [4] it was proposed the use of a hybrid encoding for the operands which is a compromise between the minimal input dependency presented by the binary encoding and the low switching characteristic of the gray encoding. A methodology for the generation of regular structures for arithmetic operators using encoded operands was presented. In this work, we have used this regular radix-4 hybrid multiplier, as well we propose a new irregular one, in order to implement a fully hybrid harmonics power line interference canceller.

A. Hybrid Code Definition

The idea of the hybrid code is to split the operands in groups of m-bits, encode each group using the gray code and use the binary approach to propagate the carry between the groups. In this manner, the number of transitions inside each group is minimized and a regular structure can be easily built. Table I exemplifies the hybrid encoding for 2´s complement 4-bit numbers and radix-4 (m=2).

Table I. 2´s complement radix-4 hybrid code repre-sentation

Decimal number 2's complement	Hybrid code representation
0	0000
1	0001
2	0011
3	0010
4	0100
5	0101
6	0111
7	0110
-8	1100
-7	1101
-6	1111
-5	1110
-4	1000
-3	1001
-2	1011
-1	1010

An additional feature of the hybrid code is that the conversion to and from binary is very simple, as indicated in Figure 3. Translating words of W bits between binary and hybrid, in any direction, all that is required are $W/2$ EXOR gates. Therefore, the overhead in terms of transcoders is low, making this encoding very desirable for the adaptive filter architecture.

In [16] it is shown that the hybrid code presents a number of transitions right between the gray and binary codes, for a counting sequence, with 33% less number of transitions than the binary code. Hence, for systems where the switched capacitance in the data buses is significant, as in adaptive filter architecture, and where the data presents a high degree of correlation, power can be saved.

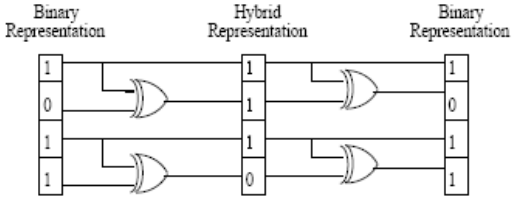


Figure 3. Conversion between binary and hybrid codes

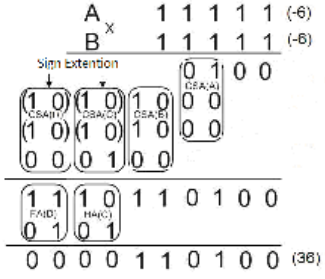


Figure 4. Example of an irregular W=5 hybrid code multiplication

B. The Hybrid Encoded Multipliers

The idea of hybrid arithmetic operators was originally introduced in [4], where it was possible to generate adders and multipliers with regular structures. In this work, these regular radix-4 (m=2) structures were used for the implementation of 18-bit and 36-bit multipliers, that are used in the structure of the fundamental frequency and harmonics generator as will be presented later. In this structure, a 23-bit multiplier is also needed. However, for this multiplier, we have proposed a new irregular hybrid code structure [10]. We have also implemented 20-bit hybrid multipliers that are used in the LMS and NLMS adaptive filters. Figure 4 shows an example of an irregular W=5 multiplication.In the example presented in Figure 4, the operands are represented in radix-4 hybrid encoding, where the most significat bit represents the signal of the operand. Therefore, the operand 11111 is converted to hybrid decimal as follows: $11111_b = -1x4^2 + 2x4^1 + 2x4^0 = -6_d$. Figure 5 shows the architecture for the example presented in Figure 4.

In the example of Figure 5, Type 2:2 is the unsigned m=2 multiplication proposed in [4]. Type 3:2 handles the 3:2 partial product of unsigned two least significant bits with 2´s complement three most significant bits. Type 3:3 operates on two signed values. Type 3:2 and 3:3 multiplier blocks are presented in Figure 6a and 6b, respectively. Note that both multiplier blocks are composed by simple regular Type 2:2 and 2:1 multiplier blocks. Type 2:1 yields the product of unsigned two least significant bits with one signed most significant bit.

We have applied Carry Save adders (CSA) in the partial product lines of the multipliers circuits in order to speed-up the carry propagation along the array, since this structure outputs the carry bits instead of propagating them to the left, as in the Ripple Carry adder (RCA). The basic idea of the CSA is that three

