2006 IEEE Congress on Evolutionary Computation
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

An Incremental Evolutionary Strategy for the Design of FIR Filters Targeting Real-Time Applications

Evangelos F. Stefatos[1] and Tughrul Arslan[1, 2]
[1]School of Engineering & Electronics,
King's Buildings, Mayfield Rd,
Edinburgh, EH9 3JL, Scotland, UK
[2]Institute for System Level Integration,
Alba Centre, Alba Campus,
Livingston, EH54 7EG, Scotland, UK

{Evangelos.Stefatos,Tughrul.Arslan}@ee.ed.ac.uk

Abstract—**This paper introduces a new methodology for the design of finite impulse response filters within an evolvable hardware platform targeting real-time adaptation. The methodology incorporates a technique, which directs the search by focusing it into regions responsible for the evolution of individual coefficients. The methodology has been evaluated with three different paradigms that include two low-pass filters (29-order and 34-order, respectively) and a 41-order pass-band finite impulse response filter. The proposed method is compared with a conventional evolutionary strategy that has been commonly used for the implementation of digital filters resulting in significant improvements in terms of quality, convergence speed and computational efficiency.**

I. Introduction

THE employment of genetic algorithms (GAs) in the field of electronics has been proved to be successful to many applications where real-time system response, autonomous reconfiguration and fault-recovery are critical issues. However, there are several factors, which are related either with the main system or the implementation of the GA that create deceptive problems to the convergence of the algorithm and prevent it from achieving a global maximum.

This paper aims to develop new GA based strategies to efficiently resolve the configuration strings of reconfigurable devices used for the real-time evolution of finite impulse response (FIR) filters. In previous research such as [1], [2] the fitness-function of the employed GA considers simultaneously the total sum of the difference between the evolved and the reference coefficients. Moreover, in [3] the authors have evolved digital filters within an embedded reconfigurable platform, where the fitness function considers the total sum of the ratio between the two coefficients. However, in all these cases, the design of the reconfigurable hardware, in which evolution is performed and the fitness functions themselves, create dependencies between adjacent coefficients and therefore it is very likely for the GA to stick at sub-optimal solutions. In [4] the authors have proven that the evolution of FIR filters in frequency domain (evolving frequency response) requires fewer generations than evolution in time domain (evolving coefficient set). However, even the frequency domain evolution [5] is negatively affected by the hardware dependencies between adjacent coefficients.

In this paper we introduce a customized GA, which is applied on a specially tailored reconfigurable substrate and presents better results for evolving digital filters in terms of accuracy, convergence speed-up and more efficient hardware exploitation. The overall concept incorporates an evolutionary technique that is based on the interconnection scheme of our reconfigurable fabric and efficiently directs the evolution by focusing it into regions, which are responsible for the realization of individual coefficients.

The rest of this paper is organized as follows. Section II presents the reconfigurable hardware mean on which the evolution is performed and explains how the coefficients are realized within the configurable hardware components. Section III emphasizes the deceptive problems, which are created on common GAs and prevent them from reaching to global optimums. Subsequently, section IV presents our proposed GA and section V demonstrates the simulation results that prove the superiority of our GA in comparison with the one used in previous research work. Finally, section VI shows the savings in power, which can be achieved by using our evolutionary strategy and section V gives the conclusions of this paper.

0-7803-9487-9/06/$20.00/©2006 IEEE

II. Reconfigurable Hardware

A. Hardware Description

For the purpose of this paper there are two versions of the reconfigurable hardware mean, which is used for evolution. The former, which is employed for the first simulation paradigm, consists of 110 configurable-arithmetic-logic-units (CALUs), while the later is used for the second and third paradigm and it consists of 200 CALUs. Furthermore, in both versions there are several 2 to 1 multiplexers, which provide additional routing capabilities between non-adjacent CALUs. Figure 1 shows the distinctive arithmetic/logic parts that compose a single CALU. It can be seen that each CALU includes 6 configuration bits. Firstly the addition/subtraction unit (A/S unit) uses 1 bit that determines whether addition or subtraction operation is to be executed. Secondly, the 2 to 1 multiplexer attached on the output of the CALU regulates the creation of a new tap. Finally, the left shifter uses 4 configuration bits and therefore each shift module can perform binary multiplication from 2 up to 16384.

