Polymorphic FIR Filters with Backup Mode Enabling Power Savings

Lukas Sekanina and Richard Ruzicka and Zbysek Gajda
Faculty of Information Technology, Brno University of Technology
Bozetechova 2, 612 66 Brno, Czech Republic
E-mail: {sekanina;ruzicka;gajda}@fit.vutbr.cz

***Abstract*—A polymorphic FIR filter is proposed which can operate in two modes. The first mode is considered as a standard mode in which the filter performs a normal operation. In the second mode, the filter operates with reduced power supply voltage (Vdd), some filter coefficients are reconfigured (as response to the change of the polymorphic gates function which is controlled by Vdd) and some parts of the filter are disconnected. Experimental results indicate that while power consumption can significantly be reduced when half of the taps is suspended the filter is still able to achieve a reasonable quality of filtering.**

***Keywords*-digital filter; polymorphic gate; evolutionary design;**

I. Introduction

FIR (finite impulse response) filters and IIR (infinite impulse response) filters represent two important classes of digital filters that are utilized in many applications [1]. Various modifications of these filters have been proposed, including adaptive filters [2], multiplierless filters [3], [4], [5] and fault-tolerant filters [6]. In the recent years, the concept of polymorphic circuits has been developed [7], [8], [9], [10], [11], [12]. Polymorphic circuits contain polymorphic gates whose logic function can be switched unconventionally, for example, as a response to changes in the level of temperature, light or power supply voltage (Vdd). The utilization of polymorphic gates allows the polymorphic circuits to exhibit various additional functions to the main function required by specification. The additional function can be employed to enhance important features of the circuit, for example, testability, robustness, adaptability or security [11], [13], [14], [15], [16], [17].

In this paper, we propose to integrate polymorphic gates to digital filters. Proposed filters will perform two functions which will be switched using bifunctional polymorphic gates. The bifunctional polymorphic gates exhibit two logic functions. For example, the polymorphic NAND/NOR gate operates as NAND for a certain level of Vdd and as NOR for another level of Vdd [9], [11]. As the implementation cost of polymorphic gates is similar to the cost of conventional gates, they can be used as a technology that is capable of performing fast and resource-saving reconfiguration of the filter.

In particular, a polymorphic FIR filter is proposed which can operate in two modes. The first mode is considered as a standard mode in which the filter performs a normal operation. In the second mode, it is assumed that the filter works with reduced power budget (e.g., in a portable music player with low battery voltage). The filter operates with reduced Vdd, some filter coefficients are reconfigured (as response to the change of the polymorphic gates function which is controlled by Vdd) and some parts of the filter are disconnected. The goal is to reconfigure the filter in such a way that its original function is approximated as precisely as possible. The reconfiguration of coefficients is implemented using multiplierless polymorphic constant multipliers (MPCM). The goal of the paper is to demonstrate that proposed concept can lead to significant power reduction for a reasonable cost (in terms of lost filtering quality and overhead area on the chip).

The paper is organized as follows. Section II provides a brief introduction to IIR and FIR filtering. In Section III, the concept of polymorphic electronics is introduced. In particular, existing polymorphic gates suitable for the polymorphic filter are overviewed. Section IV deals with the description of proposed polymorphic filter and its implementation. Section V presents a case study – design and implementation of a polymorphic low pass FIR filter. Behavior of the filter, implementation cost and power requirements are analyzed in both modes of the filter. Discussion of obtained results is presented in Section VI. Conclusions are given in Section VII.

II. Digital Filters

A general IIR (infinite impulse response) digital filter is described by equation

$$y(n) = \sum_{k=0}^{N} b_k x(n-k) - \sum_{k=1}^{J} a_k y(n-k). \qquad (1)$$

The output samples $y(n)$ are derived from current and past input samples $x(n)$ as well as from current and past output samples. Designer's task is to propose values of coefficients a_k and b_k and size of vectors N and J. In FIR (finite impulse response) filters, the current output sample, $y(n)$, is a function only of past and present values of the input, i.e.

$$y(n) = \sum_{k=0}^{N} b_k x(n-k). \qquad (2)$$

The stability and linear phase are main advantages of FIR filters. On the other hand, in order to get a really good filter many coefficients have to be considered in contrast to IIR filters. In general, IIR filters are not stable (because of feedback). FIR filters are algebraically more difficult to synthesize.