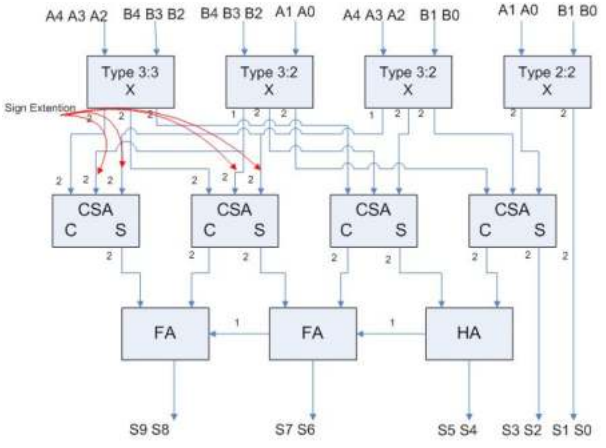


Figure 5. Hybrid code multiplier architecture for W=5

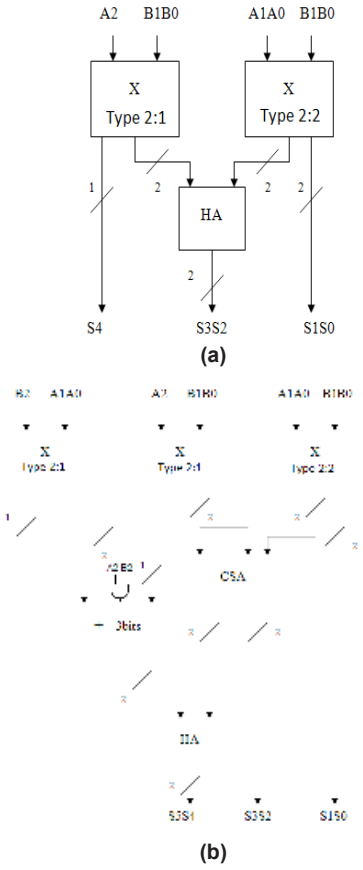


Figure 6. Type 3:2 (a), and Type 3:3 (b) multiplier blocks

numbers can be reduced to two, in a 3:2 compressor, by doing the addition while keeping the carries and the sum separate. Therefore, it should be observed that the final product for the multiplier is obtained by adding each 2-bit groups of the partial product terms in a carry save form, as shown in Figure 5 (3:3 multiplier block also uses CSA). Note that only the final recombination of the final carry and sum requires a ripple carry addition. In the RCA operation, the two most significant bits should be not considered. A sign extension technique is used, where two extra bits are used in the partial product. Note that we extend the signal as 10 that represents a negative number (-1) in the hybrid operation.

C. Multipliers Results

Table II shows area, delay and power results for the 18, 20, 23 and 36-bit multipliers. The multipliers were described in VHDL and synthesized using Cadence Encounter RTL Compiler tool with Nangate 45nm Open Cell library. Area is given in terms of number of cells, delay in ps and power in mW. For power results, leakage and dynamic values are taken into account. A total of 10,000 binary and hybrid random input samples was used as stimulus for the multipliers.

The hybrid multipliers presented larger number of cells for all multipliers, because the addition of EXOR gates for the conversion of representation. Therefore, the increase of on the number of cells leads to the larger leakage power consumption of the hybrid multipliers, as can be seen in Table II. However, the hybrid multipliers present the less delay values for all multipliers. It occurs because the internal hybrid dedicated 2-bit multiplication blocks are more regular than the binary ones, with a more reduced critical path.

This was possible to obtain, because the hybrid multiplication blocks process the hybrid code in the inputs, and the outputs are automatically converted to binary code. This is necessary because according to [16], the binary adders are more power efficient than hybrid ones. This is particularly true for the regular 18-bit, and 20-bit structures that although present more leakage power, the significant dynamic power reduction contributed for the total power reduction in these multipliers. However, this is not the same for the 23-bit hybrid multiplier, whose irregularity of the structure leads to a slightly more power consumption. However, as this hybrid multiplier presents significant less delay value, therefore when power-delay-product metric is considered, this multiplier is more efficient than the binary one.

The same aspect is presented for the 36-bit hybrid multiplier that presents more power than the binary one. We have observed that the aspect of the higher power consumption presented in the 36-bit hybrid multiplier is related to the delay value close to the binary multiplier. This means that the critical path for the 36-bit hybrid multiplier is not so small, what leads to

Table II. Results for binary and hybrid multipliers

Multiplier	#Cells	Timing (ps)	Power (mW)		
			Leakage	Dynamic	Total
18-b Binary	1444	3634	0.057	1.669	1.726
18-b Hybrid	1784	3140	0.064	1.641	1.706
20-b Binary	1894	3840	0.071	2.370	2.441
20-b Hybrid	2213	3513	0.080	2.253	2.333
23-bBinary	3056	5982	0.116	3.464	3.580
23-b Hybrid	3394	4018	0.126	3.678	3.805
36-b Binary	6537	6509	0.242	12.175	12.417
36-b Hybrid	7301	6501	0.265	12.387	12.652

the slightly more power consumption presented by this multiplier. However, when the power-delay-product aspect is taken into account, the 36-bit hybrid multiplier is slightly more efficient.

IV. HYBRID HARMONICS GENERATOR BLOCK AND ADAPTIVE FILTER ARCHITECTURES

The proposed structure consists of a simplified harmonics generator, and optimized LMS and NLMS adaptive filters, that yields a fixed-point architecture requiring only two coefficients, both operating on hybrid code.

A. Fundamental Frequency and Harmonics Generator Block

The structure that generates the fundamental frequency and its high order harmonics is presented in Figure 7. It was obtained from (4), (5), and (6) [1]. These equations were modified to operate on fixed-point. Besides the arithmetic circuits, the left and right shifting of bits was also used to perform the operations of multiplication and division, respectively. Moreover, some multipliers could be optimized by using shift-add circuits.