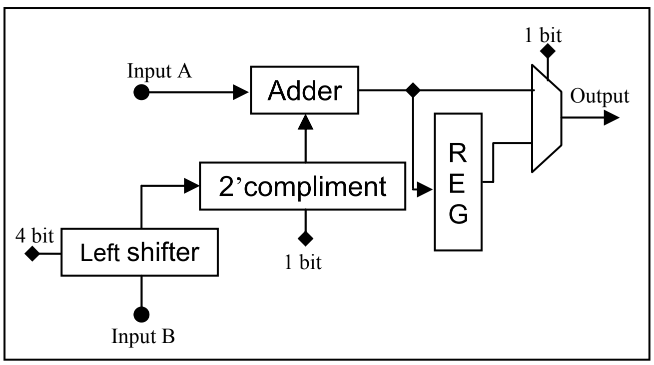

Fig. 1. CALU hardware description

The innovative part of this architecture is found on the hardware topology based on which the CALUs are interconnected. According to this the same CALU can be employed for both the accumulation and arithmetic stage of the filter.

Figure 2 illustrates how CALUs are interconnected. For better understanding of the interconnection scheme we have kept a consistent name coding for the input and output signals in figures 1 and 2. Furthermore, the usability of the multiplexers is very beneficial since the fault-robustness of the architecture is increased by providing routing flexibility and in addition to this FIR filters of different order can be realized by activating only the appropriate number of CALUs.

Another important issue in our reconfigurable fabric is that it does not create any hardware dependencies between adjacent coefficients because the realization of the coefficients does not utilize prior coefficient values.

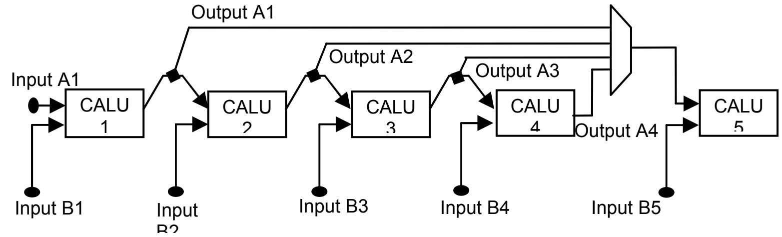

Fig. 2. Interconnection scheme between CALUs

B. Explanation of Filter Realization

In figure 2 it can be seen that each CALU has two 32-bit inputs. The upper one is always connected to input "A" of the A/S unit within the CALU as figure 1 implies and it performs the accumulation. On the other hand, the lower input of each CALU contributes to the realization of a specific coefficient, which in-turn will be accumulated. The creation of a new tap is done by storing the output of the A/S unit on the synchronous register, which is attached inside the CALU. A certain coefficient can use appropriately one or more CALUs in order to be realized. However, no matter is the total number of CALUs, which are used for the realization of the coefficient, only the last CALU utilizes the register. Hence, it can be deduced that data samples need only one clock cycle to be multiplied by any coefficient.

III. Mapping the Configuration Resolution to Common Genetic Algorithms

The employment of evolutionary methodologies to resolve the configuration string of such reconfigurable designs has been proved to be quite successful considering the complexity of the application and the autonomy that this method provides. However, in many cases conventional evolutionary strategies, which will be denoted in this paper as ES1, do not exploit efficiently the total number of the provided hardware resources and this results to the realization of electronic circuits, whose functionality deviates substantially from the ideal one. In other cases evolution reaches up to a satisfactory solution but then the convergence of the algorithm takes considerably more time than the time limits, which are set by the targeted application. The main source of this problem emerges from the objective function, which is employed to evaluate a configuration binary string for its capability to achieve the expected filter. Moreover, fitness function (1) is prone to create erroneous offsets between the ideal and the evolved coefficient in order to correct a created deviation between previously compared coefficients. This tendency usually results to the evolution of coefficient sets that give completely different frequency response compared with the expecting one.

$$fitness = \sum_{i=0}^{coef-1} \{coeff_{ideal\_i} - coeff_{evolved\_i}\} \quad (1)$$

IV. Proposed Evolutionary Strategy

This paper proposes an evolutionary strategy (ES2), which employs several techniques that suit the topology of our reconfigurable hardware, and in collaboration, the overall architecture achieves considerable improvements in terms of convergence speed-up of the algorithm, quality and power consumption of the evolved FIR filters, in comparison with ES1.