Various methods have been proposed to design digital filters (such as frequency sampling method and window method for FIR filters and pole/zero placement and bilinear z-transform for IIR filters). These methods are well developed and represent an approach to digital filter design widely adopted by industry. Digital filters are usually implemented either on DSPs or as custom circuits. Their implementation is based on multipliers and adders. The quality of output signal, speed of operations and cost of hardware implementation are important factors in the design of digital filters. The multiplier is the primary performance bottleneck when implementing filters in hardware as it is costly in terms of area, power and signal delay. Hence multiplierless filters were introduced in which multiplication is reduced to a series of bitshifts, additions and subtractions [3], [4], [5].

Evolutionary algorithms have been utilized either to optimize filter coefficients [18] or to design complete filter structures from chosen components. In particular, structures of multiplierless filters were sought by many authors [3], [6], [19], [20]. Multiplierless filters are typically composed of adders, subtracters and shifters (implementing multiplication/division by the powers of two). Miller has pioneered an evolutionary approach in which unconventional filters are constructed at the gate level without the use of the multiply–accumulate structures [21].

III. Polymorphic Circuits

Polymorphic electronics was introduced by A. Stoica's group at JPL in 2001 [7]. A polymorphic gate is capable of switching among two or more logic functions. However, the selection of the function is performed unconventionally. The logic function depends on some external factors - e.g., on the level of the power supply voltage (Vdd), temperature, light, or some other external signals [7], [8], [9], [10], [11]. For example, a polymorphic gate exists which operates as NAND when Vdd=3.3V and as NOR when Vdd=1.8V. Another gate operates as AND when temperature is 27°C and as OR when temperature is 125°C.

Other polymorphic gates were developed whose logic function can be controlled by an external logic signal. For example, Ruzicka [14] proposed NAND/XOR gate which performs the NAND function when the external signal is connected to logic Low level, and the XOR function when the external signal is connected to logic High level. Since the external signal is the third logic input of the gate, we can consider these polymorphic gates as special instances of three-input gates. The main advantage of these gates is that they can be implemented as standard complementary CMOS structures. As these implementations are optimized for area, they contain fewer transistors than the structures based on multiplexing ordinary gates. For example, while the NAND/XOR gate consists of 9 transistors, its implementation utilizing ordinary NAND, XOR and MUX gates would cost at least 10 transistors [14].

In order to demonstrate the applications of polymorphic electronics, REPOMO32 chip has been developed [12]. Figure 1 shows the structure of the chip which consists of 32 two-input Configurable Logic Elements (CLEs) organized in an array of 4 rows and 8 columns. CLE can be programmed to perform one of the following functions: AND, OR, XOR and polymorphic NAND/NOR (controlled by Vdd). When Vdd = 3.3V the NAND/NOR gate exhibits the NOR function and when Vdd = 5V the gate exhibits the NAND function. REPOMO32's logic behavior is defined by its configuration bits and the level of Vdd. The configuration bits control a set of multiplexers which are responsible for interconnecting the CLEs and selecting their logic functions. In total, 8 bits define the configuration of a single CLE. The configuration of the chip is stored in 32 8-bit latch registers. The configuration of a single CLE is performed by supplying CLE's address (conf_addr) and configuration data (conf_data) followed by activating the WE signal. The chip can be completely reconfigured in 32 configuration steps. While the data4_out outputs are connected directly to CLEs of the fourth column the data8_out outputs are connected directly to CLEs of the last column. There are no synchronization registers in REPOMO32. The chip has 28 pins and occupies the area of 2900 x 1970 μm. It was fabricated using AMIS CMOS 0.7 μm technology. The functionality of multiplierless polymorphic constant multipliers will be demonstrated using REPOMO32.

IV. Proposed Polymorphic Filters

This section describes the concept of polymorphic digital filters and its possible implementations. Polymorphic FIR filter is considered as an example in this section.

A. Filter Description

Figure 2 shows proposed polymorphic FIR filter. The filter consists of $N-1$ delay registers R, N multiplication units and an N-operand adder which is divided into two subadders whose outputs are summed in the third adder. The filter can operate either in the standard mode or backup mode. The standard mode is used during normal operational conditions of the filter. In that case, the filter is operated as any conventionally created N-tap filter with coefficients $b_0 \dots b_{N-1}$. In the backup mode, the filter is switched using signal c to the configuration which approximates the standard mode using restricted resources. In this mode, the filter utilizes only $M, M < N$ coefficients ($b_0^* \dots b_{M-1}^*$) and $M-1$ delay registers. Therefore, in the backup mode,