$$XF120 = \frac{2x^2 - a^2}{a} \tag{4}$$

$$XF180 = \frac{4x^3 - 3xa^2}{a^2} \tag{5}$$

$$XF240 = \frac{2(XF120)^2 - a^2}{a} \tag{6}$$

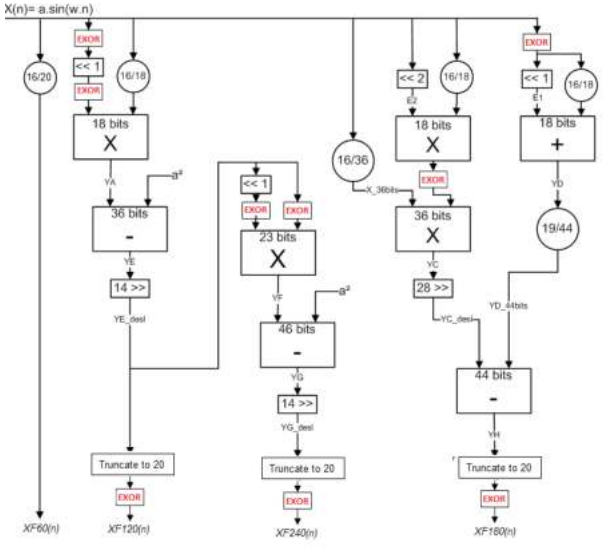


Figure 7. Structure of the fundamental frequency and harmonics generator

In (4)-(6), the term x is the sampled reference signal from the power line, i.e., 60 Hz sinusoidal signal. The terms $XF120$ (120 Hz), $XF180$ (180 Hz), and $XF240$ (240 Hz), are resultant terms from the trigonometric relations that are obtained from the 60 Hz signal. The term a is used to adjust the equations for fixed-point operations, and it is the amplitude of the reference signal $x(n) = a.sine(w,n)$. Note that in order to enable the hybrid code operation, converters from binary to hybrid code were used. They are represented by only EXOR gates in Figure 7. Therefore, the reference signal $x(n)$, and the signal $d(n)$ are both represented in hybrid code as well the step size (μ) and the term a.

All the multipliers in Figure 7 operate on hybrid code, including the new irregular 23-bit hybrid multiplier, whose structure was previously presented in Section III. However, the addition/subtracter circuits are operated on binary code, because as stated in [16], and as mentioned before, these operators do not enable power consumption reduction when using hybrid code. Thus, the outputs of the hybrid multipliers are already encoded in binary representation for the addition and subtraction operations. This aspect explains the EXOR gates used in the $XF120$, $XF180$ and $XF240$ output terms (note that the output of the fundamental 60 Hz frequency has been already represented in hybrid code from the input).

B. LMS Adaptive Filter Architecture

The hybrid adaptive filter architecture uses as basis the LMS adaptive filter architecture proposed in [14]. The architecture of one block of 2-Tap adaptive filter is presented in Figure 8 (the control part was omitted). The update of the two coefficients is performed in seven clock cycles, as can be seen in Table III, what is related to the steps of updating the coefficients of the LMS algorithm. Note that the use of EXOR gates for the hybrid code operation does not increase the number of clock cycles of the architecture.

In the adjustment of the vector of coefficients, the term μ establishes the step of adaptation and the

Table III. Operations of the LMS algorithm architecture.

Clock cycles	Operation
1°	$w_0(n)\,x_0(n) + w_1(n)\,x_1(n)$
2°	$e(n) = d(n) - y(n)$
3°	$\mu.e(n)$
4°	$w_1(n) + \mu.e(n)\,x_1(n)$
5°	$w_1(n+1)$
6°	$w_0(n) + \mu.e(n)\,x_0(n)$
7°	$w_0(n+1)$

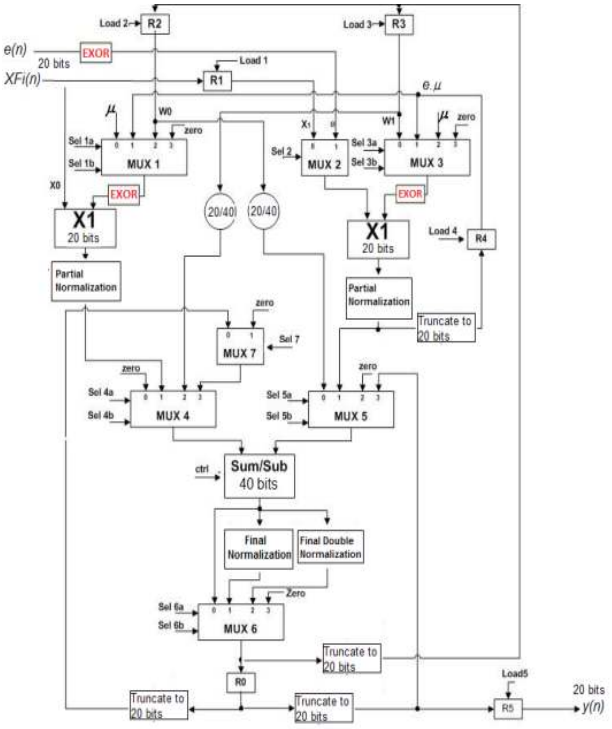


Figure 8. Adaptive filter architecture based on LMS algorithm

speed of convergence of the algorithm. The higher the step of adaptation μ, the faster will be the convergence of the coefficients. However, the value of μ depends on some criteria that can assure the stability of the LMS algorithm, such as the filter order and the power of the input signal [11].

The step of adaptation is converted to hybrid code by the EXOR gates in the output of the multiplexers 1 and 3 ('MUX 1' and 'MUX 3' in Figure 8). The two coefficients w_0 and w_1 are also converted to hybrid code by 'MUX1' and 'MUX3' outputs, respectively. One extra EXOR gate converts the signal $e(n)$ to hybrid code.

Once again the outputs of the hybrid multipliers are already internally converted to Binary to enable the addition/subtraction operation (SUM/SUB block in Figure 8) in binary code. When the outputs of the SUM/SUB block are converted to hybrid again, by the EXOR gates in the outputs of 'MUX1' and 'MUX3', new multiplications in hybrid code are allowed.