The most significant action that is hidden behind this methodology is the capability to reduce the search area of the GA by applying constraints on the part of the chromosome that is mutated during the evolution of a certain coefficient. On the other hand, crossover always occurs on the full size of the chromosome. Another difference of the fitness-function of ES2 with the one given in section III is that our algorithm considers only the difference/ratio of a single pair per time between the ideal and the evolved coefficient set that is currently realized. Hence, only when the first coefficient is fully realized the evolutionary algorithm proceeds to the next one. According to this methodology each coefficient can be precisely approached and there are no dependencies created between adjacent coefficients. Figure 3 depicts the flow diagram of ES2. The blocks within the shaded box compose the part of the algorithm that is in charge of assigning the minimum possible number of CALUs for the realizing of a coefficient. Hence, when the evolution of a new coefficient starts, ES2 always assigns two CALUs. However, if the evolved coefficient cannot approach the ideal one at a satisfactory percentage within the next 10 generations, then the algorithm increases by one the number of the assigned CALUs every 10 generations. In this manner the ES2 is prevented from utilizing redundant resources for the evolution of the entire filter. It must also be mentioned here that this algorithm cannot be employed in any reconfigurable design. In our design the CALUs are connected in a serial chain and the coefficients are starting to get evolved from the CALU on the right-end edge towards the left one. Hence, based on the registers that are used and the current maximum fitness-score, ES2 can easily calculate the total number of the employed CALUs at any point of the evolution.

Concerning the overall methodology of ES2, it utilizes the (μ+λ) strategy. According to this, the initial population consists of 50 chromosomes (parents) and then through random selection the population is extended to 100 by the addition of another 50 chromosomes (children). The rate of crossover that has been applied on the evolution of the filters with ES1 and ES2 was 90%. On the other hand the rate of mutation in the evolution of the filters differentiates because in the evolution with ES1 all the possible genes can be mutated, while in the evolution with ES2, mutation can be occurred only to a small part of the chromosome each time. Particularly, in the former scenario the rate was dynamically varying from 0 to 0.016%, while in the later the rate was varying from 0 to 0.22%. Subsequently, after evaluation and ranking the 50 survivors are selected (deterministic selection) to compose the new parent chromosomes.