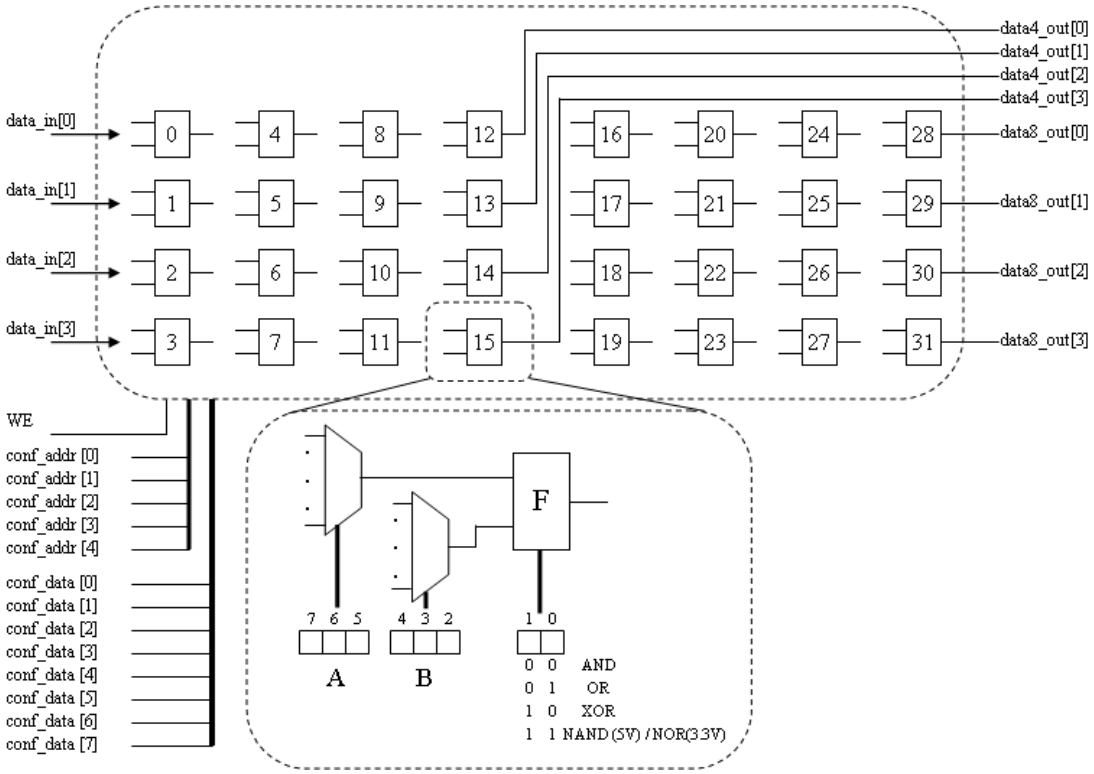

Figure 1. Architecture of REPOMO32

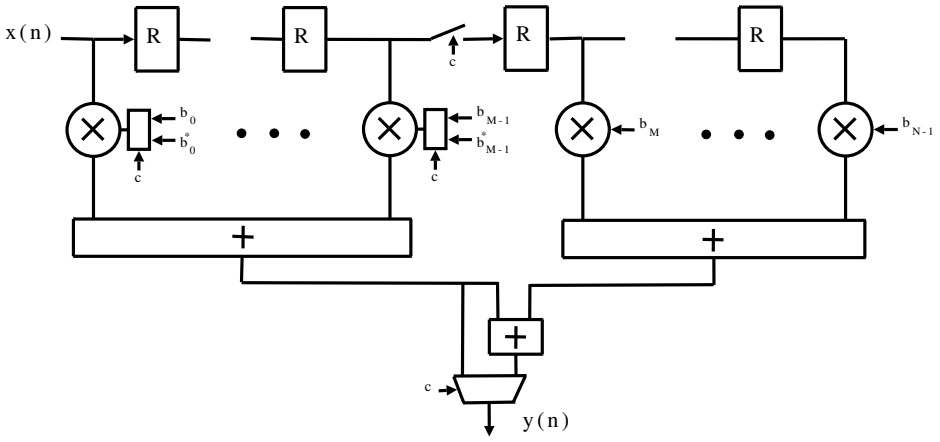

Figure 2. Polymorphic FIR filter

original coefficient values $b_0 \dots b_{M-1}$ are reconfigured and unused parts of the filter are disconnected.