Besides multiplexers, the architecture of Figure 8 also includes registers, multipliers, adders, truncation, and normalization blocks. With these blocks, the signals are adjusted to produce results close to the floating-point representation in hybrid encoding. The reference signal $XF(n)$ is obtained from the harmonics generator block in hybrid code. The normalization is divided into partial and final normalization. The normalization leads to the division of the signals by a normalization factor of 32768 (2^{15}), where Q15 format is used. The final double normalization represents a division by 2^{14}. The 2-tap adaptive filter architecture was implemented using 20-bit, because by experimental analyses was verified that it is the minimum number of bits to suppress the interferences by using only two coefficients in the filter stages.

The steps of the LMS architecture operation, shown in Table III, are summarized as follows:

- Cycle 1: Calculation of the filtering signal y(n):

Two samples x_0 and x_1 are inserted (delayed each other by the register 'R1'). The samples are multiplied ('X1' and 'X2' multipliers) by the w_0 and w_1 coefficients (initially values equal to zero). The intermediate values are added after the first normalization step. The addition result (40-bit) pass through the final normalization, and the result is kept in the register R0, until the next clock cycle.

- Cycle 2: Calculation of the error signal e(n):

As presented in Figure 2, this signal is obtained from the subtraction of the external desired signal $d(n)$ and the filtering signal $y(n)$ obtained in the previous clock cycle. However, after released by the register 'R0', the signal $y(n)$ pass through the trunc block, and it is kept in the register 'R5'. This main goal of this block is limit to 20-bit the signal $y(n)$, since in the normalization process the bits are shifted right, i.e, the most part of the representation from the processed signal is in the least significant bits.

- Cycle 3: Calculation of μ.e(n):

Represents the first part of the calculation of updating portion. The error signl obtained in the previous cycle is multiplied by the adaptation step, by using the 'X2' multiplier. As result, a 40-bit value is generated, but it is normalized and truncated to 20-bit. The result is kept in the register 'R4'. Note that in this cycle, only one partial normalization is realized.

- Cycle 4: Calculation of the next coefficient w_1:

For this calculation, the delayed sample x_1 is multiplied by the value of the register 'R4' ($\mu.e(n)$). The result is added to zero. In this step, the multiplier 'X2', and the adder (Sum/Sub block) circuits are used. After the multiplication, a new parcial normalization is performed (in a total of two consecutive partial normalization). In order to avoid any problem in the result of the operations, two final normalizations are realized (in the final double normalization block).

- Clock 5: Updating of the coefficient w_1:

After the portion of updating the coefficint w_1 is available (obtained in the previous clock cycles), this value is added to the actual value of w_1. Before the realization of the addition, the actual value of w1 is concatenated to 40-bit, and then the result of the addition is truncated to 20-bit, and the register 'R3' keeps the new updated coefficient. The only arithmetic circuit used in

this clock cycle is the adder (SUM/SUB block). Note that the truncate block guarantees that the coefficients have no more than 20-bit in the coefficients updating operation.

• Cycle 6: Calculation of the next coefficient w_0:

The structures used in this clock cycle are the same as in clock 4. The only difference is that now the multiplier 'X1' is used. The procedure for the normalization steps are slso the same as in the clock 4.

• Clock 7: Updating of the coefficient w_0:

In this clock cycle the coefficient w_0 is finally updated. The control of the state machine is returned to the first cloc cycle, and new samples $x(n)$ and $d(n)$ can be loaded.

C. NLMS Adaptive Filter Architecture

The architecture of the NLMS filter was developed by reusing most of the part of the LMS architecture, including the normalization blocks, the changing of the position of the registers as well as the addition of signals in the multiplexers. It can be seen in Figure 9. The architecture needs a divider circuit, for the normalization process. We have used the divider circuit from the Cadence tool.

The NLMS adaptive filter uses almost the same steps of operation as in LMS, as can be seen in Table IV. However, in the 4th clock cycle occurs a division of $\mu.e(n)$ by $x(n)^T x(n)$. The operation of this clock cycle adjusts the convergence step of the NLMS filter according to the estimated error, the speed of adaptation, and the power of reference signal. It should be observed

Table IV Operations of the NLMS algorithm architecture

Clock Cycle	Operation
1º	$w_0(n)x_0(n) + w_1(n)x_1(n)$
2º	$e(n) = d(n) - y(n)$
3º	$k(n)=\mu. e(n)$
4º	$\Delta(n)=k(n)/\|x(n)\|^2$
5º	$w_1(n) + \Delta(n)x_1(n)$
6º	$w_1(n+1)$
7º	$w_0(n) + \Delta(n)x_0(n)$
8º	$w_0(n+1)$

that in the division step, the result of the denominator term is given by a multiplication of the sample vector of the reference signal, in the transposed form, with the same vector, by in the original form. However, as the NLMS filter was implemented with only two coefficients, therefore the vector $x(n)$ presents only the two last samples of the reference signal.

Note that the hybrid operation of the filter is the same as in the LMS, i.e, the step of adaptation is converted to hybrid code by the EXOR gates in the output of the multiplexers 1 and 3 ('MUX 1' and 'MUX 3' in Figure 9). The two coefficients w_0 and w_1 are also converted to hybrid code by 'MUX1' and 'MUX3' outputs, respectively. One extra EXOR gate converts the signal $e(n)$ to hybrid code. As in LMS, when the outputs of the SUM/SUB block are converted to hybrid again, by the EXOR gates in the outputs of 'MUX1' and 'MUX3', new multiplications in hybrid code are allowed. We have not used divider circuit in hybrid code in this work.