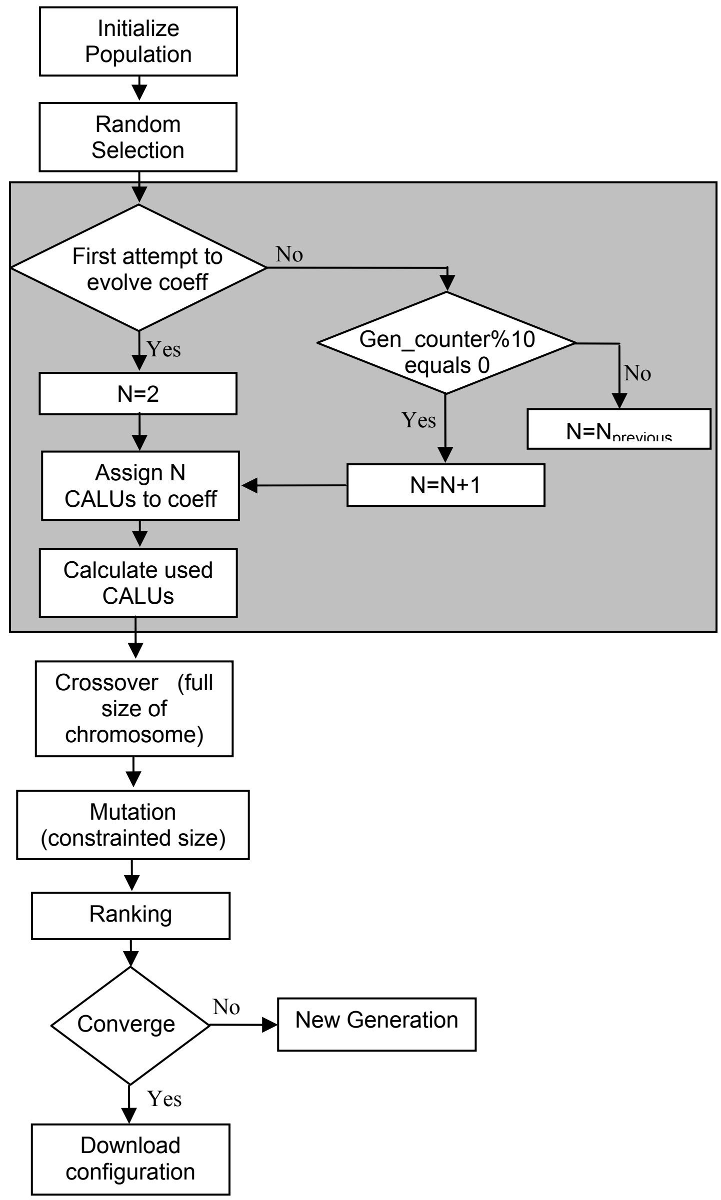

Fig. 3. Flow diagram of ES2

Moreover, figure 4 presents the encoding scheme that has been used in the proposed evolutionary methodology. According to this example CALUs 1 and 2 are employed to realize a coefficient with value 544 (32'h220). It can be seen that CALU 2 initially shifts the input data (logic "1") by 9 positions on the left and consequently it performs addition (A/S unit bit equals to 0). Furthermore, CALU 1 shifts the input data by 5 positions on the left and performs subtraction (A/S unit bit equals to 1). The multiplication of a data sample by a coefficient takes one clock cycle. Hence, only CALU 1 inserts a new tap in the realization of the filter and therefore the tap-creation bit equals to logic "1".

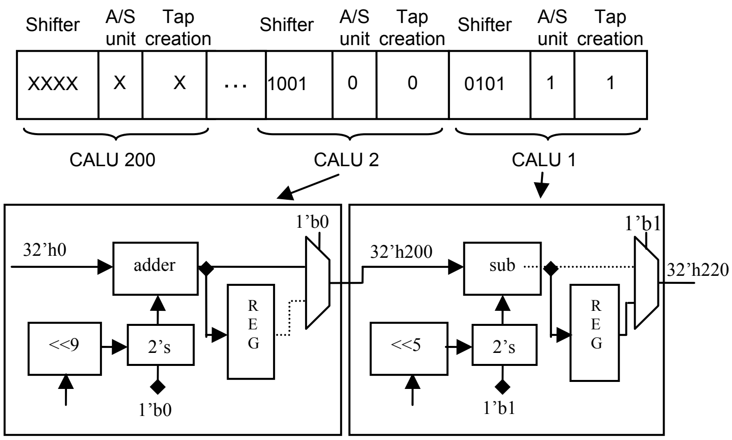

Fig. 4. Representation of the chromosome encoding scheme

V. SIMULATION RESULTS

For the comparison of the two strategies (ES1 and ES2) we employed the specification of three symmetrical FIR filters that are shown in Table 1. The specification of filter 1 and filter 2 composes two low-pass filters of 29 and 34 taps, respectively, while filter 3 is a 41-tap pass-band filter. Specifically, filter 1 presents a stop-band attenuation of 50 db. Moreover, the pass-band edge is at 0.2 rad/s, while the stop-band starts at 0.32 rad/s. Similarly, in filter 2 there is a stop-band attenuation of 78 db, the pass-band ends at 0.2 rad/s and the stop-band starts at 0.35 rad/s. Finally, filter 3 presents a stop-band attenuation of 60 and 80 db in the left and right stop-band region, respectively. Moreover, the first stop-band region ends at 0.3 rad/s and the corresponding pass-band starts at 0.42 rad/s. Subsequently, the pass-band region ends at 0.5 rad/s and the second stop-band starts at 0.62 rad/s.