B. Filter Design Principle

The coefficients of the polymorphic FIR filter can be designed using standard techniques. Note that in our case, the evolutionary algorithm is *not* utilized. Firstly, N coefficients $b_0 \dots b_{N-1}$ are calculated for a given specification which typically includes the sampling frequency, pass band frequency and stop band frequency for a low pass filter. Then, M coefficients ($b_0^* \dots b_{M-1}^*$) are calculated for the same specification; however, with M as the order of the filter.

C. Reconfiguration of the Filter

Reconfiguration, i.e. the change of the filter mode, includes the reconfiguration of the coefficients ($b_0 \rightarrow b_0^*, \dots, b_{M-1} \rightarrow b_{M-1}^*$), disconnecting remaining $N - M$ taps of the filter (including coefficients $b_M \dots, b_{N-1}$) and re-connection of the output port $y(n)$. The reconfiguration

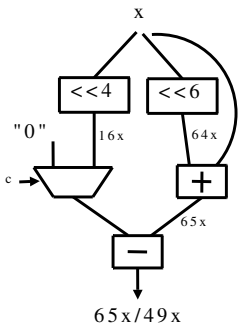

Figure 3. An example of two-mode multiplierless multiplier

of the filter coefficients can be implemented using ordinary logic circuits or polymorphic circuits. We will briefly review possible implementation techniques:

(1) The basic implementation can be based on multiplexing b_i and b_i^* using ordinary multiplexers. However, obtaining the resulting products using standard multipliers is area expensive.

(2) In order to reduce the area on the chip, digital filters are often implemented as multiplierless circuits using adders, subtractors and shifters. Figure 3 gives an example. The circuit calculates the product $65x$ and $49x$ depending on the control signal c (x is the input signal value). The cost for calculating the "backup" product ($49x$) is quite reasonable (1 shifter, 1 subtractor and 1 multiplexer). In order to even reduce the area on the chip, proposed circuit can further be optimized at the gate level.

(3) In this paper, we propose to implement the coefficient reconfiguration using polymorphic gates. Therefore, the aim is to design polymorphic circuits which calculate $b_i x$ in the first mode of polymorphic gates and $b_i^* x$ in the second mode of polymorphic gates ($i = 0 \ldots M-1$). The mode can be controlled using either a logic signal (c in Figure 2) or Vdd level. If the polymorphic gates are controlled using Vdd and unused taps can be disconnected simply by changing Vdd, the control signal c is not required and the filter mode can solely be controlled by the level of Vdd.

D. Power Reduction

It is expected that the backup mode can reduce power consumption and simultaneously provide a sufficient quality of filtering. In order to get a reduction in power consumption, the cost (in terms of power) of the M-tap filter which utilizes configurable coefficients must be lower than the cost of the original N-tap filter with invariable coefficients. This can be achieved by selecting a suitable order of the backup filter M and/or reducing Vdd in the backup mode. On the other hand, the standard mode will have slightly higher power consumption due to the utilization of reconfigurable multiplication part.

Table I
POLYMORPHIC LOW PASS FIR FILTER SPECIFICATION

sampling frequency	4000 Hz
pass band frequency	500 Hz
stop band frequency	550 Hz
standard mode	N = 20 taps
backup mode	M = 10 taps
input signal	8 bit/sample
coeff. width	8 bits

V. EXPERIMENTAL RESULTS

A simple polymorphic low pass FIR filter is presented as a case study in this section. The specification of the filter is given in Table I. Note that a relatively low filter order was chosen in order to easily implement (part of) the filter in REPOMO32.

A. Filter Description

The implementation is based on the polymorphic FIR filter shown in Figure 2. In our case, $N = 20$ and $M = 10$. Table II (part "poly. FIR") summarizes filter coefficients for both modes. These coefficients were calculated using Digital Filter Analyzer v 2.6 and quantized to 8 bits (2'complement code). Figure 4 shows the filter response for the standard mode. Once the filter mode is switched to the backup mode the filter response is changed (see Figure 5). The quality of filtering is worse than in the standard mode; however, the filter still exhibits the required low pass function.