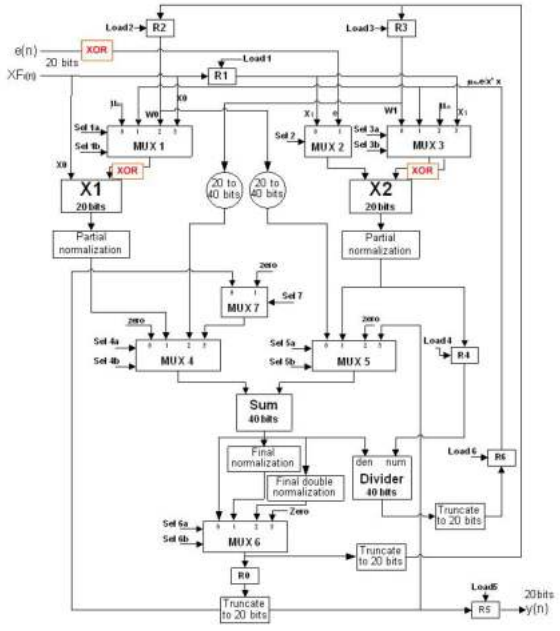


Figure 9. 2-Tap adaptive filter architecture based on NLMS algorithm.

V. VALIDATION OF THE DEVELOPED ARCHITECTURES

The proposed architectures are evaluated on ECG signals. The ECG signal contaminated with 60, 120, 180, and 240 Hz sinusoidal signals is used to simulate the architecture. The ECG signals were taken from the physiologic database Physiobank Archive Index [17] sampled at 1KHz, where 10,000 16-bit width samples were used in the simulations. The ICS was prototyped in a DE1 Altera board, where FPGA Cyclone II component was used. The methodology is presented in Figure 10 [14]. Audio CODECs from the PC Computer and from the DE1 board were used for A/D and D/A conversions. This methodology was used in order to show the graphics with the interference cancelling results (comparisions from both the hardware and the model from Matlab model).

Figure 11 shows the desired signal $d(n)$ with corrupted signal by additive noise in binary and hybrid representation. As illustrative example, Table III shows the first five 20-bit samples of the signal $d(n)$ in binary

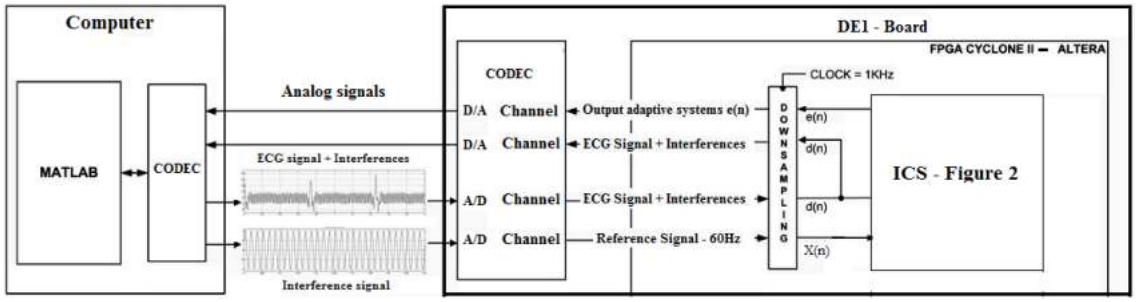
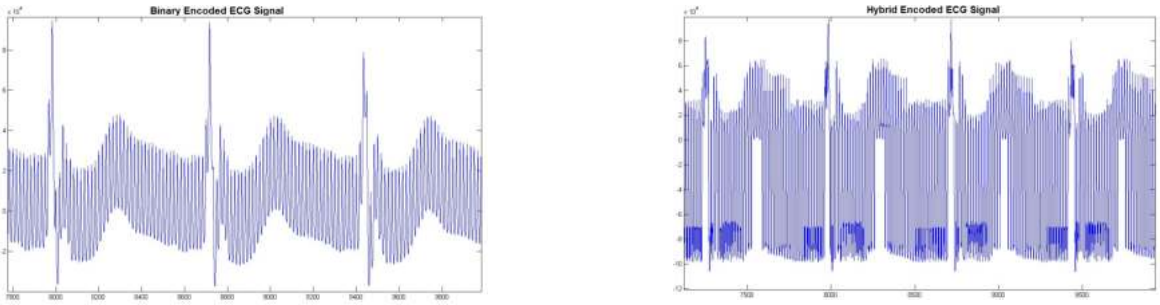


Figure 10. Methodology for the prototyped ICS architecture



(a) **(b)**

Figure 11. Desired signal *d*(*n*) with corrupted signal by additive noise – Binary signal (a) and Hybrid signal (b)

Table V. Desired signal represented in binary and hybrid encoding.

d(n) **–** Binary encoding	***d(n)*** **–** Hybrid encoding
00000100100110101101	00000100110111111001
00000010001110011110	00000011001011011011
11111111100110110001	10101010110111100001
11111101100010011110	10101001110011011011
11111100000100000101	10101000000100000101

and hybrid code. Note that the conversion for hybrid code is realized by applying an EXOR operation at each two bits, what lead to a radix-4 operation.