For the purpose of this paper the reconfigurable hardware consists of 110 CALUs for the implementation of filter 1. For the evolution of filters 2 and 3 a version of 200 CALUs has been employed. In the first paradigm, the 110 CALUs proved to be sufficient for the evolution scenario with ES2, while simulations with ES1 proved that fitness-function (1) does not utilize appropriately the given hardware resources. It must be mentioned here that although the filter is a symmetrical one, in both simulations the coefficient set has been realized as in a direct form filter.

Figure 5 shows the comparison of the frequency response between the two evolved filters and the ideal one. It is obvious from the three graphs that ES2 evolved a low-pass filter, which approaches the ideal one with great accuracy.

TABLE I: IDEAL COEFFICIENT SETS

Filter 1	Filter 2	Filter 3
$C_{0,28} = -14$	$C_{0,33} = -29$	$C_{0,40} = 8$
$C_{1,27} = 46$	$C_{1,32} = -28$	$C_{1,39} = -87$
$C_{2,26} = 121$	$C_{2,31} = 91$	$C_{2,38} = -130$
$C_{3,25} = 205$	$C_{3,30} = 459$	$C_{3,37} = 117$
$C_{4,24} = 237$	$C_{4,29} = 1118$	$C_{4,36} = 173$
$C_{5,23} = 159$	$C_{5,28} = 1902$	$C_{5,35} = -349$
$C_{6,22} = -39$	$C_{6,27} = 2385$	$C_{6,34} = -202$
$C_{7,21} = -289$	$C_{7,26} = 2031$	$C_{7,33} = 1018$
$C_{8,20} = -451$	$C_{8,25} = 538$	$C_{8,32} = 419$
$C_{9,19} = -368$	$C_{9,24} = -1800$	$C_{9,31} = -2345$
$C_{10,18} = 49$	$C_{10,23} = -3940$	$C_{10,30} = -1308$
$C_{11,17} = 756$	$C_{11,22} = -4383$	$C_{11,29} = 4184$
$C_{12,16} = 1563$	$C_{12,21} = -1862$	$C_{12,28} = 3384$
$C_{13,15} = 2204$	$C_{13,20} = 3878$	$C_{13,27} = -5855$
$C_{14} = 2448$	$C_{14,19} = 11660$	$C_{14,26} = -6732$
	$C_{15,18} = 19140$	$C_{15,25} = 6370$
	$C_{16,17} = 23724$	$C_{16,24} = 10679$
		$C_{17,23} = -5016$
		$C_{18,22} = -13930$
		$C_{19,21} = 1911$
		$C_{20} = 15193$

On the other hand the filter which has been created with the guidance of ES1 (dotted line) presents similar pass-band characteristics (amplitude) and transition band with the ideal one. However, there is an obvious deviation on the stop-band attenuation, which is considerably smaller than the specified one. Figure 6 depicts the evolution process of the low-pass filter, which has been evolved with the aid of ES2. The total evolution lasts only 350 generations. Furthermore, whenever a coefficient is fully realized the fitness-score is increased by 100. In the meantime the fitness-score has a value from 0 to 99 according to how close is the currently evolved coefficient to the ideal one. From the graph it can be deduced that the number of generations needed for the realization of each coefficient is somehow proportional to the number of the CALUs, which are assigned to it.

On the other hand the evolution of the filter with ES1 lasted 800 generations. After that point the GA was not able to obtain any better configuration for the filter even by using different rates of crossover and mutation. It must be mentioned here that in figure 6 the maximum fitness-score that equals to 100 does not correspond to the best possible fitness value but to the best obtained fitness-score.

The key answer for this significant difference in the quality of the two evolved filters is that ES1 initially satisfies with any mean (use redundant hardware resources) the biggest coefficients within the coefficient set and then when it tries to fix the smaller ones there are either not sufficient resources left or due to the created hardware dependencies between adjacent coefficients, the GA sticks at sub-optimal solutions.