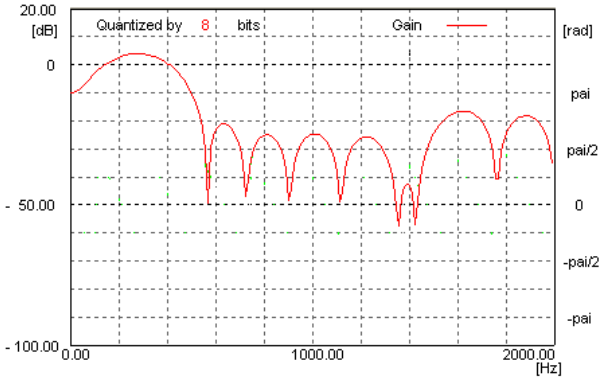

Figure 4. FIR filter response in the standard mode (N=20)

B. Design of the polymorphic filter

The circuits calculating $b_i x$ in the standard mode and $b_i^* x$ in the backup mode are implemented using polymorphic and ordinary gates. Figure 6 shows one of them which calculates $F0x/20x$h. In addition to ordinary gates, all presented MPCMs contain only the polymorphic NAND/NOR gates because only these gates are available in REPOMO32. The method used to design polymorphic circuits such as MPCMs was described in [22]. It requires the following steps to design a single MPCM:

Table II
THE COEFFICIENTS AND THE NUMBER OF GATES FOR CONVENTIONAL FIR FILTERS WITH N=20 AND N=10 AND FOR THE POLYMORPHIC FIR FILTER IN STANDARD MODE AND BACKUP MODE (ONLY ACTIVE GATES COUNTED)

tap	conv. FIR N=20		conv. FIR N=10		poly. FIR (standard)		poly. FIR (backup)	
	b_i	gates	b_i	gates	b_i/b_i^*	gates	b_i^*	gates (polymorphic)
0	00h	0	08h	0	00/08h	10	08h	10 (2)
1	F7h	67	10h	0	F7/10h	67	10h	67 (16)
2	F6h	64	18h	32	F6/18h	93	18h	93 (28)
3	F0h	26	20h	0	F0/20h	27	20h	27 (4)
4	F4h	26	22h	32	F4/22h	71	22h	71 (21)
5	F8h	26	22h	32	F8/22h	72	22h	72 (13)
6	02h	0	20h	0	02/20h	23	20h	23 (8)
7	10h	0	18h	32	10/18h	41	18h	41 (11)
8	18h	32	10h	0	18/10h	42	10h	42 (11)
9	20h	0	08h	0	20/08h	24	08h	24 (14)
10	20h	0			20h	0		
11	18h	32			18h	32		
12	10h	0			10h	0		
13	02h	0			02h	0		
14	F8h	26			F8h	26		
15	F4h	26			F4h	26		
16	F0h	26			F0h	26		
17	F6h	64			F6h	64		
18	F7h	67			F7h	67		
19	00h	0			00h	0		
multiplication		482		128		711		470
10-input adders		666		333		666		333
output adder		77				77		
mux y						32		32
registers		912		432		912		432
total		2137		893		2398		1267
relative cost		1.00		0.42		1.12		0.59

- Two (independent) circuits calculating $b_i x$ (first one) and $b_i^* x$ (second one) are designed according to Section V-C.
- The outputs of resulting circuits are interconnected using polymorphic multiplexers in order to obtain MPCM.
- Cartesian genetic programming (CGP) [23] is applied to minimize the number of gates for MPCM.

CGP was used with the following parameters:

- Population size: 15
- Circuit topology: $1 \times u$, where $u = g_1 + g_2 + 7w$, g_1 is the number of gates for $b_i x$, g_2 is the number of gates for $b_i^* x$ and w is the number of circuit outputs.
- Chromosome length: $3u + w$ integers
- Level-back parameter: u
- Max. generations: 50×10^6
- Mutation rate: 7 genes/chromosome on average

The fitness value is defined as:

$$fitness = B_1 + B_2 \tag{3}$$

where B_1 (resp. B_2) is the number of correct output bits for $b_i x$ (resp. $b_i^* x$) obtained as a response for all possible input combinations. Table II summarizes the best implementation costs for $b_i x / b_i^* x, i = 0 \ldots 9$ obtained out of 10 independent runs of CGP (the number of NAND/NOR gates is given in parenthesis in the last column of Table II).

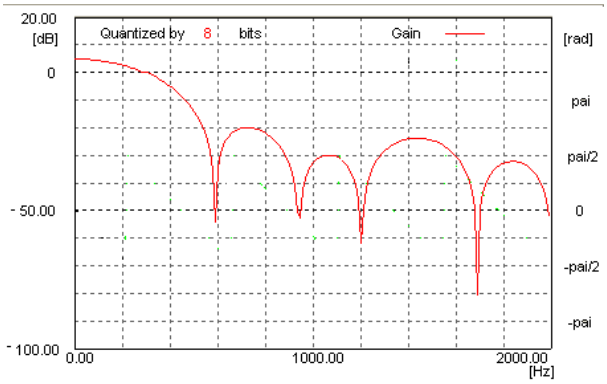

Figure 5. FIR filter response in the backup mode (N=20, M=10)