A. Simulation Results

The signals were normalized to ease the visualization and comparisons among the different graphics. In the simulation of the adaptive filter example (based on LMS filter), the adaptation step $\mu = 0.0015$. As the hardwired filter processes each sample at seven clock cycles, and the input signals were sampled at 1 KHz, hence the architecture is simulated at 7 KHz. The graphics with the last samples of the ECG signal with interference, and after the filtering process, both in Hybrid code, are presented in Figure 12.

The cancelling of the interference in the fundamental frequency and its high order harmonics is more easily verified during the power spectrum simulations, as can be seen in Figure 13. The results presented in the graphics of Figure 13 show that the proposed structure is able to cancel the interferences from the ECG signal operating on hybrid code.

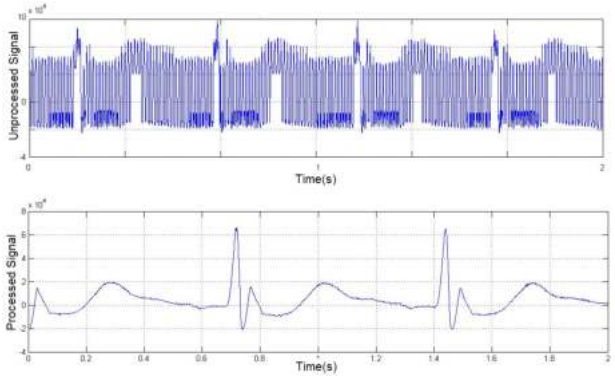


Figure 12. Hybrid signal *e*(*n*) with interference and after the filtering process

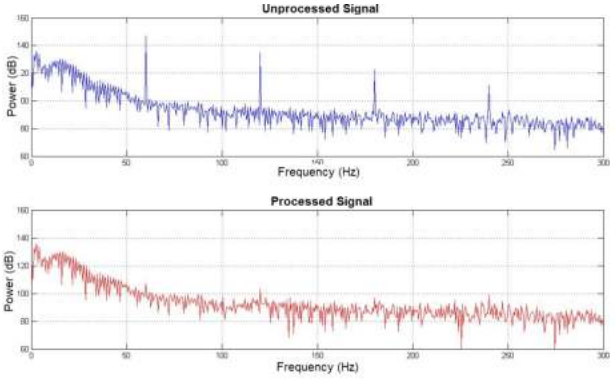


Figure 13. Power spectrum from the hybrid ECG signal. Signal with interferences in the fundamental frequency and the high order harmonics, and signal obtained from the hybrid hardwired adaptive filter

In [18] it was already shown that the filter result from the LMS fixed-point architecture is close to that obtained from the floating-point MATLAB model. Figure 14 shows that it is the same for the NLMS fixed-point architecture. Note that although it is possible for both structures converge for the minimum error, the adaptation process from the NLMS Matlab model is faster than the hardwired NLMS adaptive filter. In fact, it occurs because the hardwired NLMS filter uses internally some trucations needed for a fixed-point operation, what has delayed the adaptation protcess. However, although the delay in the convergence, the hardwired ICS structure, based on NLMS adaptive filter, is as efficient in the interference cancelling as the Matlab model, as can be seen in Figure 14.

Figure 15 shows the speed of convergence of the LMS [18] and the NLMS [14] adaptive filters for the first 4,000 samples of the estimated error signal $e(n)$. It can be seen that the NLMS filter suppress the interference faster than the LMS architecture (almost 3 seconds faster, as marked in Figure 14). This occurs because the NLMS filter adjusts the step of adaptation in the process of coefficient updating, and it contributes for the fast convergence of the coefficients to the optimal values of filtering. The faster convergence in the NLMS adaptive filter is obtained at cost of hardware increase (because the use of a divider circuit), and one clock cycle more, as can be seen in Table IV.

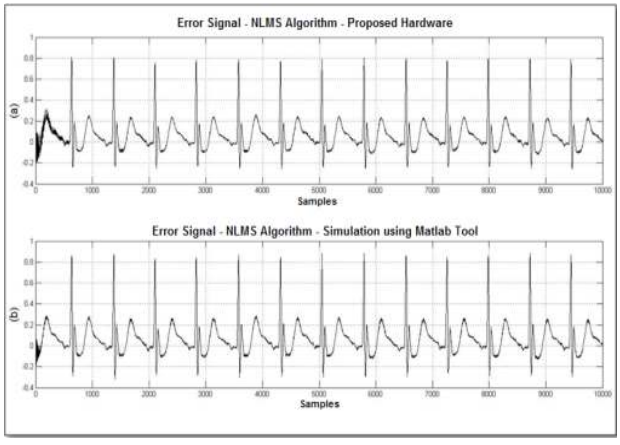


Figure 14. Error signal *e(n)* of the simulation from both the ICS (a), and from the Matlab model both using NLMS adaptive filter (b)

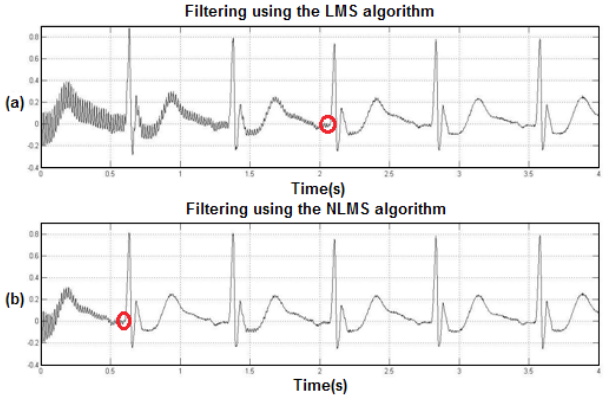


Figure 15. Simulation results of e(n) signal from hardwired adaptive filters based on: LMS [18] (a), and NLMS [14] (b)

B. Synthesis Results

Table VI shows area, delay and power results for the Binary and Hybrid harmonics power line interference cancelling structures (ICS) using both LMS and NLMS adaptive filters. The structures use internally the multipliers presented in the previous section. All the developed architectures were described in VHDL and synthesized to Nangate 45nm Open Cell library in fast operation conditions, using the Cadence Encounter RTL Compiler tool. Area (in terms of number of cells), and power consumption (in mW) are presented. For the power consumption, leakage and dynamic power are considered.