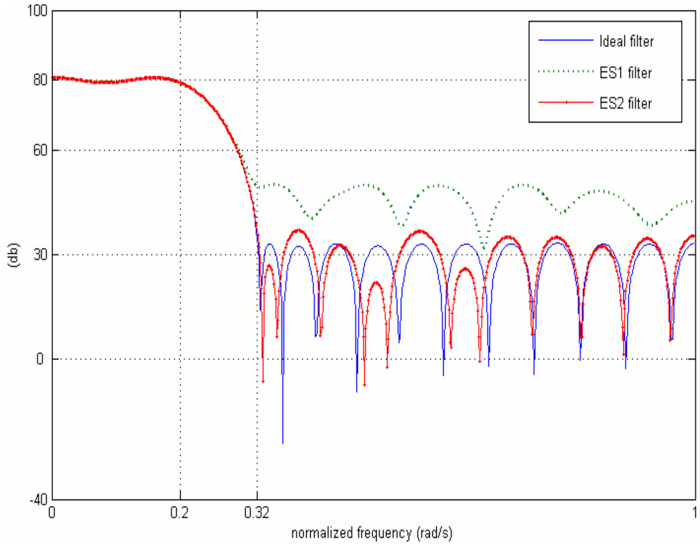

Fig. 5 Comparison of the ES1 and ES2 methodology for filter 1

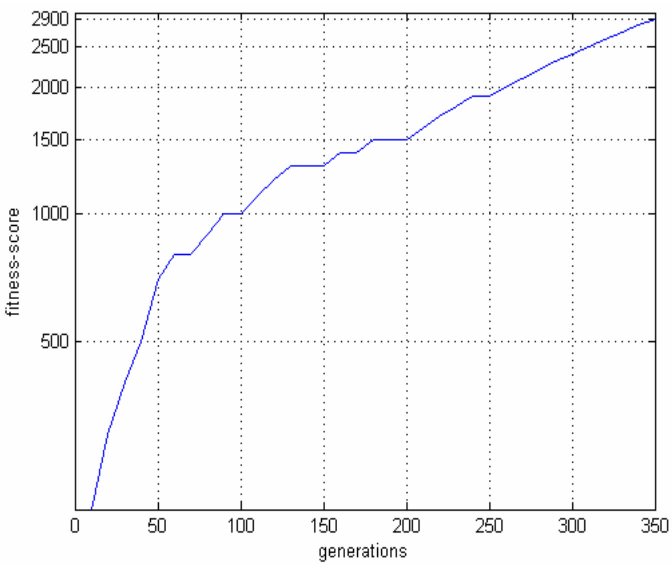

Fig. 6 Evolution process of ES2 strategy for filter 1

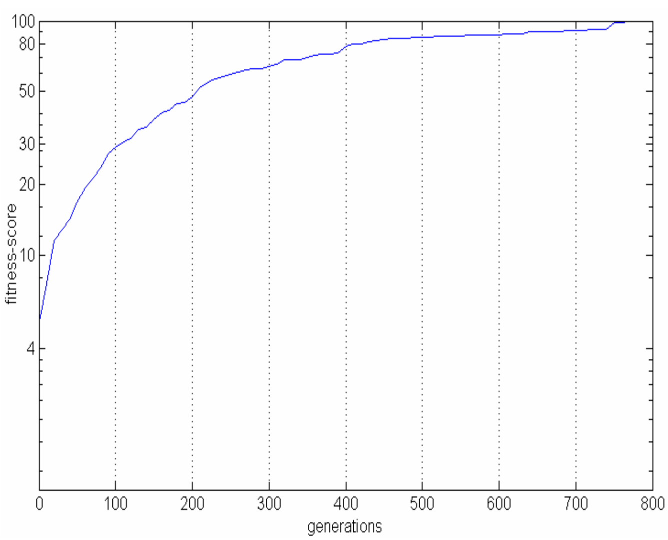

Fig. 7 Evolution process of ES1 strategy for filter 1

Moreover, the ES1 strategy completely failed to implement filter 2. In this paradigm ES1 guided the evolution to several erroneous configurations. Specifically, after 3200 generations it created several erroneous offsets between the ideal and the evolved values of coefficient pairs in order to correct offsets from previously evolved coefficients. Hence, the obtained frequency response was totally ruined and did not correspond to the requested specification. On the other hand, ES2 successfully evolved the filter 2 shown in Table I. Figure 8 shows the comparison between the frequency response of the ideal and the evolved filter. It can be seen that there is only a small deviation in the stop-band region for a very restricted number of frequencies. Finally, figure 9 depicts the capability of the two strategies (ES1 and ES2) to evolve filter 3. Again, it is obvious that ES1 fails to implement a filter that is close enough to the expected specification, although its evolution lasted almost three times more than ES2. On the other hand, it can be deduced from figure 9 that ES2 evolved a filter that accurately matches the specification. The evolution of ES2 lasted 1090 generations that is an impressive number considering the order of the filter and the bit-length of the coefficient set.