C. Implementation cost of the non-reconfigurable filter

In order to compare the proposed solution with a conventional solution, we have to calculate the cost of the original (non-reconfigurable) version of the filter (N=20). The multiplication operations are implemented using adders, inverters and shifters. We assume that shifters are for free, 5 gates are required to build a full adder (FA) and 2 gates are required to build a half adder (HA). For example, the circuit outputting $18x$ (18 is a hexa number here) can be implemented as $(x << 4) + (x << 3)$, i.e. using 32 gates (6 FAs, 1 HA). Table II (part "conv. FIR N=20")

summarizes the cost of constant coefficient multipliers for the filter with $N = 20$. This multiplication part will cost 482 two-input gates. The cost of a delay register is estimated as 6 two-input gates/bit. In total, 912 two-input gates are required to implement 19 eight-bit registers. Finally, the products $b_0x \dots b_{19}x$ are summed using two 10-operand tree adders connected together using another adder which is composed of 77 gates. Each of the 10-operand adders requires 333 gates to be implemented. In total, we can estimate the implementation cost of the non-reconfigurable filter as 2137 two-input gates. For comparison, Table II (part "conv. FIR N=10") also gives the implementation cost of a 10-tap solution.

D. Implementation cost of the polymorphic filter

Table II (part "poly. FIR (standard)") gives the number of gates of the polymorphic FIR filter. We can observe a higher cost for taps ($b_0 \dots b_9$) because these taps are now reconfigurable using polymorphic gates (711 gates). The summing part of the filter consists of three adders (333 + 333 + 77 gates). The output value $y(n)$ is obtained using a polymorphic multiplexer (32 gates). In total, the filter is composed of 2398 gates. As all these gates are active in the standard mode we can estimate that its power consumption will be higher (approx. 12%) in comparison to a conventional 20-tap filter.

Although the polymorphic filter utilizes only 10 taps in the backup mode, the number of active gates that are required for the implementation of MPCM (470 gates) is almost identical with the number of gates in the multiplication part of the 20-tap filter (482 gates). However, the main power reduction of the backup mode comes from the reduction of the number of delay registers and adders. The backup mode utilizes only one 10-operand adder and polymorphic multiplexer which represent 333+32 active gates. Other 432 two-input gates are required to implement 9 eight-bit delay registers. In total, we can estimate that the backup mode requires 1267 active gates. In comparison with a conventional 20-tap filter, the reduction is 40%. However, in comparison with a conventional 10-tap filter, the increase in the number of gates is almost 42%.

E. Power consumption

A rough estimate of power consumption can be made on the basis of the number of active gates. While 2398 two-input gates are utilized in the standard mode of the filter, the backup mode utilizes only 1267 gates. That difference would represent the reduction of 47.2% in the power consumption. Another reduction can be obtained if the backup mode operates with lower Vdd than the standard mode. By comparing power consumptions in both modes we can observe that the backup mode uses only

$$\frac{P_{3.3V}}{P_{5V}} = \frac{kq(3.3)^2}{k(5)^2} = 23.0\% \qquad (4)$$

of the original power budget, where k represents the product of dynamic capacitance and operating frequency (k is assumed to be very similar in both modes) and $q = 1267/2398$ is the reduction in the number of active gates.

VI. Discussion

We have shown that polymorphic gates can be employed to effectively implement the circuits which ensure the reconfiguration of filter coefficients. When polymorphic gates controlled by Vdd are utilized the filter reconfiguration can be performed by a simple change in the Vdd level. The filter is switched to the backup mode in which the standard mode is approximated using fewer taps. The same concept may be applied to modify the filter stop band frequency in the backup mode or even change the type of filter (in order to obtain, e.g., a high-pass filter).

A preliminary analysis has shown a significant decrease of power consumption. However, more precise power analysis based on the switching activity of gates, including dynamic analysis (logic transitions, glitches) and static analysis (leakage currents) has to be performed in order to really validate the proposed concept. Another power reduction can be achieved by reducing the power supply voltage in the backup mode. It assumes that polymorphic gates controlled by Vdd are utilized. In our calculations, we have also assumed that polymorphic gates have the same electrical properties (power consumption, area, delay) as conventional gates. That is not necessarily true for existing polymorphic gates.

The main difficulty in the design methodology is the design of area-efficient MPCMs. Increasing the number of bits of MPCMs will lead to decreasing the efficiency of the evolutionary design. Hence, more scalable design method has to be addressed in future research.