We have used a constraint of 10,000ps for the synthesis. A vector of 1,000 samples from the ECG signal contaminated with 60, 120, 180, and 240 Hz sinusoidal signals was used to simulate the architecture.

The hybrid ICS structures present more area than the binary ones, due to the extra EXOR gates used between the hybrid-binary conversions. Particularly, the ICS structure with NLMS adaptive filters present larger amount of cells due to the divider circuit used internally in the NLMS adaptive filter.

As we presented before (in Figure 8 and Figure 9), the LMS and NLMS adaptive filters are composed of two 20-bit multipliers. As the 20-bit hybrid multiplier presents less power than the binary one, as shown in Table II, therefore the use of 20-bit hybrid multipliers in the hybrid adaptive filters has largely impacted on the power reduction in the hybrid ICS structures, when compared with the binary ICS structures, as can be seen in Table VI. This aspect is enforced if we remember that the ICS structure uses internally four adaptive filters, as can be seen in Figure 2. In Table VI we can also clearly observe the main effect on reduction of the number of transitions in the hybrid adaptive filter what directly lead to its dynamic power consumption reduction.

Table VI. Results for binary and hybrid ICS structures.

Used Adaptive Filter	# Cells	Power (mW)		
		Leakage	Dynamic	Total
Binary LMS	24526	0.919	0.987	1.906
Hybrid LMS	27352	0.912	0.928	1.851
Binary NLMS	29061	1.181	3.620	4.801
Hybrid NLMS	33001	1.283	3.565	4.849

In terms of dynamic power consumption, the hybrid ICS structure using NLMS adaptive filter has presented less value than the binary ICS structure, what proves the efficiency of the hybrid signal, that present less transitions than the binary one, as stated in [16]. However, as the NLMS adaptive filter uses internally a divider circuit for the normalization process, as presented in Figure 9, this circuit was responsible for the larger leakage power value presented by the hybrid ICS structure, as compared with the binary ICS structure, as can be seen in Table VI.

It is also important to observe that the aspect of using a divider circuit leads the ICS structure with NLMS adaptive filter consumes more power when compared with the ICS with the LMS filter, as can be compared in Table VI. Note that we have no access to the internal divider circuit, since we have used the arithmetic circuit from the Cadence tool. As future work, we intend to test different dividers from the literature in order to optimize this circuit.

Another important aspect to be emphisized in Table VI is the large influence of the leakage power that is a trend in advanced technologies such that we are using in this work.

C. Comparisons with the Literature

In [18] the 2-Tap binary adaptive filter architecture was compared with the notch filter SFANC – Single-Frequency Adaptive Noise Canceller [19], and with the sequential delayed pipelined LMS-based adaptive FIR filter [20]. The results showed the smallest amount of arithmetic operators among the filters, less critical path than [19], and slightly larger critical path than [20].

The results for the 2-TAP LMS hybrid adaptive filter presented in [10] were almost the same, in terms of critical path, because although *W*/2 EXOR gates are used for the conversion between Binary and hybrid codes, only two EXOR gates are added in the critical path (one internally in the hybrid multiplier and another one in the output of 'MUX1' or 'MUX 3' in Figure 8).

In terms of the 2-TAP NLMS adaptive filter, one divider circuit is added in the critical path. However, the addittion of the EXOR gates, for the hybrid filter, is the same as in the LMS structure.

The results presented in [18] for binary and in [10] for hybrid adaptive filters were only FPGA-based. Moreover, only results for LMS adaptive filter were reported. In this work we also present results for NLMS adaptive filters. Moreover, we were able to synthesize the circuits using the commercial Cadence tool. The power results for 2-TAP binary and hybrid for both LMS and NLMS adaptive filters are presented in Table VII. The filters were synthesized to Nangate 45nm Open Cell library.

Table VII. Power results for 2-TAP binary and hybrid LMS and NLMS adaptive filters

2-TAP Adaptive Filter	Power (mW)		
	Leakage	Dynamic	Total
Binary LMS [18]	0.170	0.083	0.253
Hybrid LMS [10]	0.152	0.017	0.169
Binary NLMS [This work]	0.215	1.133	1.384
Hybrid NLMS [This work]	0.236	0.938	1.174

The results pesented in Table VII show that the hybrid encoding has enabled a large dynamic power reduction in the LMS and NLMS adaptive filters. In fact, the use of 20-bit hybrid multipliers has contributed for the large gains in power in the LMS and NLMS adaptive filters.

Note that due to the fast convergence speed, good stability, and accurate tracking capability characteristics, the NLMS adaptive filter can be a good choice for the harmonics power line interference. As we see in Table VII, the NLMS adaptive filters presented more power than the LMS adaptive filter. However, the NLMS adaptive filter presents faster convergence, as could be seen in Figure 15.