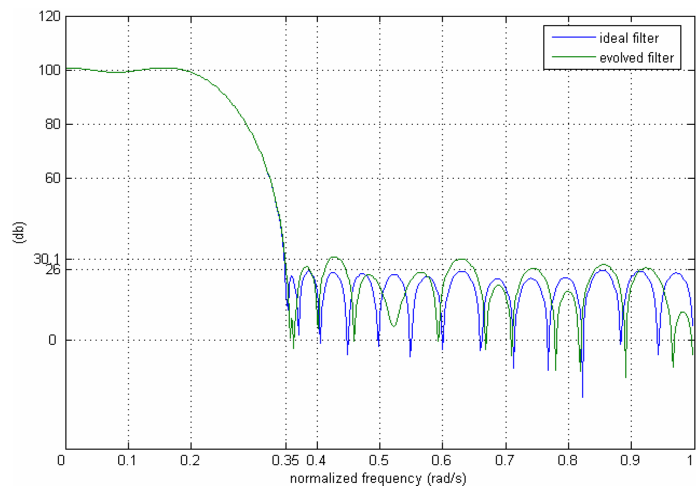

Fig. 8 Evolution process of ES2 strategy for filter 2

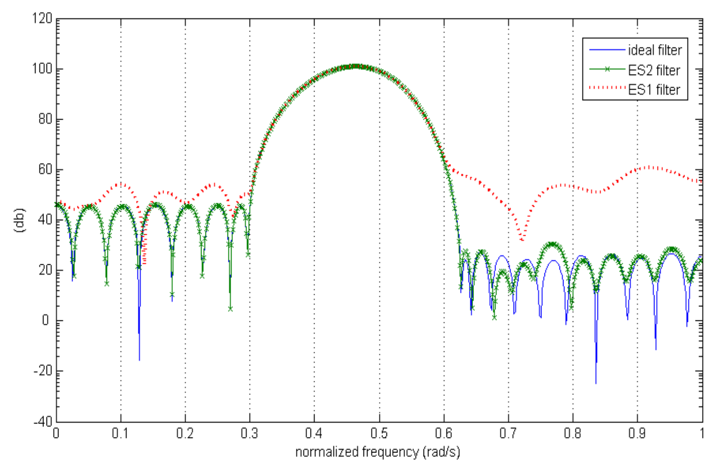

Fig. 9 Comparison of the ES1 and ES2 methodology for filter 3

VI. Power Analysis

According to the obtained results, the evolution of filter 1 with the guidance of ES2 occupies 78 CALUs in total. This implies that each coefficient needs in average 2.7 CALUs, which is a very impressive result considering the number of full-adders that a 32-bit dedicated multiplier requires. On the other hand, ES1 evolved a low-pass filter that occupies all the hardware resources (110 CALUs) of our reconfigurable design. This means that it employs 3.8 CALUs per coefficient.

In the second paradigm (filter 2) only ES2 managed to successfully evolve the targeted filter. According to this simulation, the evolved filter occupies 140 CALUs. Hence, the realization of each coefficient needs in average 4.1 CALUs. In comparison with the first paradigm there is an increase of 1.4 in the average number of CALUs. This is reasonable considering that in the first paradigm the bigger coefficient is represented by 12 bits, while in the second it needs 18 bits.

Finally, in the third paradigm ES2 evolved filter 3 by using 179 CALUs, while ES1 implemented a filter that significantly deviates from the targeted one by using 173 CALUs. Figure 10 summarizes the power that is consumed by the evolved filters. It also shows the savings in power that can be achieved when ES2 is employed for the evolution. However, the comparison in the third paradigm is not fair since ES1 did not implement an accurate filter.