VII. Conclusions

In this paper, we have presented the concept of polymorphic FIR filter equipped with a backup mode. Behavior of the filter was simulated in both modes. The implementation cost and power consumption were analyzed using a simplified methodology. An example of MPCM circuit was presented. Estimated power reduction is 77% in the backup mode for a polymorphic low pass FIR filter which utilizes 20 taps in the standard mode and 10 taps in the backup mode. Our future work will be devoted to the physical implementation and evaluation of the complete filter. Dynamic parameters of the filter will also be measured.

Acknowledgements

This research was partially supported by the Grant Agency of the Czech Republic under No. 102/07/0850 *Design and hardware implementation of a patent-invention machine* and the Research Plan No. MSM 0021630528 *Security-Oriented Research in Information Technology*.

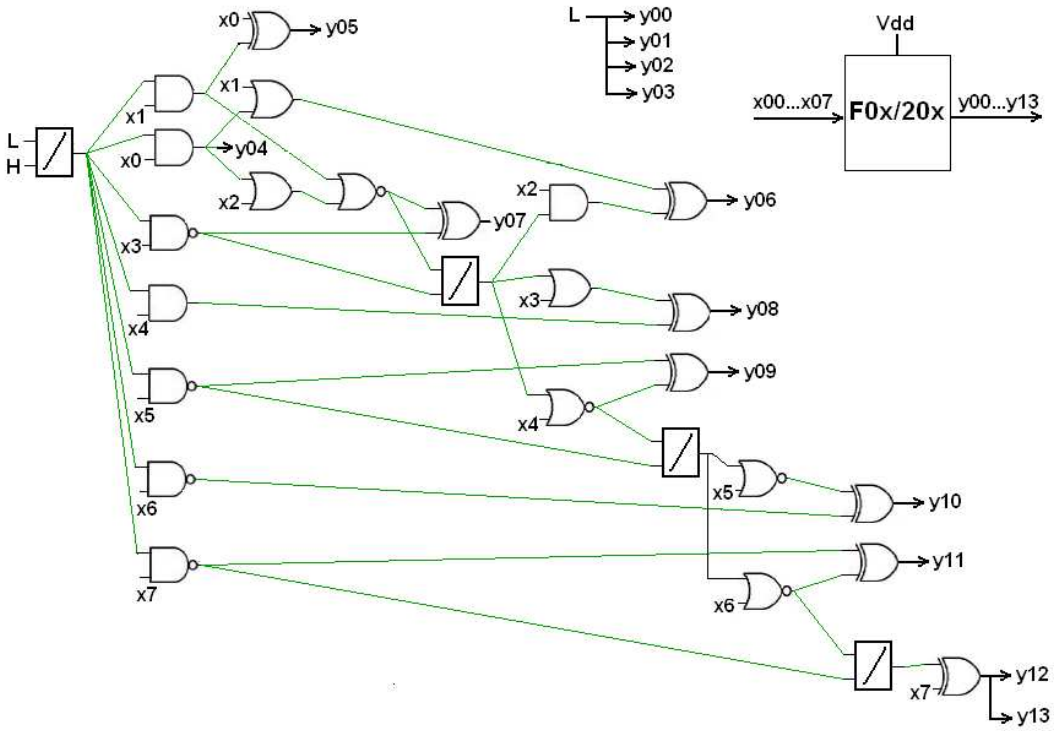

Figure 6. Polymorphic constant coefficient multiplier F0/20h. The NAND/NOR gates are shown as rectangles. L denotes Low and H denotes High voltage levels

REFERENCES

[1] E. Ifeachor and B. Jervis, *Digital Signal Processing A Practical Approach (2nd ed.)*. Prentice-Hall, 2002.

[2] B. Farhang-Boroujeny, *Adaptive Filters: Theory and Applications*. John Wiley and sons Ltd., 1998.

[3] G. Wade, A. Roberts, and G. Williams, "Multiplier-less fir filter design using a genetic algorithm," *IEE Proceedings in Vision, Image and Signal Processing*, vol. 141, no. 3, pp. 175–180, 1994.

[4] M. Martinez-Peiro, E. I. Boemo, and L. Wanhammar, "Design of high-speed multiplierless filters using a nonrecursive signed common subexpression algorithm," *IEEE Trans. Circuits Syst. II*, vol. 49, no. 3, pp. 196–203, 2002.

[5] D. L. Maskell, "Multiplierless multiple constant multiplication," *IET Circuits Devices Syst.*, vol. 1, no. 2, pp. 175–180, 2007.