V. CONCLUSIONS

We introduced a hardwired Hybrid encoded structure for cancelling interferences from the power line in fundamental frequency and its high order harmonics, using LMS and NLMS adaptive filters. For the implementation of the fully structure, a new 23-bit Hybrid multiplier was proposed for the third harmonic generation (*XF*240). The efficiency of the Hybrid proposed structure was proved by efficiently filtering the interferences from the ECG signal. The proposed Hybrid adaptive filter also proved to be efficient in terms of power consumption reduction, when compared with the Binary structure. In terms of LMS versus NLMS adaptive filters tradeoff, it could be observed that the NLMS adaptive filter consumes more power, but it presents faster convergence than the LMS adaptive filter. Finally, the presented results show that it could be practicable to implement an harmonic power line interference cancelling structure operating on Hybrid encoding.

REFERENCES

[1] M. H. Costa, M. C. Tavares, "Removing harmonic power line interference from biopotential signals in low cost acquisition systems", *Computers in Biology and Medicine,* no.39, 2009. pp.519-526.

[2] M. F. Chimeno, R. Pallàs-Areny, "A comprehensive model for power line interference in biopotential measurements", *IEEE Transactions on Biomedical Engineering*, 49(3), 2000, pp.535-540.

[3] H. El-Sherief, H. Harberts, S. Pham, N. El-Sherief, "Design of a high resolution solid state ambulatory (holter) ECG recorder", in *Proceedings of IEEE 17th Annual Conference of the Engineering in Medicine and Biology Society*, 1995, pp. 1667-1668.

[4] E. Costa, J. Monteiro, S. Bampi, "A New Architecture for 2's Complement Gray Encoded Array Multiplier", in *15th Symposium on Integrated Circuits and Systems Design*, 2002, pp. 14-19.

[5] R. Ramos, A. Mànuel-Làzaro, J. Del Río, G. Olivar, "FPGA-Based Implementation of an Adaptive Canceller for 50/60-Hz Interference in Electrocardiography", *IEEE Transactions on Instrumentation and Measurement*, 56(6), 2007, pp. 2633-2640.

[6] Z. Zhou, J. He, Z. Liu, "FPGA-Implementation of LMS Adaptive Noise Canceller for ECG Signal Using Model Based Design", in *International Symposium on Bioelectronics and Bioinformatics*, 2011, pp. 127–130.

[7] M. Bahoura, H. Ezzaidi, "FPGA-Implementation of Parallel and Sequential Architectures for Adaptive Noise Cancelation", *Circuits, Systems, and Signal Processing*, 30(6), 2011, pp. 1521–1548.

[8] I. Homana, I. Muresan, M. Topa, and C. Contan, "FPGA Implementation of LMS and NLMS Adaptive Filters for Achoustic Echo Cancellation", *ACTA Technica Napocensis – Electronics and Telecommunications*, vol. 52, no. 4, 2011, pp.13-16.

[9] J. Dai, G. Zhang, and K. Zhao, "The FPGA Implementation of NLMS Adaptive Interference Algorithm", *Applied Mechanics and Materials*, vols. 128-129,2012, pp. 1105–1108.

[10] E. Costa, S. Almeida, M. Matzenauer, "Gray Encoded Fixed-Point LMS Adaptive Filter Architecture for the Harmonics Power Line Interference Cancelling", in *26th Symposium on Integrated Circuits and Systems Design*, 2013, pp. 1-6.

[11] S. Haykin, *Adaptive Filter Theory*, Prentice-Hall, *4th* ed, 2002.

[12] S. Kuo, B. Lee, *Real-Time Digital Signal Processing: Implementations, Applications, and Experiments with the TMS320C55X*, John Wiley & Sons, 2001.

[13] B. Widrow, J. Glover Jr., J. McCool, et al., "Adaptive noise cancelling: principles and applications", *Proceedings of the IEEE,* 63(12), 1975, pp. 1692-1716.

[14] E. Costa, S. Almeida, "Design of an Efficient FPGA-Based Interference Canceller Structure Using NLMS Adaptive Algorithm". in *IEEE 20th International Conference on Elecronics, Circuits, and Systems*, 2013, pp. 779-782.

[15] A. Chandrakasan, and R. Brodersen, *Low Power Digital CMOS Design*, Kluwer Academic Publishers, 1995.

[16] E. Costa, J. Monteriro, S. Bampi, "Power Efficient Arithmetic Operand Encoding", in *14th Symposium on Integrated Circuits and Systems Design*, 2001, pp. 201-206.

[17] G. Al, A. Lan, G. Hausdorff, et al, "Physiobank, Physiotoolkit, and Physionet: Components of a new research resource for complex physiologic signals", 2000.

[18] G. Seibel, F. Itturriet, E. Costa, and S. Almeida, "Fixed-Point Adaptive Filter Architecture for the Harmonics Power Line Interference Cancelling", In: *Proceedings of IEEE Fourth Latin American Symposium on Circuits and Systems*, 2013, pp. 1-4.

[19] R. Ramos, A. Mànuel-Làzaro, J. Del Río, G. Olivar, "FPGA-Based Implementation of an Adaptive Canceller for 50/60-Hz Interference in Electrocardiography", *IEEE Transactions on Instrumentation and Measurement*, 56(6), 2007, pp. 2633-2640.

[20] M. Bahoura, H. Ezzaidi, "FPGA-Implementation of Parallel and Sequential Architectures for Adaptive Noise Cancelation", *Circuits, Systems, and Signal Processing*, 30(6), 2011, pp. 1521–1548.