The power results have been obtained by synthesizing the design of a single CALU with UMC 0.13μm technology cell library and then calculating the power consumed by each filter based on the total number of employed CALUs. It must be mentioned here that the power analysis in figure 10 corresponds to the power consumed only by the CALUs and not for the interconnections between them.

VII. Conclusion

In this paper we introduce an incremental evolutionary strategy that targets to improve the realization of digital FIR filters. This methodology incorporates techniques, which provide significant improvements in terms of accuracy, convergence speed of the algorithm and efficiency in hardware exploitation.

Simulation results have shown that our proposed evolution strategy implements more accurate filters than commonly used genetic methodologies and furthermore the algorithm needs from 1.7 to 3 times less time to converge.

Finally, the power analysis of the evolved filters shows that our incremental evolutionary strategy exploits more efficiently the hardware resources of the reconfigurable platform and hence this implies a considerable reduction in the power consumption.

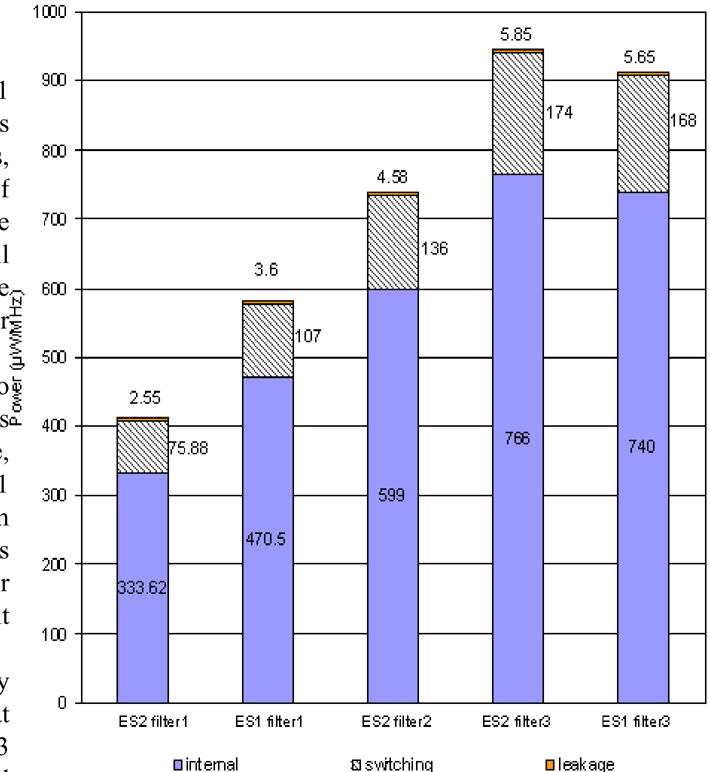

Fig. 10 Power analysis of the evolved filters using ES1 and ES2

References

[1] Vinger, K.A., Torresen, J., *Implementing evolution of FIR-filters efficiently in an FPGA,* Evolvable Hardware, 2003. Proceedings. NASA/DoD Conference on 9-11 July 2003, Page(s):26 – 29.

[2] Gunnar Tufte, Pauline C. Haddow., *Evolving an adaptive digital filter,* EH, The Second NASA/DoD Workshop on Evolvable Hardware, 2000, Page(s): 143.

[3] Hounsell B.I., Arslan T., and Thomson R., *Evolutionary design and adaptation of high performance digital filters within an embedded reconfigurable fault tolerant hardware platform,* Soft. Comp. – A Fusion of Found., Method. And Appl., Vol. 8, Issue 5, 2004, Page(s): 307-317.

[4] Stefatos E. F., Arslan T., Keymeulen D. and Ferguson I., *An EHWarchitecture for the design of unconstrained low-power FIR filters for sensor control using custom reconfigurable technology,* Evolvable Hardware. Proc. 2005 NASA/DoD Conf. on 29-1 June 2005, Page(s): 147-153.

[5] Stefatos E. F., Arslan T., Keymeulen D. and Ferguson I., *Custom reconfigurable architecture for autonomous fault-recovery of MEMS vibratory sensor electronics,* VLSID '06, 19th International Conference on VLSI Design held jointly with 5th International Conference on Embedded Systems Design, 2006, Page(s): 725-728.