[6] B. I. Hounsell, T. Arslan, and R. Thomson, "Evolutionary design and adaptation of high performance digital filters within an embedded reconfigurable fault tolerant hardware platform," *Soft Computing*, vol. 8, no. 5, pp. 307–317, 2004.

[7] A. Stoica, R. S. Zebulum, and D. Keymeulen, "Polymorphic electronics," in *Proc. of Evolvable Systems: From Biology to Hardware Conference*, ser. LNCS, vol. 2210. Springer, 2001, pp. 291–302.

[8] A. Stoica, R. S. Zebulum, D. Keymeulen, and J. Lohn, "On polymorphic circuits and their design using evolutionary algorithms," in *Proc. of IASTED International Conference on Applied Informatics AI2002*, Insbruck, Austria, 2002.

[9] A. Stoica, R. Zebulum, X. Guo, D. Keymeulen, I. Ferguson, and V. Duong, "Taking Evolutionary Circuit Design From Experimentation to Implementation: Some Useful Techniques and a Silicon Demonstration," *IEE Proc.-Comp. Digit. Tech.*, vol. 151, no. 4, pp. 295–300, 2004.

[10] R. S. Zebulum and A. Stoica, "Four-Function Logic Gate Controlled by Analog Voltage," *NASA Tech Briefs*, vol. 30, no. 3, p. 8, 2006.

[11] R. Ruzicka, L. Sekanina, and R. Prokop, "Physical demonstration of polymorphic self-checking circuits," in *Proc. of 14th IEEE International On-Line Testing Symposium*. IEEE, 2008, pp. 31–36.

[12] L. Sekanina, R. Ruzicka, Z. Vasicek, R. Prokop, and L. Fujcik, "Repomo32 – new reconfigurable polymorphic integrated circuit for adaptive hardware," in *2009 IEEE Workshop on Evolvable and Adaptive Hardware*. IEEE Computational Intelligence Society, 2009, pp. 39–46.

[13] L. Sekanina, L. Starecek, Z. Kotasek, and Z. Gajda, "Polymorphic gates in design and test of digital circuits," *International Journal of Unconventional Computing*, vol. 4, no. 2, pp. 125–142, 2008.

[14] R. Ruzicka, "On bifunctional polymorphic gates controlled by a special signal," *WSEAS Transactions on Circuits*, vol. 7, no. 3, pp. 96–101, 2008.

[15] R. S. Zebulum and A. Stoica, "Multifunctional Logic Gates for Built-In Self-Testing," *NASA Tech Briefs*, vol. 30, no. 3, p. 10, 2006.

[16] ——, "Ripple Counters Controlled by Analog Voltage," *NASA Tech Briefs*, vol. 30, no. 3, p. 2, 2006.

[17] L. Sekanina, "Evolution of Polymorphic Self-Checking Circuits," in *Proc. of the 7th Conf. on Evolvable Systems: From Biology to Hardware*, ser. LNCS, no. 4684. Wuhan, China: Springer, 2007, pp. 186–197.

[18] S. P. Harris and E. C. Ifeachor, "Automating IIR filter design by genetic algorithm," in *Proc. of the First IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA'95)*, no. 414. IEE, 1995, pp. 271–275.

[19] M. Erba, R. Rossi, V. Liberali, and A. Tettamanzi, "An Evolutionary Approach to Automatic Generation of VHDL Code for Low-Power Digital Filters," in *Proc. of the 4th European Conference on Genetic Programming EuroGP2001*, ser. LNCS, vol. 2038. Springer Verlag, 2001, pp. 36–50.

[20] Y. Voronenko and M. Puschel, "Multiplierless multiple constant multiplication," *ACM Transactions on Algorithms*, vol. 3, no. 2, pp. 1–282, 2007.

[21] J. F. Miller, "On the filtering properties of evolved gate arrays," in *Proc. of the 1st NASA/DoD Workshop on Evolvable Hardware*. IEEE Computer Society, 1999, pp. 2–11.

[22] Z. Gajda and L. Sekanina, "Gate-level optimization of polymorphic circuits using cartesian genetic programming," in *IEEE Congress on Evolutionary Computation*. IEEE Computational Intelligence Society, 2009, pp. 1–6.

[23] J. F. Miller, D. Job, and V. K. Vassilev, "Principles in the Evolutionary Design of Digital Circuits – Part I," *Genetic Programming and Evolvable Machines*, vol. 1, no. 1, pp. 8–35, 2000.